

# A Framework for Conditioning Probabilistic XML Data

Ruiming Tang\*    Dongxu Shao<sup>†</sup>    M. Lamine Ba<sup>‡</sup>  
Pierre Senellart<sup>§</sup>    Stéphane Bressan<sup>¶</sup>

January 7, 2015

A probabilistic database denotes a set of non-probabilistic instances called possible worlds, each of which has a probability (that is the confidence of this possible world being actual). This is often a compact way to represent uncertain data. In addition, direct observations and general knowledge, in the form of constraints, help refining the probabilities of the possible worlds, possibly ruling out some of them. Enforcing such constraints on the set of possible worlds, obtaining the valid possible worlds that satisfy the given constraints and refining the probability of each such possible world as the conditional probability of the possible world when the constraint is true, is called *conditioning* the probabilistic database. The problem is to find a new probabilistic database that denotes the valid possible worlds with their new probabilities. Probabilistic XML allows capturing uncertainty of both values and structure. We consider the conditioning problem for probabilistic XML with a language of formulae of independent events to express the probabilistic dependencies among the nodes of the XML tree. In the most general case, conditioning is intractable. For reference, we present an exponential algorithm. We then focus on the specific case of independent events and mutually exclusive constraints. This case goes beyond local mutually exclusive constraints as considered so far in the literature. We devise and present polynomial-time algorithms for conditioning probabilistic XML data in tractable cases.

---

\*School of Computing, National University of Singapore

<sup>†</sup>Institute for Infocomm Research, A\*STAR, Singapore

<sup>‡</sup>Institut Mines-Télécom; Télécom ParisTech; CNRS LTCI Paris, France

<sup>§</sup>Institut Mines-Télécom; Télécom ParisTech; CNRS LTCI & NUS; CNRS IPAL, Paris & Singapore

<sup>¶</sup>School of Computing, National University of Singapore

# 1 Introduction

Uncertainty of data, in its various forms, naturally arises from such applications as information extraction [9], information integration [26, 11] and version control [6].

Probabilistic databases address the problem of the management and of the representation of uncertain data by means of probabilities. A good probabilistic database model offers a generally compact and easily manageable representation of uncertain data. A probabilistic database instance denotes a set of possible deterministic database instances called *possible worlds*, each of which has a probability (that is the confidence of this possible world being actual).

Direct observations and general knowledge, in the form of constraints, can be injected into the database during a data cleaning process or during an auditing phase by domain experts, for instance. These constraints help refining the probabilities of the possible worlds, possibly ruling out some of them. Enforcing such constraints on the set of possible worlds, obtaining the valid possible worlds that satisfy the given constraints and refining the probability of each such possible world as the conditional probability of the possible world when the constraint is true, is called *conditioning* the probabilistic database. The problem is to find a new probabilistic database that denotes the valid possible worlds with their new probabilities.

Probabilistic relational databases [8, 7, 23, 12] represent uncertainties at value or tuple level, while schema is constrained. The conditioning problem in probabilistic relational databases has been studied in [19] and [24], respectively. Koch and Olteanu [19] claim that relational conditioning is NP-Hard. They present a general but exponential time algorithm as well as efficient heuristics and decomposition methods. Tang et al. [24] identify tractable scenarios for which they devise polynomial time algorithms. We recognize that the idea of probabilistic databases and the idea of conditioning are borrowed and adapted from concepts and vocabulary in artificial intelligence research [21].

In contrast, probabilistic XML databases [18], leveraging the schema independence of XML, can represent uncertainties not only at the value but also at the structural level (see Section 2 for details). The concept of *p-documents* [1, 18] is a general framework encompassing various probabilistic XML models from the literature [20, 26, 2, 16]. It provides a compact way for representing probabilistic XML databases, that is, a probability distribution over a set of possible XML documents.

In this paper, we consider the conditioning problem for probabilistic XML with a language of formulae of independent events to express the probabilistic dependencies among the nodes of the XML tree. The tree-like structure of XML data and the fact that probabilistic XML captures uncertainty of both values and structure introduce new interesting challenges for the conditioning problem.

We extend with constraints the most expressive and succinct (update-efficient [16]) family of *p-documents*, namely PrXML<sup>fie</sup> [16], which consists of annotations made of propositional formulae over a set of independent random Boolean variables. We consider an XML query language to express constraints.

Note that even though we aim at compact models (e.g., *p-document* polynomial in the size of the largest possible world), a simple counting argument shows that there are no compact representation of arbitrary probability distributions over XML documents (see also Proposition 3).

