Preference Elicitation in Prioritized Skyline Queries

Denis Mindolin · Jan Chomicki

Received: date / Accepted: date

Abstract Preference queries incorporate the notion of binary preference relation into relational database querying. Instead of returning all the answers, such queries return only the best answers, according to a given preference relation.

Preference queries are a fast growing area of database research. Skyline queries constitute one of the most thoroughly studied classes of preference queries. A well known limitation of skyline queries is that skyline preference relations assign the same importance to all attributes. In this work, we study p-skyline queries that generalize skyline queries by allowing varying attribute importance in preference relations.

We perform an in-depth study of the properties of p-skyline preference relations. In particular, we study the problems of containment and minimal extension. We apply the obtained results to the central problem of the paper: eliciting relative importance of attributes. Relative importance is implicit in the constructed p-skyline preference relation. The elicitation is based on user-selected sets of superior (positive) and inferior (negative) examples. We show that the computational complexity of elicitation depends on whether inferior examples are involved. If they are not, elicitation can be achieved in polynomial time. Otherwise, it is NP-complete. Our experiments show that the proposed elicitation algorithm has high accuracy and good scalability.

Keywords preference · preference query · skyline · p-skyline

1 Introduction

Effective and efficient user preference management is a crucial part of any successful sales-oriented business. Knowing what customers like and more importantly why they like that and what they will like in the future is an essential part of the modern risk management process. The essential components of preference management include preference specification, preference elicitation, and querying using preferences. Many preference handling frameworks have been developed [Bőrzsönyi et al(2001), Kießling and Köstler(2002), Brafman and Domshlak (2002), Chomicki(2003), P. Pu and Torrens(2003), Hansson(1995), Fishburn(1970)].

Our starting point here is the skyline framework [Bőrzsönyi et al(2001)]. The skyline preference relation is defined on top of a set of preferences over individual attributes. It represents the Pareto improvement principle: a tuple $o_1$ is preferred to a tuple $o_2$ iff $o_1$ is as good as $o_2$ according to all the attribute preferences, and $o_1$ is strictly better than $o_2$ according to at least one attribute preference. Now given a set of tuples, the set of the best tuples according to this principle is called a skyline.

Example 1 Assume the following cars are available for sale.

| make | price | year |
|------|-------|------|
| ford | 30k   | 2007 |
| bmw  | 45k   | 2008 |
| kia  | 20k   | 2007 |
| ford | 40k   | 2008 |
| bmw  | 50k   | 2006 |

Also, assume that Mary wants to buy a car and her attribute preferences are as follows:
BMW is better than Ford, Ford is better than Kia
the car should be as new as possible
the car should be as cheap as possible.

Then the skyline is

| make | price | year |
|------|-------|------|
| t1   | ford  | 30k  | 2007 |
| t2   | bmw   | 45k  | 2008 |
| t3   | kia   | 20k  | 2007 |
| t4   | ford  | 40k  | 2008 |

A large number of algorithms for computing skyline queries have been developed [Börzsönyi et al(2001), Chomicki et al(2003), Godfrey et al(2005), Lin et al(2005)]. Elicitation of skyline preference relations based on user-provided feedback has also been studied [Jiang et al(2008)].

One of the reasons of the popularity of the skyline framework is the simplicity and intuitiveness of skyline semantics. Indeed, in order to define a skyline preference relation, one needs to provide only two parameters: the set $A$ of relevant attributes and the set $H$ of corresponding preferences over each individual attribute in $A$. (In Example 1, $A = \{\text{make, price, year}\}$ and $H = \{><\text{make},><\text{price},><\text{year}\}$.)

At the same time, the simplicity of skyline semantics comes with a number of well known limitations. One of them is the inability of skyline preference relations to capture the important notion of difference in attribute importance. The Pareto improvement principle implies that all relevant attributes have the same importance. However, in real life, it is often the case that benefits in one attribute may outweigh losses in one or more attributes. For instance, given two cars that differ in age and price, for some people the age is crucial while the price is secondary. Hence, in that case, the price has to be considered only when the benefits in age cannot be obtained, i.e., when the age of the two cars is the same.

**Example 2** Assume that Mary decides that year is more important for her than make and price, which in turn are equally important. Thus, regardless of the values of make and price, a newer car is always better than an old one. At the same time, given two cars of the same age, one needs to compare their make and price to determine the better one. The set of the best tuples according to this preference relation is

| make | price | year |
|------|-------|------|
| t2   | bmw   | 45k  | 2008 |
| t4   | ford  | 40k  | 2008 |

Namely, $t_2$ and $t_4$ are better than all other tuples in year, but $t_2$ is better than $t_4$ in make, and $t_4$ is better than $t_2$ in price.

Another drawback of the skyline framework is that the size of a skyline may be exponential in the number of attribute preferences [Godfrey(2004)]. A query result of that size is likely to overwhelm the user. In interactive preference elicitation scenarios [Balke et al(2007)], user preferences are elicited in a stepwise manner. A user is assumed to analyze the set of the best tuples according to the intermediate preference relation and criticize it in some way. Clearly, if such a tuple set is too large, it is hard to expect high quality feedback from the user. The large size of a skyline is caused by the looseness of the Pareto improvement principle. Pareto improvement implies that if a tuple $o$ is better than $o'$ in one attribute, then the existence of an attribute in which $o'$ is better than $o$ makes the tuples incomparable. Thus, every additional attribute increases the number of incomparable tuples.

Here we develop the p-skyline framework which generalizes the skyline framework and addresses its limitations listed above: the inability to capture differences in attribute importance and large query results. The skyline semantics is enriched with the notion of attribute importance in a natural way. Assuming two relevant attributes $A$ and $B$ such that $A$ is more important than $B$, a tuple with a better value of $A$ is unconditionally preferred to all tuples with worse values of $A$, regardless of their values of $B$. However, given a tuple with the same value of $A$, the one with a better value of $B$ is preferred (assuming no other attributes are involved). For equally important attributes, the Pareto improvement principle applies. Therefore, skyline queries are also representable in our framework.

Relative attribute importance implicit in a p-skyline preference relation is represented explicitly as a p-graph: a graph whose nodes are attributes, and edges go from more to less important attributes. Such graphs satisfy the properties quite natural for importance relationships: transitivity and irreflexivity. We show that, in addition to representing attribute importance, p-graphs play another important role in the p-skyline framework: they can be used to determine equivalence and containment of p-skyline relations, and tuple dominance.

We notice that two p-skyline relations may differ in the following aspects:

- the set $A$ of relevant attributes,
- the set $H$ of preferences over those attributes, and
- the relative importance of the corresponding attributes, represented by a p-graph.

In this work, we are particularly interested in the class $F_{3F}$ of full p-skyline relations for which the set of relevant attributes $A$ consists of all the attributes and the set of corresponding attribute preferences is $H$. Hence, two different p-skyline relations from $F_{3F}$ are different only in the corresponding p-graphs. We show the following properties of such relations:
– the containment and equivalence of p-skyline relations are equivalent to the containment and equivalence of their p-graphs;
– four transformation rules are enough to generate all minimal extensions of a p-skyline relation;
– the number of all minimal extensions of a p-skyline relation is polynomial in \(|\mathcal{A}|\);
– every \(\subset\)-chain in \(\mathcal{F}_p\) is of polynomial length, although \(\mathcal{F}_p\) contains at least \(|\mathcal{A}|!\) relations.

The properties listed above are used to develop the elicitation algorithm and prove its correctness. Incorporating attribute importance into skyline relations allows not only to model user preferences more accurately but also to make the size of the corresponding query results more manageable.

At the same time, enriching the skyline framework with attribute importance comes at a cost. To construct a p-skyline preference relation from a skyline relation, one needs to provide a p-graph describing relative attribute importance. However, requiring users to describe attribute importance explicitly seems impractical for several reasons. First, the number of pairwise attribute comparisons required may be large. Second, users themselves may not be fully aware of their own preferences.

To address this problem, we develop a method of elicitation of p-skyline relations based on simple user-provided feedback. The type of feedback used in the method consists of two sets of tuples belonging to a given set: superior examples [Jiang et al(2008)], i.e., the desirable tuples, and inferior examples [Jiang et al(2008)] i.e., the undesirable tuples. This type of feedback is quite natural in real life: given a set of tuples, a user needs to examine them and identify some tuples she likes and dislikes most. Moreover, it is advantageous from the point of view of user interface design—a user is required to perform a number of simple “check off” actions to identify such tuples. Finally, such feedback can be elicited automatically [Holland et al(2003)].

We consider the problems related to the construction of p-skyline relations covering the given superior and inferior examples. Specifically, we need to guarantee that the superior examples are among the best tuples and that the inferior examples are dominated by at least one other tuple. Also, to guarantee an optimal fit we postulate that the constructed relation be maximal. We show that determining the existence of a p-skyline relation covering a given set of superior examples, and a polynomial-time algorithm for constructing a maximal p-skyline relation of that kind. The latter algorithm is based on checking the satisfaction of a system of negative constraints, each of which captures the fact that one tuple does not dominate another according to the p-skyline relation being constructed.

We provide two effective methods for reducing the size of systems of negative constraints and hence improving the performance of the elicitation algorithm. At the same time, we show that the problem of minimizing the size of such a system is unlikely to be efficiently solvable. The experimental evaluation of the algorithms on real life and synthetic data sets demonstrates high accuracy and scalability of the elicitation algorithm, as well as the efficacy of the proposed optimization methods.

The paper is organized as follows. In section 2, we introduce the concepts used throughout the paper. In section 3, we describe p-skylines—skylines enriched with relative attribute importance information. We also discuss the fundamental properties of such relations. In section 4, we study the problem of eliciting p-skyline relations based on superior and inferior examples. In Section 5, we show the results of the experimental evaluation of the proposed algorithms. Section 6 concludes the paper with a discussion of related and future work. The proofs of all the results presented in the paper are provided in the Appendix.

2 Basic notations

2.1 Binary relations

A binary relation \(R\) over a (finite of infinite) set \(S\) is a subset of \(S \times S\). Binary relations may be finite or infinite. To denote \((x, y) \in R\), we may write \(R(x, y)\) or \(x R y\). Here we list some typical properties of binary relations. A binary relation \(R\) is:

– \(\text{irreflexive}\) iff \(\forall x (\neg R(x, x))\),
– \(\text{transitive}\) iff \(\forall x, y, z (R(x, y) \land R(y, z) \rightarrow R(x, z))\),
– \(\text{connected}\) iff \(\forall x, y (R(x, y) \lor R(y, x) \lor x = y)\),
– \(\text{a strict partial order (SPO)}\) if it is irreflexive and transitive,
– \(\text{a weak order}\) iff it is an SPO such that

\[\forall x, y, z (R(x, y) \rightarrow R(x, z) \lor R(z, y))\],

– \(\text{a total order}\) if it is a connected SPO.
The transitive closure $TC(R)$ of a binary relation $R$ is defined as

$$(x, y) \in TC(R) \text{ iff } R^m(x, y) \text{ for some } m > 0,$$

where

$$R^1(x, y) \equiv R(x, y)$$

$$R^{m+1}(x, y) \equiv \exists z \left( R(x, z) \land R^m(z, y) \right)$$

A binary relation $R \subseteq S \times S$ may be viewed as a directed graph. The set $S$ is called the set of nodes of $R$ and denoted as $N(R)$. We say that the tuple $xy$ is an $R$-edge from $x$ to $y$ if $(x, y) \in R$. A path in $R$ (or an $R$-path) from $x$ to $y$ for an $R$-edge $xy$ is a sequence of $R$-edges such that the start node of the first edge is $x$, the end node of the last edge is $y$, and the end node of every edge (except the last one) is the start node of the next edge in the sequence. The length of an $R$-path is the number of $R$-edges in the path. An $R$-sequence is the sequence of nodes participating in an $R$-path. The length of an $R$-sequence is the number of nodes in it.

Given a directed graph $R$ and its node $x$,

- $Ch_R(x) = \{ y \mid (x, y) \in R \}$ is the set of children of $x$ in $R$,
- $Pa_R(x) = \{ y \mid (y, x) \in R \}$ is the set of parents of $x$ in $R$,
- $Desc_R(x) = \{ y \mid (x, y) \in TC(R) \}$ is the set of descendents of $x$ in $R$,
- $Anc_R(x) = \{ y \mid (y, x) \in TC(R) \}$ is the set of ancestors of $x$ in $R$.

We also write $Desc Sel_f R(x)$ and $Anc Sel_f R(x)$ as short-hand of $(Desc_R(x) \cup \{x\})$ and $(Anc_R(x) \cup \{x\})$, respectively. Similarly, we define set versions of the above definitions, e.g., $Ch_R(x) = \{ y \mid \exists x \in X \, ((x, y) \in R) \}$.

Given two nodes $x$ and $y$ of $R$ and two sets of nodes $X$ and $Y$ of $R$, we write

- $R \models x \sim y$ iff $(x, y) \notin R$ and $(y, x) \notin R$;
- $R \models x \sim y$ iff $\forall x \in X, y \in Y \, (R \models x \sim y)$;
- $(X, Y) \in R$ iff $\forall x \in X, y \in Y \, ((x, y) \in R)$.

2.2 Preference relations

Below we describe some concepts of a variant of the preference framework [Chomicki(2003)], which we adopt here.

Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be a finite set of attributes (a relation schema). Every attribute $A_i \in \mathcal{A}$ is associated with an infinite domain $\mathcal{D}_{A_i}$. The domains considered here are rationals and uninterpreted constants (numerical or categorical).

We work with the universe of tuples $\mathcal{U} = \prod_{A_i \in \mathcal{A}} \mathcal{D}_{A_i}$. Given a tuple $o \in \mathcal{U}$, we denote the value of its attribute $A_i$ as $o.A_i$.

Preference relations we consider in this paper are of two types: attribute and tuple.

**Definition 1 (Attribute preference relation)** An attribute preference relation $\succ_{A_i}$ for an attribute $A_i \in \mathcal{A}$ is a subset of $\mathcal{D}_{A_i} \times \mathcal{D}_{A_i}$, which is a total order over $\mathcal{D}_{A_i}$.

An attribute preference relation describes a preference over the values of a single attribute e.g., the red color is preferred to the blue color, or the make BMW is preferred to the make Kia.

**Definition 2 (Tuple preference relation)** A tuple preference relation $\succ$ is a subset of $\mathcal{U} \times \mathcal{U}$, which is a strict partial order over $\mathcal{U}$.

In contrast to an attribute preference relation, a tuple preference relation describes a preference over tuples, e.g., a red BMW is preferred to a blue Kia. We say that

- a tuple $o_1$ dominates (is preferred to, is better than) a tuple $o_2$, and
- $o_2$ is dominated by (is worse than) $o_1$, according to a preference relation $\succ$ iff $o_1 \succ o_2$. In the remaining part of the paper, tuple preference relations are simply referred to as preference relations.

We assume that both attribute and tuple preferences are defined as quantifier-free formulas over some appropriate signature. In this way both finite and infinite preference relations can be captured. For instance, the following formula defines an infinite tuple preference relation over the domains of the attributes make, year, and price.

$$o_1 \succ o_2 = o_1.\text{year} \geq o_2.\text{year} \land o_1.\text{make} = \text{BMW} \land o_2.\text{price} \leq o_1.\text{price} \land$$

$$\left( o_1.\text{make} = \text{BMW} \land o_2.\text{make} = \text{Ford} \loright)$$

$$\left( o_1.\text{make} = \text{Ford} \land o_2.\text{make} = \text{Kia} \loright)$$

$$\left( o_1.\text{make} = \text{BMW} \land o_2.\text{make} = \text{Kia} \loright)$$

$$\left( o_1.\text{make} = \text{make} \land o_2.\text{price} \neq o_1.\text{price} \land o_1.\text{make} \neq o_2.\text{make} \right)$$

Given a tuple preference relation, the two most common tasks are:

1. dominance testing: checking if a tuple is preferred to another one, and
2. computing the best (most preferred) tuples in a given finite set of tuples.

The first problem is easily solved by checking if the formula representing the preference relation evaluates to true for the given pair of tuples. (Nevertheless, we will revisit this problem in section 3.) To deal with the second problem, a new winnow relational algebra operator was proposed [Chomicki(2003), Kießling(2002)].

**Definition 3 (Winnow)** If $\succ$ is a tuple preference relation over $\mathcal{U}$, then the winnow operator $\omega_\succ(\mathcal{A})$ is defined as

$$\omega_\succ(r) = \{ t \in r \mid \neg \exists t' \in r \, (t' \succ t) \}.$$
3 p-skylines

Let $\mathcal{A} = \{A_1, ..., A_k\}$ be a finite set of attributes and $\mathcal{H} = \{A_1, ..., A_k\}$ be a set of the corresponding attribute preference relations. Below we define the syntax and the semantics of p-skyline relations.

**Notation:** We use “$=$” for syntactic identity of expressions and “$\equiv$” for equality of relations viewed as sets of tuples.

**Definition 4 (p-expression)** An expression $\pi$ is a p-expression if
- $\pi$ is $>_{A_i}$ for $A_i \in \mathcal{A}$, or
- $\pi = \pi_1 \otimes \pi_2$ for two p-expressions $\pi_1$ and $\pi_2$, or
- $\pi = \pi_1 \& \pi_2$ for two p-expressions $\pi_1$ and $\pi_2$.

Intuitively, expressions $\pi_1, \pi_2$ represent preference relations, and the operators $\otimes$ and $\&$ define the relative importance of the preference relations represented by $\pi_1, \pi_2$ in the preference relation represented by $\pi$. The details are shown further.

**Definition 5 ( Relevant attributes)** Given a p-expression $\pi$, the corresponding set of relevant attributes $\text{Var}(\pi)$ is:
- $\{A_i\}$, if $\pi$ is $>_{A_i}$;
- $\text{Var}(\pi_1) \cup \text{Var}(\pi_2)$ for $\pi = \pi_1 \& \pi_2$ or $\pi = \pi_1 \otimes \pi_2$, where $\pi_1$ and $\pi_2$ are p-expressions.

$$\alpha_1 \equiv_{\pi} \alpha_2 \iff \forall A \in X (\alpha_1 A = \alpha_2 A).$$

**Definition 6 ( Preference relation induced by p-expression)**
The preference relation $\succ_{\pi}$ induced by a p-expression $\pi$ is defined as
1. if $\pi$ is $>_{A_i}$ and $A_i \in \mathcal{A}$,
   $$\succ_{\pi} \equiv \{ (a, o') | a, o' \in U . o.A_i >_{A_i} o'.A_i \},$$
   and $\succ_{\pi}$ is also written as $\succ_{A_i}$, and called an atomic preference relation,
2. for $\pi = \pi_1 \& \pi_2$,
   $$\succ_{\pi} \equiv \succ_{\pi_1} \cup (\approx_{\text{Var}(\pi_1)} \cap \succ_{\pi_2}),$$
3. for $\pi = \pi_1 \otimes \pi_2$,
   $$\succ_{\pi} \equiv (\succ_{\pi_1} \cap \approx_{\text{Var}(\pi_1)}) \cup (\succ_{\pi_2} \cap \approx_{\text{Var}(\pi_2)}) \cup (\succ_{\pi_1} \cap \succ_{\pi_2}),$$

where $\succ_{\pi_1}$ and $\succ_{\pi_2}$ are preference relations induced by the p-expressions $\pi_1$ and $\pi_2$.

In the second case, we say that $\succ_{\pi} \equiv \succ_{\pi_1} \& \succ_{\pi_2}$ and in the third case, $\succ_{\pi} \equiv \succ_{\pi_1} \otimes \succ_{\pi_2}$. We also refer to the set of relevant attributes $\text{Var}(\pi)$ of $\pi$ as $\text{Var}(\succ_{\pi})$. When the context in clear, we may omit the subscript $\pi$ and refer to p-skyline relations as $\succ_{1}, \succ_{2}, ...$. Note the difference between the attribute preference relation $\succ_A$ and the tuple preference relation $\succ_{A_i}$. However, the correspondence between those two relations is straightforward.

The intuition behind Definition 6 is as follows. In the first case, $\succ_{A_i}$ is the tuple preference relation corresponding to the attribute preference relation $\succ_{A_i}$. In the second case, $\succ_{\pi}$ is composed of $\succ_{\pi_1}$ and $\succ_{\pi_2}$ a way that $\succ_{\pi_1}$ has higher importance than $\succ_{\pi_2}$: a tuple $o$ is preferred to $o'$ according to $\succ_{\pi}$ iff $o$ is preferred to $o'$ according to $\succ_{\pi_1}$ (regardless of $\succ_{\pi_2}$), or $o$ and $o'$ are equal in all the relevant attributes of $\succ_{\pi_1}$ and $o$ is preferred to $o'$ according to $\succ_{\pi_2}$. The operator $\&$ is called prioritized accumulation [Kießling(2002)]. Similarly, if $\pi = \pi_1 \otimes \pi_2$, then $\succ_{\pi_1}$ and $\succ_{\pi_2}$ are considered to be equally important in $\succ_{\pi}$. The operator $\otimes$ is called Pareto accumulation [Kießling(2002)]. Some known properties of the operators are summarized below.

**Proposition 1** [Kießling(2002)] The operators $\otimes$ and $\&$ are associative. The operator $\otimes$ is commutative.

Since accumulation operators are associative, we extend them from binary to n-ary operators.

**Proposition 2** [Kießling(2002)] A relation induced by a p-expression is an SPO, i.e., a tuple preference relation.

**Definition 7 (p-skyline relation)** A p-skyline relation $\succ_{\pi}$ is the relation induced by a p-expression $\pi$ such that for all subexpressions of $\pi$ of the form $\pi_1 \& \pi_2$ or $\pi_1 \otimes \pi_2$:
- $\text{Var}(\pi_1) \cap \text{Var}(\pi_2) = \emptyset$;
- the relations induced by $\pi_1$ and $\pi_2$ are p-skyline relations.

A p-skyline relation $\succ_{\pi}$ induced by $\pi$ is full iff $\text{Var}(\pi) = \mathcal{A}$.

Essentially, p-skyline relations are induced by those p-expressions in which every member of $\mathcal{H}$ is used at most once (exactly once in the case of full p-skyline relations). The set of all full p-skyline relations for $\mathcal{H}$ is denoted by $\mathcal{F}_{\mathcal{H}}$. Further we consider only full p-skyline relations.

A key property of p-skyline relations is that the skyline preference relation $\text{sk}_{\mathcal{H}}$ is the p-skyline relation induced by the p-expression $\succ_{A_1} \otimes \ldots \otimes \succ_{A_k}$. That is, the p-skyline framework is an extension of the skyline framework.

### 3.1 Syntax trees

Dealing with p-skyline relations, it is natural to represent the corresponding p-expressions as syntax trees. This representation is used in Section 3.4 for constructing minimal extensions of a p-skyline relation.

**Definition 8 (Syntax tree)** A syntax tree $T_{\succ_{\pi}}$ of a p-skyline relation $\succ_{\pi}$ is an ordered rooted tree representing the p-expression $\pi$. 
Every non-leaf node of the syntax tree is labeled with an accumulation operator and corresponds to the result of applying the operator to the p-skyline relations represented by its children, from left to right. Every leaf node of the syntax tree is labeled with an attribute \( A \in \mathcal{A} \) and corresponds to the attribute preference relation \( >_A \in \mathcal{H} \) (and the atomic preference relation \( >_A \)).

**Definition 9 (Normalized syntax tree)** A syntax tree is normalized iff each of its non-leaf nodes is labeled differently from its parent.

Clearly, for every p-skyline relation, there is a normalized syntax tree which may be constructed in polynomial time in the size of the original tree. To do that, one needs to find all occurrences of syntax tree nodes \( C_1 \) and their children \( C_2 \) such that \( C_1 \) and \( C_2 \) have the same label. After that, \( C_2 \) has to be removed from the list of children of \( C_1 \), and the list of children of \( C_2 \) has to be added to the list of children of \( C_1 \) in place of \( C_2 \). The correctness of this procedure follows from Proposition 1.

We note that a normalized syntax tree is not unique for a p-skyline relation. That is due to the commutativity of \( \otimes \) (Proposition 1).

**Example 3** Let a p-skyline relation \( \triangleright^1 \) be defined as

\[
\triangleright = (\triangleright_A \otimes (\triangleright_B \& \triangleright_C)) \otimes (\triangleright_D \& (\triangleright_E \otimes \triangleright_F))
\]

An unnormalized syntax tree of \( \triangleright \) is shown in Figure 1(a). Two normalized syntax trees of \( \triangleright \) are shown in Figures 1(b) and 1(c).

![Syntax trees of \( \triangleright \)](image)

Every node of a syntax tree is itself a root of another syntax tree. Let us associate with every node \( C \) of a syntax tree the set \( Var(C) \) of attributes which are descendants of \( C \) in the syntax tree or \( C \) itself (if it is a leaf). Essentially, \( Var(C) \) corresponds to \( Var(\pi_C) \) where \( \pi_C \) is the p-expression represented by the subtree with the root node \( C \).

\[\text{Fig. 1 Syntax trees of } \triangleright\]

\[\text{(a) Unnormalized} \quad \text{(b) Normalized} \quad \text{(c) Equivalent normalized}\]

3.2 Attribute importance in p-skyline relations

Recall that the p-skyline relations composed using \& (resp. \( \otimes \)) have different (resp. equal) importance in the resulting relation. However, the composed p-skyline relations do not have to be atomic and may themselves be composed using \& or \( \otimes \). The problem we discuss in this section is how to represent relative importance of attributes in different subtrees. For this purpose, we define another graphical representation of a p-skyline relation – the p-graph.

**Definition 10 (p-graph)** The p-graph \( \Gamma_\triangleright \) of a p-skyline relation \( \triangleright \) is a directed graph with the set of nodes \( N(\Gamma_\triangleright) = Var(\triangleright) \) and the set of edges \( E(\Gamma_\triangleright) \):

\[
\begin{align*}
\Gamma_\triangleright &= \emptyset, \text{ if } \triangleright \text{ is an atomic preference relation;} \\
E(\Gamma_\triangleright) &= E(\Gamma_{\triangleright_1}) \cup E(\Gamma_{\triangleright_2}), \text{ if } \triangleright = \triangleright_1 \otimes \triangleright_2; \\
E(\Gamma_\triangleright) &= E(\Gamma_{\triangleright_1}) \cup E(\Gamma_{\triangleright_2}) \cup (Var(\triangleright_1) \times Var(\triangleright_2)), \text{ if } \triangleright = \triangleright_1 & \triangleright_2,
\end{align*}
\]

for two p-skyline relations \( \triangleright_1 \) and \( \triangleright_2 \).

