

Top-k Querying of Unknown Values under Order Constraints

Antoine Amarilli¹, Yael Amsterdamer², Tova Milo², and Pierre Senellart^{1,3}

- 1 LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay; Paris, France
first.last@telecom-paristech.fr
- 2 Tel Aviv University; Tel Aviv
{yaelamst,milo}@cs.tau.ac.il
- 3 IPAL, CNRS, National University of Singapore; Singapore

Abstract

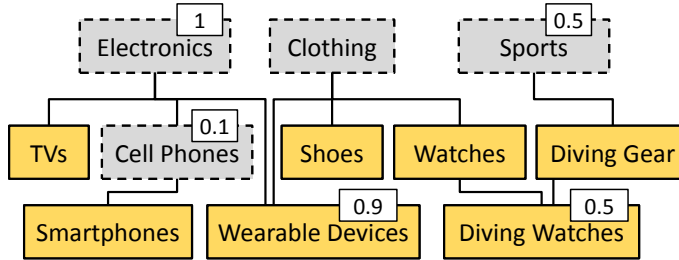
Many practical scenarios make it necessary to evaluate top- k queries over data items with partially unknown values. This paper considers a setting where the values are taken from a numerical domain, and where some *partial order constraints* are given over known and unknown values. Our work is the first to propose a principled scheme to derive the value distributions and expected values of unknown items in this setting, with the goal of computing estimated top- k results by interpolating the unknown values from the known ones. We study the complexity of this general task, and show tight complexity bounds, proving that the problem is intractable; however, we show that the values can be tractably approximated. We then consider the case of tree-shaped partial orders, and show a constructive PTIME solution to compute the exact values. We also compare our problem setting to other top- k definitions on uncertain data.

1 Introduction

Many data analysis tasks involve queries over ordered data, such as maximum and top- k queries, which must often be evaluated in presence of *unknown data values*. This problem occurs in many real-life scenarios: retrieving or computing exact data values is often expensive, but querying the partially unknown data may still be useful to obtain approximate results, or to decide which data values should be retrieved next. Application domains include, e.g., top- k queries over sensor data [18, 26], Web services [33], data mining [1], managing preference data [35], as well as *crowdsourcing* (as we will illustrate). In such contexts, we can often make use of *order constraints* relating the data values, even when they are unknown: for instance, we know that object A will be preferred to object B (though we do not know their exact rating); we know that an event happened before another one (but we do not know exact timestamps), etc.

This paper thus studies the following general problem. We consider a set of numerical values, some of which are unknown, and we assume that we have a *partial order* on these values: a comparability pair $x \geq y$ indicates that the value x (which can be known or unknown) must be greater or equal to y (which may also be unknown). Our goal is to *estimate* the unknown values, in a principled way, and to evaluate *top- k queries*: find the data items with the highest values and compute an estimation of these values when they are unknown.

Example. We consider a specific application setting where our problem occurs. Consider a scenario where products are classified in a catalog taxonomy (Figure 1) using human input:



■ **Figure 1** Sample catalog taxonomy with compatibility scores

for each product, we may ask a question¹ to obtain its *compatibility score* with any category. Our goal is to assign the product to the top- k most compatible categories among a set of *end categories* (in yellow with a solid border), as opposed to *virtual categories* (dashed border). The virtual categories generalize the end categories, and allow us to ask broader questions to experts, but we are not interested by the compatibility of objects with them: e.g., they do not have a dedicated page in our online store.

Imagine now that the product to classify is a *smartwatch*, and that we want to find the top-2 end categories for it. We asked an expert for its compatibility score with some categories (both end and virtual categories), which we indicate in Figure 1. Because expert input is costly, however, we wish to choose the top-2 end categories based on the incomplete information that we have. The naïve answer is to look only at categories with known scores, and to identify *Wearable Devices* and *Diving Watches* as the best end categories.

However, any product that belongs to a specific category (e.g., *Smartphones*) conceptually also belongs to each of the more general categories (e.g., *Cell Phones*). This implies a partial order over the (un)known category scores: if a category x is a sub-category of y , then the product’s compatibility score for x should be at most its score for y .

Under these order constraints, the scores of *Watches* and *Diving Gear*, while unknown, cannot be lower than that of *Diving Watches*; so either of the two could replace *Diving Watches* in the top-2 answer. To choose between these categories, we observe that the score of *Diving Gear* must be exactly 0.5 (which bounds it from above and below). In contrast, as the score of *Wearable Devices* is 0.9, *Clothing* has a score of at least 0.9, so the score of *Watches* can be anything between 0.5 and an unknown value which is ≥ 0.9 . A better top-2 answer is thus *Wearable Devices* and *Watches*, the latter being likely to have a higher score than *Diving Watches* (or *Diving Gear*).

Contributions. Our work introduces a general scheme to reason over partially ordered values which are not fully known, to estimate their values, and to answer top- k queries over them, in a way that generalizes the previous example. We formally define this scheme in Section 2: we consider the convex polytope of the possible value assignments under the constraints (modeled as linear inequalities), and we assume the uniform distribution over the possible *valuations*. We show in Section 3 how this natural model allows us to deduce *expected values* for the unknown variables, and actually (marginal) *probability distributions* for them: they are *not* independent, because of the order constraints, and are generally not uniform. We show that we can compute the expected values, and evaluate top- k queries, with an $\text{FP}^{\#P}$ brute-force algorithm.

¹ The questions may be asked to domain experts or to a crowd [6,30,36] of unqualified users; in the latter case, we aggregate the answers of multiple workers to obtain the compatibility score.

This scheme differs from previous work on top- k queries over incomplete or probabilistic data [10,21,33], which usually assumes that each value is given with its probability distribution, and assumes independence of the distributions (so that they cannot model order constraints between values). Our approach does not need any distribution on the unknown values, and does *not* assume independence: the (dependent and non-uniform) distribution on values are derived from our uniform prior on the assignments. Further, in contrast with prior work, our method allows us to estimate the *values* of the unknown items (not just their ranks), which amounts to performing linear interpolation on all possible total orders weighted by their probability. Thus, we propose (to our knowledge) the *first principled way* to estimate missing numerical values under partial order constraints and known values.

Our second contribution is to study the *complexity* of this task. Via a connection to expected rank computation in partial orders, we show in Section 4 that it is $\text{FP}^{\#\text{P}}$ -hard to compute the expected values, matching the upper bound on our algorithm. We also show a more technical result: it is hard to compute the top-1 item, even without computing its value. However, we also design an FPRAS to approximate the expected values. Further, in Section 5, we show that the problem is tractable when the order constraints are *tree-shaped* in some sense (i.e., in our motivating example, when the taxonomy is a tree rather than a directed acyclic graph). In this case, as we prove, the expected values can be computed in cubic time.

Last, we study how our definition of top- k matches other existing definitions (in Section 6). We survey related work in more depth in Section 7 and conclude in Section 8. Our results are provided with complete proofs which are given as an appendix for lack of space.

2 Preliminaries and Problem Statement

This section introduces the formal definitions for the problem that we study in this paper. We model known and unknown item values as *variables*, and order constraints as *equalities* and *inequalities* over them. Then we define the possible valuations for the variables via possible-world semantics, and use this semantics to define a uniform distribution where all worlds are equally likely. The problem of top- k querying over unknown values can then be formally defined with respect to the expected values of variables in the resulting distribution.

2.1 Unknown Data Values under Constraints

Our input includes a set $\mathcal{X} = \{x_1, \dots, x_n\}$ of variables with unknown values $v(x_1), \dots, v(x_n)$, which we assume² to be in the range $[0, 1]$. We consider two kinds of constraints over them:

- **order constraints**, written $x_i \leq x_j$ for $x_i, x_j \in \mathcal{X}$, encoding that $v(x_i) \leq v(x_j)$;
- **exact-value constraints** to represent variables with known values, written³ $x_i = \alpha$ for $0 \leq \alpha \leq 1$ and for $x_i \in \mathcal{X}$, encoding that $v(x_i) = \alpha$.

In what follows, a *constraint set* with constraints of both types is typically denoted \mathcal{C} . We assume that constraints in \mathcal{C} are *not contradictory* (e.g., we forbid $x = 0.1$, $y = 0.2$, $y \leq z$, and $z \leq x$), and that they are *closed under implication*: e.g., if $x = \alpha$, $y = \beta$ are given, and $\alpha \leq \beta$, then $x \leq y$ is implied and thus should also be in \mathcal{C} . We can check in PTIME that \mathcal{C} is non-contradictory by simply verifying that it does not entail a false inequality on exact

² Our results extend to other bounded, continuous ranges, because we can rescale them to fall in $[0, 1]$.

³ The number α is written as a rational number, represented by its numerator and denominator.

values (e.g., $0.2 \leq 0.1$ as in our previous example). The closure of \mathcal{C} can be found in PTIME as a transitive closure computation [20] that also considers exact-value constraints.

► **Example 1.** In the product classification example from the Introduction, the variable $x_i \in \mathcal{X}$ would represent the compatibility score of the product to the i -th category. If the score is known, we encode it as a constraint $x_i = \alpha$. In addition, \mathcal{C} contains the order constraint $x_i \leq x_j$ whenever category i is a sub-category of j , since by our definition the score of a sub-category cannot be higher than that of an ancestor category.

2.2 Possible World Semantics

The unknown data captured by \mathcal{X} and \mathcal{C} makes infinitely many valuations of \mathcal{X} possible, in addition to the true one, $v(\mathcal{X})$, that we do not know. We model these options via possible world semantics: a *possible world* w for a constraint set \mathcal{C} over $\mathcal{X} = \{x_1, \dots, x_n\}$ is a set of values $w = (v_1, \dots, v_n) \in [0, 1]^n$, intuitively corresponding to setting $v(x_i) := v_i$ for all i , such that all the constraints of \mathcal{C} hold under this valuation. The set of all possible worlds is denoted by $\text{pw}_{\mathcal{X}}(\mathcal{C})$, or simply by $\text{pw}(\mathcal{C})$ when the set of variables is clear from context.

We now notice that \mathcal{C} can be encoded as a set of *linear constraints*, i.e., a set of inequalities between linear expressions on \mathcal{X} and constants in $[0, 1]$. We can encode $x_i \leq x_j$ as $x_j - x_i \geq 0$ and $x_i = \alpha$ can be encoded as $x_i \geq \alpha$ and $x_i \leq \alpha$. The set $\text{pw}(\mathcal{C})$ then contains all valuations of \mathcal{X} into $[0, 1]^n$ for which the constraints hold.

The feasible region of the set of linear constraints \mathcal{C} , namely $\text{pw}(\mathcal{C})$, can be characterized geometrically as a *convex polytope*: writing $n := |\mathcal{X}|$, each linear constraint defines a feasible half-space of \mathbb{R}^n (e.g., the half-space where $x \leq y$), and the convex polytope $\text{pw}(\mathcal{C})$ is the intersection of all half-spaces. For instance, in 2-dimensional space, a convex polytope is a polygon enclosed by line segments such that, for any two points of the polygon, the segment connecting them is within the polygon. In our setting, since the variables only take values in $[0, 1]$, the polytope $\text{pw}(\mathcal{C})$ is bounded within $[0, 1]^n$, and it is non-empty by our assumption that \mathcal{C} is not contradictory. We call it the *admissible polytope*.

However, with exact values constraints, or with order constraints such as $x_i \leq x_j$ and $x_j \leq x_i$ for two variables $x_i, x_j \in \mathcal{X}$, it may be the case that the dimension of this admissible polytope is $< |\mathcal{X}|$. This dimension can be easily computed from \mathcal{C} , as we show:

► **Lemma 2.** *Given a set of order and exact-value constraints \mathcal{C} , we can compute in PTIME the dimension of the admissible polytope.*

► **Example 3.** Let $\mathcal{X} = \{x, y, z\}$. If $\mathcal{C} = \{x \leq y\}$, the admissible polytope has dimension 3 and is bounded by the planes defined by $x = y$, $x = 0$, $y = 1$, $z = 0$ and $z = 1$. If we add to \mathcal{C} the constraint $y = 0.3$, the admissible polytope is a 2-dimensional rectangle bounded by $0 \leq x \leq 0.3$ and $0 \leq z \leq 1$ on the $y = 0.3$ plane. We cannot add, for example, $x = 0.5$, because \mathcal{C} would become contradictory. If we add $y \leq x$, the admissible polytope becomes a 1-dimensional segment $0 \leq z \leq 1$ on the intersection of the planes $x = 0.3$ and $y = 0.3$.

2.3 Probability Distribution

Having characterized the possible worlds of $\text{pw}(\mathcal{C})$, we assume a *uniform* probability distribution over $\text{pw}(\mathcal{C})$, as indicated in the Introduction. This captures the case when all possible worlds are equally likely, and is a natural choice when we have no information about which valuations are more probable.

Since the space of possible worlds is continuous, we formally define this distribution via a probability density function (pdf), as follows. Let \mathcal{X} and \mathcal{C} define a d -dimensional polytope

$\text{pw}_{\mathcal{X}}(\mathcal{C})$ for some integer d . The d -volume (also called the Lebesgue measure [22] on \mathbb{R}^d) is a measure for continuous subsets of d -dimensional space, which coincides with length, area, and volume for dimensions 1, 2, and 3, respectively. We denote by $V_d(\mathcal{C})$ the d -volume of the admissible polytope. We also write $V(\mathcal{C})$ for brevity when d is taken to be the dimension of $\text{pw}(\mathcal{C})$, with \mathcal{X} being also implicit.

► **Definition 4.** The *uniform pdf* p is the function defined on $\text{pw}_{\mathcal{X}}(\mathcal{C})$ by $p(w) := 1/V_d(\mathcal{C})$.

2.4 Top-k Queries

We are now ready to formally define the main problem studied in this paper, namely, the evaluation of *top-k queries* over unknown data values. The queries that we consider retrieve the k items that are estimated to have the highest values, *along with their estimated values*, with ties broken arbitrarily. We further allow queries to apply a *selection operator* σ on the items before performing the top- k computation. In our example from the Introduction, this is what allows us to select the top- k categories only among the end categories. We denote the subset of \mathcal{X} selected by σ as \mathcal{X}_{σ} .

If all item values are known, the semantics of top- k queries is clear. In presence of unknown values, however, the semantics must be redefined to determine how the top- k items and their values are estimated. In this paper, we estimate unknown items by their *expected value over all possible worlds*, i.e., their expected value according to the uniform pdf p defined above on $\text{pw}(\mathcal{C})$. This corresponds to *interpolating* the unknown values from the known ones, and then querying the result. We use these interpolated values to define the top- k problem as computing the k variables with the highest expected values, but we also study on its own the interpolation problem of computing the expected values.