Let us illustrate the conditioning problem for probabilistic XML with a simple student database

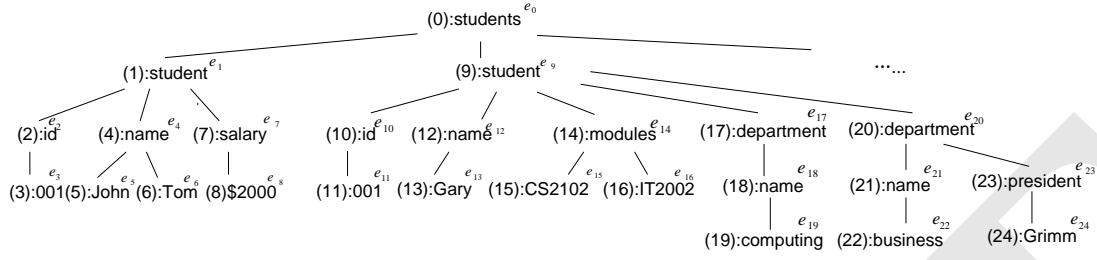


Figure 1: Example PrXML<sup>fie</sup> p-document: data about students

and mutual exclusiveness constraints. Consider the PrXML<sup>fie</sup> p-document shown in Figure 1, which represents a simple student database. The uncertainty of the values and structure is captured with independent events. The tree is in PrXML<sup>ind</sup>. For instance, the probability of event  $e_{13}$  is the probability of the student represented by the record rooted at node 9 to be called Gary when the parent node, node 12 (and consequently nodes 9 and 0) exists.

The given p-document represents prior knowledge about students and their dependencies. Further new dependencies (which might translate natural observations and new constraints) can be used to refine this knowledge. These dependencies map to natural constraints over some nodes in the p-document. Here are some examples of such dependencies representing mutual exclusiveness constraints (for simplicity “mutually exclusive constraints”):

- *Constraint 1 – Mutually Exclusive Siblings (MES)*: each student has only one name, i.e., node 5 and 6 are mutually exclusive.
- *Constraint 2 – Mutually Exclusive Ancestor-Descendant (MEAD)*: a student cannot have a salary, i.e., node 1 and node 7 are mutually exclusive.
- *Constraint 3 – Mutually Exclusive Descendance (MED)*: id is unique, i.e., node 3 and node 11 are mutually exclusive.
- *Constraint 4 – MED & AD* (combination of *MED* and *MEAD*): each student belongs to at most one department and a department cannot have a president, i.e., node 17, node 20 and node 23 are mutually exclusive.

These are precisely the kind of constraints we describe how to enforce.

The contributions of this article are as follows. (i) A general study of the problem of conditioning probabilistic XML for various types of constraints, when constraints and dependencies among nodes are arbitrary. (ii) Tractable algorithms for conditioning probabilistic XML data in the specific case of independent events and four classes of mutually exclusive constraints given in a rather simplified but reasonable query language (tree-pattern queries). This case goes beyond local mutually exclusive constraints as considered so far in the literature (i.e., PrXML<sup>ind,mux</sup> from [18]).

We start by reviewing state-of-the-art probabilistic XML models and conditioning probabilistic databases in Section 2. Then, we present in Section 3 the XML data model we consider and the extension of the PrXML<sup>fie</sup> system with constraints. In Section 4, we explain how constraints can be expressed as queries. We then proceed in Section 5 to provide general lower and upper bounds on the conditioning problem. We describe the four classes of nodes mutually exclusive constraints in Section 6. Tractable algorithms are presented for individual classes of mutually exclusive

constraints in Sections 7, 8, 9, respectively. Due to lack of space, some proofs and technical content are omitted. An extended version of this article, with all this material, is available in [25].

## 2 Related Work

This section reviews state-of-the-art probabilistic XML representation systems and the conditioning of probabilistic databases.

**Probabilistic XML models** A probabilistic XML document captures the description of a probability distribution over a space of ordinary XML documents. The p-document setting [1] is a general probabilistic XML representation system modeling this distribution in terms of a probabilistic process that generates an ordinary random XML document (seen as an unranked, labeled, and unordered tree).

Informally, a p-document is a special XML tree with *distributional nodes* in addition to the *usual* (regular) XML nodes. While distributional nodes cannot appear in ordinary XML documents, they are key structure elements used to specify a probability distribution over the subsets of their children. In other words, distributional nodes enable to define the process of obtaining random documents. As surveyed in [18, 1] existing probabilistic XML representation systems differ in the distributional nodes that they consider in their p-documents. The set of distributional nodes can be subdivided in two groups as follows.

- local distributional nodes: ind (for independent choices), mux (for mutually exclusive choices), exp (for explicit choices) and det (for deterministic choices);
- global distributional nodes: cie (for conjunction of independent events) and fie (for formula of independent events).