A p-graph represents the attribute importance relationships implicit in a p-skyline relation \( \triangleright \) in the following way: an edge in \( E(\Gamma_\triangleright) \) goes from a more important attribute to a less important attribute. This follows from Definition 10: if \( \triangleright = \triangleright_1 \otimes \triangleright_2 \) (i.e., \( \triangleright_1 \) and \( \triangleright_2 \) are equally important in \( \triangleright \)), then no new attribute importance relationships are added to \( E(\Gamma_\triangleright) \), and those which exist in \( E(\Gamma_{\triangleright_1}) \) and \( E(\Gamma_{\triangleright_2}) \) are preserved in \( E(\Gamma_\triangleright) \). Similarly, if \( \triangleright = \triangleright_1 \& \triangleright_2 \), then the attribute importance relationships in \( E(\Gamma_{\triangleright_1}) \) and \( E(\Gamma_{\triangleright_2}) \) are preserved in \( E(\Gamma_\triangleright) \), but new importance relationships are added: every attribute relevant to \( \triangleright_1 \) is more important than every attribute relevant to \( \triangleright_2 \).

**Example 4** Take the p-skyline relations \( \triangleright_1 \) and \( \triangleright_2 \) as below. Their p-graphs are shown in Figure 2.

\[
\begin{align*}
\triangleright_1 &= (\triangleright_A \otimes \triangleright_B) \& \triangleright_C \\
\triangleright_2 &= \triangleright_A \otimes (\triangleright_B \otimes \triangleright_C)
\end{align*}
\]

In the previous section, we showed that the skyline relation \( \text{sky}_{\mathcal{H}} \) is constructed as the Pareto accumulation of all the members of \( \mathcal{H} \). Hence, the following holds.

**Proposition 3** The p-graph \( \Gamma_{\text{sky}_{\mathcal{H}}} \) of the skyline relation \( \text{sky}_{\mathcal{H}} \) has the set of nodes \( N(\Gamma_{\text{sky}_{\mathcal{H}}}) = \mathcal{A} \) and the set of edges \( E(\Gamma_{\text{sky}_{\mathcal{H}}}) = \emptyset \).
Theorem 1 shows that p-graphs indeed represent attribute importance. According to the theorem, a p-skyline relation can be decomposed into “dimensions” which are attribute preference relations. This decomposition shows which attribute preferences (resp. the corresponding attributes) are less important than a given attribute preference (resp. the corresponding attribute) in a preference relation.

Theorem 1 Every p-skyline relation \( \succ \in \mathcal{F}_M \) is equal to

\[
\succ = TC \left( \bigcup_{A \in \mathcal{A}} q_A \right),
\]

where

\[
q_A \equiv \{(o_1, o_2) \mid o_1.A \succ A o_2.A \} \cap \approx_{\mathcal{A} \setminus (Ch_{\succ} (A) \cup \{A\})}.
\]

The relation \( q_A \) may be viewed as a “projection” of the p-skyline relation \( \succ \) to a “dimension” which is a preference relation over \( A \). Comparing tuples on the attribute \( A \), one needs to consider only the attributes \( \mathcal{A} - (Ch_{\succ} (A) \cup \{A\}) \) The values of the remaining attributes \( Ch_{\succ} (A) \) do not matter: those attributes are less important than \( A \). The relation \( \succ \) above can also be viewed as a relaxed ceteris paribus preference relation [Boutilier et al(2004)], for which attribute preferences are unconditioned on each other, and “everything else being equal” is replaced with “\( \mathcal{A} - (Ch_{\succ} (A) \cup \{A\}) \) being equal”.

Now let us take a closer look at the properties of p-graphs. Since p-graphs represent attribute importance implicit in p-skyline relations, there are some properties of importance relationships that p-graphs are expected to have, for example SPO. In particular:

- no attribute should be more important than itself (irreflexivity), and
- if an attribute \( A \) is more important than an attribute \( B \) which is more important than an attribute \( C \), \( A \) is expected to be more important than \( C \) too (transitivity).

As Theorem 2 shows, a p-graph is indeed an SPO\(^2\). However, a graph needs to satisfy some additional properties in order to be a p-graph of some p-skyline relation. Namely, there is a requirement that the p-expression inducing the p-skyline relation contain exactly one occurrence of each member of \( \mathcal{A} \). This requirement is captured by the Envelope property visualized in Figure 3: if a graph \( \Gamma \) has the three bold edges, then it must have at least one dashed edge.

Theorem 2 (SPO+Envelope)
A directed graph \( \Gamma \) with the set of nodes \( \mathcal{A} \) is a p-graph of some p-skyline relation iff

1. \( \Gamma \) is an SPO, and
2. \( \Gamma \) satisfies the Envelope property:

\[
\forall A, B, C, D \in \mathcal{A}, \text{all different}
\]

\[
\exists \Gamma \in \mathcal{F}_M \text{ all different}
\]

\[
(A, B) \in \Gamma \land (C, D) \in \Gamma \land (C, B) \in \Gamma \Rightarrow \exists \Gamma \in \mathcal{F}_M \text{ all different}
\]

\[
(C, A) \in \Gamma \lor (A, D) \in \Gamma \lor (D, B) \in \Gamma
\]

Fig. 3 The Envelope property

We showed above that a p-graph represents the attribute importance induced by a p-skyline relation. Hence, the SPO properties of a p-graph are quite intuitive – they capture the rationality of the importance relationship. The Envelope property of a p-graph is due to the fact that each attribute preference relation can have only one occurrence in a p-skyline p-expression. According to that property, if a graph \( \Gamma \) has the three edges shown bold in Figure 3, then it must have at least one dashed edge.

We note that so far we have introduced two graph notations for p-skyline relations: syntax trees and p-graphs. Although these notations represent different concepts, there is a correspondence between them (Proposition 4).

Proposition 4 (Syntax tree and p-graph correspondence)
Let \( A \) and \( B \) be leaf nodes in a normalized syntax tree \( T_\succ \) of a p-skyline relation \( \succ \in \mathcal{F}_M \). Then \( \Gamma \in \mathcal{F}_M \) iff the least common ancestor \( C \) of \( A \) and \( B \) in \( T_\succ \) is labeled by \&., and \( A \) precedes \( B \) in the left-to-right tree traversal.

3.3 Properties of p-skyline relations

In this section, we show several fundamental properties of p-skyline relations. These properties are used later to efficiently perform essential operations on p-skyline relations: checking equivalence and containment of relations and (tuple) dominance testing. Before going further, we note that p-skyline relations are representable as formulas constructed from the corresponding p-expressions. So one can use such formulas to perform the operations mentioned above. For example, relation containment corresponds to formula implication. However, we show below more direct ways of performing the operations on p-skyline relations. The results presented in this section are used in sections 3.4 and 4.

Recall Example 3, where we showed that a p-skyline relation may have more than one syntax tree (and hence p-expression) defining it. In contrast, as shown in the next theorem, the p-graph corresponding to a p-skyline relation is unique.

Theorem 3 (p-graph uniqueness) Two p-skyline relations \( \succ_1, \succ_2 \in \mathcal{F}_M \) are equal iff their p-graphs are identical.
Theorem 4 (p-skyline relation containment) For p-skyline relations \( \succ_1, \succ_2 \in \mathcal{F}_{\mathcal{G}}, \succ_1 \subset \succ_2 \Leftrightarrow E(\Gamma_{\succ_1}) \subseteq E(\Gamma_{\succ_2}). \)

Theorem 4 implies an important result. Recall that in Corollary 3 we showed that the edge set of the p-graph \( \Gamma_{\text{sky}} \) of the skyline preference relation \( \text{sky} \) is empty. Hence, the following facts are implied by Theorem 4.

Corollary 1 For every relation instance \( r \) and p-skyline relations \( \succ_1, \succ_2 \in \mathcal{F}_{\mathcal{G}}, s.t. \Gamma_{\succ_2} \subset \Gamma_{\succ_1}, \) we have \( \omega_{\succ_1}(r) \subseteq \omega_{\succ_2}(r) \)

The importance of Corollary 1 is that for every p-skyline relation, the winnow query result will always be contained in the corresponding skyline relation. In real life, that means that if user preferences are modeled as a p-skyline relation instead of a skyline relation, the size of the query result will not be larger than the size of the skyline, and may be smaller.

Example 5 Let \( \mathcal{A} = \{A_1, A_2, A_3\} \), and for every attribute, larger values are preferred. Consider the relations

\[
\text{sky}_{\mathcal{G}} = \succ_1 \otimes \succ_2 \otimes \succ_3
\]

\[
\succ_1 = (\succ_{A_1} \& \succ_{A_2}) \otimes \succ_{A_3}
\]

\[
\succ_2 = (\succ_{A_2} \& \succ_{A_3}) \otimes \succ_{A_1}
\]

whose p-graphs are shown in Figures 4(a), 4(b), and 4(c), respectively. Theorems 4 and 3 imply that \( \text{sky}_{\mathcal{G}} \subset \succ_1, \text{sky}_{\mathcal{G}} \subset \succ_2, \succ_1 \not\subset \succ_2, \) and \( \succ_2 \not\subset \succ_1. \) Take the relation instance \( r \) shown in Figure 4(d). Then \( \omega_{\text{sky}_{\mathcal{G}}}(r) = \{t_1, t_2, t_3\}, \omega_{\succ_1}(r) = \{t_1, t_2\}, \) and \( \omega_{\succ_2}(r) = \{t_1, t_3\}. \)

In Theorem 5, we show how one can directly test tuple containment dominance. The dominance is expressed in terms of containment constraints on attribute sets. This formulation is essential for our approach to preference elicitation (Sec. 4).

Given two tuples \( o, o' \in \mathcal{U} \), a p-skyline relation \( \succ \) and its p-graph \( \Gamma_{\succ} \), let

- \( \text{Diff}(o, o') \) be the attributes in which \( o \) differs from \( o' \):

\[
\text{Diff}(o, o') = \{A \in \mathcal{A} \mid o_1 A \neq o_2 A\},
\]

Example 6 Let \( \mathcal{A} = \{A_1, \ldots, A_7\} \), and for every attribute, larger values are preferred. Let a p-skyline relation \( \succ \) be represented by the p-graph shown in Figure 5(a). Consider the tuples \( t_1, t_2, t_3 \) shown in Figure 5(b). Let \( \mathcal{B} = \{A_1, A_2, A_3, A_4, A_5, A_7\} \), and \( \text{Diff}(t_1, t_2) = \{A_1, A_2, A_4, A_5, A_7\} \), and \( \text{Diff}(t_1, t_3) = \{A_1, A_2, A_4, A_5\} \), and \( \text{Diff}(t_2, t_3) = \{A_1, A_2, A_4, A_6\} \), and \( \text{Diff}(t_1, t_2) = \{A_1, A_2, A_4, A_5, A_7\} \), and \( \text{Diff}(t_1, t_3) = \{A_1, A_2, A_4, A_5, A_7\} \), and \( \text{Diff}(t_2, t_3) = \{A_1, A_2, A_4, A_5\} \).

In Theorem 2, we showed that p-graphs satisfy \( \text{SPO}^+ \text{Envelope} \), where the property Envelope was formulated in terms of single p-graph nodes. However, it is often necessary to deal with sets of nodes. The next theorem generalizes the Envelope property to disjoint sets of nodes.

Theorem 6 (GeneralEnvelope) Let \( \succ \) be a p-skyline relation with the p-graph \( \Gamma_{\succ} \), and \( A, B, C, D \) disjoint node sets of \( \Gamma_{\succ} \). Let the subgraphs of \( \Gamma_{\succ} \) induced by those nodes be singletons or unions of at least two disjoint subgraphs. Then

\[
(A, B) \in \Gamma_{\succ} \land (C, D) \in \Gamma_{\succ} \land (C, B) \in \Gamma_{\succ} \Rightarrow (C, A) \in \Gamma_{\succ} \lor (A, D) \in \Gamma_{\succ} \lor (D, B) \in \Gamma_{\succ}
\]
the p-graph satisfies GeneralEnvelope. That form is quite general. For instance, Var(≻) induces disjoint subgraphs if ≻ is defined as Pareto accumulation of p-skyline relations. Theorem 6 is used in the following section.

Example 7 Let \( \mathcal{A} = \{A_1, \ldots, A_7\} \). Consider the p-graph \( \Gamma_\succ \) (Figure 6) of

\[
\succ \in ((\succ A_1 \odot \succ A_2 \odot \succ A_3) \& (\succ A_4 \odot \succ A_5 \odot \succ A_6)) \odot \succ A_7
\]

Let \( \mathcal{A} = \{A_1\}, B = \{A_4\}, C = \{A_2, A_3\}, D = \{A_5, A_6\} \). Then the p-graph satisfies GeneralEnvelope because

\[
(A, B) \in \Gamma_\succ \& (C, D) \in \Gamma_\succ \& (A, D) \in \Gamma_\succ
\]

3.4 Minimal extensions
We conclude this section by studying the notion of minimal extension of a p-skyline relation. This notion is central for our approach to preference elicitation (section 4). Intuitively, we will construct a p-skyline relation that incorporates user feedback using an iterative process that starts from the skyline relation and extends it repeatedly in a minimal way.

Definition 11 (p-extension) For a p-skyline relation \( \succ \in \mathcal{F}_{\Gamma_\succ} \), a p-skyline relation \( \succ_{ext} \in \mathcal{F}_{\Gamma_\succ} \) is a p-extension of \( \succ \) if \( \succ \subset \succ_{ext} \). The p-extension \( \succ_{ext} \) is minimal if there exists no \( \succ' \in \mathcal{F}_{\Gamma_\succ} \) such that \( \succ \subset \succ' \subset \succ_{ext} \).

Theorem 4 implies that for every p-skyline relation \( \succ \), a p-extension \( \succ_{ext} \) of \( \succ \), if it exists, may be obtained by constructing an extension \( \Gamma_{\succ_{ext}} \) of the p-graph \( \Gamma_{\succ} \). Hence, the problem of constructing a minimal p-extension of a p-skyline relation can be reduced to the problem of finding a minimal set of edges that when added to \( \Gamma_{\succ} \) form a graph satisfying SPO+Envelope. However, it is not clear how to find such a minimal set of edges efficiently: adding a single edge to a graph may not be enough due to violation of SPO+Envelope, as shown in the following example.

Example 8 Take the relation \( \succ \) from Example 7 (Figure 6), and add the edge \((A_6, A_7)\). The resulting graph satisfies SPO+Envelope. However, if instead of the edge \((A_6, A_7)\), we add the edge \((A_3, A_7)\), then for preserving Envelope, it is enough to add \((A_1, A_7)\) and \((A_2, A_7)\) (other extension possibilities exist too). The resulting graph (Figure 7) satisfies SPO+Envelope. The corresponding p-expression is

\[
\succ' = (\succ A_1 \odot \succ A_2 \odot \succ A_3) \& (\succ A_4 \odot \succ A_5 \odot \succ A_6 \odot \succ A_7)
\]

The method of constructing all minimal p-extensions we propose in this paper operates directly on normalized p-expressions represented as syntax trees. In particular, we show a set of transformation rules of syntax trees such that every unique application of a rule from this set results in a unique minimal p-extension of the original p-skyline relation. If all minimal p-extensions of a p-skyline relation are needed, then one needs to apply to the syntax tree every rule in every possible way.

The transformation rules are shown in Figure 9. On the left hand side, we show a part of the syntax tree of an original p-skyline relation. On the right hand side, we show how this part is modified in the resulting relation. We assume that the rest of the syntax tree is left unchanged. All the transformation rules operate on two children \( C_i \) and \( C_{i+1} \) of a \( \odot \)-node of the syntax tree. For simplicity, these nodes are shown as consecutive children. However, in general, \( C_i \) and \( C_{i+1} \) may be any pair of children nodes of the same \( \odot \)-node. Their order is unimportant due to the commutativity of \( \odot \).

Intuitively, \( Rule_1 \) and \( Rule_2 \) push the subtree \( C_{i+1} \) of \( T_\succ \) down into the subtree \( C_i \) (denoted \( C'_i \) in the resulting trees). \( Rule_3 \) replaces two nodes \( C_{i} \) and \( C_{i+1} \) of \( R \) with the subtree \( R'_i \), having \( C_i \) and \( C_{i+1} \) as children. \( Rule_4 \) results in redistributing the subtrees of the trees \( C_i \) and \( C_{i+1} \). Instead of \( C_i \) and \( C_{i+1} \), the resulting tree has two subtrees \( \succ R'_i \) and \( R'_i \sim C \) each of which has two branches combining the former subtrees of \( C_i \) and \( C_{i+1} \).

Let us denote the original relation as \( \succ \) and the relation obtained as the result of applying one of the transformation rules as \( \succ_{ext} \). Observation 1 shows that all the rules only add edges to the p-graph of the original preference relation and hence extend the p-skyline relation.

Observation 1 If \( T_{\succ_{ext}} \) is obtained from \( T_\succ \) using some of \( Rule_1, \ldots, Rule_4 \), then \( E(\Gamma_{\succ_{ext}}) \subseteq E(\Gamma_{\succ}) \). Moreover,

- if \( T_{\succ_{ext}} \) is a result of \( Rule_1(T_\succ, C_i, C_{i+1}) \), then
  \( E(\Gamma_{\succ_{ext}}) = E(\Gamma_\succ) \cup \{(X, Y) \mid X \in \text{Var}(N_1) \wedge Y \in \text{Var}(C_{i+1})\}\)
- if \( T_{\succ_{ext}} \) is a result of \( Rule_2(T_\succ, C_i, C_{i+1}) \), then
  \( E(\Gamma_{\succ_{ext}}) = E(\Gamma_\succ) \cup \{(X, Y) \mid X \in \text{Var}(C_{i+1}) \wedge Y \in \text{Var}(N_m)\}\)
- if \( T_{\succ_{ext}} \) is a result of \( Rule_3(T_\succ, C_i, C_{i+1}) \), then
  \( E(\Gamma_{\succ_{ext}}) = E(\Gamma_\succ) \cup \{(C_i, C_{i+1})\}\)
- If $T_{\text{ext}}$ is a result of Rule4($T_{\text{e}}, C_{t}, C_{t+1}, s, t$) for $s \in [1, n-1], t \in [1, m-1]$, then $E(T_{\text{ext}}) = E(T_{\text{e}}) \cup$

\[
\{(X, Y) \mid X \in \bigcup_{p=1}^{s} \text{Var}(N_p), Y \in \bigcup_{q=t+1}^{m} \text{Var}(M_q)\} \cup
\[
\{(X, Y) \mid X \in \bigcup_{p=1}^{s} \text{Var}(N_p), Y \in \bigcup_{q=t+1}^{m} \text{Var}(N_q)\}
\]

We note that every $\&$- and $\otimes$-node in a syntax tree has to have at least two children nodes. This is because the operators $\&$ and $\otimes$ must have at least two arguments. However, as a result of a transformation rule application, some $\&$- and $\otimes$-nodes may end up with only one child node. These nodes are:

1. $R'_t$ if $k = 2$ for Rule1, Rule2, Rule3, Rule4;
2. $R'_s$ if $m = 2$ for Rule1, Rule2;
3. $R'_t$ or $R'_s$ if $s = 1$ or $s = m - 1$, respectively, for Rule4;
4. $R'_t$ or $R'_s$ if $t = 1$ or $t = n - 1$, respectively, for Rule4.

In such cases, we remove the nodes with a single child and connect the child directly to the parent (Figure 8).

![Fig 8 Single-child node elimination (δ ∈ {& , ⊗ })](image)

**Theorem 7 (minimal p-extension)** Let $\succ \in \mathcal{F}_J$, and $T_{\text{e}}$ be a normalized syntax tree of $\succ$. Then $\succ_{\text{ext}}$ is a minimal p-extension of $\succ$ iff the syntax tree $T_{\text{ext}}$ of $\succ_{\text{ext}}$ is obtained from $T_{\text{e}}$ by a single application of a rule from Rule1, ..., Rule4, followed by a single-child node elimination if necessary.

For instance, the minimal extension $\succ'$ for $\succ$ from Example 8 was computed by applying Rule1($T_{\text{e}}, C_{1}, C_{2}$), for the tree $C_1$ representinig ($\succ_{A_1} \otimes \succ_{A_2} \otimes \succ_{A_3}$) & ($\succ_{A_4} \otimes \succ_{A_5} \otimes \succ_{A_6}$) and having two children: the subtrees $N_1$ representing ($\succ_{A_1} \otimes \succ_{A_2} \otimes \succ_{A_3}$) and $N_2$ representing ($\succ_{A_4} \otimes \succ_{A_5} \otimes \succ_{A_6}$); and $C_2$ representing $\succ_{A_7}$.

Theorem 7 has two important corollaries describing properties of minimal p-extensions.

**Corollary 2** For a p-skyline relation $\succ$, with a normalized syntax tree $T_{\text{e}}$, a syntax tree $T_{\text{ext}}$ of each of its minimal p-extensions $\succ_{\text{ext}}$ may be constructed in time $O(|\mathcal{A}|)$.

In Corollary 2, we assume the adjacency-list representation of syntax trees. The total number of nodes in a tree is linear in the number of its leaf nodes [Cormen et al.(2001)], which is $|\mathcal{A}|$. Thus the number of edges in $T_{\text{e}}$ is $O(|\mathcal{A}|)$. The transformation of $T_{\text{e}}$ using every rule requires removing $O(|\mathcal{A}|)$ and adding $O(|\mathcal{A}|)$ edges.

**Corollary 3** For a p-skyline relation $\succ$, the number of its minimal p-extensions is $O(|\mathcal{A}|^4)$.

The justification for Corollary 3 is as follows. The set of minimal-extension rules is complete due to Theorem 7. Every rule operates on two nodes $C_t$ and $C_{t+1}$ of the syntax tree. Hence, the number of such node pairs is $O(|\mathcal{A}|^2)$. Rule4 also relies on some partitioning of the sequence of child nodes of $C_t$ and $C_{t+1}$. The total number of such partitionings is $O(|\mathcal{A}|^2)$. Thus, the total number of different rule applications is $O(|\mathcal{A}|^4)$. Consequently, the number of minimal p-extensions is polynomial in the number of attributes. This differs from the number of all p-extensions of a p-skyline relation, which is $\Omega(|\mathcal{A}|!)$.

The last property related to p-extensions that we consider here is as follows. By Theorem 4, a p-extension of a
p-skyline relation is obtained by adding edges to its p-graph. However, the total number of edges in a p-graph is at most $O(|A|^2)$. Hence, the next Corollary holds.

**Corollary 4** Let $S$ be a sequence of p-skyline relations

\[ \succ_1, \ldots, \succ_k \in \mathcal{F}_H \]

such that for every $i \in [1, k-1]$, $\succ_{i+1}$ is a p-extension of $\succ_i$. Then $|S| = O(|A|^2)$.

4 Elicitation of p-skyline relations

In Section 3, we proposed a class of preference relations called p-skyline relations. In this section, we introduce a method of constructing p-skyline relations based on user-provided feedback.

4.1 Feedback-based elicitation

As we showed in the previous section, the p-skyline framework is a generalization of the skyline framework. The main difference between those frameworks is that in the p-skyline framework one can express varying attribute importance. On the other hand, one of the main distinguishing properties of the skyline framework is the simplicity of representing preferences. Namely, the user needs to provide only a set of attribute preferences to specify a preference relation. For p-skylines, an additional piece of information, the relative importance of the attributes (in the form of, e.g., a p-graph or a p-expression), has to be also provided by the user. But how can relative attribute importance be specified? It seems impractical to ask the user to compare distinct attributes pairwise for importance: even though some relationships can be deduced by transitivity, the number of comparisons may still be too large. Another issue is even more serious: the users themselves may be not fully aware of their own preferences.

In this section, we propose an alternative approach to elicitation of attribute importance relationships, based on user feedback. We use the following scenario. A fixed, finite set of tuples is stored in a database relation $O \subseteq U$. All the tuples have the same set of attributes $A$. We assume that, in addition to $A$, a corresponding set of attribute preference relations $\mathcal{H}$ is given. The user partitions $O$ into three disjoint subsets: the set $G$ of tuples she confidently likes (superior examples), the set $W$ of tuples she confidently dislikes (inferior examples), and the set of remaining tuples about which she is not sure. The output of our method is a p-skyline relation $\succ$ (with the set of relevant attributes $A$), according to which all tuples in $G$ are superior and all tuples in $W$ are inferior. A tuple $o \in O$ is superior if $o$ does not contain any tuples preferred to $o$, according to $\succ$. A tuple $o \in O$ is inferior if there is at least one superior example in $O$, which is preferred to $o$. The last assumption is justified by a general principle that the user considers something bad because she knows of a better alternative.

Formally: given $A$, $\mathcal{H}$, $O$, $G$, and $W$, we want to construct a p-expression inducing a p-skyline relation $\succ \in \mathcal{F}_H$ such that

1. $G \subseteq \omega_r(O)$, i.e., the tuples in $G$ are among the most preferred tuples in $O$, according to $\succ$; and
2. for every tuple $o'$ in $W$, there is a tuple $o$ in $G$ such that $o \succ o'$, i.e., $o'$ is an inferior example.

Such a p-skyline relation $\succ$ is called favoring $G$ and disfavoring $W$ in $O$. We may also skip “in $O$” when the context is clear.

The first problem we consider is the existence of a p-skyline relation favoring $G$ and disfavoring $W$ in $O$.

**Problem DF-PSKYLINE.** Given a set of attributes $A$, a set of attribute preference relations $\mathcal{H}$, a set of superior examples $G$ and a set of inferior examples $W$ in a set $O$, determine if there exists a p-skyline relation $\succ \in \mathcal{F}_H$ favoring $G$ and disfavoring $W$ in $O$.

In most real life scenarios, knowing that a favoring/disfavoring p-skyline relation exists is not sufficient. One needs to know the contents of such a relation.

**Problem FDF-PSKYLINE.** Given a set of attributes $A$, a set of attribute preference relations $\mathcal{H}$, a set of superior examples $G$ and a set of inferior examples $W$ in a set $O$, construct a p-skyline relation $\succ \in \mathcal{F}_H$ favoring $G$ and disfavoring $W$ in $O$.