To summarize, the two formal problems that we study on constraint sets are:

Interpolation. Given a constraint set \mathcal{C} over \mathcal{X} and variable $x \in \mathcal{X}$, the *interpolation problem* for x is to compute the expected value of x in the uniform distribution over $\text{pw}_{\mathcal{X}}(\mathcal{C})$.

Top- k . Given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , and an integer k , the *top- k computation problem* is to compute the ordered list of the k maximal expected values of variables in \mathcal{X}_{σ} (or less if $|\mathcal{X}_{\sigma}| \leq k$), with ties broken arbitrarily.

We compare our choice of definition with other definitions of top- k on uncertain data [10] in Section 6, where we justify our choice of semantics.

3 Analysis of the General Case

We start by studying the general problem and investigate how the uniform distribution, defined jointly over the variables of \mathcal{X} , affects the individual behavior of each variable, i.e., their *marginal distribution*. To this end, we design an algorithm that computes the expected value of variables.

The algorithm is *brute-force* because it enumerates all the possible orderings of the variables (to be defined formally below), but it is still nontrivial: we must handle exact value constraints specifically, and we must compute the probability of each ordering to determine its weight in the overall expected value computation. Based on this algorithm, we then establish complexity upper bounds for our two problems of interpolation and top- k computation.

We obtain our bounds through the more general problem of computing the *marginal distribution* of variables, that we now define. Letting \mathcal{C} be a constraint set, and $x \in \mathcal{X}$ a variable, for a value $v \in [0, 1]$, we write $\mathcal{C}_{|x=v}$ the *marginalized* constraint set $\mathcal{C} \cup \{x = v\}$. Assuming the uniform distribution on the polytope $\text{pw}(\mathcal{C})$, we can define:

► **Definition 5.** The *marginal distribution* of x is defined as the pdf $p_x : v \mapsto V(\mathcal{C}_{|x=v})/V(\mathcal{C})$.

We will show how marginal distributions can be computed in our setting. To simplify our study, we will eliminate from the start the problem of *ties*. We say that a possible world $w = (v_1, \dots, v_n)$ of a constraint set \mathcal{C} has a *tie* if $v_i = v_j$ for some i, j . Intuitively, the only situation where ties have non-zero probability is when they are enforced by \mathcal{C} and hold in *every* possible world. In such situations, we can rewrite \mathcal{C} by merging these variables to obtain an equivalent constraint set where ties have probability 0. Formally:

► **Proposition 6.** *For any constraint set \mathcal{C} , we can construct in PTIME a constraint set \mathcal{C}' such that the probability that the possible worlds of \mathcal{C}' have a tie (under the uniform distribution) is zero, and such that any interpolation or top-k computation problem on \mathcal{C} can be reduced in PTIME to the same type of problem on \mathcal{C}' .*

Hence, we assume from now on that ties have zero probability in \mathcal{C} . Note that this implies that all of our results also hold for *strict inequality constraints*, of the forms $x < y$ and $x \neq y$.

Under this assumption, we first study in Section 3.1 the case where \mathcal{C} is a total order. We then show in Section 3.2 how to handle arbitrary \mathcal{C} by summing over the linear extensions.

3.1 Total Orders

We assume in this section that \mathcal{C} is a total order $\mathcal{C}_1^n(\alpha, \beta)$ defined as $\alpha \leq x_1 \leq \dots \leq x_n \leq \beta$, where α and β are variables with an exact-value constraint: abusing notation we also denote the corresponding values as α (which is ≥ 0) and β (which is ≤ 1). We first assume that there are no other exact value constraints.

For $1 \leq i \leq n$, we want to determine the marginal distribution and expected value of x_i . This relates to the known problem of computing *order statistics*:

► **Definition 7.** The *i-th order statistic* for n samples of a probability distribution Pr is the distribution Pr_i of the following process: draw n independent values according to Pr , and return the i -th smallest value of the draw.

► **Proposition 8 ([16], p. 63).** *The i-th order statistic for n samples of the uniform distribution on $[0, 1]$ is the Beta distribution $B(i, n + 1 - i)$.*

The connection to the marginal distribution of x_i in $\mathcal{C}_1^n(\alpha, \beta)$ is the following:

► **Observation 9.** *The marginal distribution of x_i within the admissible polytope of $\mathcal{C}_1^n(\alpha, \beta)$ is the distribution of the i-th order statistic for n samples of the uniform distribution on $[\alpha, \beta]$.*

We thus obtain that the expected value of x_i in $\mathcal{C}_1^n(\alpha, \beta)$ is the mean of the Beta distribution $B(i, n + 1 - i)$ scaled and shifted to $[\alpha, \beta]$, namely: $\alpha + \frac{i(\beta - \alpha)}{n + 1}$. Note that this corresponds to a linear interpolation of the unknown variables between α and β . The pdf of the marginal distribution $p_{x_i} : [\alpha, \beta] \rightarrow [0, 1]$ is a polynomial derived from the expression of the Beta distribution. The following proposition summarizes our findings:

► **Proposition 10.** *Given $\mathcal{C}_1^n(\alpha, \beta)$, i.e., a constraint set implying a total order bounded by α and β , the expected value and the marginal distribution of any variable x_i can be computed in PTIME, and the marginal distribution is a polynomial of degree at most n .*

Exact-Value Constraints. In the case where \mathcal{C} is allowed to contain more exact-value constraints than the ones on α and β , we observe that we can *split* the total order into sub-sequences of variables with no exact-value constraints, and then compute the expected values and marginal distributions of each sub-sequence independently.

The soundness of splitting can be proved via the possible world semantics: assume that there is an exact-value constraint on x_i . Then the possible valuations of x_1, \dots, x_{i-1} are affected only by the constraints in \mathcal{C} on x_1, \dots, x_i , and similarly for x_{i+1}, \dots, x_n . The possible worlds $\text{pw}(\mathcal{C})$ are then equivalent to the concatenation of every possible valuation of x_1, \dots, x_{i-1}, v for x_i , and every possible valuation of x_{i+1}, \dots, x_n . Formally:

► **Lemma 11.** *Let $\mathcal{C}_1^n(\alpha, \beta)$ be the constraint set on $\mathcal{X} := \{x_1, \dots, x_n\}$ defined as in the previous section, let $\mathcal{C}^=$ be a non-empty set of exact-value constraints on \mathcal{X} , and let $x_i = v$ be a constraint of $\mathcal{C}^=$. Write $\mathcal{X}_{1, \dots, i-1} := \{x_1, \dots, x_{i-1}\}$, $\mathcal{X}_{i+1, \dots, n} := \{x_{i+1}, \dots, x_n\}$, and $\mathcal{C}_{1, \dots, i-1}^=$ (resp., $\mathcal{C}_{i+1, \dots, n}^=$) the subset of $\mathcal{C}^=$ on $\mathcal{X}_{1, \dots, i-1}$ (resp., $\mathcal{X}_{i+1, \dots, n}$). Then we have:*

$$\text{pw}_{\mathcal{X}}(\mathcal{C}_1^n(\alpha, \beta) \cup \mathcal{C}^=) = \text{pw}_{\mathcal{X}_{1, \dots, i-1}}(\mathcal{C}_1^i(\alpha, v) \cup \mathcal{C}_{1, \dots, i-1}^=) \times \{v\} \times \text{pw}_{\mathcal{X}_{i+1, \dots, n}}(\mathcal{C}_{i+1}^n(v, \beta) \cup \mathcal{C}_{i+1, \dots, n}^=)$$

Hence, given a constraint set \mathcal{C} imposing a total order and possibly exact-value constraints, the expected value of x_i can be computed as follows. If x_i has an exact-value constraint, its marginal distribution and expected value are trivial. Otherwise, we consider the total order $\mathcal{C}_{p+1}^{q-1}(v_p, v_q)$, where p is the maximal index such that $0 \leq p < i$ and x_p has an exact-value constraint (where x_0 is the leftmost variable α and $v_0 = \alpha$). Similarly, q is the minimal index such that $i < q \leq n+1$ and x_q has an exact-value constraint, where x_{n+1} is β and $v_{n+1} = \beta$. The expected value and marginal distribution of x_i can be computed using the expression of the mean and pdf of Beta distributions for $\mathcal{C}_{p+1}^{q-1}(v_p, v_q)$.

3.2 General Constraint Sets

We can now extend the result for total orders to an expression of the expected value and marginal distribution for a general constraint set \mathcal{C} . We will focus on expected values, and return to the marginal distributions at the end of the section. To compute expected values, we apply the previous process to each possible total ordering of the variables. To do this, we define the notion of *linear extensions*, inspired by partial order theory:

► **Definition 12.** Given a constraint set \mathcal{C} over \mathcal{X} , we say that a constraint set \mathcal{T} is a *linear extension* of \mathcal{C} if (i) \mathcal{T} is a total order; (ii) the exact-value constraints of \mathcal{T} are exactly those of \mathcal{C} ; and (iii) $\mathcal{C} \subseteq \mathcal{T}$, namely every constraint $x \leq y$ in \mathcal{C} also holds in \mathcal{T} .⁴

We now partition the possible worlds of $\text{pw}(\mathcal{C})$ that have no ties, according to the linear extension of \mathcal{C} which is realized by their values. Worlds with ties can be neglected as having zero probability, as we assumed using Proposition 6. Algorithm 1 presents this general scheme to compute the expected value of a variable $x \in \mathcal{X}$ under an arbitrary constraint set \mathcal{C} assuming the uniform distribution on $\text{pw}(\mathcal{C})$. For each linear extension \mathcal{T} of \mathcal{C} , we compute the expected value of x in \mathcal{T} and the overall probability of \mathcal{T} in $\text{pw}(\mathcal{C})$. Since a linear extension is a total order, the *expected value of x relative to \mathcal{T}* , denoted by $\mathbb{E}_{\mathcal{T}_i}[x]$, is computed at line 7 as explained at the end of Section 3.1: perform linear interpolation within the total order of the contiguous unknown variables in \mathcal{T} that contain x . As for the

⁴ The linear extensions of \mathcal{C} in this sense are thus exactly the linear extensions of the partial order on \mathcal{X} imposed by \mathcal{C} : this partial order is indeed antisymmetric because \mathcal{C} has no ties.

Algorithm 1: Compute a variable's expected value

Input: Constraint set \mathcal{C} on variables \mathcal{X} with $n := |\mathcal{X}|$, variable $x \in \mathcal{X}$
Output: Expected value of x

- 1 **if** x has an exact value constraint to some value v in \mathcal{C} **then return** v ;
- 2 $\mathbb{E}_{\mathcal{C}}[x] \leftarrow 0$; $V(\mathcal{C}) \leftarrow 0$;
- 3 **foreach** linear extension \mathcal{T} of \mathcal{C} **do**
- 4 Write \mathcal{T} as $y_1 \leq y_2 \leq \dots \leq y_n$ and $y_{i_1} = v_1, \dots, y_{i_m} = v_m$ with $i_1 < \dots < i_m$;
- 5 Add $\{y_0 \leq y_1, y_n \leq y_{n+1}, y_0 = 0, y_{n+1} = 1\}$ to \mathcal{T} on fresh variables y_0 and y_{n+1} ;
- 6 Set $i_0 \leftarrow 0, i_{m+1} \leftarrow n+1, v_0 \leftarrow 0, v_{m+1} \leftarrow 1$;
- 7 $\mathbb{E}_{\mathcal{T}}[x] \leftarrow v_j + k \times \frac{v_{j+1} - v_j}{i_{j+1} - i_j}$ with $0 \leq j \leq m$ and $0 < k < i_{j+1} - i_j$ s.t. x is y_{i_j+k} ;
- 8 $V(\mathcal{T}) \leftarrow \prod_{j=0}^m \frac{(v_{j+1} - v_j)^{i_{j+1} - i_j - 1}}{(i_{j+1} - i_j - 1)!}$;
- 9 $V(\mathcal{C}) \leftarrow V(\mathcal{C}) + V(\mathcal{T})$;
- 10 $\mathbb{E}_{\mathcal{C}}[x] \leftarrow \mathbb{E}_{\mathcal{C}}[x] + V(\mathcal{T}) \times \mathbb{E}_{\mathcal{T}}[x]$;
- 11 **return** $\frac{\mathbb{E}_{\mathcal{C}}[x]}{V(\mathcal{C})}$;

probability of \mathcal{T} , we compute it as the volume of \mathcal{T} (line 8): following Lemma 11, we can compute it as a product of the volume of constraint sets of the form $\mathcal{C}_{p+1}^{q-1}(\alpha, \beta)$, where it is $\frac{(\beta - \alpha)^{(q-p-1)}}{(q-p-1)!}$ because the overall volume is split evenly among the $(q-p-1)!$ possible orderings.

Finally, the total volume of $\text{pw}(\mathcal{C})$ is computed as the sum of volumes of the linear extensions \mathcal{T} (line 9), and the overall expected value $\mathbb{E}_{\mathcal{C}}[x]$ of x in \mathcal{C} is the sum of expected values for each \mathcal{T} weighted by their volume (line 10) and normalized by the total volume (line 11). Hence, Algorithm 1 correctly computes the expected value of x .

The complexity of Algorithm 1 is polynomial in the number of linear extensions of \mathcal{C} , as we can enumerate them in constant amortized delay [31]). However, in the general case, there may be up to $|\mathcal{X}|!$ linear extensions. To obtain an upper bound in the general case, we note that we can rescale all constraints so that all numbers are integers, and then nondeterministically sum over the linear extensions. This yields an $\text{FP}^{\#P}$ upper bound:

► **Theorem 13.** *Given a constraint set \mathcal{C} over \mathcal{X} and $x \in \mathcal{X}$, determining the expected value of x in $\text{pw}(\mathcal{C})$ under the uniform distribution is in $\text{FP}^{\#P}$.*

This $\text{FP}^{\#P}$ membership result (proved in the Appendix) is not immediate: we compute the volume of the polytope $\text{pw}(\mathcal{C})$ in $\text{FP}^{\#P}$, but this task is not in $\text{FP}^{\#P}$ for general convex polytopes [24], so our algorithm relies on the fact that $\text{pw}(\mathcal{C})$ is defined from order constraints.

Given the expected value of each variable in \mathcal{X}_{σ} , retrieving the top- k such values can naturally be done in PTIME, and thus the following is an immediate corollary of Theorem 13.

► **Corollary 14.** *Given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , and an integer k , the top- k computation problem over \mathcal{X} , \mathcal{C} , and σ is in $\text{FP}^{\#P}$.*

We will show in Section 4 that this $\text{FP}^{\#P}$ upper bound is tight, even if we just wish to compute the top-1 variable without its expected value.