Similarly to [18, 1] we denote a given probabilistic XML data model by  $\text{PrXML}^{\mathcal{X}}$ , where  $\mathcal{X} \subseteq \{\text{ind}, \text{mux}, \text{det}, \text{exp}, \text{cie}, \text{fie}\}$  contains distributional nodes considered in this model. We mostly focus in this paper on  $\text{PrXML}^{\text{fie}}$  documents. For completeness, we review most commonly studied probabilistic XML models.

$\text{PrXML}^{\text{mux}}$ ,  $\text{PrXML}^{\text{ind}}$ ,  $\text{PrXML}^{\{\text{mux}, \text{ind}\}}$  [20, 17, 10, 26], and  $\text{PrXML}^{\text{exp}}$  [14, 13] are proposed PrXML systems built on local distributional nodes. Such probabilistic XML representation models describe local dependencies between regular nodes, that is, each distributional node selects a subset of children independently from choices of other distribution nodes.

$\text{PrXML}^{\text{cie}}$  [2] and  $\text{PrXML}^{\text{fie}}$  [16] are global dependency models handling in addition long-distance dependencies between nodes – the aforementioned two models support local dependencies as in the first family of probabilistic XML systems. Instead of assigning directly probability values to children or set of children of distributional nodes,  $\text{PrXML}^{\text{cie}}$  and  $\text{PrXML}^{\text{fie}}$  attach respectively conjunctions of independent events and formulae of independent events to them. Each event represents a Boolean random variable with a probability of being true. Different distributional nodes can share common events, therefore the choice of a distributional node might also correlate choices of some other distributional nodes. Note that  $\text{PrXML}^{\text{cie}}$  p-documents belong also to  $\text{PrXML}^{\text{fie}}$  but there is no efficient translations from the latter to the former [16].  $\text{PrXML}^{\text{fie}}$  has been shown in [16] to be the most *expressive* and *succinct* probabilistic XML

representation system in the literature. In this paper, we propose an extended PrXML<sup>fie</sup> data model with *additional constraints* integrated as *first class citizens*. The model we present caters for constraints rather than treating them as add-ons.

**Conditioning probabilistic data** There are two existing works on conditioning probabilistic relational data, namely [19] and [24]. The authors of [19] propose an approach to do conditioning probabilistic relations. They adapt algorithms and heuristics for Boolean validity checking and simplification to solve the general NP-hard conditioning problem. In [24], some of the authors of the present work identify special practical families of constraints (i.e., *observation* and *X-tuple* constraints) for which efficient algorithms are presented.

We consider in this paper XML documents annotated with independent events; this is conceptually a similar setting as in [24], where the authors consider probabilistic relations where tuples are annotated with independent events. However, as we shall see further in the paper, the difference in structure between relations and XML make tractable cases in both scenarios quite different: indeed, though one can encode a probabilistic XML document as a relation in a straightforward manner [4], resulting tuples are *not* annotated with independent events, because of the implicit dependency between the existence of a node and that of its parent. In other words, the setting we consider in this paper is more general than the one in [24].

The problem of the evaluation of constraints in probabilistic XML has only been investigated in [10]. Given a probabilistic XML document and a specific constraint in a pre-defined language, Cohen et al. study in [10] three problems: *constraint satisfaction*, *query evaluation*, and *sampling*. They do not consider, however, how to enforce the constraints into the probabilistic XML document, so that possible worlds of the updated probabilistic XML document always satisfies the constraints, which is what we do in this paper. The reason is that they are restricted to a local-dependency probabilistic XML model, i.e., PrXML<sup>{mux,ind}</sup>. In this model, materializing constraints which imply global-dependencies would result in exponential blow-up even in very simple cases.

Updating is also related to conditioning. In [16], the authors define the semantics of two elementary kinds of updates, *insertions* and *deletions*, using a *locator query*<sup>1</sup>. The main result of this study is that PrXML<sup>fie</sup> is efficient for these classes of updates. The problem of updating probabilistic XML is relevant to conditioning in the sense that insertion is actually specifying certain nodes or subtrees should be there (their corresponding formulae should be true) and deletion is specifying some nodes or subtrees should not be there (their corresponding formulae should be false). However, updating is not able to deal with other constraints as considered in conditioning.

### 3 Data Model

We now present our data model: deterministic trees, syntax and semantics of probabilistic XML documents, conditioning.

---

<sup>1</sup>A locator query is a tree-pattern query specifying the nodes where the update is to be performed.