We notice that FDF-PSKYLINE is the functional version [Papadimitriou(1994)] of DF-PSKYLINE. Namely, given subsets $G$ and $W$ of $O$, an instance of FDF-PSKYLINE outputs “no” if there is no $\succ \in \mathcal{F}_H$ favoring $G$ and disfavoring $W$ in $O$. Otherwise, it outputs some p-skyline relation $\succ \in \mathcal{F}_H$ favoring $G$ and disfavoring $W$ in $O$.

**Example 9** Let the set $O$ consist of the following tuples describing cars for sale:

| make | price | year |
|------|-------|------|
| ford | 30k   | 2007 |
| bmw  | 45k   | 2008 |
| kia  | 20k   | 2007 |
| ford | 40k   | 2008 |
| bmw  | 50k   | 2006 |

Assume also Mary wants to buy a car and her preferences over automobile attributes are as follows.

$\succ_{\text{make}}$: BMW is better than Ford, Ford is better than Kia.

$\succ_{\text{year}}$: higher values of year are preferred.

$\succ_{\text{price}}$: lower values of price are preferred.

Let $G = \{t_2\}$, $W = \{t_1\}$. We elicit a p-skyline relation $\succ$ favoring $G$ and disfavoring $W$. First, $\succ_{\text{make}}$ cannot be
more important than all other attribute preferences, since then \( t_2 \) and \( t_5 \) dominate \( t_4 \) and thus \( t_4 \) is not superior. Moreover, \( \triangleright_{\text{price}} \) cannot be more important than the other attribute preferences, because then \( t_3 \) and \( t_1 \) dominate \( t_4 \). However, if \( \triangleright_{\text{year}} \) is more important than the other attribute preferences, then \( t_4 \) dominates \( t_1, t_3, t_5 \) and \( t_2 \) does not dominate \( t_3 \) in \( \triangleright_{\text{year}} \). At the same time, both \( t_2 \) and \( t_4 \) are the best according to \( \triangleright_{\text{year}} \), but \( t_2 \) dominates \( t_4 \) in \( \triangleright_{\text{make}} \). Therefore, \( \triangleright_{\text{make}} \) should not be more important than \( \triangleright_{\text{price}} \). Thus, for example, the p-skyline relation \( \triangleright = \triangleright_{\text{year}} \& (\triangleright_{\text{price}} \otimes \triangleright_{\text{make}}) \) favors \( G \) and disfavors \( W \) in \( O \). The set of the best tuples in \( O \) according to \( \triangleright_1 \) is \( \{t_2, t_4\} \).

Generally, there may be zero, one or more p-skyline relations favoring \( G \) and disfavoring \( W \) in \( O \). When more than one such relation exists, we pick a maximal one (in the set-theoretic sense). Larger preference relations imply more dominated tuples and fewer preferred ones. Consequently, the result of \( \omega_\triangleright (O) \) is likely to get more manageable due to its decreasing size. Moreover, maximizing \( \triangleright \) corresponds to minimizing \( \omega_\triangleright (O) - G \), which implies more precise correspondence of \( \triangleright \) to the real user preferences. Thus, the next problem considered here is constructing maximal p-skyline relations favoring \( G \) and disfavoring \( W \).

Problem OPT-\text{FDF-PSKLINE}. Given a set of attributes \( \mathcal{A} \), a set of attribute preference relations \( \mathcal{H} \), a set of superior examples \( G \) and a set of inferior examples \( W \) in a set \( O \), construct a maximal p-skyline relation \( \triangleright \in \mathcal{F}_G \) favoring \( G \) and disfavoring \( W \) in \( O \).

Example 10 Take \( G, W, \) and \( \triangleright \) from Example 9. Note that to make \( t_4 \) dominate \( t_2 \), we need to make \( \text{price} \) more important than \( \text{year} \). As a result, the relation
\[
\triangleright_2 = \triangleright_{\text{year}} \& \triangleright_{\text{price}} \& \triangleright_{\text{make}}
\]
also favors \( G \) and disfavors \( W \) in \( O \) but the set of best tuples in \( O \) according to \( \triangleright_2 \) is \( \{t_4\} \). Moreover, \( \triangleright_2 \) is maximal. The justification is that no other p-skyline relation favoring \( G \) and disfavoring \( W \) contains \( \triangleright_2 \) since the p-graph of \( \triangleright_2 \) is a total order of the attributes \( \{\text{year}, \text{price}, \text{make}\} \) and thus \( \triangleright_2 \) is a maximal SPO.

Even though the notion of maximal favoring/disfavoring reduces the space of alternative p-skyline relations, there may still be more than one maximal favoring/disfavoring p-skyline relation, given \( \mathcal{A}, \mathcal{H}, G, W, \) and \( O \).

4.2 Negative and positive constraints

We formalize now the kind of reasoning from Examples 9 and 10 using constraints on attribute sets. The constraints guarantee that the constructed p-skyline relation favors \( G \) and disfavors \( W \) in \( O \).

Consider the notion of favoring \( G \) in \( O \) first. For a tuple \( o' \in G \) to be in the set of the most preferred tuples of \( O, o' \) must not be dominated by any tuple in \( O \). That is,
\[
\forall o \in O, o' \in G (o \not\approx o') \tag{1}
\]

Using Theorem 5, we can rewrite (1) as
\[
\forall o \in O, o' \in G (Ch_{\triangleright_{\mathcal{H}}} (\text{BetIn}(o, o')) \not\supseteq \text{BetIn}(o', o)), \tag{2}
\]
where \( \text{BetIn}(o_1, o_2) = \{A \in \mathcal{A} \mid o_1 \triangleright A \triangleright o_2\} \). Note that no tuple can be preferred to itself by irreflexivity of \( \triangleright \). Thus, a p-skyline relation favoring \( G \) in \( O \) should satisfy \(|O| - 1\cdot|G|\) negative constraints \( \tau \) in the form:
\[
\forall \tau: Ch_{\triangleright_{\mathcal{H}}} (\mathcal{L}_\tau) \not\supseteq \mathcal{R}_\tau
\]
where \( \mathcal{L}_\tau = \text{BetIn}(o, o'), \mathcal{R}_\tau = \text{BetIn}(o', o) \). We denote this set of constraints as \( \mathcal{K}(G, O) \).

Example 11 Take Example 9. Then some p-skyline relation \( \triangleright \in \mathcal{F}_G \) favoring \( G = \{t_3\} \) in \( O \) has to satisfy each negative constraint below
\[
\begin{array}{|c|c|}
\hline
\tau & Ch_{\triangleright_{\mathcal{H}}} (\mathcal{L}_\tau) \not\supseteq \mathcal{R}_\tau \\
\hline
\{\text{make}\} & \{\text{price}\} \\
\{\text{make}, \text{year}\} & \{\text{price}\} \\
\{\text{make}, \text{year}\} & \{\text{price}, \text{year}\} \\
\hline
\end{array}
\]

Now consider the notion of disfavoring \( W \) in \( O \). According to the definition, a p-skyline relation \( \triangleright \) favoring \( G \) disfavors \( W \) in \( O \) iff the following holds
\[
\forall o' \in W \exists o \in G (o \not\approx o'). \tag{3}
\]

Following Theorem 5, it can be rewritten as a set of positive constraints \( \mathcal{P}(W, G) \)
\[
\forall o' \in W \bigvee_{o_i \in G} \text{Ch}_{\triangleright_{\mathcal{H}}} (\text{BetIn}(o_i, o')) \supseteq \text{BetIn}(o', o_i). \tag{4}
\]

Therefore, in order for \( \triangleright \) to disfavor \( W \) in \( O \), it has to satisfy \(|W|\) positive constraints.

Example 12 Take Example 9. Then every p-skyline relation \( \triangleright \in \mathcal{F}_G \) favoring \( G = \{t_1, t_2\} \) and disfavoring \( W = \{t_4\} \) in \( O \) has to satisfy the constraint \( (t_1 \triangleright t_4 \triangleright t_5) \), which is equivalent to the following positive constraint
\[
\text{Ch}_{\triangleright_{\mathcal{H}}} (\{\text{price}\}) \supseteq \text{Ch}_{\triangleright_{\mathcal{H}}} (\{\text{year}\}) \supseteq \text{Ch}_{\triangleright_{\mathcal{H}}} (\{\text{make}\}).
\]

\footnote{Here we again replace attribute preference relations by atomic preference relations.}
Notice that positive and negative constraints are formulated in terms of relative importance of the attributes captured by the p-graph of the constructed p-skyline relation. Since p-skyline relations are uniquely identified by p-graphs (Theorem 3), we may refer to a p-skyline relation satisfying/not satisfying a system of positive/negative constraints. Formally, a p-skyline relation satisfies a system of (positive or negative) constraints iff it satisfies every constraint in the system.

Let us summarize the kinds of constraints we have considered so far. To construct a p-skyline relation favoring G and disfavoring W in O, we need to construct a p-graph Γ > that satisfies SPO+Envelope to guarantee that > is a p-skyline relation, N(G, O) to guarantee favoring G in O, and P(W, G) to guarantee disfavoring W in O. By Theorem 4, the p-graph of a maximal > is maximal among all graphs satisfying SPO+Envelope, N(G, O), and P(W, G).

4.3 Using superior and inferior examples

In this section, we study the computational complexity of the problems of existence of a favoring/disfavoring p-skyline relation and of constructing a favoring/disfavoring p-skyline relation.

**Theorem 8** DF-PSKYLINE is NP-complete.

Now consider the problems of constructing favoring/disfavoring p-skyline relations. First, we consider the problem of constructing some p-skyline relation favoring G and disfavoring W in O. Afterwards we address the problem of constructing a maximal p-skyline relation. The results shown below are based on the following proposition.

**Proposition 5** Let > be a p-skyline relation, O a finite set of tuples, and G and W disjoint subsets of O. Then the next two operations can be done in polynomial time:

1. verifying if > is maximal favoring G and disfavoring W in O;
2. constructing a maximal p-skyline relation >ext that favors G and disfavors W in O, and is an e-extension of > favoring G and disfavoring W in O.

**Theorem 9** FDF-PSKYLINE is FNP-complete

Surprisingly, the problem of constructing a maximal favoring/disfavoring p-skyline relation is not harder than the problem of constructing some favoring/disfavoring p-skyline relation.

**Theorem 10** OPT-FDF-PSKYLINE is FNP-complete

4.4 Using only superior examples

In view of Theorems 8, 9, and 10, we consider now restricted versions of the favoring/disfavoring p-skyline relation problems, where we assume no inferior examples (W = ∅). Denote as DF⁺-PSKYLINE, FDF⁺-PSKYLINE, and OPT-FDF⁺-PSKYLINE the subclasses of DF-PSKYLINE, FDF-PSKYLINE, and OPT-FDF-PSKYLINE in which the sets of inferior examples W are empty. We show now that these problems are easier than their general counterparts: they can all be solved in polynomial time.

Consider DF⁺-PSKYLINE first. We showed in Corollary 1 that the set of the best objects according to the skyline preference relation is the largest among all p-skyline relations. Hence, the next proposition holds.

**Proposition 6** There exists a p-skyline relation > ∈ F⁺ favoring G in O iff G ⊆ ωsky⁺(O).

Proposition 6 implies that to solve DF⁺-PSKYLINE, one needs to run a skyline algorithm over O and check if the result contains G. This clearly can be done in polynomial time.

FDF⁺-PSKYLINE can also be solved in polynomial time: if G ⊆ ωsky⁺(O), then sky⁺ is a relation favoring G and disfavoring W in O. Otherwise, there is no such a relation.

Now consider OPT-FDF⁺-PSKYLINE. To specify a p-skyline relation > favoring G in O, we need to construct the corresponding graph Γ > which satisfies N(G, O) and SPO+Envelope. Furthermore, to make the relation > maximal favoring G in O, Γ > has to be a maximal graph satisfying these constraints. In the next section, we present an algorithm for constructing maximal p-skyline relations.

4.4.1 Syntax tree transformation

Our approach to constructing maximal favoring p-skyline relations favoring G is based on iterative transformations of normalized syntax trees. We assume that the provided set of superior examples G satisfies Proposition 6, i.e., G ⊆ ωsky⁺(O). The idea beyond our approach is as follows. First, we generate the set of negative constraints N(G, O). The p-skyline relation we start with is sky⁺ since it is the least p-skyline relation favoring G in O. In every iteration of the algorithm, we pick an attribute preference relation in H and apply a fixed set of transformation rules to the syntax tree of the current p-skyline relation. As a result, we obtain a "locally maximal" p-skyline relation satisfying the given set N(G, O) of negative constraints. Recall that a negative constraint in N(G, O) represents the requirement that no tuple in G is dominated by a tuple in O. Eventually, this technique produces a maximal p-skyline relation satisfying N(G, O).

Let us describe what we mean by "locally maximal".

**Definition 12** Let M be a nonempty subset of A. A p-skyline relation > ∈ F⁺ that favors G in O such that E(Γ >) ⊆ M × M is M-favoring G in O.
We note that, similarly to a maximal favoring p-skyline relation, a maximal $M$-favoring p-skyline relation is often not unique for given $G$, $O$, and $M$.

Proposition 7 Let a relation process be shown in the next proposition. A maximal $M$-favoring relation is often not unique for given $G$, $O$, and $M$.

Example 13 Let $\mathcal{A} = \{A_1, A_2, A_3, A_4\}$ and $\mathcal{H} = \{>_{A_1}, >_{A_2}, >_{A_3}, >_{A_4}\}$, where a greater value of the corresponding attribute is preferred, according to every $>_{A_i}$. Let the set of objects $O$ be as shown in Figure 10(a) and $G = \{t_1\}$. Then the set of negative constraints $\mathcal{N}(G, O)$ is shown in Figure 10(b): $\tau_1, \tau_2, \tau_3$ represent $t_2 \not>_{A_1} t_1$, $t_3 \not>_{A_2} t_1$, and $t_4 \not>_{A_3} t_1$, respectively. Consider the p-skyline relation $\sigma$ represented by the p-graph $\Gamma_\sigma$ shown in Figure 10(c). It is a maximal $\{A_1, A_2, A_3\}$-favoring relation because: 1) all the edges of $\Gamma_\sigma$ go between the nodes $\{A_1, A_2, A_3\}, 2) \Gamma_\sigma$ satisfies all the constraints in $\mathcal{N}(G, O)$ and 3) every additional edge from one attribute to another attribute in $\{A_1, A_2, A_3\}$ violates $\mathcal{N}(G, O)$. In particular, the edge $(A_1, A_3)$ violates $\tau_1$ and the edge $(A_2, A_3)$ violates $\tau_2$. Every other edge between $A_1, A_2, A_3$ induces one of the two edges above.

At the same time, $\sigma$ is not a maximal $\mathcal{A}$-favoring relation because, for example, the edge $(A_3, A_1)$ may be added to $\Gamma_\sigma$ without violating $\mathcal{N}(G, O)$.

By Definition 12, the edge set of the p-graph of every maximal $M$-favoring relation is maximal among all the p-graphs of $M$-favoring relations. Note that if $M$ is a singleton, the edge set of a p-graph $\Gamma_\sigma$ of a maximal $M$-favoring relation $\sigma$ is empty, i.e., $\sigma = \text{sky}_G$. If $M = \mathcal{A}$, then a maximal p-skyline relation $M$-favoring $G$ in $O$ is also a maximal p-skyline relation favoring $G$ in $O$. Thus, if we had a method of transforming a maximal $M$-favoring p-skyline relation to a maximal $(M \cup \{A\})$-favoring p-skyline relation for each attribute $A$, we could construct a maximal favoring p-skyline relation iteratively. A useful property of such a transformation process is shown in the next proposition.

Proposition 7 Let a relation $\sigma \in \mathcal{F}_G$ be a maximal $M$-favoring relation, and a p-extension $\tau_{ext}$ of $\sigma$ be $(M \cup \{A\})$-favoring. Then every edge in $E(\Gamma_{\tau_{ext}}) - E(\Gamma_\sigma)$ starts or ends in $A$.

Example 14 Consider $\mathcal{N}(G, O)$ from Example 13 (also depicted in Figure 11(a)), and the maximal $\{A_1, A_2, A_3\}$-favoring p-skyline relation $\sigma$. Several different maximal $\mathcal{A}$-favoring p-skyline relations containing $\sigma$ exist. Two of them are $\sigma_1$ and $\sigma_2$ whose p-graphs are shown in Figures 11(b) and 11(c).

In section 3.4, we showed four syntax tree transformation rules. Rule 1, Rule 4, for extending p-skyline relations in a minimal way. Although a maximal $(M \cup \{A\})$-favoring p-skyline relation is a p-extension of a maximal $M$-favoring p-skyline relation, it is not necessary a minimal p-extension in general. However, an important property of the rule set is its completeness, i.e., every minimal p-extension can be constructed using them. Hence, a maximal $(M \cup \{A\})$-favoring p-skyline relation can be produced from a maximal $M$-favoring p-skyline relation by iterative application of the minimal extension rules. This process is illustrated by Figure 12.

We use the following idea for constructing maximal $(M \cup \{A\})$-favoring relations. We start with a maximal $M$-favoring p-skyline relation $\sigma$ and apply the transformation rules to $T_{\sigma}$ in every possible way guaranteeing that the new edges in the p-graph go only from or to $A$. In other words, we construct all minimal $(M \cup \{A\})$-favoring p-extensions of $\sigma$. We construct such p-extensions until we find the first one which does not violate $\mathcal{N}(G, O)$. When we find it (denote it as $\sigma_m$), we repeat all the steps above but for $\sigma_m$. This process continues until for some $\sigma_m$, every of its constructed minimal p-extension violates $\mathcal{N}(G, O)$. Since in every iteration we construct all minimal $(M \cup \{A\})$-favoring p-extensions, $\sigma_m$ is a maximal $(M \cup \{A\})$-favoring p-extension of $\sigma$.

There is subtle point here. We can limit ourselves to minimal p-extensions because if a minimal p-extension violates $\mathcal{N}(G, O)$, so do all non-minimal p-extensions containing it. Also, if there exists a p-extension satisfying $\mathcal{N}(G, O)$, so does some minimal one. In fact, each p-extension of a p-skyline relation can be obtained through a finite sequence of minimal p-extensions. Those properties are characteristic of negative constraints. The properties do not hold for positive
constraints and thus our approach cannot be directly generalized to such constraints. An important condition to apply Theorem 7 is that the input syntax tree for every transformation rule be normalized. At the same time, syntax trees returned by the transformation rules are not guaranteed to be normalized. Therefore, we need to normalize a tree before applying transformation rules to it.

Consider the rules Rule1 – Rule4 which can be used to construct an \((M \cup \{A\})\)-favoring p-skyline relation from an \(M\)-favoring one. By Proposition 7, such rules may only add to the p-graph the edges that go to \(A\) or from \(A\). According to Observation 1, Rule1 adds edges going to the node \(A\) if \(C_{i+1} = A\) or \(N_1 = A\). Similarly, Rule2 adds edges going from \(A\) if \(C_{i+1} = A\) or \(N_0 = A\). Rule3 adds edges going from or to \(A\) if \(C_i = A\) or \(C_{i+1} = A\) correspondingly. However, Rule4 can only be applied to a pair of \& -nodes. Hence, as we showed in section 3.4, Rule4 adds edges going from at least two nodes to at least two different nodes of a p-graph. Hence, every application of Rule4 violates Proposition 7. We conclude that Rule1, Rule2, and Rule3 are sufficient to construct every maximal \((M \cup \{A\})\)-favoring p-skyline relation.

4.4.2 Efficient constraint checking

Before going into the details of the algorithm of p-skyline relation elicitation, we consider an important step of the algorithm: testing if a p-extension of a p-skyline relation satisfies a set of negative constraints. We propose now an efficient method for this task.

Recall that a negative constraint is of the form \(\tau : Ch_{\Gamma_{\neg}}(L_x) \not\supseteq R_x\).

It can be visualized as two layers of nodes \(L_x\) and \(R_x\). For a p-skyline relation \(\succ \in F_{\phi}\) satisfying \(\tau\), its p-graph \(\Gamma_{\neg}\) may contain edges going between the nodes of the layers \(L_x\) and \(R_x\). However, in order for \(\succ\) to satisfy \(\tau\), there should be at least one member of \(R_x\) with no incoming edges from \(L_x\).

The method of efficient checking of negative constraints against a p-graph that we propose here is based on the fact that the edge set of the p-graph of a transformed p-skyline relation monotonically increases. Therefore, while we transform a p-skyline relation \(\succ\), we can simply drop the elements of \(R_x\) which already have incoming edges from \(L_x\). If we do so after every transformation of the p-skyline relation \(\succ\), the negative constraint \(\tau\) will be violated by \(\Gamma_{\neg}\) only if \(R_x\) is empty. The next proposition says that such a modification of negative constraints is valid.

**Proposition 8** Let a relation \(\succ \in F_{\phi}\) satisfy a system of negative constraints \(\mathcal{N}\). Construct the system of negative constraints \(\mathcal{N}'\) from \(\mathcal{N}\) in which every constraint \(\tau' \in \mathcal{N}'\) is created from a constraint \(\tau\) of \(\mathcal{N}\) in the following way:

- \(L_x = L_t\)
- \(R_x = R_t - \{B \in R_t \mid \exists A \in L_t ((A, B) \in \Gamma_{\neg})\}\)

Then every p-extension \(\succ'\) of \(\succ\) satisfies \(\mathcal{N}'\) iff \(\succ'\) satisfies \(\mathcal{N}\)’.

A constraint \(\tau'\) constructed from \(\tau\) as shown in Proposition 8 is called a minimal negative constraint w.r.t. \(\succ\). The corresponding system of negative constraints \(\mathcal{N}'\) is called a system of minimal negative constraints w.r.t. \(\succ\).

Minimization of a system of negative constraints is illustrated in the next example.

![Example 15](image)

**Example 15** Consider the system of negative constraints \(\mathcal{N}\) and the p-skyline relation \(\succ\) from Example 13 (they are shown in Figures 13(a) and 13(b) correspondingly). The result \(\mathcal{N}'\) of minimization of \(\mathcal{N}\) w.r.t. \(\succ\) is shown in Figure 13(c). Only the constraint \(\tau'\) of \(\mathcal{N}'\) is different from \(\tau_2\) because \((A_2, A_3) \in \Gamma_{\neg}\) and \((A_2, A_3) \notin \mathcal{N}'\).

The next proposition summarizes the constraint checking rules over a system of minimal negative constraints.

**Proposition 9** Let a relation \(\succ \in F_{\phi}\) satisfy a system of negative constraints \(\mathcal{N}\) and \(\mathcal{N}\) be minimal w.r.t. \(\succ\). Let \(\succ'\) be a p-extension of \(\succ\) such that every edge in \(E(\Gamma_{\neg}) - E(\Gamma_{\neg})\) starts or ends in \(A\). Denote the new parents and children of \(A\) in \(\Gamma_{\neg}\), as \(P_A\) and \(C_A\) correspondsingly. Then \(\succ'\) violates \(\mathcal{N}\) iff there is a constraint \(\tau \in \mathcal{N}\) such that

1. \(R_x = \{A\} \wedge P_A \cap L_x \neq \emptyset\), or
2. \(A \in L_t \wedge R_x \subseteq C_A\)

Proposition 9 is illustrated in the next example.

**Example 16** Take the system of minimal negative constraints \(\mathcal{N}'\) w.r.t. \(\succ\) from Example 15. Construct a p-extension \(\succ'\) of \(\succ\) such that every edge in \(E(\Gamma_{\neg}) - E(\Gamma_{\neg})\) starts or ends in \(A_4\). Consider possible edges going to \(A_4\). Use Proposition 9 to check if a new edge violates \(\mathcal{N}'\). The edge \((A_1, A_4)\) is not allowed in \(\Gamma_{\neg}\) because then \(A_1 \in L_{\neg}\) and \(\{A_1\} = R_{\neg}\) (and thus the constraint \(\tau\) is violated). The edge \((A_3, A_4)\) is not allowed in \(\Gamma_{\neg}\) because \(A_3 \in L_{\neg}\) and \(\{A_3\} = R_{\neg}\). However, the edge \((A_2, A_4)\) is allowed in \(\Gamma_{\neg}\). The p-graph of the resulting \(\succ'\) is shown in Figure 14. One can analyze the edges going from \(A_4\) in a similar fashion.
In this section, we show an algorithm for p-skyline relation elicitation which exploits the ideas developed in the previous sections.

The function `elicit` (Algorithm 1) is the main function of the algorithm. It takes four arguments: the set of superior examples $G$, the entire set of tuples $O$, the set of attribute preferences $\mathcal{H}$, and the set of all relevant attributes $\mathcal{A}$. It returns a normalized syntax tree of a maximal p-skyline relation favoring $G$ in $O$. Following Proposition 6, we require $G$ to be a subset of $\omega_{sky}(O)$. First, we construct the set of negative constraints $\mathcal{N'}$ for the superior tuples $G$. We start with $sky_{\mathcal{H}}$ as the initial p-skyline relation favoring $G$ in $O$. After that, we take the set $M$ consisting of a single attribute. In every iteration, we enlarge it and construct a maximal $M$-favoring p-skyline relation. As a result, the function returns a maximal p-skyline relation favoring $G$ in $O$. The construction of a maximal $(M \cup \{A\})$-favoring relation from a maximal $M$-favoring relation is performed in the `repeat/until` loop (lines 5-8). Here we use the function `push` which constructs a minimal $(M \cup \{A\})$-favoring p-extension of the relation represented by the syntax tree $T$. It returns `true` if $T$ has been (minimally) extended to a relation not violating $\mathcal{N'}$, and further p-extensions are feasible (though they may still violate $\mathcal{N'}$). Otherwise, it returns `false`. The syntax tree $T$ passed to `push` has to be normalized. Hence, after extending the relation, we normalize its syntax tree (line 7) using the normalization procedure sketched in Section 3.1. The `repeat/until` loop terminates when all minimal extensions of $T$ violate $\mathcal{N'}$.