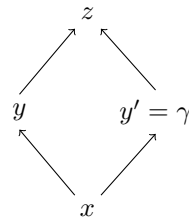
We close the section with three concluding observations. First, we observe that Algorithm 1 generalizes to the computation of the *marginal distributions* of the variables. Second, we note that the expected values computed by Algorithm 1 relate to the *center of mass* of the polytope $\text{pw}(\mathcal{C})$. Third, we provide a complete example to illustrate the constructions of this section.

Marginal Distributions. We note that Algorithm 1 can be adapted to compute the *marginal distribution* of variable x instead of its expected value, so that our results generalize to other tasks, such as computing other moments of the variables, their median value, etc. To do so, let $v_0 = 0 < v_1 < \dots < v_m < v_{m+1} = 1$ be the different values that occur in exact value constraints of \mathcal{C} . We adapt Algorithm 1 by replacing line 7 to compute instead the polynomial marginal distribution of x in \mathcal{T} on variables $x_{i_j} \dots x_{i_{j+1}}$: according to Proposition 10, this marginal distribution is polynomial and defined on the range $[v_j, v_{j+1}]$. We modify line 10 to compute the sum of the marginal distributions instead, seeing them as *piecewise polynomial* functions (which are zero outside of their range), and still weighted by the volume of \mathcal{T} . We deduce that the marginal distribution of x in \mathcal{C} (computed by modifying line 11) is a *piecewise polynomial* function with at most $|\mathcal{X}|$ pieces and degree at most $|\mathcal{X}|$.

Center of Mass. We can show that the expected values computed by Algorithm 1 coincide with the *center of mass* of the admissible polytope. The *center of mass* of a polytope is the point G such that all vectors relative to G originating at points within the polytope sum to zero: if the polytope is convex, G is located within the polytope. We show:

► **Proposition 15.** *For any constraint set \mathcal{C} on \mathcal{X} , if \bar{x}_i is the expected value of x_i for all $x_i \in \mathcal{X}$, then $\bar{\mathbf{x}}$ is the center of mass of the admissible polytope $\text{pw}(\mathcal{C})$.*

Full Example. We exemplify our scheme on variables $\mathcal{X} = \{x, y, y', z\}$ and on the constraint set \mathcal{C} generated by the order constraints $x \leq y$, $y \leq z$, $x \leq y'$, $y' \leq z$ and the exact-value constraint $y' = \gamma$ for some fixed $0 < \gamma < 1$. Remember that we necessarily have $0 \leq x$ and $z \leq 1$ as well. The constraints of \mathcal{C} are closed under implication, so they also include $x \leq z$. The figure shows the Hasse diagram of the partial order defined by \mathcal{C} on \mathcal{X} . Note that ties have a probability of zero in $\text{pw}(\mathcal{C})$.



The two linear extensions of \mathcal{C} are $\mathcal{T}_1 : x \leq y \leq y' \leq z$ and $\mathcal{T}_2 : x \leq y' \leq y \leq z$. By Lemma 11, we have $\text{pw}(\mathcal{T}_1) = \text{pw}_{\{x,y\}}(\mathcal{C}') \times \{\gamma\} \times [\gamma, 1]$ where \mathcal{C}' is defined on variables $\{x, y\}$ by $0 \leq x \leq y \leq \gamma$. We can compute the volume of $\text{pw}(\mathcal{T}_1)$ as $\alpha_1 = \frac{\gamma^2}{2} \times (1 - \gamma)$, and similarly the volume of $\text{pw}(\mathcal{T}_2)$ is $\alpha_2 = \gamma \times \frac{(1-\gamma)^2}{2}$.

Let us compute the marginal distribution and expected value of y for \mathcal{C} . We do so by computing the marginal distribution of y for \mathcal{T}_1 and \mathcal{T}_2 . We first study \mathcal{T}_1 , where by Lemma 11 it suffices to determine the marginal distribution of y for \mathcal{C}' : we compute by Proposition 10 that it has pdf $f_1 : t \mapsto \frac{2}{\gamma} \times \frac{t}{\gamma}$ for $t \in [0, \gamma]$. Its expected value is $\mu_1 := \frac{2}{3}\gamma$. For \mathcal{T}_2 we determine in the same way that the pdf of y is $f_2 : t \mapsto \frac{2}{1-\gamma} \times \frac{1-t}{1-\gamma}$ for $t \in [\gamma, 1]$ which has expected value $\mu_2 := \frac{1+2\gamma}{3}$. We deduce that the marginal distribution of y for \mathcal{C} has pdf $\alpha_1 f_1$ on $[0, \gamma]$ and $\alpha_2 f_2$ on $[\gamma, 1]$, and that its expected value is $\mu = \frac{\alpha_1 \mu_1 + \alpha_2 \mu_2}{\alpha_1 + \alpha_2}$.

4 Hardness and Approximations

We next show that the intractability of Algorithm 1 in Section 3 is probably unavoidable, by proving matching lower bounds for interpolation and top- k in Section 4.1. However, we show in Section 4.2 that it is tractable to approximate expected values. We will show in the next section that exact computation for our problems can be tractable under some assumptions on the input constraint sets.

4.1 Hardness of Exact Computation

We now analyze the complexity of computing an exact solution to our two main problems.

Interpolation. In the previous section, we have shown that the interpolation problem is in $\text{FP}^{\#P}$ (Theorem 13). We now show a matching lower bound, implying that the problem is $\text{FP}^{\#P}$ -complete.

► **Theorem 16.** *Given a set \mathcal{C} of order constraints with variable set \mathcal{X} and $x \in \mathcal{X}$, determining the expected value of x in $\text{pw}(\mathcal{C})$ under the uniform distribution is $\text{FP}^{\#P}$ -hard.*

Proof sketch. This follows from the known hardness of computing the *expected rank* of an element in a partial order without exact values [7]. See the Appendix for the full proof. ◀

Top-k computation. If top- k queries must return the expected values of their output, then the previous interpolation result immediately implies the hardness of top- k . However, we can show a more interesting result: computing a top- k (or top-1) query is still $\text{FP}^{\#P}$ -hard, even without returning the expected value.

► **Theorem 17.** *Given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , and an integer k , the top- k computation problem over \mathcal{X} , \mathcal{C} and σ is $\text{FP}^{\#P}$ -hard even if k is fixed to be 1, $|\mathcal{X}_\sigma|$ is 2, and the top- k answer does not include the expected value of the variables.*

Proof sketch. To prove hardness in this case, we reduce from interpolation. We show that a top-1 computation oracle can be used as a comparison oracle to compare the expected value of a variable x to any other rational value α , by adding a fresh element with an exact-value constraint to α . What is more technical is to show that, given such a comparison oracle, we can perform the reduction and determine *exactly* the expected value v of x (a rational number) using only a *polynomial* number of comparisons to other rationals. This follows from a bound on the denominator of v , and by applying the rational number identification scheme of [29]. See the Appendix for full details. ◀

A simple corollary of Theorem 17 is then the intractability of the top- k *decision* problem, that we show by reducing from the top- k *computation* problem that we previously studied:

► **Corollary 18.** *The top- k decision problem of deciding, given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , an integer k , and a set $X \subseteq \mathcal{X}$ of size k , whether the k items of \mathcal{X}_σ that have maximal expected values are exactly those of X , is NP-hard.*

4.2 Complexity of Approximate Computation

In light of the previous hardness results, we now consider approximation algorithms.

Interpolation. First, we show that the *interpolation* problem admits a fully polynomial-time randomized approximation scheme (FPRAS):

► **Proposition 19.** *Let \mathcal{C} be a set of constraints with variable set \mathcal{X} and $x \in \mathcal{X}$. There is an FPRAS that determines an estimate $\hat{\mathbb{E}}_{\mathcal{C}}[x]$ of the expected value $\mathbb{E}_{\mathcal{C}}[x]$ of x in $\text{pw}(\mathcal{C})$ under the uniform distribution.*

Proof sketch. This relies on the characterization of the set of possible worlds in our setting as a convex polytope: we use the algorithm by Kannan, Lovász, and Simonovits to tractably sample from a convex body [23]. The sampling is *almost uniform* in the sense that it has a bounded *total variation distance* to the uniform distribution, which we can use to bound the error on the expected value caused by this non-uniformity; we combine this with a bound on the error from the sampling itself, obtained from Hoeffding’s inequality. See Appendix. ◀

This result is mostly of theoretical interest, as the polynomial is in $|\mathcal{X}|^7$, but recent theoretical improvements on the underlying sampling algorithm [28] may ultimately yield a practical approximate interpolation technique for general constraint sets (see [15, 27]).

Top- k computation. We now study approximations for *top- k computation*. There are two natural ways to define randomized approximations for this problem:

- We can define the *approximate top- k* as an ordered list of k items whose expected value does not differ by more than some $\epsilon > 0$ from that of the item in the actual top- k at the same rank. An FPRAS for this definition of approximate top- k , with the same reserve on practicality, can be obtained from that of Proposition 19. Indeed, if we approximate within ϵ the expected value of each item in \mathcal{X}_σ with probability $\geq (1 - \delta)^{\frac{1}{|\mathcal{X}_\sigma|}}$, the list of top- k results will be an ϵ -approximation of the actual top- k with probability $\geq 1 - \delta$.
- We can look for a PTIME algorithm that returns the *actual top- k* with high probability. However, if we could decide the top- k problem with high probability in PTIME, then this problem would be in the BPP (bounded-error probabilistic time) complexity class. However, as it is NP-hard by Corollary 18, we would have $\text{NP} \subseteq \text{BPP}$, so that the polynomial hierarchy collapses at the second level (Theorem 10 of [39]): this is open but considered to be highly unlikely.

5 Tractable Cases

Given the hardness results in the previous section and the impracticality of approximation, we now study whether *exact* interpolation and top- k computation can be tractable on restricted classes of constraint sets. We consider tree-shaped constraints (defined formally below) and generalizations thereof: they are relevant for practical applications (e.g., classifying items into tree- or forest-shaped taxonomies), and we will show that our problems are tractable on them (we will even compute marginal distributions, as in Section 3). We start by a splitting lemma that generalizes Lemma 11, and then define and study our tractable class.

5.1 Splitting Lemma

We will formalize the cases in which the valuations of two variables in \mathcal{X} are probabilistically dependent (the variables *influence* each other), according to \mathcal{C} . This, in turn, will enable us to define *independent subsets of the variables* and thus *independent subsets of the constraints* over these variables. In what follows, we use $x_i \prec x_j$ to denote the *covering relation* of the partial order \leq , i.e., $x_i \leq x_j$ is in \mathcal{C} but there exists no $x_k \notin \{x_i, x_j\}$ such that $x_i \leq x_k$ and $x_k \leq x_j$ are in \mathcal{C} .

► **Definition 20.** We say that x *influences* x' , written $x \rightarrow x'$, if there is a path $x_1 \prec \dots \prec x_k$ with $x_1 = x$, $x_k = x'$, such that none of the x_i has exact-value constraints.

The *influence relation* $x \leftrightarrow x'$ is the symmetric, reflexive, and transitive closure of \rightarrow .

The *uninfluenced classes* of \mathcal{X} under \mathcal{C} are the classes of the equivalence relation \leftrightarrow consisting of variables with no exact-value constraints, and singleton classes for each variable with an exact-value constraint.

Intuitively, the uninfluenced classes of \mathcal{X} under \mathcal{C} are just the connected components of the relation “ x and y co-occur in some order constraint” (forgetting about the direction of the order) when the variables with an exact value constraint are removed. We write $\mathcal{C}_1, \dots, \mathcal{C}_m$ for the uninfluenced classes of \mathcal{X} under \mathcal{C} . We call the *uninfluence decomposition* the set $\mathcal{C}_1, \dots, \mathcal{C}_m$ of subsets of \mathcal{C} , such that \mathcal{C}_i contains the (implication closure of) constraints on the variables of \mathcal{C}_i . We can show that the possible worlds of \mathcal{C} can be decomposed following the uninfluence decomposition:

► **Lemma 21.** *There exists a bijective correspondence between $\text{pw}(\mathcal{C}_1) \times \cdots \times \text{pw}(\mathcal{C}_m)$ and $\text{pw}(\mathcal{C})$, obtained by merging the variables with exact-value constraints.*

Note that this lemma subsumes Lemma 11 (and generalizes it beyond total orders). We will use it to analyse restricted classes of constraint sets in the next section.

5.2 Tree-Shaped Constraints

We define the first restricted class of constraints that we consider: *tree-shaped* constraints. Recall that a *Hasse diagram* is a representation of a partial order as a directed acyclic graph, whose nodes correspond to \mathcal{X} and there is an edge (x, y) if $x \prec y$ (used, e.g., in the diagram in Section 3.2).

► **Definition 22.** A constraint set \mathcal{C} over \mathcal{X} is *tree-shaped* if the probability of ties is zero, the Hasse diagram of the partial order induced on \mathcal{X} by \mathcal{C} is a directed tree, the root has exactly one child, and exactly the root and leaves have exact-value constraints. \mathcal{C} thus imposes a global minimal value, and maximal values at each leaf, and no other exact-value constraint.

We call \mathcal{C} *reverse-tree-shaped* if the reverse of the Hasse diagram (obtained by reversing the direction of the edges) satisfies the requirements of being tree-shaped.

► **Example 23.** Consider the taxonomy of Figure 1 (without the indicated compatibility scores). Remove *Wearable Devices* and *Diving Watches* and add dummy variables with exact-value constraints to 0 and 1 as respectively the leaves and root, to materialize the fact that all variables are assumed to be ≥ 0 and ≤ 1 . The resulting monotonicity constraints can be expressed as a reverse-tree-shaped constraint set.

We now show that for a tree-shaped constraint set \mathcal{C} , unlike in the general case, we can tractably compute exact expressions of the marginal distributions and expected values of variables. In the next two results, we assume arithmetic operations on rationals to have unit cost, e.g., they are performed up to a fixed numerical precision. Otherwise, the complexities remain polynomial but the degrees may be larger.