**Trees and XML documents** Given an unordered, directed tree  $t$ , we consider  $V(t)$  and  $E(t)$  as, respectively, the set of nodes and edges of  $t$  – the special node  $\text{root}(t)$  refers to the root node of this tree. A given node  $n_i$  in  $t$  has (i) a unique identifier and; (ii) a (possibly shared) label which we denote by  $\text{label}(n_i)$ . Any two nodes  $n_1, n_2 \in V(t)$  such that  $(n_1, n_2) \in E(t)$  are in *parent-child relationship*, that is,  $n_1$  is the parent of  $n_2$  and  $n_2$  is a child of  $n_1$ . We use  $\text{parent}(n)$  to represent the parent of node  $n$ . Two nodes  $n_2, n_3$  are *siblings* if  $(n_1, n_2), (n_1, n_3) \in E(t)$ . The node  $n_1$  is an ancestor of  $n_2$  (or  $n_2$  is a descendant of  $n_1$ ) if there exists a path from  $n_1$  to  $n_2$ . The parent-child relationship is a special case of ancestor-descendant relationship. We use  $\text{path\_node}(n_i, n_j)$  to represent the set of nodes along the path from  $n_i$  to  $n_j$ , where  $n_i$  is an ancestor of  $n_j$ . Finally, we define  $\text{LCA}(N)$  as the *lowest common ancestor* of the set of nodes  $N$  in the tree.

We model an *XML document* as an unordered directed tree with node labels. Throughout this paper, for ease of presentation, we use tree and XML document interchangeably.

**Probabilistic XML** We adapt the probabilistic relational data model in [24] to a probabilistic XML data model.

**Definition 1.** (*Events, complex events and formulae*) Let  $E$  be a set of symbols called **events** ( $e$ ). A **formula** is a **complex event** ( $ce$ ), which is a well formed formula of propositional logic in which events are propositions:  $ce = e \mid ce \vee ce \mid ce \wedge ce \mid ce \rightarrow ce \mid \neg ce$ . We denote by  $\mathbb{F}(E)$ , the set of formulae formed with the events in  $E$ .

**Definition 2.** (*Probabilistic XML document*) A **probabilistic XML document** (or **p-document**) is a quintuple  $\langle D, E, f, C, p \rangle$ , where  $D$  is a normal XML document,  $E$  is a set of events,  $f$  is a function from  $V(D)$  to  $\mathbb{F}(E)$  (associating each node in the document with a formula),  $C$  – the constraint – is an element of  $\mathbb{F}(E)$ , and  $p$  is a probability function from  $E$  to  $(0, 1]$ . If  $C$  is empty, the probabilistic XML document is said to be **unconstrained**.

The function  $f$  has a different semantics from that of [24] where a given tuple  $t$  is actual if and only if  $f(t)$  is true. Here, a node  $n$  is actual if and only if  $\text{parent}(n)$  exists and  $f(n)$  is true.

**Definition 3.** (*Interpretation, model*) An **interpretation** of a formula  $ce$  is an assignment of each event in  $ce$  to  $\{\text{true}, \text{false}\}$ . A **model** of a formula  $ce$  is an interpretation of  $ce$  that makes  $ce$  true. The set of models of  $ce$  is denoted as  $\mathcal{M}(ce)$ .

**Definition 4.** (*Probability of a formula*) Given a probability function  $p$  over a set  $E$  of events, The **probability** of a formula  $ce \in \mathbb{F}(E)$ , denoted as  $p(ce)$  is

$$p(ce) = \sum_{I \in \mathcal{M}(ce)} \left( \prod_{I(e)=\text{true}} p(e) \prod_{I(e)=\text{false}} (1 - p(e)) \right)$$

where an assignment  $e$  ranges over all the events in  $ce$ .

For ease of presentation, in this paper, we use  $\text{PrXML}^{\text{ind}}$  (respectively,  $\text{PrXML}^{\text{fie}}$ ) to denote a probabilistic XML data model, such that in a probabilistic XML document  $\langle D, E, f, C, p \rangle$ ,  $f$  assigns an independent event (respectively, a formula of events) to each node in  $D$ .

**Possible worlds** A possible world of a probabilistic XML document is an ordinary XML document, obtained as a subtree of the probabilistic XML document such that (1) for each existing node, the formulae of this node and its ancestors are true; (2) for each non-existing node, at least one formula of this node or its ancestors are false; (3) the constraint is true. The probability of a possible world is the probability of the model(s) satisfying the above conditions.