**Algorithm 1** `elicit(G, O, \mathcal{H}, \mathcal{A})`

1. $\mathcal{N'} = \mathcal{N}(G, O)$
2. $T = \text{a normalized syntax tree of } sky_{\mathcal{H}}$
3. $M = \text{set containing an arbitrary attribute from } \mathcal{A}$
4. for each attribute $A$ in $\mathcal{A} - M$ do
5.   `repeat`
6.   $r = \text{push}(T, M, A, \mathcal{N'});$  
7.   `normalizeTree(root of T);`
8.   `until` $r$ is `false`
9.   $M = M \cup \{A\}$
10. `end for`
11. `return T`

Let us now take a closer look at the function `push` (Algorithm 2). It takes four arguments: a set $M$ of attributes, a normalized syntax tree $T$ of an $M$-favoring p-skyline relation $\succ$, the current attribute $A$, and a system of negative constraints $\mathcal{N}$ minimal w.r.t. $\succ$. It returns `true` if a transformation rule $q \in \{Rule_1, Rule_2, Rule_3\}$ has been applied to $T$ without violating $\mathcal{N}$, and `false` if no transformation rule can be applied to $T$ without violating $\mathcal{N}$. When `push` returns `true`, $\mathcal{N'}$ and $T$ have been changed. Now $\mathcal{N'}$ is minimal w.r.t. the p-skyline relation represented by the modified syntax tree, and $T$ has been modified by the rule $q$ and is normalized.

The goal of `push` is to find an appropriate transformation rule which adds to the current p-graph edges going from $M$ to $A$ or vice versa. The function has two branches: the first for the parent of the node $A$ in the syntax tree $T$ being a $\&$-node (i.e., we may apply `Rule_1` where $N_1$ is $A$ or `Rule_2` where $N_m$ is $A$), and the second for it being a $\otimes$-node (i.e., we may apply `Rule_1` or `Rule_2` where $C_{r+1}$ is $A$, or `Rule_3` where $C_r$ or $C_{r+1}$ is $A$). In the first branch (line 2-14), we distinguish between applying `Rule_1` (line 3-8) and `Rule_2` (line 9-14). It is easy to notice that, with the parameters specified above, the rules are exclusive, but the application patterns are similar. First, we find an appropriate child $C_{r+1}$ of $R$ (lines 4 and 10). (It is important for `Var(C_{r+1})` to be a subset of $M$ because we want to add edges going from $M$ to $A$ or from $A$ to $M$.) Then we check if the corresponding rule application does not violate $\mathcal{N'}$ using the function `checkConstr` (lines 5 and 11), as per Proposition 9. If the rule application does not violate $\mathcal{N'}$, we apply the corresponding rule to $T$ (lines 6 and 12) and minimize $\mathcal{N'}$ w.r.t. the p-skyline relation which is the result of the transformation (Proposition 8) using the function `minimize`.

The second branch of `push` is similar to the first one and different only in the transformation rules applied. So it is easy to notice that `push` checks every possible rule application not violating $\mathcal{N'}$ and adds to the p-graph only edges going from $A$ to the elements of $M$ or vice versa.

In our implementation of the algorithm, all sets of attributes are represented as bitmaps of fixed size $|\mathcal{A}|$. Similarly, every negative constraint is represented as a pair of bitmaps corresponding to $L_\tau$ and $R_\tau$. With every node $C_i$ of the syntax tree, we associate a variable storing `Var(C_i)`. Its value is updated whenever the children list of $C_i$ is changed.

**Theorem 11** The function `elicit` returns a syntax tree of a maximal p-skyline relation favoring $G$ in $O$. Its running time is $O(|\mathcal{N'}| \cdot |\mathcal{A}|^3)$.

The order in which the attributes are selected and added to $M$ in `elicit` is arbitrary. Moreover, the order of rule application in `push` may be also changed. That is, we currently try to apply `Rule_1` (line 21) first and `Rule_2` (line 25) afterwards. However, one can apply the rules in the opposite order. The same observation applies to `Rule_3(T,A,C)` and `Rule_3(T,C,A)` (lines 30 and 34, respectively). If the algo-
Algorithm 2 \( \text{push}(T, M, A, A') \)

**Require:** \( T \) is normalized

1. if the parent of \( A \) in \( T \) is of type &
2. \( C_i \leftarrow \) parent of \( A \) in \( T \); \( R_i \leftarrow \) parent of \( C_i \) in \( T \);
3. if \( R \) is defined, and \( A \) is the first child of \( C_i \)
4. for each child \( C_{i+1} \) of \( R \) s.t. \( \text{Var}(C_{i+1}) \subseteq M \)
5. if \( \text{checkConstr}(A', A, \text{Var}(C_{i+1})) \)
6. apply Rule \((T, C, C_{i+1})\)
7. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(C_{i+1})) \)
8. return true

9. else if \( R \) is defined, and \( A \) is the last child of \( C_i \)
10. for each child \( C_{i+1} \) of \( R \) s.t. \( \text{Var}(C_{i+1}) \subseteq M \)
11. if \( \text{checkConstr}(A', A, \text{Var}(C_{i+1}), \emptyset) \)
12. apply Rule \((T, C, C_{i+1})\)
13. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(C_{i+1})) \)
14. return true

15. else if the parent of \( A \) in \( T \) is of type \( \otimes \)
16. \( R \leftarrow \) parent of \( A \) in \( T \);
17. for each child \( C_i \) of \( R \) s.t. \( \text{Var}(C_i) \subseteq M \)
18. if \( C_i \) is of type &
19. \( N_i \leftarrow \) first child of \( C_i \), \( N_m \leftarrow \) last child of \( C_i \)
20. if \( \text{checkConstr}(A', A, \text{Var}(N_i), \emptyset) \)
21. apply Rule \((T, C, A)\)
22. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(N_i)) \)
23. return true
24. else if checkConstr \((A', A, \emptyset, \text{Var}(N_m)) \)
25. apply Rule \((T, C, A)\)
26. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(N_m)) \)
27. return true
28. else if \( C_i \) is a leaf node, since \( T \) is normalized
29. if \( \text{checkConstr}(A', A, \text{Var}(C_i), \emptyset) \)
30. apply Rule \((T, C, A)\)
31. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(C_i)) \)
32. return true
33. else if checkConstr \((A', A, \emptyset, \text{Var}(C_i)) \)
34. apply Rule \((T, A, C)\)
35. \( \mathcal{N} \leftarrow \text{minimize}(\mathcal{N}, \text{Var}(A), \text{Var}(C_i)) \)
36. return true
37. return false

Algorithm 3 \( \text{checkConstr}(\mathcal{N}, A, P_A, C_A) \)

for each \( \tau \) in \( \mathcal{N} \)
do
\( \mathcal{R}_\tau = \{A\} \times P_A \cap L_\tau \neq \emptyset \) or \( A \in L_\tau \land \mathcal{R}_\tau \subseteq C_A \) then
return false
end if
end for
return true

Algorithm 4 \( \text{minimize}(\mathcal{N}, U, D) \)

1. for each constraint \( \tau \) in \( \mathcal{N} \) do
2. if \( U \cap L_\tau \neq \emptyset \) then
3. \( \mathcal{R}_\tau \leftarrow \mathcal{R}_\tau - D \)
4. end if
5. end for
6. return \( \mathcal{N} \)

The algorithm is changed along those lines, the generated p-skylines relation may be different. However, even if the p-skylines relation is different, it will still be a maximal p-skylines relation favoring \( G \) in \( O \). Note also that due to the symmetry of \( \otimes \), the order of children nodes of a \( \otimes \)-node may be different in normalized p-skylines of equivalent p-skylines relations. Hence, the order in which the leaf nodes are stored in the normalized syntax tree of \( \text{sky}_H \) (line 2 of \( \text{elicit} \)) also affects the resulting p-skylines relation.

![Fig. 15 Example 17](image-url)

**Example 17** Take \( O \) and \( H \) from Example 9, and \( G \) from Example 11. Then the corresponding system of negative constraints \( \mathcal{N} = \mathcal{N}(G, O) \) (Example 11) is shown in Figure 15(a). Consider the attributes in the following order: \( \text{make} \), \( \text{price} \), \( \text{year} \). Run \( \text{elicit} \). The tree \( T \) (line 2) is shown in Figure 15(b). The initial value of \( M \) is \{\text{make}\}. First, call \( \text{push}(T, \{\text{make}\}, \text{price} \), \( \mathcal{N} \). The parent of \( \text{price} \) is a \( \otimes \)-node (Figure 15(b)), so we go to line 16 of \( \text{push} \), where \( R \) is set to the \( \otimes \)-node (Figure 15(b)). After \( C_i \) is set to the node \text{make} in line 17, we go to line 29 because it is a leaf node. The \( \text{checkConstr} \) test in line 29 fails because \( \mathcal{N} \) prohibits the edge (\text{make}, \text{price})]. Hence, we go to line 33 where the \( \text{checkConstr} \) test succeeds. We apply \( \text{Rule}_3(T, \text{price}, C_i) \), \( \text{push} \) returns true, and the resulting syntax tree \( T \) is shown in Figure 15(c). Next time we call \( \text{push}(T, \{\text{make}\}, \text{price}, \mathcal{N}) \) in the line 6 of \( \text{elicit} \), we get to the line 4 of \( \text{push} \). Since \( \text{year} \notin M \), we immediately go to line 37 and return false. In \( \text{elicit} \), \( M \) is set to \{\text{make, price}\} and \( \text{push}(T, \{\text{make, price}\}, \text{year}, \mathcal{N}) \) is called. There we go to line 16 (\( R \) is set to the \( \otimes \)-node in Figure 15(c)), \( C_i \) is set to the \&-node (Figure 15(c)), we apply \( \text{Rule}_1(T, C_i, \text{year}) \) (the resulting tree \( T \) is shown in Figure 15(d)), and true is returned. When \( \text{push}(T, \{\text{make, price}\}, \text{year}, \mathcal{N}) \) is called the next time, we first go to line 16, \( R \) is set to the \( \otimes \)-node (Figure 15(d)), and \( C_i \) to the node \text{make}. Then \( \text{Rule}_1(T, C_i, \text{year}) \) is applied (line 30) resulting in the tree \( T \) shown in Figure 15(e), and true is returned. Now \( \text{push}(T, \{\text{make, price}\}, \text{year}, \mathcal{N}) \) gets called once again from \( \text{elicit} \) and returns false, and thus the tree in Figure 15(e) is the final one. According to the corresponding p-skylines relation, \( t_s \) dominates all other tuples in \( O \).
relations. This is because $\mathcal{N}$ effectively contained only one constraint (all constraints are implied by $\tau_2$, as shown below). When more constraints are involved, an elicited p-skyline relation may also have occurrences of $\ominus$.

4.5 Reducing the size of systems of negative constraints

As we showed in Theorem 11, the running time of the function $\text{elicit}$ linearly depends on the size of the system of negative constraints $\mathcal{N}$. If $\mathcal{N} = \mathcal{N}(G, O)$, then $\mathcal{N}$ contains $|O| \cdot |G|$ constraints. A natural question which arises here is whether we really need all the constraints in $\mathcal{N}$ to elicit a maximal p-skyline relation satisfying $\mathcal{N}$. In particular, can we replace $\mathcal{N}$ with an equivalent subset of $\mathcal{N}$?

We define equivalence of systems of negative constraints in a natural way.

**Definition 13** Given two systems of negative constraints $\mathcal{N}_1$ and $\mathcal{N}_2$, and two negative constraints $\tau_1, \tau_2$:

- $\mathcal{N}_1$ (resp. $\tau_1$) implies $\mathcal{N}_2$ (resp. $\tau_2$) if every $\tau \in \mathcal{F}_H$ satisfying $\mathcal{N}_1$ (resp. $\tau_1$) also satisfies $\mathcal{N}_2$ (resp. $\tau_2$);
- $\mathcal{N}_1$ (resp. $\tau_1$) strictly implies $\mathcal{N}_2$ (resp. $\tau_2$) if every $\tau \in \mathcal{F}_H$ satisfying $\mathcal{N}_1$ (resp. $\tau_1$) also satisfies $\mathcal{N}_2$ (resp. $\tau_2$), but $\mathcal{N}_2$ (resp. $\tau_2$) does not imply $\mathcal{N}_1$ (resp. $\tau_1$);
- $\mathcal{N}_1$ (resp. $\tau_1$) is equivalent to $\mathcal{N}_2$ (resp. $\tau_2$) if $\mathcal{N}_1$ (resp. $\tau_1$) implies $\mathcal{N}_2$ (resp. $\tau_2$) and vice versa.

In particular, a subset of $\mathcal{N}(G, O)$ from Example 17 that is equivalent to $\mathcal{N}(G, O)$ is $\mathcal{N}' = \{\tau_2\}$: first, $\mathcal{N}'$ clearly implies $\mathcal{N}(G, O)$; second, $\{\tau_2\}$ is trivially implied by $\{\tau_2\}$, $\{\tau_1\}$ is implied by $\{\tau_2\}$ if price is not a child of either make or year, it is not a child of make, and $\{\tau_1\}$ is implied by $\{\tau_2\}$ (if price is a child of neither make nor year, then both price and year cannot be children of make).

Below we propose a number of methods for computing an equivalent subset of a system of negative constraints.

4.5.1 Using $\text{sky}_{\mathcal{F}_H}(O)$ instead of $O$

The first method of reducing the size of a system of negative constraints is based on the following observation. Recall that each negative constraint is used to show that a tuple should not be preferred to a superior example. We also know that the relation $\text{sky}_{\mathcal{F}_H}$ is the least p-skyline relation. By definition of the winnow operator, for every $o' \in (O - \omega_{\text{sky}_{\mathcal{F}_H}}(O))$ there is a tuple $o \in \omega_{\text{sky}_{\mathcal{F}_H}}(O)$ s.t. $o$ is preferred to $o'$ according to $\text{sky}_{\mathcal{F}_H}$. Since $\text{sky}_{\mathcal{F}_H}$ is the least p-skyline relation, the same $o$ is preferred to $o'$ according to every p-skyline relation. Thus, to guarantee favoring $G$ in $O$, the system of negative constraints needs to contain only the constraints showing that the tuples in $\omega_{\text{sky}_{\mathcal{F}_H}}(O)$ are not preferred to the superior examples. Hence, the following proposition holds.

**Proposition 10** Given $G \subseteq \omega_{\text{sky}_{\mathcal{F}_H}}(O)$, $\mathcal{N}(G, O)$ is equivalent to $\mathcal{N}(G, \omega_{\text{sky}_{\mathcal{F}_H}}(O))$.

Notice that $\mathcal{N}(G, \omega_{\text{sky}_{\mathcal{F}_H}}(O))$ contains $|\omega_{\text{sky}_{\mathcal{F}_H}}(O)| \cdot |G|$ negative constraints. Proposition 10 also imply an important result: if a user considers a tuple $t$ superior based on the comparison with $\omega_{\text{sky}_{\mathcal{F}_H}}(O)$, comparing $t$ with the tuples in $(O - \omega_{\text{sky}_{\mathcal{F}_H}}(O))$ does not add any new information.

4.5.2 Removing redundant constraints

The second method of reducing the size of a negative constraint system is based on determining the implication of distinct negative constraints in a system. Let two $\tau_1, \tau_2 \in \mathcal{N}$ be such that $\mathcal{L}_{\tau_2} \subseteq \mathcal{L}_{\tau_1}$, $\mathcal{R}_{\tau_1} \subseteq \mathcal{R}_{\tau_2}$. It is easy to check that $\tau_1$ implies $\tau_2$. Thus, the constraint $\tau_2$ is redundant and may be deleted from $\mathcal{N}$. This idea can also be expressed as follows: $\tau$ implies $\tau'$ iff $\mathcal{L}_{\tau'} \subseteq \mathcal{L}_{\tau} \land (\mathcal{A} - \mathcal{R}_{\tau'}) = (\mathcal{A} - \mathcal{R}_{\tau})$.

Let us represent $\tau$ as a bitmap representing $(\mathcal{A} - \mathcal{R}_{\tau})$ appended to a bitmap representing $\mathcal{L}_{\tau}$. We assume that a bit is set to 1 if the corresponding attribute is in the corresponding set $(\mathcal{L}_{\tau} \land (\mathcal{A} - \mathcal{R}_{\tau'}))$, resp. Denote such a representation as $\text{bitmap}(\tau)$.

**Example 18** Let $\mathcal{L}_{\tau} = \{A_1, A_3, A_5\}$, $\mathcal{R}_{\tau} = \{A_2\}$, $\mathcal{L}_{\tau'} = \{A_2, A_4\}$. Let $\mathcal{A} = \{A_1, \ldots, A_5\}$. As a result, $\text{bitmap}(\tau) = 10101$ and $\text{bitmap}(\tau') = 10001$.

Consider $\text{bitmap}(\tau)$ as a vector with $2 \cdot |\mathcal{A}|$ dimensions. From the negative constraint implication rule, it follows that $\tau$ strictly implies $\tau'$ iff $\text{bitmap}(\tau)$ and $\text{bitmap}(\tau')$ satisfy the Pareto improvement principle, i.e., the value of every dimension of $\text{bitmap}(\tau)$ is greater or equal to the corresponding value in $\text{bitmap}(\tau')$, and there is at least one dimension whose value in $\text{bitmap}(\tau)$ is greater than in $\text{bitmap}(\tau')$. Therefore, the set of all non-redundant constraints in $\mathcal{N}$ corresponds to the skyline of the set of bitmap representations of all constraints in $\mathcal{N}$. Moreover, $\text{bitmap}(\tau)$ can have only two values in every dimension: 0 or 1. Thus, algorithms for computing skylines over low-cardinality domains (e.g. [Morse et al. (2007)]) can be used to compute the set of non-redundant constraints.

4.5.3 Removing redundant sets of constraints

The method of determining redundant constraints in the previous section is based on distinct constraint implication. A more powerful version of this method would compute and discard redundant subsets of $\mathcal{N}$ rather then redundant distinct constraints. However, as we show in this section, that problem appears to be significantly harder.

**Problem** \textbf{SUBSET-EQUIV.} Given systems of negative constraints $\mathcal{N}_1$ and $\mathcal{N}_2$ s.t. $\mathcal{N}_2 \subseteq \mathcal{N}_1$, check if $\mathcal{N}_2$ is equivalent to $\mathcal{N}_1$. \


To determine the complexity of \( \text{SUBSET-EQUIV} \), we use a helper problem.

**Problem NEG-SYST-IMPL.** Given two systems of negative constraints \( \mathcal{N}_1 \) and \( \mathcal{N}_2 \), check if \( \mathcal{N}_1 \) implies \( \mathcal{N}_2 \).

It turns out that the problems \( \text{NEG-SYST-IMPL} \) and \( \text{SUBSET-EQUIV} \) are intractable in general.

**Theorem 12** \( \text{NEG-SYST-IMPL} \) is co-NP complete

**Theorem 13** \( \text{SUBSET-EQUIV} \) is co-NP complete

We notice that even though the problem of minimizing the size of a system of negative constraints is intractable in general, the methods of reducing its size we proposed in sections 4.5.2 and 4.5.1 result in a significant decrease in the size of the system. This is illustrated in Section 5.

### 5 Experiments

We have performed extensive experimental study of the proposed framework. The algorithms were implemented in Java. The experiments were run on Intel Core 2 Duo CPU 2.1 GHz with 2.0GB RAM under Windows XP. We used four data sets: one real-life and three synthetic.

#### 5.1 Experiments with real-life data

In this subsection, we focus on experimenting with the accuracy of the \( \text{elicit} \) algorithm and the reduction of winnow result size, achieved by modeling user preferences using p-skyline relations. We use a data set \( \text{NHL} \) which stores statistics of NHL players [nhl(2008)], containing 9395 tuples. We consider three sets of relevant attributes \( \mathcal{A} \) containing 12, 9, and 6 attributes. The size of the corresponding skylines is 568, 114, and 33, respectively.

##### 5.1.1 Precision and recall

The aim of the first experiment is to demonstrate that the \( \text{elicit} \) algorithm has high accuracy. We use the following scenario. We assume that the real, hidden preferences of the user are modeled as a p-skyline relation \( \succ_{\text{hid}} \). We also assume that the user provides the set of relevant attributes \( \mathcal{A} \), the set of corresponding attribute preferences \( \mathcal{H} \), and a set \( \mathcal{G}_{\text{hid}} \) of tuples which she likes most in \( \text{NHL} \) (i.e., \( \mathcal{G}_{\text{hid}} \) are superior examples and \( \mathcal{G}_{\text{hid}} \subseteq \omega_{\succ_{\text{hid}}} \)). We use \( \mathcal{G}_{\text{hid}} \) to construct a maximal p-skyline relation \( \succ \) favoring \( \mathcal{G}_{\text{hid}} \) in \( \text{NHL} \). To measure the accuracy of \( \text{elicit} \), we compare the set of the best tuples \( \omega_{\succ} \) with the set of the best tuples \( \omega_{\succ_{\text{hid}}} \). The latter is supposed to correctly reflect user preferences.

To model user preferences, we randomly generate 100 p-skyline relations \( \succ_{\text{hid}} \). For each \( \omega_{\succ_{\text{hid}}} \), we randomly pick 5 tuples from it, and use them as superior examples \( \mathcal{G}_{\text{hid}} \) to elicit three different maximal p-skyline relations \( \succ \) favoring \( \mathcal{G}_{\text{hid}} \) in \( \text{NHL} \). Out of those three relations, we pick the one resulting in \( \omega_{\succ} \) of the smallest size. Then we add 5 more tuples from \( \omega_{\succ_{\text{hid}}} \) to \( \mathcal{G}_{\text{hid}} \) and repeat the same procedure. We keep adding tuples to \( \mathcal{G}_{\text{hid}} \) from \( \omega_{\succ_{\text{hid}}} \) until \( \mathcal{G}_{\text{hid}} \) reaches \( \omega_{\succ_{\text{hid}}} \).

To measure the accuracy of the \( \text{elicit} \) algorithm, we compute the following three values:

1. Precision of the p-skyline elicitation method:
   \[
   \text{precision} = \frac{|\omega_{\succ}(\text{NHL}) \cap \omega_{\succ_{\text{hid}}}(\text{NHL})|}{|\omega_{\succ}(\text{NHL})|},
   \]
2. Recall of the p-skyline elicitation method:
   \[
   \text{recall} = \frac{|\omega_{\succ}(\text{NHL}) \cap \omega_{\succ_{\text{hid}}}(\text{NHL})|}{|\omega_{\succ_{\text{hid}}}(\text{NHL})|},
   \]
3. \( F \)-measure which combines precision and recall:
   \[
   F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
   \]

We plot the average values of those measures in Figures 16(a), 16(b), and 16(c). As can be observed, precision of the \( \text{elicit} \) algorithm is high in all experiments: it is greater than 0.9 in most cases. However, recall starts from a low value when the number of superior examples is low. This is due to the fact that \( \text{elicit} \) constructs a maximal relation favoring \( \mathcal{G}_{\text{hid}} \) in \( \text{NHL} \), and small \( \mathcal{G}_{\text{hid}} \) is not sufficient to capture the preference relation \( \succ_{\text{hid}} \) and thus the ratio of false negatives is rather high. When we increase the number of superior examples, recall consistently grows.

In Figure 16(d), we plot the values of the \( F \)-measure with respect to the share of the skyline used as superior examples. The value of \( F \) starts from a comparatively low value of 0.7 but quickly reaches 0.9 via a small increase of the size of \( \mathcal{G}_{\text{hid}} \). The value of \( F \) is generally inversely dependent on the number of relevant attributes (given the same ratio of superior examples used). This is justified by the following observation. To construct a p-skyline relation favoring \( \mathcal{G}_{\text{hid}} \) in \( \text{NHL} \), the algorithm uses a set of negative constraints \( \mathcal{N} \). Intuitively, the constructed p-skyline relation \( \succ \) will match the original relation \( \succ_{\text{hid}} \) better if the set \( \mathcal{N} \) captures \( \succ_{\text{hid}} \) sufficiently well. The number of constraints in \( \mathcal{N} \) depends not only on the number of superior examples but also on the skyline size. Skyline sizes are generally smaller for smaller sets of \( \mathcal{A} \), and more superior examples are needed for smaller \( \mathcal{A} \) to capture \( \succ_{\text{hid}} \).

#### 5.1.2 Winnow result size

In Section 1, we discussed a well known deficiency of the skyline framework: skylines are generally of large size for large sets of relevant attributes \( \mathcal{A} \). The goal of the experiments in this section is twofold. First, we demonstrate that using p-skyline relations to model user preferences results in
larger skyline size. That is, drawing superior examples is more difficult than for those from \( G_{\text{rand}} \). Notice that \( \text{winnow-size-ratio} \) is always smaller when it contains only a few tuples; and increasing its size decreases the similarity of the tuples, resulting in a quick growth of \( \text{winnow-size-ratio} \). Since superior examples correspond to a real p-skyline relation, they share some similarity expressed using the attribute importance relationships. For \( G_{\text{rand}} \), such similarity exists when it contains only a few tuples; and increasing its size decreases the similarity of the tuples, resulting in a quick growth of \( \text{winnow-size-ratio} \).

5.2 Experiments with synthetic data

Here we present experiments with synthetic data. The main goals of the experiments is to demonstrate that the proposed p-skyline relation elicitation approach is scalable and allows effective optimizations. We use three synthetic data sets here: correlated \( S_1 \) (based on linear dependence), anti-correlated \( S_2 \) (based on Zipf distribution), and uniform \( S_1 \). Each of them contains 50000 tuples. We use three different sets \( \mathcal{A} \) of 10, 15, and 20 relevant attributes. For each of those sets, we pick a different set of superior examples \( G \). Sets \( G \) are constructed of similar tuples, similarity being measured as Euclidean distance. As before, given a set \( G \), we use \textit{elicit} to construct maximal p-skyline relations \( \succ \) favoring \( G \). This setup is supposed to model an automated process of identifying superior objects \( G \), in which a user is involved indirectly.

5.2.1 Scalability

In this section, we show that the \textit{elicit} algorithm is scalable with respect to various parameters. In Figure 18, we plot the dependence of the average running time of \textit{discover} on the number of superior examples \( |G| \) used to elic
a p-skyline relation (Figure 18(a), \(|S_i| = 50000, |\mathcal{A}| = 20\)), the size of \(S_i\) for \(i = 1, \ldots, 3\) (Figure 18(b), \(|G| = 50, |\mathcal{A}| = 20\)), and the number \(|\mathcal{A}|\) of relevant attributes (Figure 18(c), \(|S_i| = 50000, |G| = 50\)). The measured time does not include the time to construct the system of negative constraints and find the non-redundant constraints in it. According to our experiments, the preprocessing time predominantly depends on the performance of the skyline computation algorithm.

According to Figure 18(a), the running time of the algorithm increases until the size of \(G\) reaches 30, and does not vary much after that. This is due to the fact that the algorithm performance depends on the number of negative constraints used. We use only non-redundant constraints for elicitation. As we show further (Figure 19(a)), the dependence of the size of a system of non-redundant constraints on the number of superior examples has a pattern similar to Figure 18(a).