► **Theorem 24.** *For any tree-shaped constraint set \mathcal{C} over \mathcal{X} , we can compute $V(\mathcal{C})$ in time $O(|\mathcal{X}|^2)$.*

Proof sketch. We process the tree bottom-up, propagating a piecewise polynomial function expressing the volume of the subpolytope on the subtree rooted at each node as a function of the value of the parent node: we compute it using Lemma 21 from the child nodes. ◀

See Appendix for the complete proof. This result can be applied to prove the tractability of computing marginal probabilities in a tree-shaped constraint set:

► **Theorem 25.** *For any tree-shaped constraint set \mathcal{C} on variable set \mathcal{X} and variable $x \in \mathcal{X}$ with no exact-value constraint, the marginal distribution for x is piecewise polynomial and can be computed in time $O(|\mathcal{X}|^3)$.*

Proof sketch. We proceed similarly to the proof of Theorem 24 but with two functions: one for x and its descendants, and one for all other nodes. The additional factor in $|\mathcal{X}|$ is because the second function depends on how the value given to x compares to the tree leaves. ◀

We last note that our results on tree-shaped constraint sets extend to reverse-tree-shaped constraint sets: any reverse-tree-shaped constraint set \mathcal{C} can be transformed to a tree-shaped constraint set \mathcal{C}' such that $\mathbb{E}_{\mathcal{C}'}(x) = 1 - \mathbb{E}_{\mathcal{C}}(x)$ for every $x \in \mathcal{X}$, by reversing order

constraints and replacing exact-value constraints $x = \alpha$ with $x = 1 - \alpha$. Further, if \mathcal{C} is not (reverse-)tree-shaped but each \mathcal{C}_i in its uninfluence decomposition $\mathcal{C}_1, \dots, \mathcal{C}_m$ is, then we can compute the expected and top- k values in \mathcal{C} from each \mathcal{C}_i using Lemma 21. We deduce:

► **Corollary 26.** *The top- k and interpolation problems can be solved in PTIME for constraint sets whose uninfluence decomposition contains only tree-shaped or reverse-tree-shaped constraint sets.*

6 Other Variants

We have defined top- k computation on constraint sets by considering the *expected value* of each variable under the uniform distribution. However, many other definitions of top- k on unknown values have been studied in previous work. We now compare them to our definition, which we call *local-top- k* , for the constraint sets that we study.

U-top- k . The *U-top- k* variant does not study *individual* variables but defines the top- k as a choice between variable *sequences*, namely, the ordered sequence of k variables with the highest probability of being the k largest variables (in decreasing order) among those of \mathcal{X}_σ , for the uniform distribution on $\text{pw}(\mathcal{C})$. We call this alternative definition *U-top- k* by analogy with [10, 34]. Note that U-top- k does not give a score *per variable*, but instead considers the global probability that the k variables returned are indeed the top- k . We show that the U-top- k and local-top- k definitions sometimes disagree in our setting:

► **Lemma 27.** *There is a constraint set \mathcal{C} and selection predicate σ such that local-top- k and U-top- k do not match, even for $k = 1$ and without returning expected values or probabilities.*

We can easily design an algorithm to compute U-top- k in PSPACE and in polynomial time in the number of linear extensions of \mathcal{C} : compute the probability of each linear extension as in Algorithm 1, and then sum on linear extensions depending on which top- k sequence they realize (on the variables selected by σ), to obtain the probability of each answer. Hence:

► **Proposition 28.** *For any constraint set \mathcal{C} over \mathcal{X} , integer k and selection predicate σ , the U-top- k query for \mathcal{C} and σ can be computed in PSPACE and in time $O(\text{poly}(N))$, where N is the number of linear extensions of \mathcal{C} .*

Unlike Theorem 13, however, this does not imply $\text{FP}^{\#\text{P}}$ -membership: when selecting the most probable sequence, the number of candidate sequences may not be polynomial (as k is not fixed). We leave to future work an investigation of the precise complexity of U-top- k .

We can also show that, in our setting, U-top- k does not satisfy the *containment property* of [10]. We define the containment property as follows, taking variable order into account:

► **Definition 29.** A top- k definition satisfies the *containment property* if for any constraint set \mathcal{C} on variables \mathcal{X} , for any predicate σ (where we write \mathcal{X}_σ the selected variables), and for any $k < |\mathcal{X}_\sigma|$, letting S_k and S_{k+1} be the top- k and top- $(k+1)$ variables (i.e., without scores), S_k is a strict prefix of S_{k+1} .

The containment property is a natural desideratum: computing the top- k for some $k \in \mathbb{N}$ should not give different variables or a different order for the top- k' with $k' < k$. Our local-top- k clearly satisfies the containment property by definition, except in the case of ties. However:

► **Lemma 30.** *There is a constraint set \mathcal{C} without ties such that U-top- k does not satisfy the containment property for the uniform distribution on $\text{pw}(\mathcal{C})$.*

We see this as a drawback of U-top- k compared to our local-top- k definition.

Global-top- k . We now study the *global-top- k* definition [40], and show that it does not respect the containment property either, even though it is defined on individual variables:

► **Definition 31.** The *global-top- k query*, for a constraint set \mathcal{C} , selection predicate σ , and integer k , returns the k variables that have the highest probability in the uniform distribution on $\text{pw}(\mathcal{C})$ to be among the variables with the k highest values, sorted by decreasing probability.

► **Lemma 32.** *There is a constraint set \mathcal{C} without ties such that global-top- k does not satisfy the containment property for the uniform distribution on $\text{pw}(\mathcal{C})$.*

Other Variants. Additional variants of top- k have been studied, see [10, 40]. However, in the context of [10], these definitions do not satisfy the containment property either, except for two. The first, U-kRanks [34], does not satisfy the natural property that top- k answers always contain k different variables. The second, expected ranks [10], resembles local-top- k but uses ranks instead of values, so the definition is *value-independent*. While this makes sense for top- k queries designed to return tuples, as in [10], we argue it is less sensible when focusing on the numerical value of variables; this justifies our focus on local-top- k .

7 Related Work

We extend the discussion about related work from the Introduction.

Ranking queries over uncertain databases. A vast body of work has focused on providing semantics and evaluation methods for order queries over uncertain databases, including top- k and ranking queries (e.g., [10, 12, 18, 19, 21, 25, 32, 33, 37, 38]). Such works consider two main uncertainty types: *tuple-level uncertainty*, where the existence of tuples (i.e., variables) is uncertain, and hence affects the query results [10, 12, 19, 21, 25, 32, 37, 38]; and *attribute-level uncertainty*, more relevant to our problem, where the data tuples are known but some of their values are unknown or uncertain [10, 18, 21, 33]. Top- k queries over uncertain data following [33] was recently applied to crowdsourcing applications in [8]. These studies are relevant to our work as they identify multiple possible semantics for order queries in presence of uncertainty, and specify desired properties for such semantics [10, 21]; our definition of top- k satisfies the desiderata that are relevant to attribute-level uncertainty [21].

We depart from this existing work in two main respects. First, existing work assumes that each variable is given with an *independent function* that describes its probability distribution. We do not assume this, and instead *derive* non-independent marginal distributions for the variables in a principled way from a uniform prior on the possible worlds. Our work is thus well-suited to the many situations where probability distributions on variables are not known, or where they are not independent (e.g., when order constraints are imposed on them). For this reason, the problems that we consider are generally computationally harder. For instance, [33] is perhaps the closest to our work, since they consider the total orders compatible with given partial order constraints. However, they assume independent marginal distributions, so they can evaluate top- k queries by only considering k -sized prefixes of the linear extensions; in our setting even computing the top-1 element is hard (Theorem 17).

The second key difference is that other works *do not try to estimate the top- k values*, because they assume that the marginal distribution is given: they only focus on ranks. In our context, we need to compute missing values, and need to account, e.g., for exact-value constraints and their effect on the probability of possible worlds and on expected values (Section 3).

We also mention our previous work [2] which considers the estimation of uncertain values (expectation and variance), but only in a *total order*, and did not consider complexity issues.

Partial order search. Another relevant research topic, *partial order search*, considers queries over elements in a partially ordered set to find a subset of elements with a certain property [1, 11, 14, 17, 30]. This relates to many applications, e.g., crowd-assisted graph search [30], frequent itemset mining with the crowd [1], and knowledge discovery, where the unknown data is queried via oracle calls [17]. These studies are *complementary to ours*: when the target function can be phrased as a top- k or interpolation problem, if the search is stopped before all values are known, we can use our method to estimate the complete output.

Interpolation. While various interpolation methods are known for totally ordered data, we are the first, to our knowledge, to propose a principled interpolation scheme for partially ordered values. A non-linear interpolation method based on spline curves was proposed in [13] for partial orders, but with no possible world semantics or complexity analysis.

Tree-shaped partial orders. Our analysis of tractable schemes for tree-shaped partial orders is reminiscent of the well-known tractability of probabilistic inference in tree-shaped graphical models [5], and of the tractability of probabilistic query evaluation on trees [9] and treelike instances [4]. However, we study continuous distributions on numerical values, and the influence between variables when we interpolate does not simply follow the tree structure; so our results do not seem to follow from these settings.

Algebra for partial orders. Some of the present authors are also authors of another submission to the same venue [3] dealing with incompleteness in ordered data. The problem considered there is however very different: [3] proposes an algebra for complex queries over ordered relational data, where order is not fully known. In contrast to our work, [3] does not consider querying *unknown data values*, nor their interpolation. Instead its goal is to develop a formalism to represent and query all compatible orders on tuples, and study the complexity of possible and certain answers, with no quantitative notion of their probability.

8 Conclusion

In this paper, we have studied the problems of top- k computation and interpolation for data with unknown values and order constraints. We have provided foundational solutions, including a general computation scheme, complexity bounds, and analysis of tractable cases.

One natural direction for future work is to study whether our tractable cases (tree-shaped orders, sampling) can be covered by more efficient PTIME algorithms, or whether more general tractable cases can be identified. Another interesting question is to extend our scheme to request additional values from the crowd, as in [1, 8], and reduce the expected error on the interpolated values or top- k query, relative to a user goal. In such a setting, how should we choose which values to retrieve, and could we update incrementally the results of interpolation when we receive new exact-value constraints? Another question is whether our results generalize to arbitrary linear constraints, or to different prior distributions on the polytope.

We also note that the uniform interpolation scheme that we have described, while intuitive, fails to respect a natural *stability* property. Intuitively, an interpolation scheme is *stable* if adding exact-value constraints to fix some variables to their interpolated values does not change the interpolated values of the other variables. We can show that this property is not respected by our scheme (see Appendix F). We leave for future work the study of alternative interpolation schemes on partial orders that would be stable in this sense.

References

- 1 A. Amarilli, Y. Amsterdamer, and T. Milo. On the complexity of mining itemsets from the crowd using taxonomies. In *ICDT*, 2014.
- 2 A. Amarilli, Y. Amsterdamer, and T. Milo. Uncertainty in crowd data sourcing under structural constraints. In *UnCrowd*, 2014.
- 3 A. Amarilli, M. L. Ba, D. Deutch, and P. Senellart. Possible and certain answers for queries over order-incomplete data. <http://pierre.senellart.com/publications/amarilli2016possible.pdf>, 2016. Submitted for publication.
- 4 A. Amarilli, P. Bourhis, and P. Senellart. Provenance circuits for trees and treelike instances. In *ICALP*, 2015.
- 5 C. M. Bishop. Graphical models. In *Pattern Recognition and Machine Learning*, chapter 8. Springer, 2006.
- 6 J. Bragg, Mausam, and D. S. Weld. Crowdsourcing multi-label classification for taxonomy creation. In *HCOMP*, 2013.
- 7 G. Brightwell and P. Winkler. Counting linear extensions. *Order*, 8(3), 1991.
- 8 E. Ciceri, P. Fraternali, D. Martinenghi, and M. Tagliasacchi. Crowdsourcing for top-k query processing over uncertain data. *Knowledge and Data Engineering, IEEE Transactions on*, 28(1):41–53, 2016.
- 9 S. Cohen, B. Kimelfeld, and Y. Sagiv. Running tree automata on probabilistic XML. In *PODS*, 2009.
- 10 G. Cormode, F. Li, and K. Yi. Semantics of ranking queries for probabilistic data and expected ranks. In *ICDE*, 2009.
- 11 S. B. Davidson, S. Khanna, T. Milo, and S. Roy. Using the crowd for top-k and group-by queries. In *ICDT*, 2013.
- 12 L. Detwiler, W. Gatterbauer, B. Louie, D. Suciu, and P. Tarczy-Hornoch. Integrating and ranking uncertain scientific data. In *ICDE*, 2009.
- 13 T. Došlić and D. J. Klein. Splinoid interpolation on finite posets. *JCAM*, 177(1), 2005.
- 14 U. Faigle, L. Lovasz, R. Schrader, and G. Turán. Searching in trees, series-parallel and interval orders. *SIAM J. Comput.*, 15(4), 1986.
- 15 C. Ge and F. Ma. A fast and practical method to estimate volumes of convex polytopes. In *FAW*, 2015.
- 16 J. E. Gentle. *Computational Statistics*. Springer, 2009.
- 17 D. Gunopulos, R. Khardon, H. Mannila, S. Saluja, H. Toivonen, and R. S. Sharma. Discovering all most specific sentences. *TODS*, 28(2), 2003.
- 18 P. Haghani, S. Michel, and K. Aberer. Evaluating top-k queries over incomplete data streams. In *CIKM*, 2009.
- 19 M. Hua, J. Pei, and X. Lin. Ranking queries on uncertain data. *VLDB J.*, 20(1), 2011.
- 20 Y. E. Ioannidis and R. Ramakrishnan. Efficient transitive closure algorithms. In *VLDB*, 1988.
- 21 J. Jestes, G. Cormode, F. Li, and K. Yi. Semantics of ranking queries for probabilistic data. *TKDE*, 23(12), 2011.
- 22 F. Jones. *Lebesgue Integration on Euclidean Space*. Jones & Bartlett Learning, 2001.
- 23 R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $o^*(n^5)$ volume algorithm for convex bodies. *Random Struct. Algorithms*, 11(1), 1997.
- 24 J. Lawrence. Polytope volume computation. *Mathematics of Computation*, 57(195), 1991.
- 25 J. Li, B. Saha, and A. Deshpande. A unified approach to ranking in probabilistic databases. *PVLDB*, 2(1), 2009.
- 26 X. Lian and L. Chen. A generic framework for handling uncertain data with local correlations. *VLDB*, 4(1), 2010.