**Definition 5.** (*Possible world*) Let  $\mathcal{D}=\langle D, E, f, C, p \rangle$  be a probabilistic XML document.  $D'$  is a **possible world** of  $\mathcal{D}$  if and only if the following formula  $F$  has non-zero probability.

$$F = \bigwedge_{n \in D'} \bigwedge_{n_i \in \text{path\_node}(\text{root}(D'), n)} f(n_i) \wedge \bigwedge_{n \in D - D'} \neg \left( \bigwedge_{n_i \in \text{path\_node}(\text{root}(D'), n)} f(n_i) \right) \wedge C.$$

We denote  $p_{\mathcal{D}}(D') = p(F | C) = \frac{p(F)}{p(C)}$  the probability of the possible world  $D'$ , and  $P(\mathcal{D})$  the set of possible worlds of  $\mathcal{D}$ .

With all these definitions in, we need to introduce the important notion of consistency of a constraint with respect to the structural annotations of a probabilistic XML document.

**Definition 6.** (*Consistency*) Let  $\mathcal{D}=\langle D, E, f, C, p \rangle$  be a probabilistic XML document.  $\mathcal{D}$  is **consistent** (resp., **inconsistent**) if and only if there exists a possible world (resp., there does not exist a possible world) of  $\mathcal{D}$ , i.e.,  $P(\mathcal{D}) \neq \emptyset$  (resp.,  $P(\mathcal{D}) = \emptyset$ ).

It is easy to see that consistency only depends on the constraint:

**Lemma 1.** Let  $\mathcal{D}=\langle D, E, f, C, p \rangle$  be a probabilistic XML document.  $\mathcal{D}$  is inconsistent if and only if  $C$  is always evaluated to be false.

It is sometimes convenient to talk about the probability of an individual node:

**Definition 7.** (*Probability of a node*) Given a probabilistic XML document  $\mathcal{D}=\langle D, E, f, C, p \rangle$ , the **probability** of a node  $n \in V(D)$ , denoted by  $p_{\mathcal{D}}(n)$ , is defined as:

$$p_{\mathcal{D}}(n) = \sum_{D' \in P(\mathcal{D}), n \in V(D')} p_{\mathcal{D}}(D').$$

**Equivalent probabilistic XML documents** We introduce an equivalence relation between probabilistic XML documents under the *possible world semantics*.

**Definition 8.** (*World equivalence*) Given two probabilistic XML documents  $\mathcal{D}_1=\langle D, E_1, f_1, C_1, p_1 \rangle$  and  $\mathcal{D}_2=\langle D, E_2, f_2, C_2, p_2 \rangle$ , we say that  $\mathcal{D}_1$  and  $\mathcal{D}_2$  are **world-equivalent**, denoted by  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$ , if and only if

$$D' \in P(\mathcal{D}_1) \iff D' \in P(\mathcal{D}_2) \quad \text{and} \quad p_{\mathcal{D}_1}(D') = p_{\mathcal{D}_2}(D').$$

According to Definition 8,  $\equiv_w$  is an equivalence relation, because it is reflexive (i.e.,  $\mathcal{D}_1 \equiv_w \mathcal{D}_1$ ), symmetric (i.e., if  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$  then  $\mathcal{D}_2 \equiv_w \mathcal{D}_1$ ) and transitive (i.e., if  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$ ,  $\mathcal{D}_2 \equiv_w \mathcal{D}_3$  then  $\mathcal{D}_1 \equiv_w \mathcal{D}_3$ ).

A fundamental property, that allows considering the conditioning operation, is as follows:

**Theorem 1.** Let  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  be a probabilistic XML document. If  $\mathcal{D}_1$  is consistent, then there exists an unconstrained probabilistic XML document  $\mathcal{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$  such that  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$ .

*Proof.* This is essentially a corollary of Proposition 5.7 in [1] which states that an arbitrary finite probability distributions over XML documents can be represented by a probabilistic XML document using mux and det nodes. However, the number of nodes of the resulting probabilistic XML document is exponentially larger than that of individual possible worlds because all possible worlds are represented. We give a slightly different proof, that maintain the same document  $D$  (but may introduce exponentially large node annotations).

We normalize  $C$  to be its full disjunctive normal form (an exponential blowup may occur here) and denote each conjunct by  $K_i$  for  $1 \leq i \leq q$ . Each conjunct  $K_i$  maps to one possible world  $pwd(K_i)$  (where  $pwd(K_i)$  is generated by the model of  $K_i$ ). For each node  $n \in V(D)$ , we need to check whether  $n$  is in a possible world  $pwd(K_i)$  by verifying whether the model of  $K_i$  is a model of  $\bigwedge_{n' \in \text{path\_node}(\text{root}(D), n)} f_1(n')$ .

We set  $f_2(n) = \bigvee_{n \in pwd(K_i)} K_i$ . This makes sure that  $\mathcal{D}_2$  has the same set of possible worlds as  $\mathcal{D}_1$ . The probability of  $K_i$  is computed as  $p_2(K_i) = \frac{p_1(K_i)}{p(C)}$ . This equation guarantees that  $\mathcal{D}_2$  and  $\mathcal{D}_1$  have the same probability for the same possible world.