The growth of the running time with the increase in the data set size (Figure 18(b)) is due to the fact that the number of negative constraints depends on skyline size (Section 4.5). For the data sets used in the experiment, the skyline size grows with the size of the data set. The running time of the algorithm grows with the number of relevant attributes (Figure 18(c)) for the same reasons.

We conclude that the \texttt{elicit} algorithm is efficient and its running time scales well with respect to the number of superior examples, the size of the data set, and the number of relevant attributes used.

5.2.2 Reduction in the number of negative constraints

In this section, we demonstrate that the algorithm \texttt{elicit} allows effective optimizations. Recall that the running time of \texttt{elicit} depends linearly (Theorem 11) on the number of negative constraints in the system \(N\). Here we show that the techniques proposed in Section 4.5 result in a significant reduction in the size of \(N\).

In Figure 19(a), we show how the number of negative constraints depends on the number of superior examples used to construct them. For every data set, we plot two values: the number of unique negative constraints in \(N(G, \omega_{\text{sky}}(S_i))\) for \(i = 1, \ldots, 3\), and the number of unique non-redundant constraints in the corresponding system. We note that the reduction in the number of constraints achieved using the methods we proposed in Section 4.5 is significant. In particular, for the anti-correlated data set and \(G\) of size 150, the total number of constraints in \(N(G, S_i)\) is approximately \(7.5 \cdot 10^6\). Among them, about \(5.5 \cdot 10^6\) are unique in \(N(G, \omega_{\text{sky}}(S_i))\). However, less than 1% of them (about \(12 \cdot 10^3\)) are non-redundant.

5.2.3 Winnow result size

In Section 5.1, we showed how the size of p-skyline query result depends on the number of relevant attributes and the size of the skyline. In this section, we show that another parameter which affects the size of winnow query result is data distribution. In Figure 18(b), we demonstrate how the size of the p-skyline query result varies with the number of superior examples. We compare this size with the size of the corresponding skyline and plot the value of \texttt{winnow-size-ratio} defined in the previous section. Here we use anti-correlated, uniform, and correlated data sets of 50000 tuples each. The number of relevant attributes is 20. The size of the corresponding skylines is: 41716 (anti-correlated), 37019 (uniform), and 33888 (correlated). For anti-correlated and uniform data sets, the values of \texttt{winnow-size-ratio} quickly reach a certain bound and then grow slowly with the number of superior examples. This bound is approximately 1% of the skyline size (i.e., about 350 tuples) for both data sets. At the same time, the growth of \texttt{winnow-size-ratio} for correlated data set is faster. Note that the values of \texttt{winnow-size-ratio} are generally lower for synthetic data sets, in comparison to the real-life data set \textit{NHL}, due to the larger set of relevant attributes and larger skyline sizes in the current experiment.

We conclude that the experiments that we have carried out show that incorporating relative attribute importance into skyline relations in the form of p-skyline relations results in a significant reduction in query result size. The proposed algorithm \texttt{elicit} for eliciting a maximal p-skyline relation
favoring a given set of superior examples has good scalability in terms of the data set size and the number of relevant attributes. The algorithm has high accuracy even for small sets of superior examples.

6 Related work

In this section, we discuss related work that has been done in the areas covered in the paper: modeling preferences as skyline relations and preference elicitation.

6.1 Modeling preferences as skyline relations

The p-skyline framework is based on the preference constructor approach proposed in [Kießling(2002)]. That approach was extended in [Kießling (2005)] by relaxing definitions of the accumulation operators and by using SV-relations, instead of equality, as indifference relations. [Kießling (2005)] showed that such an extension preserves the SPO properties of the resulting preference relations. The resulting relations were shown to be larger (in the set theoretic sense) than the relations composed using the equality-based accumulation operators. However, relative importance of attributes implicit in such relations was addressed neither in [Kießling(2002)] nor [Kießling (2005)]. Containment of preference relations and minimal extensions were also not considered in these works.

[Börzsönyi et al(2001)] proposed the original skyline framework. That paper introduced an extension of SQL in which the skyline queries can be formulated. The paper also proposed a number of skyline computation algorithms. Since then, many algorithms for that task have been developed ([Tan et al(2001),Kossmann et al(2002),Chomicki et al(2003), Lee et al(2007),Godfrey et al(2007)] and others).

[Godfrey et al(2005)] showed that the number of skyline points in a dataset may be exponential in the number of attributes. Since then, a number of approaches have been developed for reducing the size of skylines by computing only the most representative skyline objects.

[Chan et al(2006)] proposed to compute the set of k-dominant skyline points instead of the entire skyline. Another variant of the skyline operator was presented in [Lin et al(2007)]. That operator computes k most representative tuples of a skyline. [Lin et al(2007)] showed that when the number of attributes involved is greater than two, the problem is NP-hard in general. For such cases, [Lin et al(2007)] proposed a polynomial time approximation algorithm.

More recently, [Tao et al(2009)] proposed the distance-based representative skyline operator. This approach is based on the observation that if a skyline of a dataset consists of clusters, then in many cases, a user is interested in seeing only good representatives from each skyline cluster rather than the entire skyline (which may be quite large). If interested, the user may drill down to each cluster further on. The representativeness here is measured as the maximum of the distance from the cluster center to each object of the cluster. The authors studied the problem of computing k most representative skyline objects and proposed an efficient approximation algorithm for datasets with arbitrary dimensionality.

Another recent work in the area of skyline-size reduction is [Zhao et al(2010)]. There, the authors proposed the order-based representative skyline operator. The approach is based on a well-known fact that an object is in a skyline iff it maximizes some monotone utility function. As a measure of skyline object similarity, the authors used the similarity between (possibly infinite) sets of orders which favor the corresponding objects. The authors developed an algorithm for computing representatives of clusters of similar objects. They also proposed a method of eliciting user preferences which allows to drill down to clusters in an iterative manner.

Another direction of research using the skyline framework concerns subspace skyline computation [Pei et al(2005), Yuan et al(2005)]. An interesting problem in this framework is how to identify the subspaces to whose skylines a given tuple belongs. [Pei et al(2005)] showed an approach to that problem, which uses the notion of decisive subspace. A subspace skyline can be computed using every skyline algorithm. However, to compute k subspace skylines (for k different attribute sets), an algorithm for efficient computing of all subspace skylines at once [Pei et al(2005), Yuan et al(2005)] may be more efficient. [Yuan et al(2005)] introduced the related notion of skyline cube. The skyline cube approach was used in [Lee et al(2009)] to find the most interesting subspaces given an upper bound on the size of the corresponding skyline and a total order of attributes, the latter representing the importance of the attributes to the user.

We notice that the framework based on subspace skylines is, in a sense, orthogonal to the p-skyline framework proposed here. Both of them extend the skyline framework. In the subspace skyline framework, the relative importance of attributes is fixed (i.e., all considered attributes are of equal importance) while the sets of the relevant attributes

\[
\text{Fig. 19 Synthetic data experiments}
\]
may vary. In the p-skyline approach, the set of relevant attributes is fixed while the relative importance of them may vary. However, given a set of attribute preference relations, all subspace skylines and the results of all full p-skyline relations are subsets of the (full-space) skyline (assuming the distinct value property for subspace skylines).

[Zhang et al(2010)] studied the properties of skyline preference relations and showed that they are the only relations satisfying the introduced properties of rationality, transitivity, scaling robustness, and shifted robustness. The authors analyzed these properties and the outcome of their relaxation in skyline preference relations. They also showed how to adapt existing skyline computation algorithms to relaxed skylines. This work is particular interesting in the context of the current paper, since it gives some insights to possible approaches for computing p-skyline winnow queries.

6.2 Preference elicitation

An approach to elicit preferences aggregated using the accumulation operators was proposed in [Holland et al(2003)]. Web server logs were used there to elicit preference relations. The approach was based on statistical properties of log data – more preferable tuples appear more frequently. The mining process was split into two parts: eliciting attribute preferences and eliciting accumulation operators which aggregate the attribute preferences. Attribute preferences to be elicited were in the form of predefined preference constructors such as LOWEST, HIGHEST, POS, NEG etc. [Holland et al(2003)] used a heuristic approach to elicit the way attribute preferences are aggregated (using Pareto and prioritized accumulation operators). The case when more than one different combination of accumulation operators may be elicited in the same data was not addressed. Moreover, no criteria of optimality of elicited preference relations were defined.

A framework for preference elicitation which is complementary to the approach we have developed here was presented in [Jiang et al(2008)]. In that work, preferences are modeled as skyline relations. Given a set of relevant attributes and a set of attribute preferences over some of them, the objective is to determine attribute preferences over the remaining attributes. The elicitation process is based on user feedback in terms of a set of superior and a set of inferior examples. The work is focused on eliciting minimal (in terms of relation size) attribute preference relations. [Jiang et al(2008)] showed that the problem of existence of such relations is NP-complete, and the computation problem is NP-hard. Two greedy heuristic algorithms were provided. The algorithms are not sound, i.e., for some inputs, the computed preferences may fail to be minimal. That approach and the approach we presented here are different in the following sense. First, [Jiang et al(2008)] dealt with skyline relations, and thus all attribute preferences are considered to be equally important. In contrast, the focus of our work is to elicit differences in attribute importance. Second, [Jiang et al(2008)] focused on eliciting minimal attribute preferences. In contrast, we are interested in constructing maximal tuple preference relations, since such relations guarantee a better fit to the provided set of superior examples. At the same time, our work and [Jiang et al(2008)] complement each other. Namely, when attribute preferences are not provided explicitly by the user, the approach of [Jiang et al(2008)] may be used to elicit them.

Another approach to preference relation elicitation in the skyline framework was introduced in [Lee et al(2008)]. It proposed to reduce skyline sizes by revising skyline preference relations by supplying additional tuple relationships: preference and equivalence. Such relationships are obtained from user answers to simple questions.

In quantitative preference frameworks [Fishburn(1970)], preferences are represented as utility functions: a tuple $t$ is preferred to another tuple $t'$ iff $f(t) > f(t')$ for a utility function $f$. Attribute priorities are often represented here as weight coefficients in polynomial utility functions. A number of methods have been proposed to elicit utility functions – some of them are [Chajewska et al(2000),Boutilier(2002)]. Utility functions were shown to be effective for reasoning with preferences and querying databases with preferences (Top-K queries) [Fagin et al(2001),Das et al(2006),Bacchus and Grove(1996)]. Some work has been performed on eliciting utility functions for preferences represented in other models [McGeachie and Doyle(2002)].

[Domshlak and Joachims(2007)] described another model of preference elicitation in the form of utility functions. The authors proposed a framework for constructing a utility function consistent with a set of comparative statements about preferences (e.g., “A is better than B” or “A is as good as B”). That approach does not rely on any structure of preference relations. [Vu Ha(1999)] proposed an approach to composing binary preference relations and multi-linear utility functions. A quantitative framework for eliciting binary preference relations based on knowledge based artificial neural network (KBANN) was presented in [Haddawy et al(2003)]. [Viappiani et al(2006)] studied the problems of incremental elicitation of user preference based on user provided example critiques.

7 Conclusion and future work

In this work, we explored the p-skyline framework which extends skylines with the notion of attribute importance captured by p-graphs. We studied the properties of p-skyline relations – checking dominance, containment and equality of such relations – and showed efficient methods for performing the checks using p-graphs. We proposed a complete set of transformation rules for efficient computation of minimal extensions of p-skyline relations.
The main problem studied here was the elicitation of \( p \)-skyline relations based on user-provided feedback in the form of superior and inferior examples. We showed that the problems of existence and construction of a maximal \( p \)-skyline relation favoring and disfavoring given sets of superior and inferior examples are intractable in general. For restricted versions of these problems – when the provided inferior example sets are empty – we designed polynomial time algorithms. We also identified some bottlenecks of constructing maximal \( p \)-skyline relations: the system of negative constraints used may be quite large in general, which directly affects the algorithm performance. To tackle that problem, we proposed several optimization techniques for reducing the size of such systems. We also showed that the problem of minimization of such systems is unlikely to be soluble in polynomial time in general. We conducted experimental studies of the proposed elicitation algorithm and optimization techniques. The study shows that the algorithm has good scalability in terms of the data set size and the number of relevant attributes, and high accuracy even for small sets of superior examples.

At the same time, we note that our framework has a number of limitations that can be addressed in future work. First, we focused on full \( p \)-skyline relations. An interesting direction of future work would be to study the properties of partial \( p \)-skyline relations (i.e., defined on top of sets \( A \) and \( H \) of variable size).

Second, attribute preference relations considered in this work are limited to total orders. There are several reasons for this limitation:

- the limitation is natural in many contexts;

- attribute preferences in skyline relations are also typically total orders (although there are several papers, e.g., [Chan et al(2005), Balke et al(2006)], in which this limitation is lifted);

- some of our results require the assumption that attribute preferences are total orders, e.g., Theorem 5.

It would be interesting to see how our results can be generalized if the restriction of attribute preferences to total orders is relaxed. (To avoid any possible confusion, we emphasize that tuple preference relation considered in our work are not limited to total orders.)

Third, the DIFF attributes, discussed in the original skyline paper [Börzsönyi et al(2001)], were also not considered in this paper. This is another possible generalization.

Fourth, we developed elicitation algorithms for a particular scenario in which we know which tuples the user likes but do now know which ones he dislikes. Clearly, another restriction of the problem is possible – it is known which tuples the user dislikes (the set of inferior tuples is non-empty), but unknown which ones he confidently likes (the set of superior tuples is empty). The latter scenario has not been considered in this paper.

Fifth, the type of user feedback for \( p \)-skyline relation elicitation – superior and inferior examples – may not fit some real-life scenarios. So a potentially promising direction is to adapt the \( p \)-skyline elicitation approach to other types of feedback. For that, one should study appropriate classes of attribute set constraints.

Finally, the problem of computing winnow queries with \( p \)-skyline relations is left for future work.

During the preparation of the final version of this paper, we learned [Ciaccia(2011)] about some relevant results obtained in [Valdes et al(1982)]. The notion of \( p \)-graph (Definition 10) corresponds to the notion of series-parallel graph and the Envelope property, to the notion of \( N \)-free graphs. [Valdes et al(1982)] established the connection between series-parallel and \( N \)-free graphs, captured by our Theorem 2. Also, the latter paper proposes the notion of canonical decomposition tree, which corresponds to our notion of normalized syntax tree (Definition 9).

References

[nhl(2008)] (2008) NHL.com Player Stats. http://www.nhl.com/lce/playerstats.htm

[Bacchus and Grove(1996)] Bacchus F, Grove A (1996) Utility independence in a qualitative decision theory. In: Proceedings of 5th International Conference on Principles of Knowledge Representation and Reasoning (KR), Morgan Kaufmann, pp 542–552

[Balke et al(2006)] Balke WT, Gntzer U, Siberski W (2006) Exploiting indifference for customization of partial order skylines. In: Proceedings of the 10th International Database Engineering and Applications Symposium (IDEAS), Delhi, India, pp 80–88

[Balke et al(2007)] Balke WT, Gunter U, Loﬁ C (2007) Incremental Trade-Off Management for Preference-Based Queries. International Journal of Computer Science & Applications (IJCSA), 4(2):75–91

[Börzsönyi et al(2001)] Börzsönyi S, Kossmann D, Stocker K (2001) The skyline operator. In: Proceedings of the 17th International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, pp 421–430

[Boutilier(2002)] Boutilier C (2002) A POMDP formulation of preference elicitation problems. In: Proceedings of the 18th national conference on Artificial intelligence, AAAI Press, Menlo Park, CA, USA, pp 239–246

[Boutilier et al(2004)] Boutilier C, Brafman R, Domshlak C, Hoos H, Poole D (2004) CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. Journal of Artiﬁcial Intelligence Research, 21:135–191

[Brafman and Domshlak(2002)] Brafman RI, Domshlak C (2002) Introducing variable importance tradeoffs into CP-nets. In: Proceedings of the 18th Conference in Uncertainty in Artiﬁcial Intelligence, Morgan Kaufmann, Edmonton, Alberta, Canada, pp 69–76

[Chajewska et al(2000)] Chajewska U, Koller D, Parr R (2000) Making rational decisions using adaptive utility elicitation. In: Proceedings of the 17th National Conference on Artiﬁcial Intelligence, AAAI Press, Austin, TX, USA, pp 363–369

[Chan et al(2005)] Chan CY, Eng PK, Tan KL (2005) Stratified computation of skylines with partially-ordered domains. In: Proceedings of the ACM SIGMOD Conference, ACM, Baltimore, Maryland, USA, pp 203–214

[Chan et al(2006)] Chan CY, Jagadish HV, Tan KL, Tung AKH, Zhang Z (2006) Finding k-dominant skylines in high dimensional...
space. In: Proceedings of the ACM SIGMOD Conference, ACM, Chicago, Illinois, USA, pp 503–514

[Ciai11] Ciaccia P (2011) Personal Communication

[Chomicki03] Chomicki J (2003) Preference formulas in relational queries. ACM Transactions on Database Systems (TODS), 28(4):427–466

[Godfrey06] Godfrey P, Gryz J, Liang D (2006) Skyline query with presorting. In: Proceedings of the 19th International Conference on Data Engineering (ICDE), IEEE Computer Society, Bangalore, India, pp 717–816

[Haddawy03] Haddawy P, Restificar A, Geisler B, Miyamoto M (2003) Skyline cardinality for relational processing. In: Foundations of Information and Knowledge Systems (FoIKS), Springer, Lecture Notes in Computer Science, vol 2942, pp 78–97

[Kießling05] Kießling W (2005) Foundations of preferences in database systems. In: Foundations of Information and Knowledge Systems (FoIKS), Springer, Lecture Notes in Computer Science, vol 2942, pp 78–97

[Zhao10] Zhao F, Das G, Tan KL, Tung AKH (2010) Call to Startings: The k Most Representative Skyline Operator. In: Proceedings of the 32nd International Conference on Very Large Data Bases, VLDB Endowment, pp 451–462

[Domshlak07] Domshlak C, Joachims T (2007) Efficient and non-parametric reasoning over user preferences. User Modeling and User-Adapted Interaction 17(1-2):41–69

[McGeachie02] McGeachie and Doyle J (2002) Efficient utility functions for ceteris paribus preferences. In: Proceedings of the 18th national conference on Artificial intelligence, AAAI Press, Menlo Park, CA, USA, pp 279–284

[Morse07] Morse MD, Patel JM, Jagadish HV (2007) Efficient skyline computation over low-cardinality domains. In: Proceedings of the 33rd International Conference on Very Large Data Bases, ACM, Vienna, Austria, pp 279–290

[Rao09] Rao H, Das G, Tan KL, Tung AKH (2009) Call to Startings: Best Views of Skyline. In: Proceedings of the 32nd International Conference on Very Large Data Bases, VLDB, Morgan Kaufmann, Hong Kong, China, pp 311–322

[Chomicki05] Chomicki J, Godfrey P, Gryz J, Liang D (2005) Skyline with presorting. In: Proceedings of the 19th International Conference on Data Engineering (ICDE), IEEE Computer Society, Tokyo, Japan, pp 502–513

[Godfrey04] Godfrey P (2004) Skyline cardinality for relational processing. In: Foundations of Information and Knowledge Systems (FoIKS), Springer, Lecture Notes in Computer Science, vol 2942, pp 78–97

[Chomicki03] Chomicki J (2003) Preference formulas in relational databases. In: Proceedings of the 32nd International Conference on Very Large Data Bases, VLDB Endowment, pp 451–462

[Pei05] Pei J, Jin W, Ester M, Tao Y (2005) Cutting the Best Views of Skyline: A Semantic Approach Based on Decisive Subspaces. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB), ACM, Trondheim, Norway, pp 253–264

[Tan01] Tan KL, Eng PK, Ooi BC (2001) Efficient Progressive Skyline Computation. In: Proceedings of 27th International Conference on Very Large Data Bases (VLDB), Morgan Kaufmann, Roma, Italy, pp 301–310

[Tao09] Tao Y, Ding L, Lin X, Pei J (2009) Distance-based representative skyline. In: Proceedings of the 25th International Conference on Data Engineering (ICDE), Shanghai, China, pp 892–903

[Valdes82] Valdes J, Tarjan R.E., Lawler E.L. (1982) The Recognition of Series Parallel Digraphs. SIAM Journal on Computing 11:298–313

[Viappiani06] Viappiani P, Faltins B, Pu P (2006) Preference-based search using example-critiquing with suggestions. Journal of Artificial Intelligence Research 27:465–503

[VuHa99] Vu Ha PH (1999) A hybrid approach to reasoning with partial preference models. In: Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, AUAI Press, Helsinki, Finland, pp 263–270

[Yuan05] Yuan Y, Lin X, Liu Q, Wang W, Yu JX, Zhang Q (2005) Efficient computation of the skyline cube. In: Proceedings of the 31st Conference on Very Large Data Bases (VLDB), ACM, Trondheim, Norway, pp 241–253

[Zhang10] Zhang Z, Lu H, Ooi BC, Tung AK (2010) Understanding the meaning of a shifted sky: a general framework for extending skyline query. The VLDB Journal 19(2):181–201

[Zhao10] Zhao F, Das G, Tan KL, Tung AKH (2010) Call to order: a hierarchical browsing approach to eliciting users’ preference. In: Proceedings of the ACM SIGMOD Conference, ACM, Indianapolis, Indiana, USA, pp 27–38
Appendix: Proofs

Before going into the proofs, we introduce \((\mathcal{W}^0, \mathcal{A}^0)\)-structures. A \((\mathcal{W}^0, \mathcal{A}^0)\)-structure is based on the set of attributes \(\mathcal{A}^0\) and a function \(\mathcal{W}^0 = \{W_A : A \in \mathcal{A}^0\}\) mapping \(\mathcal{W}^0\) to subsets of \(\mathcal{A}^0\).

**Definition 14** ((\(\mathcal{W}^0, \mathcal{A}^0)\)-structure) Let \(\mathcal{W}^0\) and \(\mathcal{A}^0\) be as discussed above and such that for every \(A \in \mathcal{A}^0\), \(A \notin W_A\). Then the \((\mathcal{W}^0, \mathcal{A}^0)\)-structure is a tuple \((\mathcal{W}^0, \mathcal{A}^0)\), and the relation generated by \((\mathcal{W}^0, \mathcal{A}^0)\) is
\[
\succ \equiv (\bigcup_{A \in \mathcal{A}^0} q_A),
\]
where
\[
q_A \equiv \{(o_1, o_2) \mid o_1.A >_A o_2.A \} \cap \approx_{\mathcal{A} - (W_A \cup \{A\})},
\]
and \(>_A\) is the attribute preference relation for \(A\) in \(\mathcal{H}\).

Let a tuple \(o\) dominate a tuple \(o'\) according to the relation \(\succ\) generated by \((\mathcal{W}^0, \mathcal{A}^0)\). By Definition 14, this is possible if there exist a sequence of tuples \(\Sigma_{o,o'} = (o_1, o_2, \ldots, o_m, o_{m+1})\) such that \(o_i \equiv o, o_{m+1} = o'\), and a sequence of attributes \(\Psi_{o,o'} = (A_{i_1}, \ldots, A_{i_m})\), all in \(\mathcal{A}^0\), such that
\[
q_{A_{i_1}}(o_1, o_2), \ldots, q_{A_m}(o_m, o_{m+1})
\]
Then the pair \((\Sigma_{o,o'}, \Psi_{o,o'})\) is called a derivation sequence for \(o \succ\) \(o'\). Given a pair of tuples, the corresponding derivation sequence is not unique in general.

We notice that the \((\mathcal{W}^0, \mathcal{A}^0)\)-structures are an efficient tool used here to prove some theorems describing properties of p-skyline relations. Now, Theorem 1 can be reformulated as follows:

**Theorem 1'** Every p-skyline relation \(\succ \in \mathcal{F}_\mathcal{H}\) can be represented as a relation \(\succ\) generated by a \((\mathcal{W}', \mathcal{A}')\)-structure such that for every \(A \in \mathcal{A}, W_A = \text{Chr}_\succ(A)\).

**Proof of Theorem 1'**. We show here that for every \(\succ \in \mathcal{F}_\mathcal{H}\),
\[
\succ \equiv \succ(\mathcal{W}', \mathcal{A}') \equiv TC(\bigcup_{A \in \text{Var}(\succ)} q_A)
\]
where \(W_A = \text{Chr}_\succ(A)\) for \(A \in \text{Var}(\succ)\). We prove the theorem by induction on the sizes of \(\mathcal{H}\) and \(\mathcal{A}\).