- 27 L. Lovász and I. Deák. Computational results of an $O^*(n^4)$ volume algorithm. *European Journal of Operational Research*, 216(1), 2012.
- 28 L. Lovász and S. Vempala. Hit-and-run from a corner. *SIAM J. Comput.*, 35(4), 2006.
- 29 C. H. Papadimitriou. Efficient search for rationals. *Information Processing Letters*, 8(1), 1979.
- 30 A. Parameswaran, A. Sarma, H. Garcia-Molina, N. Polyzotis, and J. Widom. Human-assisted graph search: it's okay to ask questions. *PVLDB*, 4(5), 2011.
- 31 G. Pruesse and F. Ruskey. Generating linear extensions fast. *SIAM J. Comput.*, 23(2), 1994.
- 32 C. Re, N. N. Dalvi, and D. Suciu. Efficient top-k query evaluation on probabilistic data. In *ICDE*, 2007.
- 33 M. A. Soliman, I. F. Ilyas, and S. Ben-David. Supporting ranking queries on uncertain and incomplete data. *VLDB J.*, 19(4), 2010.
- 34 M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang. Top-k query processing in uncertain databases. In *ICDE*, 2007.
- 35 J. Stoyanovich, S. Amer-Yahia, S. B. Davidson, M. Jacob, T. Milo, et al. Understanding local structure in ranked datasets. In *Proc. CIDR*, 2013.
- 36 C. Sun, N. Rampalli, F. Yang, and A. Doan. Chimera: Large-scale classification using machine learning, rules, and crowdsourcing. *PVLDB*, 7(13), 2014.
- 37 C. Wang, L. Yuan, J. You, O. R. Zaïane, and J. Pei. On pruning for top-k ranking in uncertain databases. *PVLDB*, 4(10), 2011.
- 38 K. Yi, F. Li, G. Kollios, and D. Srivastava. Efficient processing of top-k queries in uncertain databases. In *ICDE*, 2008.
- 39 S. Zachos. Probabilistic quantifiers and games. *JCSS*, 36(3), 1988.
- 40 X. Zhang and J. Chomicki. Semantics and evaluation of top-k queries in probabilistic databases. *DAPD*, 26(1), 2009.

A Proofs for Section 2 (Preliminaries and Problem Statement)

► **Lemma 2.** *Given a set of order and exact-value constraints \mathcal{C} , we can compute in PTIME the dimension of the admissible polytope.*

Proof. The result follows from the more general fact that we can compute the dimension of a convex polyhedron in PTIME from the inequalities that define it, by determining which constraints are redundant and then computing the rank of the matrix of redundant constraints (see [Sch86]). However, for completeness, we will give a self-contained proof of this fact in our context.

We can compute this dimension of $\text{pw}_{\mathcal{X}}(\mathcal{C})$ by viewing the set of constraints as a graph. First, consider $\tilde{\mathcal{C}}$, the version of \mathcal{C} extended with $0 \leq x \leq 1$ constraints for all $x \in \mathcal{X}$. Second, build a directed graph $G_{\mathcal{C},\mathcal{X}}$ of all variables in \mathcal{X} and all constant values within $\tilde{\mathcal{C}}$, with an edge between u and v if $u \leq v$ holds in $\tilde{\mathcal{C}}$, for any variables or constant values u and v . Since \mathcal{C} is not contradictory, $G_{\mathcal{C},\mathcal{X}}$ has no cycle containing two different constant values. The dimension is then the number of strongly connected components of $G_{\mathcal{C},\mathcal{X}}$ that do not contain any constant α . ◀

► **Example 33.** Refer back to Example 3. For the original \mathcal{C} , the graph $G_{\mathcal{C},\mathcal{X}}$ has 5 strongly connected components, 3 of which have no constants. Once we add $y = 0.3$, the graph $G_{\mathcal{C},\mathcal{X}}$ still has 5 strongly connected components, but the one of y also contains the constant 0.3, so the dimension becomes 2. When we add $y \leq x$, the graph $G_{\mathcal{C},\mathcal{X}}$ loses one strongly connected component, and the dimension becomes 1.

B Proofs for Section 3 (Analysis of the General Case)

We first define formally the notion of tied values.

► **Definition 34.** Given a constraint set \mathcal{C} , we say that x and y have a *persistent tie* if \mathcal{C} contains $x \leq y$ and $y \leq x$. (Remember that \mathcal{C} is closed under implication; so in particular x and y have a persistent tie if they have an exact-value constraint to the same value.)

We write $x \sim y$ if x and y have a persistent tie. It is immediate that \sim is an equivalence relation.

We say that \mathcal{C} has a *persistent tie* if there are $x \neq y$ such that $x \sim y$.

► **Lemma 35.** *For any constraint set \mathcal{C} which has no persistent tie, the mass under the uniform distribution of possible worlds with ties is 0.*

Proof. Let d be the dimension of $\text{pw}(\mathcal{C})$ and $n := |\mathcal{X}|$. Let W be the subset of $\text{pw}(\mathcal{C})$ of the possible worlds that have ties. We write W as $\bigcup_{\substack{x_i, x_j \in \mathcal{X} \\ i \neq j}} W_{i,j}$, where $W_{i,j}$ is the subset of $\text{pw}(\mathcal{C})$ of the possible worlds where there is a tie between x_i and x_j . We clearly have:

$$V_d(W) \leq |\mathcal{X}|^2 \max_{\substack{x_i, x_j \in \mathcal{X} \\ i \neq j}} V_d(W_{i,j})$$

Hence, it suffices to show that for any $x_i, x_j \in \mathcal{X}$, $i \neq j$, $V_d(W_{i,j}) = 0$.

Hence, fix $x_i, x_j \in \mathcal{X}$, $i \neq j$. Remember that we have assumed that we only consider constraint sets \mathcal{C} such that $\text{pw}(\mathcal{C})$ is non-empty. Because \mathcal{C} is closed under implication, it cannot be the case that $t_i = t_j$ in every possible world $\mathbf{t} \in \text{pw}(\mathcal{C})$; otherwise this implies that the constraints $x_i \leq x_j$ and $x_j \leq x_i$ are implied by \mathcal{C} , so they hold in \mathcal{C} , and \mathcal{C} has a persistent tie, contradicting our assumption. Thus $\text{pw}(\mathcal{C})$ is a convex polytope of dimension d

in $[0, 1]^n$ which is not included in the hyperplane defined by the equation $x_i = x_j$. We deduce that $W_{i,j}$, the projection of $\text{pw}(\mathcal{C})$ to that hyperplane, has dimension $d - 1$, so that its volume $V_d(W_{i,j}) = \int_{W_{i,j}} \frac{d\mu_d}{V_d(\mathcal{C})}$ is 0. This concludes the proof. ◀

In the situation with persistent ties, we observe that, informally, tied variables can be collapsed to single variables, so as to remove the persistent ties without changing substantially the possible worlds and their probability. Formally:

► **Definition 36.** We define \mathcal{X}/\sim as the set of equivalence classes of \mathcal{X} for \sim , and we define \mathcal{C}/\sim to be the constraint set where every occurrence of a variable x_i of \mathcal{X} is replaced by the variable of \mathcal{X}/\sim corresponding to its equivalence class.

The following is immediate:

► **Lemma 37.** *For any constraint set \mathcal{C} on variable set \mathcal{X} with uniform distribution p_u , the constraint set \mathcal{C}/\sim is without persistent ties, can be computed in PTIME, and there is a bijection f from $\text{pw}(\mathcal{C})$ to $\text{pw}(\mathcal{C}/\sim)$.*

Proof. The computation of \mathcal{C}/\sim can be performed by computing in PTIME the strongly connected components of the graph of the \sim relation, which we can clearly compute in PTIME from \mathcal{C} . The absence of persistent ties is by observing that any persistent tie in the quotient between X_i and X_j in \mathcal{X}/\sim would imply the existence of $x_i, x'_i \in X_i$ and $x_j, x'_j \in X_j$ such that \mathcal{C} contains $x_i \leq x_j$ and $x'_j \leq x'_i$, from which we deduce the existence of a persistent tie between x_i and x_j as \mathcal{C} contains $x_j \leq x'_j, x'_i \leq x_i$ by definition of $x_i \sim x'_i, x_j \sim x'_j$, so that we should have identified x_i and x_j when constructing \mathcal{X}/\sim .

The bijection between the possible worlds is obtained by mapping any possible world $\mathbf{t} \in \text{pw}(\mathcal{C})$ to \mathbf{t}' obtained by choosing, for every $X_i \in \mathcal{X}/\sim$, the value t'_i as t_j such that $x_j \in X_i$. (The choice of representative does not matter as the definition of \sim implies that $t_j = t_{j'}$ for every $\mathbf{t} \in \text{pw}(\mathcal{C})$ whenever $x_j \sim x_{j'}$.) ◀

We conclude by showing how problems on \mathcal{C} can be reduced to problems on \mathcal{C}/\sim , concluding the proof of Proposition 6.

The interpolation problem for a variable x_i on \mathcal{C} clearly reduces to interpolation for variable X_j on \mathcal{C}/\sim .

The top- k computation problem on \mathcal{C} for a selection predicate σ reduces to the top- k computation problem on \mathcal{C}/\sim for the selection predicate σ' that selects variables X_i such that some $x_j \in X_i$ is selected by σ . Given the top- k result (X_1, \dots, X_k) on \mathcal{C}/\sim , we rewrite it to the top- k result on \mathcal{C} by replacing each X_i by all the variables $x_j \in X_i$ selected by σ (there is at least one of them), and truncating to length k .

B.1 Total Orders (Section 3.1)

► **Observation 9.** *The marginal distribution of x_i within the admissible polytope of $\mathcal{C}_1^n(\alpha, \beta)$ is the distribution of the i -th order statistic for n samples of the uniform distribution on $[\alpha, \beta]$.*

Proof. The distribution on n uniform independent samples in $[\alpha, \beta]$ can be described as first choosing a total order for the samples, uniformly among all permutations of $\{1, \dots, n\}$ (with ties having a probability of 0 and thus being neglected). Then, the distribution for each total order is exactly that of x_i when variables are relabeled. ◀

B.2 General Constraint Sets (Section 3.2)

► **Theorem 13.** *Given a constraint set \mathcal{C} over \mathcal{X} and $x \in \mathcal{X}$, determining the expected value of x in $\text{pw}(\mathcal{C})$ under the uniform distribution is in $\text{FP}^{\#P}$.*

Proof. Let \mathcal{C} be an arbitrary constraint set with order constraints and exact-value constraints on a variable set \mathcal{X} , and let $n := |\mathcal{X}|$. Use Proposition 6 to ensure that there are no ties. To simplify the reasoning, we will make all values occurring in exact-value constraints be integers that are multiples of $(n + 1)!$ as follows: let Δ be $(n + 1)!$ times the product of the denominators of all exact-value constraints occurring in \mathcal{C} , which can be computed in PTIME, and consider \mathcal{C}' the constraint set defined on $[0, \Delta]^n$ by keeping the same variables and order constraints, and replacing any exact-value constraint $x_i = v$ by $x_i = v\Delta$; the constraint set \mathcal{C}' is computable in PTIME from \mathcal{C} , and the polytope $\text{pw}(\mathcal{C}')$ is obtained by scaling $\text{pw}(\mathcal{C})$ by a factor of Δ along all dimensions; hence, if we can compute the expected value of x_i in \mathcal{C}' (which is the coordinate of the center of mass of $\text{pw}(\mathcal{C}')$ on the component corresponding to x_i), we can compute the expected value of x_i in \mathcal{C} by dividing by Δ . So we can thus assume that $\text{pw}(\mathcal{C}')$ is a polytope of $[0, \Delta]^n$ where all exact-value constraints are integers which are multiples of $(n + 1)!$.

We use Lemma 5.2 of [ACK⁺11] to argue that the volume of $\text{pw}(\mathcal{C})$ can be computed in $\#P$. The PTIME generating Turing machine T , given the constraint set \mathcal{C} , chooses nondeterministically a linear extension of $(\mathcal{X}, \leq_{\mathcal{C}})$, which can clearly be represented in polynomial space and checked in PTIME. The PTIME-computable function g computes, as a rational, the volume of the polytope for that linear extension, and does so according to the scheme of Section 3: the volume is the product of the volumes of each $\mathcal{C}_1^n(\alpha, \beta)$, whose volume is $\frac{(\beta - \alpha)^n}{n!}$. This is clearly PTIME-computable, and as α and β are values occurring in exact-value constraints, they are integers and multiples of $n!$, so the result is an integer, so the overall result is a product of integers, hence an integer. By Lemma 5.2 of [ACK⁺11], $V(\mathcal{C}')$, which is the sum of $V(\mathcal{T})$ across all linear extensions \mathcal{T} of \mathcal{C}' (because there are no ties), is computable in $\#P$.

We now apply the same reasoning to show that the sum, across all linear extensions \mathcal{T} , of $V(\mathcal{T})$ times the expected value of x_i in \mathcal{T} , is computable in $\#P$. Again, we use Lemma 5.2 of [ACK⁺11], with T enumerating linear extensions, and with a function g that computes the volume of the linear extension as above, and multiplies it by the expected value of x_i , by linear interpolation in the right \mathcal{C}_1^n as in the previous section (it is an integer, as all values of exact-value constraints are multiples of $n + 1$). So this concludes, as the expected value v of x_i is $\frac{1}{V(\mathcal{C}')} \sum_{\mathcal{T}} V(\mathcal{T}) v_{\mathcal{T}}$ where $v_{\mathcal{T}}$ is the expected value of x_i for \mathcal{T} , and we can compute both the sum and the denominator in $\#P$. Hence, the result of the division, and reducing back to the answer for the original \mathcal{C} , can be done in $\text{FP}^{\#P}$. ◀

► **Proposition 15.** *For any constraint set \mathcal{C} on \mathcal{X} , if \bar{x}_i is the expected value of x_i for all $x_i \in \mathcal{X}$, then $\bar{\mathbf{x}}$ is the center of mass of the admissible polytope $\text{pw}(\mathcal{C})$.*

Proof. Fix \mathcal{C} and a variable x_i . We show that the expected value \bar{x}_i of x_i is the coordinate of the center of mass $\bar{\mathbf{x}}$ of $\text{pw}(\mathcal{C})$ along vector \mathbf{u}_i , where \mathbf{u}_i is the unit vector for the i -th coordinate.