Note that the new events  $K_i$ 's are mutually exclusive. We can use a set of  $(q-1)$  independent events  $a_i$ 's to represent  $K_i$ 's, as:  $K_1 = a_1, K_i = \neg a_1 \wedge \dots \wedge \neg a_{i-1} \wedge a_i (i = 2, \dots, q-1), K_q = \neg a_1 \wedge \dots \wedge \neg a_{q-1}$ . The probabilities  $p_2(a_i), i \in [1, q-1]$  can be easily computed by knowing  $p_2(K_i), i \in [1, q]$ .

The last step is replacing mutually exclusive events  $K_i$ 's by independent events  $a_i$ 's in  $f_2$ . By the construction above, we obtain  $\mathcal{D}_2$  such that  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$ .  $\square$

**Conditioning** Assume the p-document  $\mathcal{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$  together with a set of constraints  $C$ . Conditioning  $\mathcal{D}_1$  with  $C$  consists in enforcing  $C$  into  $\mathcal{D}_1$ , that is, computing the resulting probabilistic XML document  $\mathcal{D}_c = \langle D, E_1, f_1, C_1 \cup C, p_1 \rangle$ .

This conditioning problem is solved by finding a world-equivalent probabilistic XML document with empty constraint, given the probabilistic XML document with a constraint.

**Definition 9.** (The conditioning problem) Given  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$ , **the conditioning problem** is to find an unconstrained probabilistic database  $\mathcal{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$  such that  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$ .

The existence of such a probabilistic XML document with no constraint is a direct consequence of Theorem 1.

**Corollary 1.** Let  $\mathcal{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$  be a consistent probabilistic XML document. The conditioning problem over  $\mathcal{D}_1$  always has a solution.

## 4 From Query to Constraint



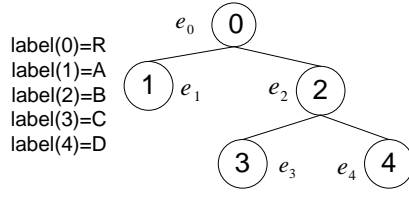


Figure 2: p-Document with independent choices

This section shows how constraints are naturally expressed as semantic constraints among nodes of a document identified by a query. In this paper, we mainly focus on mutually exclusive constraints among nodes .

The query language we consider is *twig* queries [5] (or *tree pattern* queries), also known as the *downward navigational* fragment of XPath. We occasionally reuse XPath notation.

A *mutually exclusive constraint* (or *mutex* constraint for short) over a set  $\{x_1, x_2, \dots, x_q\}$  of nodes indicates that only one of the nodes in this set exists. The corresponding query consists of two parts: (1) at least one of these  $q$  nodes exists; (2) any two nodes cannot exist together. Given that, the expression of this query can be specified within our query language as  $Q = (\bigvee_{i=1}^q q_i) \wedge \bigwedge_{i,j \in [1,q] \wedge i < j} \neg(q_i \wedge q_j)$ , where  $q_i$  is the twig query expressing that node  $x_i$  exists, i.e., the node existence constraint previously referred to.

**Example 1.** Consider the probabilistic XML document in Figure 2. Consider the constraint that one of node 1 and node 3 exists. The query is  $((/R/A) \vee (/R/B/C)) \wedge \neg((/R/A) \wedge (/R/B/C))$ . The global formula is:

$$((e_0 \wedge e_1) \vee (e_0 \wedge e_2 \wedge e_3)) \wedge \neg(e_0 \wedge e_1 \wedge e_2 \wedge e_3).$$

## 5 General case

In this section, we establish general lower and upper bounds for the conditioning problem. We consider two subproblems: tractability of the conditioning, in terms of time complexity; compactness of representation of an unconstrained probabilistic XML document equivalent to a constrained one.

**Time complexity** We first consider the time complexity of the conditioning problem. In the general case where both the constraint  $C$  and formulae associated to nodes are general elements of  $\mathbb{F}(E)$ , an EXPTIME upper bound is easy to obtain: enumerate all possible worlds, and construct an unconstrained probabilistic XML document that regroups all such documents, with their corresponding probabilities. A slightly more subtle approach, but that still yields an exponential-time algorithm, was introduced in [19]. It relies on the notion of *ws-sets* (which corresponds, in our terminology, to constraints in disjunctive normal) and *ws-trees* (a tree structure used to compute the probability of constraints and to perform conditioning). We present this approach, relying on the results of [19], as Algorithm 1.