**Base step.** Let \(\mathcal{H} = \{A\}\) and \(\mathcal{A} = \{A\}\). Then \(\mathcal{F}_\mathcal{H}\) consists of a single atomic p-skyline relation \(\succ\) induced by \(\succ\). Let \(W_A = \text{Chr}_\succ(A)\) = \(\emptyset\). Then
\[
\succ = \succ(\mathcal{W}', \mathcal{A}') \equiv TC(q_A)
\]
where
\[
q_A \equiv \{(o_1, o_2) \mid o_1.A >_A o_2.A \} \cap \approx_{\mathcal{A} - (W_A \cup \{A\})}.
\]
**Inductive step.** Now assume that the theorem holds for \(\mathcal{H}\) and \(\mathcal{A}\) of size up to \(n\). Prove that it holds for \(\mathcal{H}\) and \(\mathcal{A}\) of size \(n + 1\). Let \(\succ = \succ(\mathcal{W}_1, \mathcal{A}1) \cap \succ(\mathcal{W}_2, \mathcal{A}_2)\) (the case of \(\succ(\mathcal{W}_1, \mathcal{A}1) \cap \succ(\mathcal{W}_2, \mathcal{A}_2)\) is similar). By the definition of induced p-skyline relations,
\[
\succ \equiv \succ(\mathcal{W}_1, \mathcal{A}_1) \equiv TC(\bigcup_{A \in \text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1))} q_A) \quad (5)
\]
\[
\succ \equiv \succ(\mathcal{W}_2, \mathcal{A}_2) \equiv TC(\bigcup_{A \in \text{Var}(\succ(\mathcal{W}_2, \mathcal{A}_2))} q_A) \quad (6)
\]
where
\[
q_A \equiv \{(o_1, o_2) \mid o_1.A >_A o_2.A \} \cap \approx_{\mathcal{A} - (W_A \cup \{A\})} \quad (7)
\]
\[
q_A \equiv \{(o_1, o_2) \mid o_1.A >_A o_2.A \} \cap \approx_{\mathcal{A} - (W_A \cup \{A\})} \quad (8)
\]
Since \(\succ\) is a p-skyline relation, \(\text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1)) \cap \text{Var}(\succ(\mathcal{W}_2, \mathcal{A}_2)) = \emptyset\). \quad (9)
(9), (5), and (6) imply
\[
\succ \equiv TC\left(\bigcup_{A \in \text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1))} q_A\right) \cap \approx_{\mathcal{A} - (W_A \cup \{A\})} \quad (10)
\]
or equivalently
\[
\succ \equiv TC\left(\bigcup_{A \in \text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1))} q_A\right) \cap \approx_{\mathcal{A} - (W_A \cup \{A\})} \quad (11)
\]
Construct the function \(W\) as follows
\[
W_A = \begin{cases} W_{A1}, & \text{if } A \in \text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1)) \\ W_{A2}, & \text{if } A \in \text{Var}(\succ(\mathcal{W}_2, \mathcal{A}_2)). \end{cases}
\]
Let \(\mathcal{A}_1 = \text{Var}(\succ(\mathcal{W}_1, \mathcal{A}_1)) \cup \text{Var}(\succ(\mathcal{W}_2, \mathcal{A}_2))\) and \(\succ(\mathcal{W}, \mathcal{A})\) be generated by such \((\mathcal{W}', \mathcal{A}')\)
\[
\succ(\mathcal{W}', \mathcal{A}') \equiv TC\left(\bigcup_{A \in \mathcal{A}} q_A\right) \quad (12)
\]
We prove that \( \succ_{(\psi',\mathcal{A})} \) is equal to \( \succ \). Before going into the proof, notice that (11) can be rewritten as

\[
\succ \equiv \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_1)} q^1_A \right) \cup \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_2)} q^2_A \right) \cup \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_3)} q^3_A \right) \cap \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_4)} q^4_A \right) \cap \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_5)} q^5_A \right).
\]

(14)

1. Let \( o \succ_{(\psi',\mathcal{A})} o' \). Let \( (\Sigma_{o,o'}, \Psi_{o,o'}) \) be some derivation sequence for \( o \succ_{(\psi',\mathcal{A})} o' \). W.l.o.g. let \( \Psi_{o,o'} = (A_1, \ldots, A_m) \), \( \Sigma_{o,o'} = (o = o_1, o_2, \ldots, o_m, o_{m+1} = o') \), and

\[ q^1_A(o_1, o_2), q^2_A(o_2, o_3), \ldots, q^m_A(o_m, o_{m+1}). \]

(15)

By construction, each attribute \( A_i \in \Psi_{o,o'} \) is either in \( \text{Var}(\succ_1) \) or \( \text{Var}(\succ_2) \). For every such \( A_i \), \( q^i_A \) implies \( o_i \succ o_{i+1} \) by (14). Therefore, (15) implies

\[ o_1 \succ o_2, o_2 \succ o_3, \ldots, o_m \succ o_{m+1}. \]

(16) Transitivity of p-skyline relations implies \( o_1 \succ o_{m+1} \), i.e., \( o \succ o' \).

2. Let \( o \succ o' \). Then (14) leads to three cases

(a) \( (o,o') \in \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_1)} q^1_A \right) \). Then \( o \succ_{(\psi',\mathcal{A})} o' \) by (12).

(b) \( (o,o') \in \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_2)} q^2_A \right) \). Then \( o \succ_{(\psi',\mathcal{A})} o' \) by the same reasoning.

(c) \( (o,o') \in \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_3)} q^3_A \right) \cap \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_4)} q^4_A \right) \).

In this case, (9) implies that there is an object \( o'' \) whose values of \( \text{Var}(\succ_2) \) are equal to those of \( o \), and the values of \( \text{Var}(\succ_1) \) are equal to those of \( o' \). Then we have

\[ (o,o'') \in \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_1)} q^1_A \right) \cap \approx_{\text{Var}(\succ_1)} \]

or equivalently

\[ (o'',o') \in \text{TC} \left( \bigcup_{A \in \text{Var}(\succ_2)} q^2_A \right) \cap \approx_{\text{Var}(\succ_1)} \]

which implies by (13) and (12)

\[ o \succ_{(\psi',\mathcal{A})} o'', o'' \succ_{(\psi',\mathcal{A})} o'. \]

The transitivity of \( \succ_{(\psi',\mathcal{A})} \) implies \( o \succ_{(\psi',\mathcal{A})} o' \).
Let a subgraph $\Gamma_1$ of $\Gamma$ have the following set of nodes

$$N(\Gamma_1) = \{A\} \cup PAT(\{A\} \cup CH_T(\{A\} \cup CH_T(\{A\} \cup PA_{\Gamma}(A)))$$

and the subgraph $\Gamma_2$ of $\Gamma$ have the nodes $N(\Gamma_2) = N(\Gamma) - N(\Gamma_1)$. Assuming that $B \in N(\Gamma_1)$ leads to contradiction by case analysis. So $B \in N(\Gamma_2)$. We conclude that both $\Gamma_1$ and $\Gamma_2$ are nonempty. Also, by case analysis we show that $\Gamma \equiv N(\Gamma_1) \sim N(\Gamma_2)$. □

**Lemma 2** A directed graph $\Gamma$ satisfying SPO+Envelope with at least two nodes has a $\rightarrow$-partition or a $\sim$-partition $\langle \Gamma_{\rightarrow}, \Gamma_{\sim}\rangle$ such that $\Gamma_{\rightarrow}$ and $\Gamma_{\sim}$ satisfy SPO+Envelope.

**Proof.** We assume that no $\sim$-partition of $\Gamma$ exists and show that there exists a $\rightarrow$-partition. Since $\Gamma$ is a finite SPO, there exists a nonempty set $Top \subseteq N(\Gamma)$ of all the nodes which have no incoming edges. If $Top$ is a singleton, then $Top$ dominates every node in $N(\Gamma) - Top$, and we get the $\rightarrow$-partition $(Top, N(\Gamma) - Top)$. Assume $Top$ is not singleton. Pick two nodes $T_1, T_2 \in Top$, $T_1$ and $T_2$ have no incoming edges, and Lemma 1 implies that there exists a node $Z_1$ such that $(T_1, Z_1) \in \Gamma \wedge (T_2, Z_1) \in \Gamma$. If $|Top| > 2$, pick some node $T_k (T_k \neq T_1, T_k \neq T_2)$ from $Top$. Since $T_k$ has no incoming edges either, Lemma 1 implies that either $T_k$ is a parent of $Z_1$ or they have a common child (which is also a child of $T_1$ and $T_2$ by the transitivity of $\Gamma$). Therefore, by picking every node of $Top$, we can show that there exists at least one node $Z$ which is a child of all nodes in $Top$. Denote as $M$ the set of all the nodes dominated by every node in $Top$. Above we showed that $M$ contains at least one node.

Now let us show that if a node $X$ is not in $M$ then $(X, M) \in \Gamma$. Clearly, if $X \in Top$, then $(X, M) \in \Gamma$. So let $X \notin Top$. By definition of $Top$, there is a node $T \in Top$ such that $(T_1, X) \in \Gamma$. Assume there is a node $Z \in M$ such that $(X, Z) \notin \Gamma$. By definition of $M$, $(T_1, Z) \in \Gamma$. Now pick some node $T (T \neq T_1)$ of $Top$. By definition of $M$, $(T, Z) \in \Gamma$. Let us apply Envelope:

$$(T, Z) \in \Gamma \wedge (T_1, Z) \in \Gamma \wedge (T_1, X) \in \Gamma \Rightarrow (T, T_1) \in \Gamma \vee (T, Z) \in \Gamma \vee (X, Z) \in \Gamma.$$ 

The first and the last disjuncts in the right-hand-side of the expression contradict the assumptions $(X, Z) \notin \Gamma$ and $T \in Top$. Therefore, the only choice is $(T, X) \in \Gamma$. However, $T$ is an arbitrary node in $Top$. Therefore, $(T, X) \in \Gamma$ and thus $X \in M$ by definition of $M$. We conclude that $\langle N(\Gamma) - M, M \rangle$ is a $\rightarrow$-partition of $\Gamma$.

Finally, it is easy to check that every subgraph of an SPO+Envelope graph satisfies SPO+Envelope. □

**Proof of Theorem 2.** By induction on the structure of the p-expression inducing a given p-skyline relation, it is easy to show that SPO+Envelope is satisfied by p-graphs. Now we show that every directed graph satisfying SPO+Envelope is a p-graph of some p-skyline relation. Given such a graph, we construct the corresponding p-skyline relation recursively. If $\Gamma$ contains a single node, then the corresponding p-skyline relation is the atomic preference relation induced by the attribute preference relation of the corresponding attribute. If $\Gamma$ has more than one node, then by Lemma 2, $\Gamma$ has either a $\rightarrow$-partition or a $\sim$-partition $(\Gamma_1, \Gamma_2)$ into nonempty subgraphs satisfying SPO+Envelope. If $(\Gamma_1, \Gamma_2)$ is a $\rightarrow$-partition ($\sim$-partition), then the corresponding p-skyline relation is a prioritized (Pareto, respectively) accumulation of the p-skyline relations corresponding to $\Gamma_1$ and $\Gamma_2$. This recursive construction exactly corresponds to the construction of $W$ shown in Theorem 1. □

**Proposition 4.** Let $A$ and $B$ be leaf nodes in a normalized syntax tree $T_r$ of a p-skyline relation $\succ \in F_{\mathcal{D}}$. Then $(A, B) \in \Gamma_\succ$ iff the least common ancestor $C$ of $A$ and $B$ in $T_r$ is labeled by $\&$, and $A$ precedes $B$ in the left-to-right tree traversal.

**Proof of Proposition 4.**

Let $\succ_C$ be a p-skyline relation represented by the syntax tree with the root node $C$. Definition 10 implies $(A, B) \in \Gamma_{\succ_C}$ and $E(\Gamma_{\succ_C}) \subseteq E(\Gamma_{\succ})$.

Let $(A, B) \in \Gamma_{\succ}$. If $C$ is of type $\&$, then $(A, B)$ is labeled by $\&$, and $A$ precedes $B$ in the left-to-right tree traversal, therefore Definition 10 implies $(A, B) \in \Gamma_{\succ_C}$.

**Theorem 3.** Two p-skyline relations $\succ_1, \succ_2 \in F_{\mathcal{D}}$ are equal iff their p-graphs are identical.

To prove the theorem, we use the next lemma.

**Lemma 3.** Assume that $\succ_1$ (resp. $\succ_2$) are p-skyline relations in $F_{\mathcal{D}}$, generated by $(W^1, \mathcal{A})$ and $(W^2, \mathcal{A})$, respectively. If for some $A \in \mathcal{A}$, $W^1_A - W^2_A \neq \emptyset$, then there is a pair $o, o' \in \mathcal{U}$ such that

$$o \succ_1 o' \text{ and } o \not\succ_2 o'.$$

**Proof.** We construct two tuples $o$ and $o'$ such that $o \succ (W^1, \mathcal{A}) o'$ (and thus $o \succ_1 o'$), and $o \not\succ (W^2, \mathcal{A}) o'$ (and thus $o \not\succ_2 o'$).

For every attribute $A_i \in \mathcal{A}$, pick two values $v_{A_i}, v'_{A_i} \in \mathcal{D}_{A_i}$ such that $v_{A_i} \succ_A v'_{A_i}$. Construct the tuples $o$ and $o'$ as follows:

$$o.A_i = \begin{cases} v_{A_i} & \text{if } A_i = A, \\ v'_{A_i} & \text{otherwise} \end{cases} \text{and} \quad o'.A_i = \begin{cases} v'_{A_i} & \text{if } A_i = A, \\ v_{A_i} & \text{otherwise} \end{cases}.$$
By construction, it is clear that
\[(o, o') \in \{(o_1, o_2) \mid o_1 \succ A o_2\} \cap \approx_{R - \{(A), W_A^1\}}\]
and thus \(o \succ (W^1_A) o'\) and \(o \succ A o'\). Now assume \(o \succ (W^2_A) o'\)
(and thus \(o \succ A o'\)), i.e.
\[(o, o') \in TC \left( \bigcup_{A_i \in \mathcal{A}} q_{A_i} \right)\]
(17)

where
\[q_{A_i} \equiv \{(o_1, o_2) \mid o_1 \succ A o_2\} \cap \approx_{R - \{(A), W_A^2\}}\]
(18)

(17) implies that there should exist a derivation sequence \((\Sigma_{o,o'}, \Psi_{o,o'})\) for \(o \succ (W^2_A) o'\). That is, \(\Sigma_{o,o'} = (o_1 = o, o_2, \ldots, o_m, o_{m+1} = o')\) is a sequence of tuples, and \(\Psi_{o,o'} = (A_1, \ldots, A_{m+1})\) is a sequence of attributes such that
\[q_{A_{m+1}}(0_1, 0_2), \ldots, q_{A_m}(o_m, o_{m+1}).\]
(19)

Note that by (18), \(o_{ik}\) may be worse than any \(o_{ik+1}\) in the values of \(W_{A_{ik}}\).

First, we prove that \(\Psi_{o,o'} \subseteq W_{A^2}^2 \cup \{A\}\). For the sake of contradiction, assume \(M = \Psi_{o,o'} - (W_{A^2}^2 \cup \{A\})\) is nonempty. Pick an element \(A_{top} \in M\) which has no ancestors in \(M\) in \(\Gamma_{\succ 2}\) (such an element exists due to acyclicity of \(\Gamma_{\succ 2}\)). Since \(q_{A_{top}}\) is in the chain (19), we get
\[o \succ A_{top} \succ A_{top}o',\]

By construction of \(o, o'\) that implies \(A_{top} = A\), which is a contradiction. Thus, \(\Psi_{o,o'} \subseteq W_{A^2}^2 \cup \{A\}\).

Second, we prove \(o \not\in (W^2_A) o'\). For that, pick \(B \in W_{A^2}^1 - W_{A^2}^2\). By construction of \(o, o', o', B \succ o.B\). That implies that there is a pair of tuples \(o_k, o_{k+1}\) in \(\Sigma_{o,o'}\) in which the value of \(B\) is changed from a less preferred to a more preferred one. That is possible only if \(B \in W_{A^2}^2\) for some attribute \(C \in \Psi_{o,o'} \subseteq W_{A^2}^2 \cup \{A\}\). By Transitivity of \(\Gamma_{\succ 2}\) (Theorem 2), \(B \in \text{BetIn}_2(A, C) \cup \{A\}\). By transitivity of \(\Gamma_{\succ 2}\) (Theorem 2), \(B \in \text{BetIn}_2(A, C)\) i.e., \(B \in W_{A^2}^2\), which contradicts the definition of \(B\). Hence, \(o \not\in (W^2_A) o'\).

Now we go back to the proof of Theorem 3. Proof of Theorem 3.

\(\Rightarrow\) Every two \(p\)-skyline relations which have the same \(p\)-graph are represented by the same structure \((W, \mathcal{A})\), by the definition of \(p\)-graph. Therefore, the \(p\)-skyline relations are equal.

\(\Rightarrow\) Pick two equal \(p\)-skyline relations \(\succ 1\) and \(\succ 2\). Let the structures \((W^1, \mathcal{A}), (W^2, \mathcal{A})\) and the \(p\)-graphs \(\Gamma_{\succ 1}, \Gamma_{\succ 2}\) represent \(\succ 1\) and \(\succ 2\), respectively. Clearly, the node sets of \(\Gamma_{\succ 1}\) and \(\Gamma_{\succ 2}\) are equal to \(\mathcal{A}\). If their edge sets are different, then the functions \(W^1\) and \(W^2\) are different. Pick \(A \in \mathcal{A}\) such that \(W^1_A \neq W^2_A\). Without loss of generality, we can assume \(W^1_A - W^2_A \neq \emptyset\). Lemma 3 implies that \(\succ 1\) and \(\succ 2\) are not equal, which is a contradiction.

\(\Rightarrow\) Let the structures \((W^1, \mathcal{A}), (W^2, \mathcal{A})\) generate relations \(\succ (W^1_A)\) and \(\succ (W^2_A)\) equal to \(\succ 1\) and \(\succ 2\), correspondingly. \(E(\Gamma_{\succ 1}) \subseteq E(\Gamma_{\succ 2})\) implies that for all \(A \in \mathcal{A}\), \(W^1_A \subseteq W^2_A\). Hence, \(\succ (W^1_A) \subseteq \succ (W^2_A)\) and \(\succ 1 \subseteq \succ 2\). Therefore, \(E(\Gamma_{\succ 1}) \subseteq E(\Gamma_{\succ 2})\), and for some \(A\) we have \(W^2_A - W^1_A \neq \emptyset\). Lemma 3 implies \(\succ 1 \not\subseteq \succ 2\), which is a contradiction.

Theorem 4. For \(p\)-skyline relations \(\succ 1, \succ 2 \in \mathcal{F}_2\), \(\succ 1 \subset \succ 2\) if and only if \(E(\Gamma_{\succ 1}) \subseteq E(\Gamma_{\succ 2})\).

Proof. Let the structures \((W^1, \mathcal{A}), (W^2, \mathcal{A})\) generate relations \(\succ (W^1_A)\) and \(\succ (W^2_A)\) equal to \(\succ 1\) and \(\succ 2\), correspondingly. \(E(\Gamma_{\succ 1}) \subseteq E(\Gamma_{\succ 2})\) implies that for all \(A \in \mathcal{A}\), \(W^1_A \subseteq W^2_A\). Hence, \(\succ (W^1_A) \subseteq \succ (W^2_A)\) and \(\succ 1 \subseteq \succ 2\). Therefore, \(E(\Gamma_{\succ 1}) \subseteq E(\Gamma_{\succ 2})\), and for some \(A\) we have \(W^2_A - W^1_A \neq \emptyset\). Lemma 3 implies \(\succ 1 \not\subseteq \succ 2\), which is a contradiction.

Theorem 5. Let \(o, o' \in \cup s.t. o \neq o'\) and \(\succ \in \mathcal{F}_2\). Then the following conditions are equivalent:

1. \(o \succ o'\);
2. \(\text{BetIn}(o, o') \supseteq \text{Top}_\succ(o, o')\);
3. \(\text{Ch}_\Gamma(\text{BetIn}(o, o')) \supseteq \text{BetIn}(o, o')\).

Proof. Let the structure \((W, \mathcal{A})\) generate a relation equal to \(\succ\), i.e.
\[\succ \equiv (W, \mathcal{A}) \equiv TC \left( \bigcup_{A \in \mathcal{A}} q_A \right)\]
where
\[q_A \equiv \{(o_1, o_2) \mid o_1.A \succ A o_2.A \} \cap \approx_{R - \{(A), W_A\}}\]

\(\Rightarrow\) Let \(\text{Ch}_\Gamma(\text{BetIn}(o, o')) \supseteq \text{BetIn}(o, o')\). W.l.o.g., take \(\text{BetIn}(o, o') = \{A_1, \ldots, A_k\}\). It is easy to check that the sequence \((\Sigma_{o,o'}, \Psi_{o,o'})\) constructed as follows is a derivation sequence for \(o \succ (W^2) o'\). Let \(\Psi_{o,o'} = \text{BetIn}(o, o') = \{A_1, \ldots, A_k\}\). Let the values of all the attributes \(\mathcal{A} - \{\text{BetIn}(o, o') \cup \text{BetIn}(o, o')\}\) in \(\Sigma_{o,o'}\) be equal to those in \(o\) which are also equal to those in \(o'\). Set \(o_1\) to \(o\). Now pick \(i\) from \(2\) to \(k\) consecutively and set the values of \(\{A_i\} \cup \{W_{A_i} \cap \text{BetIn}(o, o')\}\) in \(o_i\) to those in \(o'\). Since \(W_{A_i} = \text{Ch}_\Gamma(A_i)\) (Theorem 1), the value of every attribute in \(o_i\) will be equal to the corresponding value in \(o'\).

Now assume \(\text{Ch}_\Gamma(\text{BetIn}(o, o')) \supseteq \text{BetIn}(o, o')\). Thus, the set \(\text{BetIn}(o, o') - \text{Ch}_\Gamma(\text{BetIn}(o, o'))\) is nonempty. Similarly to the proof of Lemma 3, it can be shown that no derivation sequence exists for \(o \succ (W^2) o'\).

2 implies 3 by definition of \(\text{Top}_\succ(o, o')\). Proof that \(3 \Rightarrow 2\). Assume that \(3\) holds but \(\exists A \in \text{Top}_\succ(o, o') = \text{BetIn}(o, o')\). Since \(\succ A\) is a total order, \(A \in \text{BetIn}(o, o')\), and \(3\) implies that \(A \not\in \text{Top}_\succ(o, o')\), which is a contradiction.
Proof. We prove the theorem by contradiction. Let

\((A, B) \in \Gamma_\succ \land (C, D) \in \Gamma_\succ \land (C, B) \in \Gamma_\succ \land (C, A) \not\in \Gamma_\succ \land (A, D) \not\in \Gamma_\succ \land (D, B) \not\in \Gamma_\succ\).

The second part is equivalent to the following:

\[\exists C \in C, A_1, A_2 \in A, D_1, D_2 \in D, B \in B\]

\[((C, A_2) \not\in \Gamma_\succ \land (A_1, D_1) \not\in \Gamma_\succ \land (D_2, B) \not\in \Gamma_\succ)\]

and from the first part

\[(A_1, B) \in \Gamma_\succ\]

\[(A_2, B) \in \Gamma_\succ\]

\[(C, D_1) \in \Gamma_\succ\]

\[(C, D_2) \in \Gamma_\succ\]

Note that the fact that the subgraphs of \(\Gamma_\succ\) induced by \(A, B, C, D\) are singletons or unions of at least two disjoint subgraphs implies the following four cases for \(A_1\) and \(A_2\):

\[\Gamma_\succ \models A_1 \sim A_2\]

\[(A_1, A_2) \in \Gamma_\succ \land \exists A_3 \in A (\Gamma_\succ \models A_1 \sim A_3 \land \Gamma_\succ \models A_2 \sim A_3)\]

\[(A_2, A_1) \in \Gamma_\succ \land \exists A_3 \in A (\Gamma_\succ \models A_1 \sim A_3 \land \Gamma_\succ \models A_2 \sim A_3)\]

\[A_1 \equiv A_2\]

Similarly, we have four cases for \(D_1, D_2\):

\[\Gamma_\succ \models D_1 \sim D_2\]

\[(D_1, D_2) \in \Gamma_\succ \land \exists D_3 \in D (\Gamma_\succ \models D_1 \sim D_3 \land \Gamma_\succ \models D_2 \sim D_3)\]

\[(D_2, D_1) \in \Gamma_\succ \land \exists D_3 \in D (\Gamma_\succ \models D_1 \sim D_3 \land \Gamma_\succ \models D_2 \sim D_3)\]

\[D_1 \equiv D_2\]

Notice that by our initial assumption, there exist two attributes \(A_1, A_2 \in A\) and two attributes \(D_1, D_2 \in D\). Case A4 and D4 are due to the fact that \(A_1, A_2\) and \(D_1, D_2\) may corresponding to the same attributes in \(A\) and \(D\), respectively.

Totally we have sixteen different cases, and we need to show that all of them lead to contradictions. One can show that all of them contradict the Envelope property. We demonstrate it for the case (A3-D2), while the other cases are handled similarly. In Figure 21, we show instances of the Envelope property. Recall that the Envelope property says that if a graph has certain three edges, it must have at least one of the other three edges. The instances we show below lead to only one possible edge while the other two violate some conditions above. The violated condition is shown below each corresponding edge. Finally, we show that there is an unsatisfiable instance of the Envelope property.

We have exhaustively tested the other fifteen cases and showed that similar contradictions can be derived for them, too. \(\square\)

**Fig. 21 Case A3-D2**

**Theorem 7.** Let \(\succ \in \mathcal{F}_M\), and \(T_\succ\) be a normalized syntax tree of \(\succ\). Then \(\succ_{ext}\) is a minimal p-extension of \(\succ\) iff the syntax tree \(T_{\succ_{ext}}\) of \(\succ_{ext}\) is obtained from \(T_\succ\) by a single application of a rule from Rule 1, \ldots, Rule 4, followed by a single-child node elimination if necessary.

To prove Theorem 7 we introduce the notions of frontier nodes, and top and bottom components in a syntax tree.

**Definition 17** The top and bottom components of a p-skyline relation \(\succ\) are defined as follows:

1. if \(\succ\) is the atomic preference relation induced by an attribute preference relation, then top = bottom = \(\succ\);
2. if \(\succ = \succ_1 \land \ldots \land \succ_m\), then top = \(\succ_1\) and bottom = \(\succ_m\).

Note that the notions of top and bottom components are undefined for p-skyline relations defined as Pareto accumulations of p-skyline relations.

**Definition 18** Let \(T_\succ\) be a normalized syntax tree of a p-skyline relation \(\succ\). Let also \(C_1\) and \(C_2\) be two different children nodes of a \(\odot\) node \(C\) in \(T_\succ\). Let \(\succ_{ext}\) be a p-extension
of $\succ$. Moreover, let the subgraphs of $\Gamma_\succ$ and $\Gamma_{\succ ext}$ induced by $\text{Var}(C_1)$ be equal, as well as those induced by $\text{Var}(C_2)$. Let $X \in \text{Var}(C_1)$, $Y \in \text{Var}(C_2)$ be such that

$$(X, Y) \in \Gamma_{\succ ext}.$$ 

Then $(C_1, C_2)$ is a frontier pair of $T_\succ$ w.r.t. $T_{\succ ext}$.

Given a frontier pair $(C_1, C_2)$ of $T_\succ$ w.r.t. $T_{\succ ext}$, note that $\Gamma_{\succ}$ w.r.t. $T_{\succ ext}$, note that $\Gamma_{\succ} \models \text{Var}(X) \sim \text{Var}(Y)$ by Proposition 4. By definition, a p-skyline relation is constructed in a recursive way: a higher-level relation is defined in terms of lower-level relations. Hence, the intuition behind the frontier pair is as follows. When $\succ$ and $\succ ext$ are constructed, the lower-level relations $\succ_1$ and $\succ_2$ are present in both $\succ$ and $\succ ext$. However, the next-level relations defined using $\succ_1$ and $\succ_2$ in $\succ$ and $\succ ext$ are different since $\Gamma_{\succ ext}$ has an edge from a member of $\text{Var}(\succ_1)$ to a member of $\text{Var}(\succ_2)$, which is not present in $\Gamma_{\succ}$. The next lemma shows some properties of frontier pairs.