The vector of the center of mass of $\text{pw}(\mathcal{C})$ is given by: $\bar{\mathbf{x}} := \frac{1}{V(\mathcal{C})} \int_{\text{pw}(\mathcal{C})} \mathbf{r} \, dw$ where \mathbf{r} is the vector that gives the coordinates of each point in the basis of the \mathbf{u}_i 's. Hence, $\bar{x}_i = \frac{1}{V(\mathcal{C})} \int_{\text{pw}(\mathcal{C})} \mathbf{r} \cdot \mathbf{u}_i \, dw$. Since the coordinate of the center of mass along \mathbf{u}_i must be in $[0, 1]$, because $\text{pw}(\mathcal{C}) \subseteq [0, 1]^n$, we can split the integral to an integration along \mathbf{u}_i (for t on the finite interval $[0, 1]$), and then on the section of $\text{pw}(\mathcal{C})$ projecting to coordinate t

on \mathbf{u}_i ; in other words, the intersection of $\text{pw}(\mathcal{C})$ with the hyperplane with coordinate t on axis \mathbf{u}_i . Formally, $\bar{x}_i = \frac{1}{V(\mathcal{C})} \int_{t=0}^1 tV_t dt$, where V_t is the volume of the section of $\text{pw}(\mathcal{C})$ along the hyperplane with coordinate t on axis \mathbf{u}_i , i.e., $\bar{x}_i = \frac{1}{V(\mathcal{C})} \int_{t=0}^1 tV(\mathcal{C}|_{x=t}) dt$. But this is exactly the definition of the expected value of the marginal distribution of Definition 5, which concludes the proof. \blacktriangleleft

C Proofs for Section 4 (Hardness and Approximations)

C.1 Hardness of Exact Computation (Section 4.1)

► **Theorem 16.** *Given a set \mathcal{C} of order constraints with variable set \mathcal{X} and $x \in \mathcal{X}$, determining the expected value of x in $\text{pw}(\mathcal{C})$ under the uniform distribution is $\text{FP}^{\#P}$ -hard.*

To prove this result, we will rely on the following known result, in the context of partial orders without known values:

► **Theorem 38 ([BW91], Theorems 6 and 7).** *For any input partial order (\mathcal{X}, \leq_P) and two elements x and y of \mathcal{X} , it is $\#P$ -complete to compute the expected rank of x among linear extensions of (\mathcal{X}, \leq_P) , or the probability that $x < y$ holds in linear extensions of (\mathcal{X}, \leq_P) .*

We now prove Theorem 16:

Proof. We prove hardness by showing a reduction from the problem of computing the expected rank of an element in a partial order, which is $\#P$ -hard by Theorem 38. Let $P = (\mathcal{X}, \leq_P)$ be a partial order, and let $\mathcal{C} = \{x \leq y \mid x, y \in \mathcal{X}, x \leq_P y\}$ be the set of constraints over \mathcal{X} corresponding to \leq_P . Note that \mathcal{C} contains no exact-value constraints, and that we can assume ties have a zero probability. We claim that, writing $n := |\mathcal{X}|$, if the expected value of x is v , then the expected rank r of x in P is $(n+1)v$, which can clearly be computed in PTIME from v . We now justify that this reduction is correct.

Let $\mathcal{T}_1, \dots, \mathcal{T}_N$ be the linear extensions of P . Consider the partition of the set of possible worlds $\text{pw}(\mathcal{C})$ into subsets W_1, \dots, W_N , where $W_i := \text{pw}(\mathcal{T}_i)$ for each i . Their union is indeed exactly $\text{pw}(\mathcal{C})$, as every possible world of \mathcal{C} realizes some linear extension of P (possibly many, if there are ties), and as we assumed that the probability of a tie is 0, this is indeed a partition of $\text{pw}(\mathcal{C})$ up to worlds that have a total probability of 0. Now, it is easily seen by symmetry that all \mathcal{T}_i have the same volume. Hence, denoting v the (unknown) expected value of x in \mathcal{C} , we have $v = \frac{1}{N} \sum_{i=1}^N v_i$, where v_i is the expected value of x in \mathcal{T}_i . As there are no exact-value constraints, by Section 3.1, we know that, following linear interpolation between 0 and 1, for any i , we have $v_i = \frac{r_i}{n+1}$, where r_i is the rank of x in \mathcal{T}_i (that is, its index in the total order \mathcal{T}_i , between 1 and n). Hence, we have $v = \frac{1}{N \cdot (n+1)} \sum_{i=1}^N r_i$. Now, the expected rank of x in P is defined as $r = \frac{1}{N} \sum_{i=1}^N r_i$, so it is clear that $r = (n+1)v$, showing the correctness of the reduction. \blacktriangleleft

► **Theorem 17.** *Given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , and an integer k , the top- k computation problem over \mathcal{X} , \mathcal{C} and σ is $\text{FP}^{\#P}$ -hard even if k is fixed to be 1, $|\mathcal{X}_\sigma|$ is 2, and the top- k answer does not include the expected value of the variables.*

Proof. We will perform a reduction from the interpolation problem (i.e., expected value computation): given a constraint set \mathcal{C} and a variable x_i in the variable set \mathcal{X} of \mathcal{C} , determine the expected value of x_i in \mathcal{C} , which is $\#P$ -hard by Theorem 16. From the proof of that theorem, we can further assume that \mathcal{C} contains no exact-value constraints, and hardness

still holds. Write $n := |\mathcal{X}|$. Assume using Proposition 6 that the probability of ties in $\text{pw}(\mathcal{C})$ is zero.

We first observe, as in the proof of Theorem 16, that the expected value v of x_i can be written as $\frac{1}{N(n+1)} \sum_{\mathcal{T}} r_{\mathcal{T}}$, where the sum is over all the linear extensions \mathcal{T} of \mathcal{C} , $r_{\mathcal{T}} \in \{1, \dots, n\}$ is the rank of x_i in the linear extension \mathcal{T} , and $N \leq n!$ is the number of linear extensions. This implies that v can be written as a rational p/q with $0 \leq p \leq q$ and $0 \leq q \leq M$, where we write $M := (n+1)!$.

We determine this fraction p/q using the algorithm of [Pap79], that proceeds by making queries of the form “is $p/q \leq p'/q'$ ” with $0 \leq p', q' \leq M$, and runs in time logarithmic in the value M , so polynomial in the input \mathcal{C} . To do so, we must describe how to decide in PTIME whether $v \leq p'/q'$ for $0 \leq p', q' \leq M$, using an oracle for the top-1 computation problem that does not return the expected values.

Fix $v' = p'/q'$ the query value and let $v = p/q$ be the unknown target value, the expected value of x_i . We illustrate how to decide whether $v \leq v'$. The general idea is to add a variable with exact-value constraint to v' and compute the top-1 between x_i and the new variable, but we need a slightly more complicated scheme because the top-1 answer variable can be arbitrary in the case where $v = v'$ (i.e., we have a tie in computing the top-1). Let $\epsilon := 1/(2(M^2 + 1))$, which is computable in PTIME in the value of n (so in PTIME in the size of the input \mathcal{C}). Construct \mathcal{C}' (resp., \mathcal{C}'_- , \mathcal{C}'_+) by adding an exact-value constraint $x' = v'$ (resp., $x'_- = v' - \epsilon$, $x'_+ = v' + \epsilon$) for a fresh variable x' (resp., x'_- , x'_+). Now use the oracle for \mathcal{C}' (resp., \mathcal{C}'_- , \mathcal{C}'_+) and the selection predicate that selects x_i and x' (resp., x'_- , x'_+), taking $k = 1$ in all cases. The additional variables do not affect the expected value of x_i in \mathcal{C}' , \mathcal{C}'_+ , and \mathcal{C}'_- , so it is also v in them. Further, we know that $v = p/q$, $v' = p'/q'$, with $0 \leq p, q, p', q' \leq M$, hence either $v = v'$, or $|v - v'| \geq \frac{1}{M^2}$. Hence, letting $S = \{v', v' - \epsilon, v' + \epsilon\}$, there are three possibilities: $v = v'$, $v < v''$ for all $v'' \in S$, or $v > v''$ for all $v'' \in S$. Thus, if the top-1 variable in all oracle calls is always x_i (resp., never x_i), then we are sure that $v > v'$ (resp., $v < v'$). If some oracle calls return x_i but not all of them, we are sure that $v = v'$. Hence, we can find out in PTIME using the oracle whether $v \leq v'$. This concludes the proof, as we then have an overall PTIME reduction from the $\text{FP}^{\#\text{P}}$ -hard problem of interpolation (i.e., expected value computation) to the top-1 computation problem, showing that the latter is also $\text{FP}^{\#\text{P}}$ -hard. \blacktriangleleft

► **Corollary 18.** *The top-k decision problem of deciding, given a constraint set \mathcal{C} over \mathcal{X} , a selection predicate σ , an integer k , and a set $X \subseteq \mathcal{X}$ of size k , whether the k items of \mathcal{X}_{σ} that have maximal expected values are exactly those of X , is NP-hard.*

Proof. We reduce from the top- k computation problem, which is $\#\text{P}$ -hard, and hence NP-hard. As per Theorem 17, we can fix an instance $(\mathcal{C}, \mathcal{X}, \sigma, k)$ of that problem with $k = 1$ and $|\mathcal{X}_{\sigma}| = 2$. Let x be one of the two variables of \mathcal{X}_{σ} . Then $(\mathcal{C}, \mathcal{X}, \sigma, k, x)$ is an accepted instance of the top-1 decision problem if and only if x is the answer to the top-1 computation problem. \blacktriangleleft

C.2 Complexity of Approximate Computation (Section 4.2)

► **Proposition 19.** *Let \mathcal{C} be a set of constraints with variable set \mathcal{X} and $x \in \mathcal{X}$. There is an FPRAS that determines an estimate $\hat{\mathbb{E}}_{\mathcal{C}}[x]$ of the expected value $\mathbb{E}_{\mathcal{C}}[x]$ of x in $\text{pw}(\mathcal{C})$ under the uniform distribution.*

Proof. We rely on a result by Kannan, Lovász, and Simonovits (Theorem 2.2 of [KLS97]) that shows that sampling a point *almost uniformly* from a convex body in dimension n can be

done by $\tilde{O}(n^5)$ calls to an oracle deciding membership in the body, and an additional factor of $\tilde{O}(n^2)$ arithmetic operations (here, $\tilde{O}(\cdot)$ is the *soft-O* notation, where polylogarithmic factors are ignored). By “almost uniformly” we mean that the *total variation distance* between the uniform distribution and the actual distribution realized by the sampling is less than any fixed number $\epsilon' > 0$. The dependency of the running time in ϵ' is logarithmic and thus hidden in the polylogarithmic complexity analysis.

Let $\epsilon, \delta > 0$ be two reals. For some number N that we define further, we first apply N times consecutively the sampling algorithm of [KLS97] with $\epsilon' := \frac{\epsilon}{4}$ to obtain N independent samples p_1, \dots, p_N . The algorithm uses an oracle for membership to the polytope, which in our setting can be determined in time $O(|\mathcal{C}|)$; hence, as the dimension of $\text{pw}(\mathcal{C})$ is bounded by $|\mathcal{X}|$, by the complexity analysis of [KLS97], this process has a running time of $O(N \times |\mathcal{X}|^7 \times |\mathcal{C}|)$.

We then consider the projection of every point p_i to the dimension defined by the variable x , resulting in N values v_1, \dots, v_N . We then compute the estimate $\hat{\mathbb{E}}_{\mathcal{C}}[x]$ as $\frac{1}{N} \sum_{i=1}^N v_i$. Clearly this has no impact on the running time. It only remains to show that this estimate satisfies the given bounds, that is, that we have $|\hat{\mathbb{E}}_{\mathcal{C}}[x] - \mathbb{E}_{\mathcal{C}}[x]| \leq \epsilon$ with probability at least $1 - \delta$.

Let $f : \text{pw}(\mathcal{C}) \rightarrow [0; 1]$ be the probability density function of the distribution realized by the algorithm of [KLS97], according to which the independent samples were drawn. By definition of the total variation distance, we know that $\frac{1}{2} \int_{\text{pw}(\mathcal{C})} \left| \frac{1}{V(\mathcal{C})} - f(p) \right| dp \leq \frac{\epsilon}{4}$. If we denote by μ_v the average of the (independent) x -value samples v_1, \dots, v_N , and we denote by $f_{x=t}$ the function obtained from f by setting to t the coordinate corresponding to x , we have:

$$\begin{aligned} |\mu_v - \mathbb{E}_{\mathcal{C}}[x]| &= \left| \int_0^1 t \int_{\text{pw}(\mathcal{C}_{x=t})} f_{x=t}(p) dp dt - \int_0^1 t \int_{\text{pw}(\mathcal{C}_{x=t})} \frac{1}{V(\mathcal{C})} dp dt \right| \\ &= \left| \int_0^1 t \int_{\text{pw}(\mathcal{C}_{x=t})} \left(f_{x=t}(p) - \frac{1}{V(\mathcal{C})} \right) dp dt \right| \\ &\leq \int_0^1 1 \times \int_{\text{pw}(\mathcal{C}_{x=t})} \left| \frac{1}{V(\mathcal{C})} - f_{x=t}(p) \right| dp dt \\ &= \int_{\text{pw}(\mathcal{C})} \left| \frac{1}{V(\mathcal{C})} - f(p) \right| dp \leq \frac{\epsilon}{2}. \end{aligned}$$

Now, as the v_i are independent and identically distributed, we obtain by Hoeffding’s inequality [Hoe63]:

$$\Pr \left(\left| \hat{\mathbb{E}}_{\mathcal{C}}[x] - \mu_v \right| \geq \frac{\epsilon}{2} \right) \leq 2 \exp \left(\frac{-\epsilon^2 N}{2} \right).$$

Therefore, setting $N := \frac{2 \ln(\frac{2}{\delta})}{\epsilon^2}$, we have:

$$\Pr \left(\left| \hat{\mathbb{E}}_{\mathcal{C}}[x] - \mathbb{E}_{\mathcal{C}}[x] \right| \leq \epsilon \right) \geq \Pr \left(\left| \hat{\mathbb{E}}_{\mathcal{C}}[x] - \mu_v \right| \leq \frac{\epsilon}{2} \wedge \left| \mu_v - \mathbb{E}_{\mathcal{C}}[x] \right| \leq \frac{\epsilon}{2} \right) \geq 1 - \delta. \quad \blacktriangleleft$$

D Proofs for Section 5 (Tractable Cases)

► **Lemma 21.** *There exists a bijective correspondence between $\text{pw}(\mathcal{C}_1) \times \dots \times \text{pw}(\mathcal{C}_m)$ and $\text{pw}(\mathcal{C})$, obtained by merging the variables with exact-value constraints.*

Proof. It is immediate that any $\mathbf{t} \in \text{pw}(\mathcal{C})$ yields a tuple of possible worlds of $\mathcal{C}_1, \dots, \mathcal{C}_m$. Conversely, consider any tuple of possible worlds of $\mathcal{C}_1, \dots, \mathcal{C}_m$. Note first that, indeed, those tuples match on the variables that occur in multiple sets (which are exactly the variables with exact-value constraints, which occur in all sets). Let us show that combining them yields a possible world of \mathcal{C} . Assume by contradiction that some constraint is violated. Clearly it cannot be an exact-value constraint, because all are reflected in all \mathcal{C}_i , so it must be an order constraint of the form $x \leq x'$. Clearly x and x' cannot be part of the same uninfluenced class \mathcal{C}_i , otherwise the constraint is reflected in \mathcal{C}_i ; so we must have $x \not\leq x'$. Now, as $x \leq x'$, there must be a path $x = x_1 \prec \dots \prec x_n = x'$, but as $x \not\leq x'$ there must be a variable x_k in the sequence with an exact-value constraint. Hence, the constraint $x \leq x_k$ is reflected in the uninfluence class of x , and the constraint $x_k \leq x'$ is reflected in the uninfluence class of x' , so we deduce that the constraint $x \leq x'$ is respected overall. \blacktriangleleft

► **Theorem 24.** *For any tree-shaped constraint set \mathcal{C} over \mathcal{X} , we can compute $V(\mathcal{C})$ in time $O(|\mathcal{X}|^2)$.*

Proof. Let T be the tree with vertex set \mathcal{X} which is the Hasse diagram of the order constraints imposed by \mathcal{C} . For any variable $x \in \mathcal{X}$ that has no exact-value constraint (so it is not the root of T or a leaf of T), let \mathcal{C}_x be the constraint set obtained as a subset of \mathcal{C} by keeping only constraints between x and its descendants in T , as well as between x and its parent. For $v \in [0, 1]$, we call $V_x(v)$ the d -volume of $\text{pw}(\mathcal{C}_x \cup \{x' = v\})$ where x' is the parent of x and d is the dimension of $\text{pw}(\mathcal{C}_x)$. In other words, $V_x(v)$ is the d -volume of the admissible polytope for the subtree $T|_x$ of T rooted at x , as a function of the minimum value on x imposed by the exact-value constraint on the parent of x . It is clear that, letting x'_r be the one child of the root x_r of T , we have $V(\mathcal{C}) = V_{x'_r}(v_r)$, where v_r is the exact value imposed on x_r .