The conditioning algorithm in [19] takes a probabilistic database and ws-sets (i.e., a constraint) as input, and outputs a probabilistic database that includes only the possible worlds in the ws-sets, with re-defined probabilities. Before conditioning, we encode a probabilistic XML document as a probabilistic relation, as: (1) constructing a probabilistic relation with only one attribute which stores the identifiers of node, (2) the formula associated with the tuple representing node  $n$  being  $F_1(n) = \bigwedge_{n' \in \text{path\_node}(\text{root}(D), n)} f_1(n')$  (line 2 in Algorithm 1). This probabilistic relation gives the probabilistic conditions of all nodes of the probabilistic XML document  $\mathcal{D}_1$ . The parent-child relationship in  $\mathcal{D}_1$  is reflected in the formula of the probabilistic relation. We use the conditioning algorithm in [19] (line 3 in Algorithm 1) in this probabilistic relational setting.

---

**Algorithm 1:** General conditioning algorithm for probabilistic XML data

---

**Data:**  $\mathcal{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$

**Result:**  $\mathcal{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$  such that  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$

- 1 **foreach** node  $n$  in *breadth-first-traversal of  $D$*  **do**
  - 2   |  $F_1(n) \leftarrow \bigwedge_{n' \in \text{path\_node}(\text{root}(D), n)} f_1(n')$ ;
  - 3  $F_2 \leftarrow$  result of the conditioning algorithm in [19] applied with  $F_1(n)$  as the formulae for a tuple representing node  $n$ ;
  - 4 **foreach** node  $n$  in *breadth-first-traversal of  $D$*  **do**
  - 5   | Let  $\alpha_n$  such that  $F_2(n) = f_2(\text{parent}(n)) \wedge \alpha_n$ ;
  - 6   |  $f_2(n) \leftarrow \alpha_n$ ;
- 

The exponential-time running time is a direct consequence of the running time of the conditioning algorithm of [19]. We now show the correction of this algorithm:

**Proposition 1.** *Algorithm 1 solves the conditioning problem, as long as the input document is consistent.*

*Proof.* For each  $D' \in \mathcal{P}(\mathcal{D}_2)$ , according to Theorem 5.3 of [19],

$$p_{\mathcal{D}_2}(D') = \frac{p_1(\bigwedge_{n \in D'} F_1(n) \wedge \bigwedge_{n \notin D'} \neg F_1(n) \wedge C)}{p_1(C)}.$$

But then, according to Definition 5,

$$\frac{p_1(\bigwedge_{n \in D'} F_1(n) \wedge \bigwedge_{n \notin D'} \neg F_1(n) \wedge C)}{p_1(C)} = p_{\mathcal{D}_1}(D').$$

Therefore  $p_{\mathcal{D}_2}(D') = p_{\mathcal{D}_1}(D')$ . Similarly, we show that  $\forall D' \in \mathcal{P}(\mathcal{D}_1)$ , we have  $p_{\mathcal{D}_1}(D') = p_{\mathcal{D}_2}(D')$ .  $\square$

Conditioning still requires checking consistency. This operation itself is actually intractable for any non-trivial query, which leaves little hope of having a polynomial-time conditioning algorithm in the general case where node annotations are arbitrary (which correspond to the PrXML<sup>fie</sup> case).

**Proposition 2.** *Checking the consistency of the constraint obtained by any satisfiable node existence query  $Q$  over a probabilistic XML document is NP-hard.*

*Proof.* We simply reduce from SAT. Let  $\phi$  be an arbitrary propositional formula. We consider a deterministic model of the query  $Q$  (supposed to exist as the query is satisfiable), where the target node of the query is annotated with  $\phi$  and all other nodes are annotated with true. Then the global constraint is  $\phi$  and thanks to Lemma 1, checking consistency amounts to checking satisfiability of  $\phi$ .  $\square$

**Compactness of representation** We now turn to compactness of the conditioned probabilistic XML document. Does there always exist an unconstrained probabilistic XML document that has comparable size to the input document? We first show thanks to a simple counting argument that if constraints are completely arbitrary and their size is not counted as part of the input, conditioning can result in trees of exponential size:

**Proposition 3.** *For all  $k \geq 1$  there exists a probabilistic XML document  $\mathcal{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$  with*

$$|D| + |E_1| + \sum_{n \in D} |f_1(n)| = O(k)$$

*such that every unconstrained probabilistic XML document  $\mathcal{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$  with  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$  satisfies  $\sum_{n \in D} |f_2(n)| = \Omega(2^k)$ .*