**Lemma 4** Let $\succ ext$ be a p-extension of $\succ \in F_{\mathcal{I}}$, and $T_\succ$ be a normalized syntax tree of $\succ$. Let also $(C_1, C_2)$ or $(C_2, C_1)$ be a frontier pair of $T_\succ$ w.r.t. $T_{\succ ext}$. Denote the top and the bottom components of $C_1$ as $A_1, B_1$, and the top and the bottom components of $C_2$ as $A_2, B_2$. Then

$$(\text{Var}(A_1), \text{Var}(B_1)) \in \Gamma_{\succ ext} \vee (\text{Var}(A_2), \text{Var}(B_2)) \in \Gamma_{\succ ext}$$

**Proof.** We consider the case of $(C_1, C_2)$ being a frontier pair of $T_\succ$ w.r.t. $T_{\succ ext}$. The case of $(C_2, C_1)$ is symmetric. Since $(C_1, C_2)$ is a frontier pair of $T_\succ$ w.r.t. $T_{\succ ext}$, there are $X \in \text{Var}(C_1)$ and $Y \in \text{Var}(C_2)$ such that

$$(X, Y) \in \Gamma_{\succ ext}$$

Note that we have the following cases for $X \in \text{Var}(C_1)$

| Case | $\text{Var}(C_1)$ |
|------|------------------|
| $\phi_1$ | $\{X\}$, i.e. $(C_1 = A_1 \equiv B_1)$ |
| $\phi_2$ | $(C_1 = A_1 \& \ldots \& B_1), X \not\in \text{Var}(A_1)$ |
| $\phi_3$ | $(C_1 = A_1 \& \ldots \& B_1), \text{Var}(A_1) = \{X\}$ |
| $\phi_4$ | $(C_1 = A_1 \& \ldots \& B_1)$, $A_1 = A_1^1 \oplus A_1^2 \ldots, X \in \text{Var}(A_1^1)$ |

and for $Y \in \text{Var}(C_2)$

| Case | $\text{Var}(C_2)$ |
|------|------------------|
| $\lambda_1$ | $\{Y\}$, i.e. $(C_2 = A_2 \equiv B_2)$ |
| $\lambda_2$ | $(C_2 = A_2 \& \ldots \& B_2), Y \not\in \text{Var}(B_2)$ |
| $\lambda_3$ | $(C_2 = A_2 \& \ldots \& B_2), \text{Var}(B_2) = \{Y\}$ |
| $\lambda_4$ | $(C_2 = A_2 \& \ldots \& B_2)$, $B_2 = B_2^1 \oplus B_2^2 \ldots, Y \in \text{Var}(B_2^1)$ |

The cases $\phi_1, \phi_2,$ and $\phi_3$ imply either $(\text{Var}(A_1), X) \in \Gamma_{\succ ext}$ or $\text{Var}(A_1) = \{X\}$ and as a result $(\text{Var}(A_1), Y) \in \Gamma_{\succ ext}$ by transitivity of $\Gamma_{\succ ext}$. Similarly, the cases $\lambda_1, \lambda_2,$ and $\lambda_3$ imply either $\text{Var}(B_2) = \{Y\}$ or $(Y, \text{Var}(B_2)) \in \Gamma_{\succ ext}$. Thus every combination of these cases implies $(\text{Var}(A_1), \text{Var}(B_2)) \in \Gamma_{\succ ext}$. Now consider the other combinations of the cases. All of them are handled similar to the case $(\phi_4, \lambda_4)$, so we consider it in detail.

Take the case $\lambda_3$. Take $Y' \in \text{Var}(B_2) - \text{Var}(B_2^1)$ and apply GeneralEnvelope to $\Gamma_{\succ ext}$:

$$(\text{Var}(A_2), Y') \in \Gamma_{\succ ext} \wedge (\text{Var}(A_2), Y) \in \Gamma_{\succ ext} \wedge (X, Y') \in \Gamma_{\succ ext}$$

which implies

$$(\text{Var}(A_2), X) \in \Gamma_{\succ ext} \vee (X, Y') \in \Gamma_{\succ ext} \vee (Y', Y) \in \Gamma_{\succ ext}.$$ 

$(Y', Y) \not\in \Gamma_{\succ ext}$ follows from Proposition 4 and the fact that the subgraphs of $\Gamma_{\succ ext}$ and $\Gamma_{\succ}$ are equal, as well as those induced by $\text{Var}(C_2)$ following from $\phi_4$ imply $(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext}$, which is what we need. Hence, $(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext} \wedge (X, Y') \in \Gamma_{\succ ext}$ for all $Y' \in \text{Var}(B_2) - \text{Var}(B_2^1)$. Consider $(X, Y') \in \Gamma_{\succ ext}$ and pick $Y'' \in \text{Var}(B_1)$. For such $Y''$ we have $(Y', Y'') \not\in \Gamma_{\succ ext}$ by Proposition 4. Therefore, we get a condition for GeneralEnvelope similar to the one above:

$$(\text{Var}(A_2), Y'') \in \Gamma_{\succ ext} \wedge (\text{Var}(A_2), Y') \in \Gamma_{\succ ext} \wedge (X, Y') \in \Gamma_{\succ ext}$$

$(Y'', Y') \not\in \Gamma_{\succ ext}$ by the same argument as above. Similarly to the above, $(\text{Var}(A_2), X) \in \Gamma_{\succ ext}$ and $(X, \text{Var}(B_1)) \in \Gamma_{\succ ext}$ imply $(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext}$, which is what we need. As a result, we have $(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext}$ or $(X, Y') \in \Gamma_{\succ ext} \wedge (X, Y'') \in \Gamma_{\succ ext}$ for all $Y' \in \text{Var}(B_2) - \text{Var}(B_2^1), Y'' \in \text{Var}(B_1)$, that is equivalent to

$$(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext} \vee (X, \text{Var}(B_2)) \in \Gamma_{\succ ext}.$$ 

Elaborating the case $\phi_4$ above gives that

$$(\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext} \vee (\text{Var}(A_1), Y) \in \Gamma_{\succ ext}.$$ 

After combining these two results and applying GeneralEnvelope to members of $A_1$ and $B_2$, we get

$$(\text{Var}(A_1), \text{Var}(B_2)) \in \Gamma_{\succ ext} \vee (\text{Var}(A_2), \text{Var}(B_1)) \in \Gamma_{\succ ext}.$$ 

Now we go back to the proof of Theorem 7.

**Proof of Theorem 7**

Let $\succ ext$ be a minimal p-extension of $\succ$. We show here that there is $\succ' \in F_{\mathcal{I}}$ obtained using a transformation rule $\text{Rule}_1, \ldots, \text{Rule}_4$ such that

$$\succ \subset \succ' \subseteq \succ ext.$$ (20)

By the minimal p-extension property of $\succ ext$ that implies

$$\succ' = \succ ext.$$
Theorem 4 implies that there are $X$ and $Y$ such that $(X, Y) \in E(\Gamma_{\text{ext}}) - E(\Gamma_\prec)$. Let $(C_1, C_2)$ be a frontier pair of $T_{\text{ext}}$ w.r.t. $\Gamma_{\text{ext}}$, such that $X \in \text{Var}(C_1)$ and $Y \in \text{Var}(C_2)$. Lemma 4 implies that

$$\text{(Var}(A_1),\text{Var}(B_2)) \in \Gamma_{\text{ext}} \lor \text{(Var}(A_2),\text{Var}(B_1)) \in \Gamma_{\text{ext}}$$

(21)

for the top $A_1, A_2$ and the bottom $B_1, B_2$ components of $C_1$ and $C_2$, correspondingly. Consider all possible types of $C_1$ and $C_2$. (i) Let $C_1, C_2$ be leaf nodes. Then $\succ'\forall$ for which (20) holds may be obtained by applying $\text{Rule}_3(T_\prec, C_1, C_2)$ (if the first disjunct of (21) holds) or $\text{Rule}_2(T_\prec, C_1, C_2)$ (if the second disjunct of (21) holds). (ii) Let $C_1$ be a $\&$-node and $C_2$ be a leaf node. Then $\succ'$ may be obtained by applying $\text{Rule}_1(T_\prec, C_1, C_2)$ (if the first disjunct of (21) holds) or $\text{Rule}_2(T_\prec, C_1, C_2)$ (if the second disjunct of (21) holds). Case (iii) when $C_1$ is a leaf node and $C_2$ is a $\&$-node is similar to the previous case. Consider case (iv) when $C_1$ and $C_2$ are $\&$-nodes. Let the first disjunct of (21) hold. The case of the second disjunct is analogous. We note that $(\text{Var}(A_1),\text{Var}(B_1)) \in \Gamma_{\text{ext}}$ and $(\text{Var}(A_2),\text{Var}(B_2)) \in \Gamma_{\text{ext}}$. This with (21) is a condition for $\text{GeneralEnvelope}$:

$$\text{(Var}(A_1),\text{Var}(A_2)) \in \Gamma_{\text{ext}} \lor \text{(Var}(A_2),\text{Var}(B_1)) \in \Gamma_{\text{ext}} \lor \text{(Var}(B_2),\text{Var}(B_1)) \in \Gamma_{\text{ext}}$$

(22)

If the first disjunct of (22) holds, then $\succ'$ can be obtained by applying $\text{Rule}_1(T_\prec, C_1, C_2)$. If the last disjunct of (22) holds, then $\succ'$ can be obtained by applying $\text{Rule}_2(T_\prec, C_1, C_2)$. Let the second disjunct of (22) hold, i.e. $(\text{Var}(A_2),\text{Var}(B_1)) \in \Gamma_{\text{ext}}$. Let the child nodes of $C_1$ and $C_2$ be the sequences $(A_1 = N_1, \ldots, N_m = B_1)$ and $(A_2 = M_1, \ldots, M_n = B_2)$ correspondingly. The fact that $C_1$ and $C_2$ are $\&$-nodes implies $(\text{Var}(N_1),\text{Var}(N_j)) \in \Gamma_{\text{ext}}$ and $(\text{Var}(M_i),\text{Var}(M_j)) \in \Gamma_{\text{ext}}$ for all $i < j$. Since $\succ \subseteq \succ'$, the same edges are present in $\Gamma_{\text{ext}}$. Note that $(M_1, N_m) \in \Gamma_{\text{ext}}$. Pick every child of $C_2$ in its list of children from right to left and find the first index $t$ such that $(\text{Var}(N_1),\text{Var}(M_1)) \notin \Gamma_{\text{ext}}$ but $(\text{Var}(N_1),\text{Var}(M_{t+1})) \in \Gamma_{\text{ext}}$. If no such $t$ exists, then $(\text{Var}(N_1),\text{Var}(M_1)) \in \Gamma_{\text{ext}}$ and $\succ'$ may be obtained by applying $\text{Rule}_1(T_\prec, C_1, C_2)$. Assume $t \in [1, n]$. Similarly, let $s$ be the first index such that $(\text{Var}(M_1),\text{Var}(N_1)) \notin \Gamma_{\text{ext}}$ but $(\text{Var}(M_1),\text{Var}(N_{s+1})) \in \Gamma_{\text{ext}}$. If $s$ does not exist, then $\succ'$ may be obtained by applying $\text{Rule}_2(T_\prec, C_1, C_2)$. So assume $s \in [1, m]$. If both $s$ and $t$ are equal to 1, then $\succ'$ may be obtained using $\text{Rule}_4(T_\prec, C_1, C_2, s, t)$. In all other cases, $\text{GeneralEnvelope}$ can be used to show that for all $i \in [1, s], j \in [t + 1, n] \text{(Var}(N_i),\text{Var}(M_j)) \in \Gamma_{\text{ext}}$ and for all $i \in [1, t], j \in [s + 1, m] \text{(Var}(M_i),\text{Var}(N_j)) \in \Gamma_{\text{ext}}$. Hence $\text{Rule}_{a}(T_\prec, C_1, C_2, s, t)$ may be used to construct $\succ'_{\text{ext}}$. Show that every valid application of $\text{Rule}_1, \ldots, \text{Rule}_4$ results in a minimal extension. We do it by case analysis.

Take $\text{Rule}_3$, which results in adding the edge from $C_i$ to $C_{i+1}$ to the $p$-graph. This is clearly a minimal extension of the $p$-graph and hence the resulting $p$-skyline relation is a minimal extension of $\succ$. The analysis pattern for the remaining rules is as follows. We assume that some $p$-extension $\succ_{\text{ext}}$ obtained by an application of $\text{Rule}_1, \text{Rule}_2$, or $\text{Rule}_4$ to $\succ$ is not minimal, i.e., there is $\succ'$ s.t. $\succ \subseteq \succ' \subseteq \succ_{\text{ext}}$. After that, we derive a contradiction that $\Gamma_{\succ} = \Gamma_{\text{ext}}$. Take $\text{Rule}_3$. Since $\succ'$ is an extension of $\succ$ contained in $\succ_{\text{ext}}$, there must be an edge from some $A \in \text{Var}(N_1)$ to some $B$ in the bottom component of $C_{i+1}$. Clearly, if $\text{Var}(N_1) = \{A\}$ and $\text{Var}(C_{i+1}) = \{B\}$, then $\Gamma_{\succ} = \Gamma_{\text{ext}}$ and we get the contradiction we want. So assume $\text{Var}(C_{i+1}) \neq \{B\}$. Then applying $\text{GeneralEnvelope}$ to

$$(\text{Var}(N_2)) \in \Gamma_{\succ} \land (\text{Var}(B)) \in \Gamma_{\succ} \land$$

$$(\text{Var}(T_{i+1}),\text{Var}(B)) \in \Gamma_{\succ}$$

(23)

where $T_{i+1}$ is the top component of $C_{i+1}$ results in $(A, \text{Var}(N_2)) \in \Gamma_{\succ}$ (and hence $(A, \text{Var}(C_{i+1})) \in \Gamma_{\succ}$ by transitivity of $\Gamma_{\succ}$). The other alternatives are impossible; the corresponding edges are missing in $\Gamma_{\text{ext}}$ (and hence in $\Gamma_{\succ}$, too). Clearly, if $\text{Var}(N_1) = \{A\}$, then we get the contradiction we need: $\Gamma_{\succ} = \Gamma_{\text{ext}}$. So assume $\text{Var}(N_1) \neq \{A\}$. Denote $S = \text{Var}(N_1) - \{A\}$. Then applying $\text{GeneralEnvelope}$ to

$$(S, \text{Var}(N_2)) \in \Gamma_{\succ} \land (A, \text{Var}(N_2)) \in \Gamma_{\succ} \land$$

$$(A, \text{Var}(C_{i+1})) \in \Gamma_{\succ}$$

results in $(S, \text{Var}(C_{i+1})) \in \Gamma_{\succ}$. The other alternatives are prohibited because the corresponding $p$-graph edges are not in $\Gamma_{\text{ext}}$ (and hence not in $\Gamma_{\succ}$). That results in $(\text{Var}(N_1), \text{Var}(C_{i+1})) \in \Gamma_{\succ}$ and the contradiction that $\Gamma_{\text{ext}} = \Gamma_{\succ}$. The case analysis for $\text{Rule}_4$ is similar.

Now let $\succ_{\text{ext}}$ be obtained from $\succ$ by applying $\text{Rule}_4$, and consider a $p$-extension $\succ'$ of $\succ$ s.t. $\succ \subseteq \succ' \subseteq \succ_{\text{ext}}$. Because of this assumption, $\Gamma_{\succ}$ has an edge from some $A \in \text{Var}(N_1)$ to some $B \in \text{Var}(M_1)$ or from some $C \in \text{Var}(M_1)$ to some $D \in \text{Var}(N_m)$. Since these cases are completely symmetric, take $(A, B) \in \Gamma_{\succ}$. Applying $\text{GeneralEnvelope}$ to

$$(\text{Var}(N_{i+1})) \in \Gamma_{\succ} \land (A, B) \in \Gamma_{\succ} \land$$

$$(\text{Var}(M_1),\text{Var}(M_1)) \in \Gamma_{\succ}$$

results in

$$(\text{Var}(M_1),\text{Var}(N_{i+1})) \in \Gamma_{\succ}$$

since all the other alternatives are impossible – the corresponding $p$-graph edges are not in $\Gamma_{\text{ext}}$ – and hence not in $\Gamma_{\succ}$. Now apply $\text{GeneralEnvelope}$ to

$$(\text{Var}(M_1),\text{Var}(M_{i+1})) \in \Gamma_{\succ} \land (\text{Var}(M_1),\text{Var}(N_{i+1})) \in \Gamma_{\succ} \land$$

$$(\text{Var}(N_1),\text{Var}(N_{i+1})) \in \Gamma_{\succ}$$
which results in
\[(\text{Var}(N_i), \text{Var}(M_{i+1})) \in \Gamma_{\succ r}\]
(24)
since all the other alternatives are impossible – the corresponding p-graph edges are not in \(\Gamma_{\succ r}\) and hence not in \(\Gamma_{\succ r}\). (23), (24), and the transitivity of \(\Gamma_{\succ r}\) implies that \(\Gamma_{\succ r} = \Gamma_{\succ r}\), which is a contradiction. □

**Theorem 8.** DF-PSKYLINE is NP-complete.

**Proof.** The favoring/disfavoring p-skyline existence problem is in NP since checking if a p-skyline relation \(\succ\) favors \(G\) and disfavors \(W\) in \(O\) can be done in polynomial time by evaluating \(\omega_r(O)\), checking \(G \subseteq \omega_r(O)\), and checking if for every member of \(W\) there is a member of \(W\) dominating it.

To show the hard result, we do a polynomial-time reduction from SAT. This is a two-step reduction. First, we show that for every instance \(\phi\) of SAT there are corresponding instances of positive \(\Pi\) and negative \(\mathcal{N}\) constraints, and \(\phi\) has a solution iff \(\Pi\) and \(\mathcal{N}\) are satisfiable. Second, we show that for every \(\Pi\) and \(\mathcal{N}\) there are corresponding instances of \(G, W,\) and \(O\).

Consider instances of SAT in the following form
\[\phi(x_1, \ldots, x_n) = \psi_1(x_1, \ldots, x_n) \land \ldots \land \psi_m(x_1, \ldots, x_n)\]
where
\[\psi_i(x_1, \ldots, x_n) = \chi_{i_0} \lor \ldots \lor \chi_{i_j}\]

For every instance of \(\phi\), construct \(\mathcal{A} = \{c, y_1, \overline{y}_1, y_1', \ldots, y_n, \overline{y}_n, y_n', \overline{y}_n'\}\). The sets of positive and negative constraints are constructed as follows. Let \(\Gamma\) be a graph. For every variable \(x_i\),

1. Create positive constraints
   \[\chi_i : (y_i, c) \in \Gamma \lor (\overline{y}_i, c) \in \Gamma\]
   \[\pi_i : (y_i', c) \in \Gamma\]

2. Create negative constraints
   \[\lambda_1 : (y_i, \overline{y}_i) \not\in \Gamma\]
   \[\lambda_2 : (y_i, y_i') \not\in \Gamma\]
   \[\lambda_3 : (y_i', c) \not\in \Gamma\]

Now, for every \(\psi_i(x_1, \ldots, x_n) = \chi_{i_0} \lor \ldots \lor \chi_{i_j}\) of \(\phi\) construct the following positive constraint
\[\mu_i : (\overline{y}_i, c) \in \Gamma \lor \ldots \lor (\overline{y}_i, c) \in \Gamma\]
where
\[\overline{y}_i = \begin{cases} y_i & \text{if } \overline{x}_i = x_i \\ \overline{y}_i & \text{if } \overline{x}_i = \overline{x}_i \end{cases}\]

We claim that there is a satisfying assignment \((v_1, \ldots, v_n)\) for \(\phi\) iff there is a p-graph satisfying all the constraints above.

First, assume there is a p-graph \(\Gamma\) satisfying all the constraints above. Construct the assignment \(v = (v_1, \ldots, v_n)\) as follows:

\[v_i = \begin{cases} 0 & \text{if } (\overline{y}_i, c) \in \Gamma \\ 1 & \text{if } (y_i, c) \in \Gamma \end{cases}\]

Since \(\Gamma\) satisfies all \(\chi_i\), for every \(i\) we have \((y_i, c) \in \Gamma\) or \((\overline{y}_i, c) \in \Gamma\). Thus, every \(v_i\) will be assigned to some value according to the rule above. Now prove that \(v_i\) is assigned to only one value, i.e., we cannot have both \((y_i, c) \in \Gamma\) and \((\overline{y}_i, c) \in \Gamma\). Since \(\Gamma\) satisfies \(\pi_i\), we have \((\overline{y}_i, y_i') \in \Gamma\). Thus having both \((y_i, c) \in \Gamma\) and \((\overline{y}_i, c) \in \Gamma\) and Envelope implies
\[(\overline{y}_i, y_i) \in \Gamma \lor (y_i', c) \in \Gamma \lor (y_i', c) \in \Gamma\]
However, the expression above violates the constraints \(\lambda_1, \lambda_2, \lambda_3\). Therefore, exactly one of \((y_i, c) \in \Gamma, (\overline{y}_i, c) \in \Gamma\) holds.

Take every \(\mu_i\). Since it is satisfied by \(\Gamma\), the corresponding \(\psi_i\) must be also satisfied by the construction of \(\mu_i\). Therefore, \(\phi\) is also satisfied.

Now assume that there is an assignment \((v_1, \ldots, v_n)\) satisfying \(\phi\). Show that there is a p-graph \(\Gamma_{\succ r}\) satisfying all the constraints above. Here we construct such a graph.

For every \(i \in [1, n]\), draw the edge
\[(y_i, c) \in \Gamma_{\succ r} \text{ if } v_i = 1, \quad (\overline{y}_i, c) \in \Gamma_{\succ r}, \text{ otherwise}\]
This satisfies the constraint \(\chi_i\). Moreover, all the constraints \(\mu_i\) are satisfied by the construction. Now, for every \(i \in [1, n]\), draw the edge
\[(\overline{y}_i, y_i') \in \Gamma_{\succ r}\]
which satisfies the constraint \(\pi_i\). As a result, all positive constraints are satisfied. Moreover, none of the edges above violates any negative constraints. Thus, all the constraints above are satisfied.

In addition to the edges above, let us draw the following edges

1. for every \(i, j \ (i \neq j)\) such that \(v_i = 0, v_j = 0\), draw the edge
   \[(\overline{y}_i, y_j') \in \Gamma_{\succ r}\] (P4)
   It is clear that these edges do not violate any negative constraints above.

2. for every \(i, j \) such that \(v_i = 0, v_j = 1\), draw the edge
   \[(\overline{y}_i, y_j') \in \Gamma_{\succ r}\] (P5)
Since \(i \neq j\), this edge does not violate any negative constraints above.
It is easy to verify that the constructed graph $\Gamma_3$ satisfies SPO+Envelope and all the negative and positive constraints above.

Now let us show that there exist sets of objects $O, G$ and $W$ which can be used to construct the constraints $\chi_1, \pi_1, \lambda_1, \lambda_1, \mu$. Assume that for every attribute in $A \in \mathcal{A}$, its domain contains at least three numbers $\{-1, 0, 1\}$, and greater values are to be preferred in the attribute preference relation $\succ_A$. Here we construct the sets $G, W, M$, and $O = G \cup W \cup M$ that generate the positive and negative constraints above.

1. Let $G$ consist of a single object $g$ with all attributes values equal to 0.
2. Let $W = \{b_1, \ldots, b_n, u_1, \ldots, u_m, w_1, \ldots, w_m\}$ be constructed as follows:
   - for every $i \in [1, \ldots, n]$, let all the attributes of $b_i$ be equal to 0, except for the value of $y_i$, which is 1, and the value of $y_i$, which is 1.
   - for every $i \in [1, \ldots, n]$, let all the attributes of $u_i$ be equal to 0, except for the value of $y_i$, which is 1, and the value of $c$, which is 1.
   - for every $i \in [1, \ldots, m]$, let $\mu_i : (y_i, c) \in \Gamma \lor \ldots \lor (y_i, c) \in \Gamma$, where $y_i \in \{y, \bar{y}\}$. Let all attributes of $w_i$ be equal to 0, except for the value of $c$, which is 1, and the values of $y_i, y_i$ (whatever they are), which are 1.
3. Let $M = \{m_1, m_1, m_3, \ldots, m_{m_1}, m_{m_1}, m_{m_1}\}$ be constructed as follows. For all $i \in [1, \ldots, n],$
   - Let all attributes of $m_1$ be 0, except for the value $y_i$, which is 1, and the value of $\bar{y}_i$, which is 1.
   - Let all attributes of $m_2$ be 0, except for the value of $y_i$, which is 1, and the value of $\bar{y}_i$, which is 1.
   - Let all attributes of $m_3$ be 0, except for the value of $y_i$, which is 1, and the value of $c$, which is 1.