We show by induction on T that, for any node x of T , letting m_x be the minimum, among all leaves that are descendants of x , of the values to which those leaves have an exact-value constraint, the function V_x is zero in the interval $[m_x, 1]$ and can be expressed in $[0, m_x]$ as a polynomial whose degree is at most the number of nodes in $T|_x$, written $|T|_x$. Since the probability of ties is 0, we have $m_x > 0$ for all x .

The base case is for a node x of T which has only leaves as children; in this case it is clear that $V_x(v)$ is $m_x - v$ for $v \in [0, m_x]$, and is zero otherwise. For the inductive case, let x be a variable. It is clear that $V_x(v)$ is 0 for $v \in [m_x, 1]$. Otherwise, let $v' \in [0, m_x]$ be the value of the parent x' of x . For every value $v' \leq v \leq m_x$ of x , consider the constraint set $\mathcal{C}_{x,v,v'} = \mathcal{C}_x \cup \{x' = v', x = v\}$. By Lemma 21, we have $V(\mathcal{C}_{x,v,v'}) = \prod_i V_{x_i}(v)$ where x_1, \dots, x_l are the children of x . Hence, by definition of the volume, we know that $V_x(v) = \int_{v'}^{m_x} \prod_i V_{x_i}(v) dv$. Now, we use the induction hypothesis to deduce that $V_{x_i}(v)$, for all i , in the interval $[0, m_x]$, is a polynomial whose degree is at most $|T|_{x_i}$. Hence, as the product of polynomials is a polynomial whose degree is the sum of the input polynomials, and integrating a polynomial yields a polynomial whose degree is one plus that of the input polynomial, V_x in the interval $[0, m_x]$ is a polynomial whose degree is at most $|T|_x$.

Hence, we have proved the claim by induction, and we use it to determine $V(\mathcal{C})$ as explained in the first paragraph.

We now prove that the computation is quadratic. We first assume that the tree T is binary. We show by induction that there exists a constant $\alpha \geq 0$ such that the computation of the polynomial V_{x_i} in expanded form has cost less than αn_i^2 , where n_i is $|T|_{x_i}$. The claim is clearly true for nodes where all children are leaves, because the cost is linear in the number of child nodes as long as α is at least the number of operations per node α_0 . For the induction step, if x_i is an internal node, let x_p and x_q be the two children. By induction

hypothesis, computing V_{x_p} and V_{x_q} in expanded form has cost $\leq \alpha(n_p^2 + n_q^2)$. Remembering that arithmetic operations on rationals are assumed to take unit time, computing the product of V_{x_p} and V_{x_q} in expanded form has cost linear in the product of the degrees⁵ of V_{x_p} and V_{x_q} which are less than n_p and n_q , so the cost of computing the product is $\leq \alpha_1 n_p n_q$ for some constant α_1 . Integrating has cost linear in the degree of the resulting polynomial, that is, $n_p + n_q$. So the total cost of computing V_{x_i} is $\leq \alpha(n_p^2 + n_q^2) + \alpha_1 n_p n_q + \alpha_2(n_p + n_q) + \alpha_3$ for some constants α_2, α_3 . Now, as $n_q = n_i - n_p - 1$, computing V_{x_i} costs less than:

$$\begin{aligned} & \alpha n_i^2 + 2\alpha n_p^2 + \alpha - 2\alpha n_i n_p - 2\alpha n_i + 2\alpha n_p + \alpha_1 n_p n_q + \alpha_2 n_i - \alpha_2 + \alpha_3 \\ &= \alpha n_i^2 + (\alpha_1 - 2\alpha) n_p n_q + (\alpha_2 - 2\alpha) n_i + \alpha - \alpha_2 + \alpha_3 \end{aligned}$$

As long as α is set to be $\geq \max(\frac{\alpha_1}{2}, \frac{\alpha_2}{2})$, the second and third terms are negative, which means (since $n_p n_q$ and n_i are both ≥ 1) that V_{x_i} costs less than:

$$\begin{aligned} & \alpha n_i^2 + \alpha_1 - 2\alpha + \alpha_2 - 2\alpha + \alpha - \alpha_2 + \alpha_3 \\ &= \alpha n_i^2 - 3\alpha + \alpha_1 + \alpha_3 \leq \alpha n_i^2 \end{aligned}$$

if $\alpha \geq \frac{\alpha_1 + \alpha_3}{3}$. This concludes the induction case, by setting α to any arbitrary value which is greater than $\max(\alpha_0, \frac{\alpha_1}{2}, \frac{\alpha_2}{2}, \frac{\alpha_1 + \alpha_3}{3})$. Hence the claim is proven if T is binary.

If T is not binary, we use the associativity of product to make T binary, by adding virtual nodes that represent the computation of the product. In so doing, the size of T increases only by a constant multiplicative factor (recall that the number of internal nodes in a full binary tree is one less than the number of leaves, meaning that the total number of nodes in a binary expansion of a n -ary product is less than twice the number of operands of the product). So the claim also holds for arbitrary T . \blacktriangleleft

► Theorem 25. *For any tree-shaped constraint set \mathcal{C} on variable set \mathcal{X} and variable $x \in \mathcal{X}$ with no exact-value constraint, the marginal distribution for x is piecewise polynomial and can be computed in time $O(|\mathcal{X}|^3)$.*

Proof. Recall that $\mathcal{C}_{|x=v}$ is \mathcal{C} plus the exact-value constraint $x = v$. For any variable x' , we let $m_{x'}$ be the minimum, among all leaves reachable from x' , of the values to which those leaves have an exact-value constraint. By definition, the marginal distribution for x is $v \mapsto \frac{1}{V(\mathcal{C})} V(\mathcal{C}_{|x=v})$. We have seen in Theorem 24 that $\frac{1}{V(\mathcal{C})}$ can be computed in quadratic time; we now focus on the function $V(\mathcal{C}_{|x=v})$.

By Lemma 21, letting x_1, \dots, x_k be the children of x , D_1, \dots, D_k be their descendants (the x_i included), and D be all variables except x and its descendants, that is, $D := \mathcal{X} \setminus (\{x\} \cup \bigsqcup_i D_i)$, we can express $V(\mathcal{C}_{|x=v})$ as $V'_x(v) \times \prod_i V_{x_i}(v)$, where V_{x_i} is as in the proof of Theorem 24, and $V'_x(v)$ is the volume of the constraint set $\mathcal{C}'_{x,v}$ over D obtained by keeping all constraints in \mathcal{C} about variables of D , plus the exact-value constraint $x = v$. Indeed, the uninfluenced classes of $\mathcal{C}_{|x=v}$ are clearly D, D_1, \dots, D_k , except that the root and leaves are in singleton classes because they have exact-value constraints.

We know by the proof of Theorem 24 that V_{x_i} , in the interval $[0, m_{x_i}]$, is a polynomial whose degree is at most $|T_{|x_i}|$, and that it can be computed in $O(|T_{|x_i}|^2)$. Hence, the product of the $V_{x_i}(v)$ can be computed in cubic time overall and has linear degree. We thus focus on $\mathcal{C}'_{x,v}$, for which it suffices to show that $V(\mathcal{C}'_{x,v})$ is a piecewise polynomial function with linearly many pieces, each piece having a linear degree and being computable in quadratic

⁵ We could compute the product of the polynomials more efficiently using the FFT, but this would not improve the overall complexity bound.

time. Indeed, this suffices to justify that computing the product of $V(\mathcal{C}'_{x,v})$ with $\prod_i V_{x_i}(v)$, and integrating to obtain the marginal distribution, can be done in cubic time, and that the result is indeed piecewise polynomial.

For any node x_i of D with no exact-value constraint, we let $V'_{x_i,x}(v, v')$ be the volume of the constraint set obtained by restricting $\mathcal{C}'_{x,v'}$ to the descendants of the parent x'_i of x_i and adding the exact-value constraint $x'_i = v$. We let (v_1, \dots, v_q) be the values occurring in exact-value constraints in \mathcal{C} , in increasing order. We show by induction on D the following claim: for any $1 \leq i < q$, for any variable x_i in D with no exact-value constraint, in the intervals $v \in [0, m_{x_i}]$ and $v' \in [v_i, v_{i+1}]$, $V'_{x_i}(v, v')$ can be expressed as $P(v) + v'P'(v)$, where P and P' are polynomials of degree at most $|T_{x_i}|$ and can be computed in quadratic time.

The proof is the same as in Theorem 24: for the base case where all children of x_i have exact-value constraints, $V_{x_i}(v, v')$ is either $m_{x_i} - v$ if x is not reachable from x_i or $v_i \geq m_x$, or $v' - v$ otherwise. For the inductive case, we do the same argument as before, noting that, clearly, taking the product of the $V'_{x_i}(v, v')$ among the children of x_i , the variable v' occurs in at most one of them, namely the one from which x is reachable. We conclude that $V'_x(v)$ is indeed a piecewise polynomial function with linearly many pieces that have a linear degree, by evaluating $V'_{x'',x}(v'', v)$, where x'' is the one child of the root of T and v'' is the value to which it has an exact-value constraint, and the overall computation time is cubic. \blacktriangleleft

E Proofs for Section 6 (Other Variants)

► **Lemma 27.** *There is a constraint set \mathcal{C} and selection predicate σ such that local-top- k and U-top- k do not match, even for $k = 1$ and without returning expected values or probabilities.*

Proof. Let $\mu = 2/3$, $m = 1/\sqrt{2}$, and any v such that $\mu < v < m$. Consider variables x , x' and y , with the constraint set that imposes $x' \leq x$ and $y = v$. Fix $k = 1$ and consider the predicate σ that selects all variables. It is immediate by linear interpolation that the expected value of x is μ . Further, it is easily computed that the marginal distribution p_x of x has the pdf $p_x : t \mapsto 2t$ on $[0, 1]$, for which we can check that $\int_0^m p_x(t)dt = \int_m^1 p_x(t)dt$. Hence, as $v < m$, the probability that x is larger than v in $\text{pw}(\mathcal{C})$ is $> 1/2$. This implies that the U-top-1 answer is x . By contrast, as $\mu < m$, the local-top-1 answer is y . \blacktriangleleft

► **Lemma 30.** *There is a constraint set \mathcal{C} without ties such that U-top- k does not satisfy the containment property for the uniform distribution on $\text{pw}(\mathcal{C})$.*

Proof. Consider variables x_1 , x_h , x_f^+ , and x_f^- , and the constraint set \mathcal{C} that imposes $x_1 \leq x_h$, $x_f^+ = .7$, and $x_f^- = .69$. Consider the selection predicate σ that selects all variables. The total volume of the constraint set is clearly $V = 1/2$.

We first set $k = 1$. The first possible answer is (x_f^+) with probability $\frac{.7 \times .7}{2 \cdot V} = .49$, and the second is (x_h) with probability $.51$, so the U-top-1 is (x_h) .

We then set $k = 2$. There are four possible answers. The first possible answer is (x_f^+, x_f^-) with probability $\frac{.69 \times .69}{2 \cdot V} = .4761$. The second possible answer is (x_h, x_f^+) with probability $\frac{.3 \times .7}{V} = .42$. The third possible answer is (x_h, x_1) with probability $.09$. The fourth possible answer is (x_f^+, x_h) with probability $\frac{.01 \times .69 + .01 \times .01 \times .5}{V} = .0139$. Hence, the U-top-2 is (x_f^+, x_f^-) .

Hence, the U-top-1 variable does not occur in the U-top-2. \blacktriangleleft

► **Lemma 32.** *There is a constraint set \mathcal{C} without ties such that global-top- k does not satisfy the containment property for the uniform distribution on $\text{pw}(\mathcal{C})$.*

Proof. Consider variables x_s , x_f , x_1 and x_h and the constraint set \mathcal{C} that imposes $x_1 \leq x_h$, $x_1 = .45$ and $x_f = .73$. Consider σ that selects all variables.

Set $k := 1$. Variable x_h has the highest value with probability $\frac{1}{V}(.73 \times (1 - .73) + (1 - .73)^2/2)$. Variable x_f has the highest value with probability $\frac{1}{V}(.73 - .45) \times .73$, which is less. The probability for x_s is also less. So the global-top-1 is (x_h) .