It can be easily shown that these sets of objects induce the set of constructed constraints (see Example 19). □

**Example 19** Take $n = 3$ and

$$\phi(x_1, x_2, x_3) = (x_1 \lor x_2 \lor \bar{x}_3) \land (\bar{x}_1 \lor x_2 \lor x_3).$$

Then $\mathcal{A} = \{c, y_1, y_1, y_2, y_2, y_2, y_3, y_3, y_3\}$. The constraints $\mu_1, \mu_2$ are

$$\mu_1 : (y_1, c) \in \Gamma \lor (y_2, c) \in \Gamma \lor (y_3, c) \in \Gamma
\mu_2 : (y_3, c) \in \Gamma \lor (y_2, c) \in \Gamma \lor (y_3, c) \in \Gamma$$

Take the assignment $v = (1, 0, 1)$ satisfying $\phi$. By construction above, we get the graph $\Gamma$ as in Figure 22. Now let us construct the sets $G, W, M$ as above.

|   | $y_1$ | $y_2$ | $y_3$ | $\bar{y}_1$ | $\bar{y}_2$ | $\bar{y}_3$ | $\bar{y}$ | $c$ |
|---|---|---|---|---|---|---|---|---|
| $g$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $b_1$ | 0 | -1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $b_2$ | 0 | 0 | 0 | -1 | 1 | 0 | 0 | 0 |
| $b_3$ | 0 | 0 | 0 | 0 | 0 | -1 | 1 | 0 |
| $u_1$ | -1 | -1 | 0 | 0 | 0 | 0 | 0 | 1 |
| $u_2$ | 0 | 0 | -1 | -1 | 0 | 0 | 0 | 1 |
| $u_3$ | 0 | 0 | 0 | 0 | 0 | -1 | 1 | 0 |
| $w_1$ | -1 | 0 | 0 | -1 | 0 | 0 | 0 | 1 |
| $w_2$ | 0 | -1 | 0 | -1 | 0 | 0 | 0 | 0 |
| $m_1$ | -1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $m_2$ | 1 | 0 | -1 | 0 | 0 | 0 | 0 | 0 |
| $m_3$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $m_4$ | 0 | 0 | -1 | 1 | 0 | 0 | 0 | 0 |
| $m_5$ | 0 | 0 | 0 | 1 | 0 | -1 | 0 | 0 |
| $m_6$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| $m_7$ | 0 | 0 | 0 | 0 | 0 | -1 | 1 | 0 |
| $m_8$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | -1 |
| $m_9$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Then $G = \{g\}, W = \{b_1, b_2, b_3, u_1, u_2, u_3, w_1, w_2\}, M = \{m_1, \ldots, m_3\}$. For $W$ to be a set of inferior examples, $g$ must be preferred to each member of $W$. Take for instance, $g \succ b_1$. By Theorem 5, that is equivalent to $(\bar{y}, y_1) \in \Gamma_\succ$, which corresponds to $\pi_1$. Similarly, $g \succ u_1$ results in $(y_1, c) \in \Gamma_\succ \lor (\bar{y}, y_1) \in \Gamma_\succ$, which corresponds to $\chi_1$. $g \succ w_1$ results in $(y_1, c) \in \Gamma_\succ \lor (y_2, c) \in \Gamma_\succ \lor (\bar{y}, y_1) \in \Gamma_\succ$, which corresponds to $\mu_1$. The other members of $W$ are handled similarly (resulting in the remaining positive constraints).

For $g$ to be superior, no member of $M \cup W$ must be preferred to $g$ according to $\succ$. Clearly, for a p-skyline relation $\succ$ (which is an SPO), this is equivalent to saying that no member of only $M$ must be preferred to $g$; above we already have constraints that $g$ is preferred to every member of $W$, and $\succ$ is irreflexive. $m_j \succ g$ results in $(\bar{y}, y_1) \notin \Gamma_\succ$, which corresponds to $\lambda_1$. The other members of $M$ are handled similarly, resulting in the remaining negative constraints.

**Proposition 5.** Let $\succ$ be a $p$-skyline relation, $O$ a finite set of tuples, and $G$ and $W$, disjoint subsets of $O$. Then the next two operations can be done in polynomial time:

1. verifying if $\succ$ is maximal favoring $G$ and disfavoring $W$ in $O$;
2. constructing a maximal $p$-skyline relation $\succ_{ed}$ that favors $G$, disfavors $W$ in $O$ and is a $p$-extension of $\succ$ (un-
der the assumption that \( \succ \) favors \( G \) and disfavors \( W \) in \( O \).

**Proof.** To check if \( \succ \) favors \( G \) and disfavors \( W \) in \( O \), we need to compute \( \omega_0(O) \), check \( G \subseteq \omega_0(O) \), and verify that for every \( o \in W \), there is \( o' \in G \) such that \( o' \succ o \). All those tasks can clearly be performed in polynomial time. If some of these conditions fails, \( \succ \) is obviously not maximal. Otherwise, we need to check if each of its minimal \( p \)-extensions favors \( G \) and disfavors \( W \). Note that since \( \succ \) disfavors \( W \) in \( O \), each of its \( p \)-extensions also disfavors \( W \) in \( O \). Hence, \( \succ \) is not maximal if at least one minimal \( p \)-extension favors \( G \) in \( O \), and it is maximal otherwise. Corollaries 2 and 3 imply that all minimal \( p \)-extensions of \( \succ \) can be constructed in polynomial time.

To construct a maximal \( p \)-extension \( \succ' \) of \( \succ \), we take \( \succ \), construct all of its minimal \( p \)-extensions and verify if at least one of them favors \( G \) in \( O \). If some of them does, we select it and repeat for it the same procedure. We do it until for some \( \succ' \) none of its minimal \( p \)-extensions favors \( G \) in \( O \). This implies that \( \succ' \) is a maximal \( p \)-skyline relation favoring \( G \) and disfavoring \( W \) in \( O \). Moreover, \( \succ' \) is a superset of \( \succ \) by construction. Corollaries 2, 3, and 4 imply that such a computation can be done in polynomial time. \( \square \)

**Theorem 9.** \( \text{FDF-PSKYLINE} \) is \( \text{FNP-complete} \)

**Proof.** Given two disjoint subsets \( G \) and \( W \) of \( O \) and \( \succ \in \mathcal{F}_G \), checking if \( \succ \) favors \( G \) and disfavors \( W \) in \( O \) can be done in polynomial time (Lemma 5). Hence, \( \text{FDF-PSKYLINE} \) is \( \text{FNP} \).

Now show that \( \text{FDF-PSKYLINE} \) is \( \text{FNP-hard} \). To do that, we use a reduction from \( \text{FSAT} \). In particular, we find functions \( R \) and \( S \), both computable in logarithmic space, such that 1) for each instance \( x \) of \( \text{FSAT} \), \( R(x) \) is an instance of \( \text{FDF-PSKYLINE} \), and 2) for each correct output \( z \) of \( R(x) \), \( S(z) \) is a correct output of \( x \). For such a reduction, we use the construction from the proof of Theorem 8. There we showed how a relation (denote it as \( \succ \)) satisfying all the constraints (and thus favoring/disfavoring the constructed \( G \) and \( W \)) may be obtained. In the current reduction, if there is a \( p \)-skyline relation favoring \( G \) and disfavoring \( W \) in \( O \), then the relation \( \succ \) itself is returned. Otherwise, "no" is returned.

The function \( R \) mentioned above has to convert an instance of \( \text{FSAT} \) to an instance of \( \text{FDF-PSKYLINE} \) (i.e., \( G, W \), and \( O \)). In the reduction shown in the proof of Theorem 8, such a transformation is done via a set of constraints. However, it is easy to observe that such a construction can be performed using the corresponding instance of \( \text{FSAT} \). By the construction, the sets \( G, M \), and the subset \( \{b_1, \ldots, b_a, u_1, \ldots, u_b \} \) of \( W \) are common for every instance of \( \text{FSAT} \) with \( n \) variables. To construct the subset \( \{w_1, \ldots, w_m \} \) of \( W \), one can use the expression \( \psi \) instead of the corresponding constraint \( \mu \). It is clear that the function \( R \) performing such a transformation can be evaluated in logarithmic space.

We construct the function \( S \) as follows. If the instance of \( \text{FDF-PSKYLINE} \) returns "no", \( S \) returns "no". Otherwise, it constructs the satisfying assignment \( (v_1, \ldots, v_n) \) in the following way: for every \( i \), \( v_i \) is set to 1 if the p-graph contains the edge \( (y_i, e) \in \Gamma, \) and 0 otherwise. It is clear that such a computation may be done in logarithmic space. \( \square \)

**Theorem 10.** \( \text{OPT-FDF-PSKYLINE} \) is \( \text{FNP-complete} \)

**Proof.** Given \( \succ \in \mathcal{F}_G \), checking if it is maximal favoring \( G \) and disfavoring \( W \) can be done in polynomial time (Proposition 5). Hence, \( \text{OPT-FDF-PSKYLINE} \) is in \( \text{FNP} \).

We reduce from \( \text{FDF-PSKYLINE} \) to show that it is \( \text{FNP-hard} \). Here we construct the function \( F \) that takes a \( p \)-skyline relation or "no" and returns a \( p \)-skyline relation or "no". \( F \) returns "no" if its input is "no". If its input is a \( p \)-skyline relation \( \succ \) and disfavors \( W \) in \( O \), then \( \succ \) returns a maximal \( p \)-extension of \( \succ \) as shown in Proposition 5. As a result, \( F \) returns a maximal favoring/disfavoring \( p \)-skyline relation iff the corresponding favoring/disfavoring \( p \)-skyline relation exists. The functions \( R \) and \( S \) transforming inputs of \( \text{FDF-PSKYLINE} \) to inputs of \( \text{OPT-FDF-PSKYLINE} \) and outputs of \( \text{OPT-FDF-PSKYLINE} \) to outputs of \( \text{FDF-PSKYLINE} \) correspondingly are trivial and hence are computable in logspace. Therefore, the problem \( \text{OPT-FDF-PSKYLINE} \) is \( \text{FNP-complete} \). \( \square \)

**Proposition 7.** Let a relation \( \succ \in \mathcal{F}_G \) be a maximal \( M \)-favoring relation, and a \( p \)-extension \( \succ_{\text{ext}} \) of \( \succ \) (\( (M \cup \{A\}) \)-favoring). Then every edge in \( E(\Gamma_{\succ_{\text{ext}}}) - E(\Gamma_{\succ}) \) starts or ends in \( A \).

**Proof.** Take \( \Gamma_{\succ_{\text{ext}}} \) and construct \( \Gamma' \) from it by removing all edges going from or to \( A \). Clearly, \( \Gamma' \) is an SPO. Now consider the Envelope property. Pick four nodes of \( \Gamma' \) different from \( A \). Since \( \Gamma_{\succ_{\text{ext}}} \) is a p-graph, the Envelope property holds for the graph induced by these four nodes in \( \Gamma_{\succ_{\text{ext}}} \). Envelope also holds for the corresponding subgraph of \( \Gamma' \). Thus, \( \Gamma' \) satisfies the Envelope property as well, i.e., it’s a p-graph of a p-skyline relation \( \succ' \). Moreover, \( E(\Gamma_{\succ}) \subseteq E(\Gamma_{\succ_{\text{ext}}}) \) since \( \Gamma_{\succ} \) has no edges from/to \( A \) and \( E(\Gamma_{\succ_{\text{ext}}}) \subseteq E(\Gamma_{\succ_{\text{ext}}}) \). Since \( \succ \) is maximal \( M \)-favoring, \( E(\Gamma_{\succ_{\text{ext}}}) = E(\Gamma_{\succ}) \). Therefore, all edges in \( E(\Gamma_{\succ_{\text{ext}}}) - E(\Gamma_{\succ}) \) go from or to \( A \). \( \square \)

**Proposition 8.** Let a relation \( \succ \in \mathcal{F}_G \) satisfy a system of negative constraints \( \mathcal{N} \). Construct the system of negative constraints \( \mathcal{N}' \) from \( \mathcal{N} \) in which every constraint \( \tau \in \mathcal{N}' \) is created from a constraint \( \tau \in \mathcal{N} \) in the following way:

\[
\begin{align*}
- \quad L_\tau' & = L_\tau \setminus A, \\
- \quad R_\tau' & = R_\tau \setminus \{B \in R_\tau \mid \exists A \in L_\tau ((A, B) \not\in \Gamma_{\succ})\}.
\end{align*}
\]

Then every \( p \)-extension \( \tau' \) of \( \tau \) satisfies \( \mathcal{N}' \) iff \( \tau' \) satisfies \( \mathcal{N} \).

**Proof.** Take \( \tau' \) from \( \mathcal{N}' \) with the corresponding \( \tau \in \mathcal{N} \). By construction, \( L_\tau' = L_\tau, R_\tau' \subseteq R_\tau \). Now assume \( \tau' \) satisfies \( \mathcal{N} \).

This means that

\[
\exists B \in R_\tau' \forall A \in L_\tau' ((A, B) \not\in \Gamma_{\tau'})
\]

(25)
Now recall that $R_{\ell} \subseteq R_{\ell}$. Thus $B \in R_{\ell}$. This together with $L_{\ell} = L_{\ell}$ and (25) gives

$$\exists B \in R_{\ell} \forall A \in L_{\ell}, ((A, B) \notin \Gamma_{\ell})$$

i.e., $\Gamma_{\ell}$ satisfies $\tau$.

Conjecture: Assume that there is no constraint $\nabla_1 \nabla_2$, $\ell$. $\Gamma_{\ell}$ satisfies $\tau$. This means

$$\exists B \in R_{\ell} \forall A \in L_{\ell}, ((A, B) \notin \Gamma_{\ell})$$

(26)

Since $\nabla \subseteq \nabla'$, $E(\Gamma_{\ell}) \subseteq E(\Gamma_{\ell})$. Thus, if there is no edge from $L_{\ell}$ to $B$ in $\Gamma_{\ell}$, then there is no such edge in its subset $\Gamma_{\ell}$. Recall that $\tau'$ is a minimized version of $\tau$ w.r.t. $\nabla$. Thus, the lack of edge from $L_{\ell}$ to $B$ in $\Gamma_{\ell}$ implies $B \in R_{\ell}$. This together with $L_{\ell} = L_{\ell}$ and (26) gives

$$\exists B \in R_{\ell} \forall A \in L_{\ell}, ((A, B) \notin \Gamma_{\ell})$$

i.e., $\Gamma_{\ell}$ satisfies $\tau'$. \(\square\)

**Proposition 9.** Let a relation $\nabla \in \mathcal{F}_{\ell}$ satisfy a system of negative constraints $\mathcal{N}_{\ell}$ and $\mathcal{N}_{\ell}$ be minimal w.r.t. $\nabla$. Let $\nabla'$ be a p-extension of $\nabla$ such that every edge in $E(\Gamma_{\ell}) \subseteq E(\Gamma_{\ell})$ starts or ends in $A$. Denote the new parents and children of $A$ in $\Gamma_{\ell}$ as $P_A$ and $C_A$ correspondingly. Then $\nabla'$ violates $\mathcal{N}_{\ell}$ iff there is a constraint $\tau \in \mathcal{N}_{\ell}$ such that

1. $R_{\ell} = \{A\} \cap P_A \cap L_{\ell} \neq 0$, or
2. $A \in L_{\ell} \wedge R_{\ell} \subseteq C_A$

**Proof.** $\square$

Trivial since the two conditions above imply violation of $\mathcal{N}_{\ell}$ by $\nabla$.

Assume that there is no constraint $\tau$ for which the two conditions hold, but some $\tau' \in \mathcal{N}_{\ell}$ is violated, i.e., $Ch_{\nabla}(L_{\ell}) = R_{\ell}$.

By Theorem 4, $E(\Gamma_{\ell}) \subseteq E(\Gamma_{\ell})$. We also know that all the new edges in $\Gamma_{\ell}$ start or end in $A$. Since $\Gamma_{\ell}$ satisfies $\tau'$ but $\Gamma_{\ell}$ does not, we get that either $A \in L_{\ell}$ or $A \in R_{\ell}$. If $A$ is in $R_{\ell}$ then the fact that $\tau'$ is violated by $\Gamma_{\ell}$ implies that $R_{\ell} = \{A\}$. Moreover, the fact that $\tau'$ is minimal w.r.t. $\nabla$ implies $P_A \cap L_{\ell} \neq 0$. If $A \in L_{\ell}$, then the minimality of $\tau'$ implies that $\tau$ is violated because of $R_{\ell} \subseteq C_A$. \(\square\)

**Theorem 11.** The function $elicit$ returns a syntax tree of a maximal p-skyline relation favoring $G$ in $O$. Its running time is $O(|\mathcal{N}| \cdot |A|^3)$. The running time of the sub-procedures is $O(|\mathcal{N}| \cdot |A|)$. The time needed to modify the syntax tree using a transformation rule is $O(|A|)$: every rule creates, deletes, and modifies a constant number of nodes of a syntax tree, but updating their Var-variables is done in $O(|A|)$. Similarly, syntax tree normalization runs in time $T_{\text{normalizeTree}} = O(|A|)$ for such modified syntax trees. As a result, the time needed to execute the bodies of the loops (lines 5-8, 11-14, 18-36) of $push$ is $T_{\text{rule}} = O(|\mathcal{N}| \cdot |A|)$.

Now let $T$ be a syntax tree of a maximal $M$-favoring p-skyline relation. Consider the way $push$ is used in $elicit$ to construct a maximal $M$-favoring p-skyline relation. The state diagram of this process is shown in Figure
23. It has three states: $S_0$ and $S_k$ which correspond to $T$ in which $A$ is a child of a \&- and \&-node, respectively; and $S_3$ which corresponds to the case when no transformation rule can be applied to $T$, or every rule application violates $\mathcal{N}_2$.

The starting state is $S_0$. In the starting $T$, $A$ is a child of the topmost $\otimes$-node. After applying the transformation rules Rule1 and Rule2 in lines 21 and 25 respectively, $A$ becomes a child node of another $\otimes$-node of the modified $T$. After applying Rules3 (lines 30 and 34), $A$ becomes a child of a \&-node in the modified $T$, and we go to the state $S_k$. When in $S_k$, we can only apply Rule1 or Rule2 from lines 6 and 12 respectively. Note that after applying these rules, $A$ is still a child of the same \&-node in the modified $T$. When no rule can be applied to $T$ at some state, we go to the accepting state $S_1$ and return false.

Consider the total number of nodes of $T$ enumerated in the loops (lines 4-8, 10-14, and 17-36) of push to construct a maximal $(M \cup \{A\})$-favoring p-skyline relation. Note that when we go from $S_0$ to $S_0$ by applying Rule1 or Rule2, $A$ becomes a descendant of the $\otimes$-node whose child it was originally. Hence, when in $S_0$ we enumerate the nodes $C_t$ to apply Rule1 or Rule2 to, we never pick any $C_t$ which we picked in the previous calls of push. In the process of going from $S_k$ to itself via an application of Rule1 or Rule2, we may enumerate the same node $C_{t+1}$ more than once because A does not change its parent \&-node as a result of these applications. To avoid checking these rules against the same nodes $C_{t+1}$ more than once, one can keep track of the nodes which have already been picked and tested.

The total number of nodes in a syntax tree is $O(|\mathcal{A}|)$, hence the tests $\text{Var}(C_{t+1}) \subseteq M$ (lines 4, 10) and $\text{Var}(C_t) \subseteq M$ (line 17) are performed $O(|\mathcal{A}|)$ times and the rules are applied to the tree $O(|\mathcal{A}|)$ times. Each of the containment tests above requires time $O(|\mathcal{A}|)$, given the bitmap representation of sets. Hence, to compute the syntax tree of a maximal $(M \cup \{A\})$-favoring p-skyline relation, we need time $O(|\mathcal{A}| \cdot |\mathcal{A}|^2)$. Finally, the running time of elicit is $O(|\mathcal{A}| \cdot |\mathcal{A}|^2)$. □

**Theorem 12.** $\text{NEG}$-$\text{SYST}$-$\text{IMPL}$ is co-NP complete

**Proof.** We show that checking the existence of $\tau \in \mathcal{F}_H$ satisfying $\mathcal{N}_0$ but not satisfying $\mathcal{N}_2$ is NP-complete. Clearly, this problem is in NP: we can guess $\tau \in \mathcal{F}_H$ and in polynomial-time check if it satisfies every $\tau \in \mathcal{N}_0$ (i.e., if there is a member of $\mathcal{N}_2$ which has no parent in $\mathcal{L}_2$) but violates some $\tau \in \mathcal{N}_2$. Note that checking if there’s $\tau$ satisfying $\mathcal{N}_0$ but violating $\mathcal{N}_2$ is NP-hard.

Here we show the reduction from SAT. Consider instances of SAT in the following form

$$\phi(x_1, \ldots, x_n) = \phi_1(x_1, \ldots, x_n) \land \ldots \land \phi_m(x_1, \ldots, x_n)$$

where

$$\phi_i(x_1, \ldots, x_n) = \bar{x}_i \lor \ldots \lor \bar{x}_j$$

and $\bar{x}_i \in \{x_i, \bar{x}_i\}$. For every instance $\phi$, we construct

$$\mathcal{A} = \{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n, T, F\}.$$ 

Construct $\mathcal{N}_0$ as follows:

1. for every $\phi_i(x_1, \ldots, x_n) = \bar{x}_1 \lor \ldots \lor \bar{x}_j$, create a constraint $\tau_i^1$ as follows:
   $$\mathcal{L}_i^1 = \{\bar{F}\}$$
   $$\mathcal{R}_i^1 = \{\bar{x}_1, \ldots, \bar{x}_j\}$$
   and
   $$\mathcal{L}_i^1 = \{\bar{F}\}$$
   $$\mathcal{R}_i^1 = \{\bar{x}_1, \ldots, \bar{x}_j\}$$

2. for every variable $x_i$ of $\phi$, create two constraints $\tau_i^2$ and $\tau_i^3$:
   $$\mathcal{L}_i^2 = \{T\}$$
   $$\mathcal{R}_i^2 = \{x_i, \bar{x}_i\}$$
   and
   $$\mathcal{L}_i^3 = \{F\}$$
   $$\mathcal{R}_i^3 = \{x_i, \bar{x}_i\}$$

Now we construct $\mathcal{N}_0$ consisting of a single constraint $\kappa$ as follows.

$$\mathcal{L}_k = \{T, F\}$$
$$\mathcal{R}_k = \{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$$

We prove that there is a satisfying assignment to $\phi$ iff there is a p-graph $\Gamma$ satisfying $\mathcal{N}_0$ and not satisfying $\mathcal{N}_2$. First, assume that there is a satisfying assignment $y = (y_1, \ldots, y_n)$ to $\phi$. We construct the graph $\Gamma$ as follows. For every $i \in [1, n]$:

1. if $y_i = 1$, then $(T, x_i) \in \Gamma$ and $(F, \bar{x}_i) \in \Gamma$
2. if $y_i = 0$, then $(F, x_i) \in \Gamma$ and $(T, \bar{x}_i) \in \Gamma$
3. $\Gamma$ has no other edges.

Clearly, $\Gamma$ satisfies $\text{SPO}$ (every node has either an incoming or outgoing edge, but not both) and $\text{Envelope}$ (every node has at most one incoming edge) and hence is a p-graph. We show that $\Gamma$ satisfies $\mathcal{N}_0$.

1. Consider every constraint $\tau_i^1$ for every $\phi_i(x_1, \ldots, x_n) = \bar{x}_1 \lor \ldots \lor \bar{x}_j$. Since $\Gamma$ satisfies $\phi_i$, at least one of the conjuncts of $\phi_i$ (say, $x_{i_0}$) is 1. If $x_{i_0} = x_i$, then $y_i = 1$, and $(F, x_i) \notin \Gamma$ by construction. If $x_{i_0} = \bar{x}_i$, then $y_i = 0$ and $(F, \bar{x}_i) \notin \Gamma$. Hence, $\tau_i^1$ is satisfied.
2. Consider $\tau_i^2$ and $\tau_i^3$ for every $x_i$. By construction of $\Gamma$, they are satisfied because it cannot be the case that $(T, x_i) \in \Gamma$ and $(T, \bar{x}_i) \in \Gamma$ or $(F, x_i) \in \Gamma$ and $(F, \bar{x}_i) \in \Gamma$. Hence, $\tau_i^2$ and $\tau_i^3$ are satisfied.
Now consider $\mathcal{N}_2$ and the constraint $\kappa$. By construction, for every $i \in [1,n]$, the component $y_i$ of $y$ is set to 0 or 1. Hence, $(T,x_i) \in \Gamma$ and $(F,\overline{x_i}) \in \Gamma$ or $(T,x_i) \in \Gamma$ and $(F,x_i) \in \Gamma$. Therefore, $\kappa$ is violated by $\Gamma$.

Now we show that if $\mathcal{N}_1$ is satisfied by a p-graph $\Gamma$ and $\mathcal{N}_2$ is not, then there is a satisfying assignment $y$ to $\phi$. Take such a p-graph $\Gamma$. We construct $y$ as follows:

$$y_i = \begin{cases} 1 & \text{if } (T,x_i) \in \Gamma \\ 0 & \text{if } (F,x_i) \in \Gamma \end{cases}$$

First, we show that $y_i$ is well defined, i.e., exactly one of the following holds for every $i \in [1,n]$:

(28) $(T,x_i) \in \Gamma$ and $(F,\overline{x_i}) \in \Gamma$ or $(T,x_i) \in \Gamma$ and $(F,x_i) \in \Gamma$ or $(T,x_i) \in \Gamma \lor (F,\overline{x_i}) \in \Gamma$

(29) $(T,x_i) \in \Gamma \land (F,\overline{x_i}) \in \Gamma$

(30) $(T,x_i) \in \Gamma \land (F,\overline{x_i}) \in \Gamma$

Since $\mathcal{N}_1$ is satisfied,

(28) $\forall i \in [1,n] \exists (T,x_i) \notin (T,\overline{x_i}) \notin \Gamma$,

which follows from the satisfaction of $\tau_i^2$, and

(29) $\forall i \in [1,n] \exists (F,x_i) \notin (F,\overline{x_i}) \notin \Gamma$,

which follows from the satisfaction of $\tau_i^1$. Therefore, (27), (28), and (29) imply

(30) $\forall i \in [1,n] \exists (T,\overline{x_i}) \notin (T,x_i) \notin \Gamma$

$\land (F,\overline{x_i}) \notin (F,x_i) \notin \Gamma$

Now we show that $y$ satisfies $\phi$. Since every $\tau_i^1$ is satisfied, at least one of conjuncts of $\phi_i$ (say, $\overline{x_i}$) does not have an incoming edge from $F$. If $\overline{x_i} \in x_i$ (i.e., $(F,x_i) \notin \Gamma$) then by (30) $(T,x_i) \in \Gamma$ and hence $y_i = 1$. Thus $\phi_i$ is satisfied. Similarly, if $\overline{x_i} = \overline{x_i}$ then $(F,x_i) \in \Gamma$ and hence $y_i = 0$. Thus $\phi_i$ is satisfied. Finally, $\phi$ is satisfied. Hence, we proved coNP-completeness of $\text{NEG-SYST-IMPL}$.

\textbf{Theorem 13.} $\text{SUBSET-EQUIV}$ is co-NP complete

\textbf{Proof.} The co-NP-completeness of $\text{SUBSET-EQUIV}$ follows from the co-NP-completeness of $\text{NEG-SYST-IMPL}$. Namely, the membership test is the same as in $\text{NEG-SYST-IMPL}$. To show co-NP-hardness of $\text{SUBSET-EQUIV}$, we reduce from $\text{NEG-SYST-IMPL}$. We use the observation that $\mathcal{N}_1$ implies $\mathcal{N}_2$ iff $\mathcal{N}_1 \cup \mathcal{N}_2$ is equivalent to $\mathcal{N}_1$. \hfill \square