Now, set $k := 2$. Variable x_h has one of the two highest values in all cases except for $x_f \geq x_s \geq x_h \geq x_1$ and $x_s \geq x_f \geq x_h \geq x_1$, so it has one of the two highest values with probability $1 - \frac{1}{V}((.73 - .45) \times (1 - .73) + (.73 - .45)^2/2)$. However, variable x_f has one of the two highest values in all cases except for $x_h \geq x_s \geq x_f \geq x_1$ and $x_s \geq x_h \geq x_f \geq x_1$, so it has one of the two highest values with probability $1 - \frac{1}{V}(1 - .73)^2$, which is more. Hence the first variable of the global-top-2 is x_f and not x_h . ◀

F Alternative Interpolation Scheme

We now consider variants of the interpolation problem, thus far performed by considering the expected value under the uniform distribution. Since we are not aware of candidate variants in previous work, for interpolating over partial orders with exact value constraints, we propose a new alternative variant. Rather than imposing the connection to the uniform prior, we present natural desiderata for an interpolation scheme on partial orders. We show that they are not respected by our current definition, and show that a definition that respects them can be proposed for tree-shaped partial orders (Definition 22), it is in fact unique, and it can be computed tractably.

We first define the notion of interpolation scheme.

► **Definition 39.** An *interpolation scheme* is a function that maps any constraint set \mathcal{C} on variables \mathcal{X} to a mapping from \mathcal{X} to its interpolated value in $[0, 1]$.

For instance, the interpolation scheme that we have studied thus far maps each variable to its expected value under the uniform distribution on $\text{pw}(\mathcal{C})$. We refer to this scheme as **Uniform** in the sequel.

We define the first natural desideratum for interpolation schemes, stability: intuitively, an interpolation scheme is stable if assigning variables to their interpolated value does not change the result of interpolation elsewhere. Formally,

► **Definition 40.** An interpolation scheme \mathcal{S} is *stable* if, for every constraint set \mathcal{C} over \mathcal{X} and every $x \in \mathcal{X}$, \mathcal{S} assigns the same mapping $f : \mathcal{X} \rightarrow [0, 1]$ to both \mathcal{C} and $\mathcal{C} \cup \{x = f(x)\}$.

This property can be shown to be respected, e.g., by linear interpolation on total orders. However, a counterexample shows that the stability property is not respected by the uniform scheme:

► **Lemma 41.** *The Uniform scheme is not stable, even on tree-shaped constraint sets.*

Proof. Consider the set of variables $\{x_r, x_a, x_b, x_c, x_d, x_e\}$ and the constraint set \mathcal{C} formed of the order constraints $x_r \leq x_a, x_a \leq x_b \leq x_c, x_a \leq x_d \leq x_e$, and the exact-value constraints $x_r = 0, x_c = .5$ and $x_e = 1$. We can compute that the interpolated values for x_a and for x_b are $3/20$ and $13/40$ respectively. However, adding the exact-value constraint $x_b = 13/40$, the interpolated value for x_a becomes $611/4020$, which is different from $3/20$. ◀

Let us use stability as a guide to design a different interpolation scheme. We impose another desideratum to act as a *base case*, specifying what one expects from an interpolation scheme when there is only a single unknown variable:

► **Definition 42.** Let \mathcal{C} be a (non-contradictory) constraint set such that x is the only unknown variable; y_1, \dots, y_n are variables with exact value constraints such that $y_i \leq x$; and z_1, \dots, z_m are variables with exact value constraints such that $x \leq z_i$ (having $n, m \geq 1$). We say an interpolation scheme is *balanced* if, for each such \mathcal{C} , its interpolated value for x is $\frac{\max_i(v(y_i)) + \min_i(v(z_i))}{2}$.

In particular, the Uniform scheme is balanced in this sense. However, we would like to find a scheme that is *both balanced and stable*. For the case of general constraint sets, this problem remains open, and we leave it for future work. For tree-shaped constraint sets, we next not only show such a scheme, but also prove it is unique, as follows.

► **Proposition 43.** *There is at most one interpolation scheme on trees that is both stable and balanced.*

Proof. We first observe that, because of the balanced requirement, in any constraint set \mathcal{C} , for any unknown variable x whose parent y was interpolated to value v and whose children z_i were interpolated to w_i , x must have been interpolated to $(v + (\min_i w_i))/2$. Indeed, considering the constraint set \mathcal{C}' where y and z_i have been set to those values, by stability, the interpolation value of x does not change. Hence, as the interpolation scheme is balanced, we conclude that the claimed property holds.

We now show that the resulting set of equations always has at most one solution on any tree-shaped constraint sets. Indeed, assume that there are two stable and balanced interpolation schemes f and g which yield different results on a tree-shaped constraint set \mathcal{C} . For all variables x of \mathcal{C} , let $d(x) := g(x) - f(x)$. Calling x_r the root variable of \mathcal{C} , we must have $d(x_r) := 0$ for the root, because it has an exact value constraint by definition of tree-shaped constraint sets.

Now, as f and g differ on \mathcal{C} , there must be a variable x where $d(x) \neq 0$. Without loss of generality, we have $d(x) > 0$. Hence, let us consider a variable x with parent y so that we have $d(x) > d(y)$: as $d(x_r) = 0$, we can find such a variable x by picking a variable which is as high as possible in the tree, such that $d(x) > 0$ but $d(y) = 0$. Necessarily x is not a leaf (as they have exact value constraints, so $d(x) = 0$), so x has children. We show that x has a child x_g such that $d(x_g) > d(x)$.

Consider x_f the child of x such that $f(x_f)$ is minimal among children of x , and x_g defined in the analogous manner for g . Now, as f and g are balanced, by our preliminary observation we have $f(x) = (f(y) + f(x_f))/2$ hence $f(x_f) = 2 \cdot f(x) - f(y)$, and likewise $g(x_g) = 2 \cdot g(x) - g(y)$. But then, by minimality of $g(x_g)$, we have $g(x_g) - f(x_f) \leq g(x_f) - f(x_f)$. Now, we have $g(x_g) - f(x_f) = 2 \cdot d(x) - d(y)$. Now, as we have $d(y) < d(x)$, we have $d(x_f) > d(x)$, which is what we wanted to show.

Now, repeating the argument on x_f , we obtain a child x_f^2 of x_f such that $d(x_f^2) > d(x_f)$. Repeating the argument, we thus build a descending chain of variables x in the tree-shaped constraint set \mathcal{C} along which d is strictly increasing. When we reach the leaves, we obtain a contradiction. This implies that we must have $d(x) = 0$ for all $x \in \mathcal{X}$, so that $f = g$ on \mathcal{C} . Hence, there cannot be two different stable and balanced interpolation schemes on tree-shaped constraint sets. ◀

We now prove the *existence* of a stable and balanced interpolation scheme on trees, which we dub **Stable**, and show that expected values under this scheme can be computed in linear time:

► **Proposition 44.** *There exists a stable and balanced interpolation scheme on tree-shaped constraint sets, and we can compute the interpolated values of all variables according to this scheme in time $O(|\mathcal{X}|^2)$.*

Proof. We compute the interpolation scheme on a tree-shaped constraint set \mathcal{C} top-down. For each variable x which has no exact value constraint or interpolated value, but whose parent y has an exact value or an interpolated value v_y , we consider all leaves z which are descendants of x (and have an exact value constraint to some value v_z), and set the interpolated value of x to be the minimum of linear interpolation from y to z ; namely, letting $d_x(z)$ be the depth of leaf z in the subtree rooted at x , we set $v_x := \min_z \left(v_y + \frac{v_z - v_y}{d_x(z) + 1} \right)$. This can clearly be done in the indicated time bound.

We now show that the resulting interpolation scheme is indeed balanced and stable. It is immediate to observe that this scheme is balanced. We now show that it is stable. Towards this, let us first show that for every variable x with parent y , if z is a leaf that achieved the minimum when interpolating x to its value, then for all variables on the path from x to z , z was also a leaf that achieved the minimum when interpolating their value. Indeed, it suffices to show the claim for the first variable x' of this path, a child of x , and then repeat the argument. We know that, from our choice when interpolating x , by minimality of the interpolation result for x using z , we have $\frac{v_z - v_y}{d_x(z) + 1} \leq \frac{v_{z'} - v_y}{d_x(z') + 1}$, where v_y is the value of the parent y of x ; let us call the first quantity δ and the second δ' . By definition of the interpolation of x , we then have $v_x := v_y + \delta$. Now, consider any leaf z' reachable from x' . We must show that z achieves the minimum when interpolating x' ; in other words, we must compare $\eta := \frac{v_z - v_{x'}}{d_{x'}(z) + 1}$ and $\eta' := \frac{v_{z'} - v_{x'}}{d_{x'}(z') + 1}$, and show that $\eta \leq \eta'$; note that $d_{x'}(z) + 1 = d_x(z)$ and $d_{x'}(z') + 1 = d_x(y)$. The quantity η can then be rewritten as $\frac{d_x(z) + 1}{d_x(z)} \times \frac{1}{d_x(z) + 1} (v_z - v_y - \delta)$, i.e., $\frac{d_x(z) + 1}{d_x(z)} \times \left(\delta - \frac{\delta}{d_x(z) + 1} \right)$, which simplifies to δ : hence, $\eta = \delta$. The quantity η' can be written as $\frac{d_x(z') + 1}{d_x(z')} \times \frac{1}{d_x(z') + 1} (v_{z'} - v_y - \delta)$, i.e., $\frac{d_x(z') + 1}{d_x(z')} \times \left(\delta' - \frac{\delta}{d_x(z') + 1} \right)$, which simplifies to $\frac{(d_x(z') + 1)\delta' - \delta}{d_x(z')}$. Now, as $\delta' \geq \delta$, we deduce that $\eta' \geq \frac{(d_x(z') + 1)\delta - \delta}{d_x(z')}$, so that $\eta' \geq \delta$. Hence, we have $\eta' \geq \eta$, so that the leaf z also achieves the minimum for variable x' . Repeating the argument on the path from x to z , we have shown the claim.

From this initial claim, we deduce the following (*): for any variable x , letting z be a leaf that achieves the minimum when interpolating x (once the value of its parent y is known), then the variables in the path from y to z are interpolated according to linear interpolation on that chain. This is immediate by the previous claim, as all variables on that path are interpolated using linear interpolation from their parent to that same leaf (or another minimal leaf that sets them to the same value).

We now show a similar auxiliary claim. Let us define, once we have interpolated in \mathcal{C} , the function u that maps each non-root variable x to $u(x)$ defined as the interpolated value of x minus that of its parent. We show that (**): for any variables y, x, x' , where y is the parent of x and x is the parent of x' , then $u(x) \leq u(x')$. Indeed, let v_y, v_x and $v_{x'}$ be the interpolated values, and let z' be the witness leaf used to interpolate for x' . By definition of the scheme, we have $u(x') = \frac{v_{z'} - v_x}{d_{x'}(z') + 1}$. Furthermore, letting z be the witness leaf used to interpolate for x , we have $u(x) = \frac{v_z - v_y}{d_x(z) + 1}$. Using the notation above, note that $u(x') = \eta'$ and $u(x) = \delta$. By the same reasoning as for claim (*) to show $\delta = \eta \leq \eta'$, we conclude that $u(x) \leq u(x')$.

We are now ready to show that the scheme is stable. Consider the initial tree-shaped constraint set \mathcal{C} , and let us set a variable x to its interpolated value v_x , yielding \mathcal{C}' . Note that \mathcal{C}' is no longer tree-shaped, but it can be rewritten by Lemma 21 to two tree-shaped constraint sets. It is then clear that all variables that are descendants of x in \mathcal{C} are interpolated in the same manner in \mathcal{C}' and in \mathcal{C} , as the scheme proceeds top-down and the value of x in \mathcal{C}' is by definition the same as its interpolated value in \mathcal{C} . We now show that the ancestors of x in \mathcal{C}

are interpolated in the same way in \mathcal{C}' than in \mathcal{C} , which is clearly sufficient to justify the claim that *all* variables in \mathcal{C}' are interpolated in the same way as in \mathcal{C} . Let us therefore pick an ancestor x' of x , which is neither x nor the root variable, otherwise the claim is trivial; we pick it as high as possible in the tree, so the interpolated value v_y of its ancestor y is the same in \mathcal{C} and in \mathcal{C}' .

We first show that the interpolated value for x' in \mathcal{C}' is no higher than in \mathcal{C} . Assuming to the contrary that it is, then it must be the case that x' was interpolated in \mathcal{C} using as minimal leaf z some leaf which is a descendant of x in \mathcal{C} , as otherwise we can still interpolate using z in \mathcal{C}' and obtain the same result. Now, if x' was interpolated in \mathcal{C} using z as minimal leaf, then, by our preliminary claim (*), x was interpolated in \mathcal{C} following linear interpolation between the parent y of x' and the leaf z . Hence, using the new leaf x in \mathcal{C}' to interpolate x' in \mathcal{C}' yields the same result as the interpolation in \mathcal{C} . Contradiction.

Second, we show that the interpolated value for x' in \mathcal{C}' is no lower than in \mathcal{C} . Assuming to the contrary that it is, then, if x' was interpolated in \mathcal{C}' following a leaf z which is not x , then we immediately reach a contradiction as we should have used the same leaf z to interpolate to the same value in \mathcal{C} . Hence, we must have interpolated x' in \mathcal{C}' using the new leaf x , and x' was interpolated in \mathcal{C}' following linear interpolation between y and x . Let γ be the value difference between two consecutive nodes in \mathcal{C}' on this path, and l the length of the path. Calling $u(x)$ for a variable x the difference between its interpolated value and the value of its parent in \mathcal{C} , we must then have $u(x') > \gamma$, because the value of x' in \mathcal{C} is strictly greater than in \mathcal{C}' , and y has same value in \mathcal{C} and \mathcal{C}' . By preliminary claim (**), we have reached a contradiction, because then the function u always takes values which are $> \gamma$ on the path from x' to x in \mathcal{C} , so that when we reach x we know that the value of x in \mathcal{C} is $> l \cdot \gamma$, contradicting the fact that it is $l \cdot \gamma$, as we know from \mathcal{C}' . \blacktriangleleft

REFERENCES FOR THE APPENDIX

- ACK⁺11** S. Abiteboul, T.-H. H. Chan, E. Kharlamov, W. Nutt, and P. Senellart. Capturing continuous data and answering aggregate queries in probabilistic XML. *TODS*, 36(4), 2011.
- BW91** G. Brightwell and P. Winkler. Counting linear extensions. *Order*, 8(3), 1991.
- Hoe63** W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301), 1963.
- KLS97** R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $o^*(n^5)$ volume algorithm for convex bodies. *Random Struct. Algorithms*, 11(1), 1997.
- Pap79** C. H. Papadimitriou. Efficient search for rationals. *Information Processing Letters*, 8(1), 1979.
- Sch86** A. Schrijver. The structure of polyhedra. In *Theory of Linear and Integer Programming*, chapter 8. Wiley-Interscience, 1986.