*Proof.* For a fixed  $k$ ,  $D$  is a tree with  $k$  distinct children, each of them being annotated with an independent event  $e_i$ , each having probability  $\frac{1}{2}$ . There are  $2^k$  possible worlds for  $\langle D, E_1, f_1, \emptyset, p_1 \rangle$ . There are therefore  $2^k - 1$  consistent probabilistic documents of the form  $\langle D, E_1, f_1, C, p_1 \rangle$  when  $C$  varies. Necessarily, it is not possible to describe each of these documents using  $o(2^k)$  bits for all such constraints.  $\square$

Assuming the constraint is not part of the input, or that the query that generated the constraint can be completely arbitrary, is unreasonable however. What we want is to limit the expressiveness by considering a fixed query language (such that the one of Section 4) and determine whether for this query language, a blowup can occur when conditioning. The following result relates this problem to a long-standing open problem:

**Proposition 4.** *Assume there exists an NP-definable query  $Q$  satisfying the following: for all  $k \geq 1$  there is a consistent probabilistic XML document  $\mathcal{D}_1$  of size  $k$  constrained by  $Q$  such that no unconstrained document  $\mathcal{D}_2$  having the same set of possible worlds has representation size in  $O(k^2)$ . Then there exists an NP problem for which all circuits are supra-linear.*

*The probabilistic document  $\mathcal{D}_1$  can always be taken to be of depth 1 with independent events on all nodes.*

*Proof.* We assume by contraposition that all NP problems have linear-sized circuits.

Let  $L$  be the language of an arbitrary NP problem, and  $(C_k)_{k \geq 1}$  the corresponding family of linear-sized circuit; we fix an arbitrary instance size  $k \geq 1$ . Let  $\mathcal{D}_1$  be a p-document of depth 1 and  $k$  nodes  $n_1, \dots, n_k$ , with independent events  $(e_i)_{1 \leq i \leq k}$  on all nodes, corresponding to the inputs of the circuit  $C_k$ , and let  $Q$  be a query that expresses the NP problem over this document (the

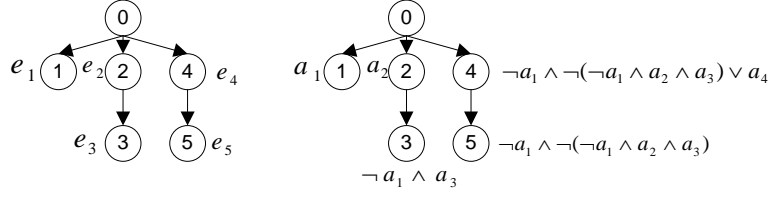


Figure 3: Two probabilistic XML documents in Example 2

result of the query is true in a possible world of  $\mathcal{D}_1$  if and only if the circuit evaluates to true for the corresponding valuation of its inputs).

Since  $\mathcal{D}_1$  is consistent, it has at least one possible world. Let  $S$  be the set of events  $e_i$  that are true in one such possible world.

We use the folklore trick of coding circuits to Boolean expressions linear in the size of the circuit [22]. Let  $\phi$  be the corresponding encoding of  $C_k$  as a Boolean expression. We take for  $\mathcal{D}_2$  a p-document with the same structure as  $\mathcal{D}_1$  and with node  $n_i$  annotated with condition  $e_i \wedge \phi$  if  $e_i \notin S$ , and  $(e_i \wedge \phi) \vee \neg\phi$  otherwise. Since  $\mathcal{D}_2$  is of the same structure size as  $\mathcal{D}_1$ , and each of its node is annotated with a linear-sized formula, it has representation size quadratic in  $k$ . Observe that  $\mathcal{D}_2$  has exactly the same set of possible worlds as the conditioning of  $\mathcal{D}_1$  by  $Q$  (in particular, when  $\phi$  is not satisfied by a valuation, we obtain a valid possible world).  $\square$

The existence of an NP (or even PTIME) problem with a supra-linear circuit is a long-standing open problem [15] that has potential applications to the  $P \stackrel{?}{=} NP$  problem [3].

There is therefore little hope to find reasonable queries that would force a blowup worse than quadratic of any unconstrained document having the same set of possible worlds.

However, note that having the same set of possible worlds is not enough: one also needs to get the right probability distributions. There are cases where an unconstrained document has the same set of possible worlds, but is not world-equivalent:

**Example 2.** Consider the probabilistic XML document  $\mathcal{D}_1$  presented on the left-hand-side of Figure 3 and its constraint is that only one of node 1, 3, 5 exists. The probabilistic XML document  $\mathcal{D}_2$  is on the right-hand-side in Figure 3.

$\mathcal{D}_1$  and  $\mathcal{D}_2$  have the same set of possible worlds as presented in Figure 4. The corresponding formulae of possible worlds in  $\mathcal{D}_1$  and  $\mathcal{D}_2$  are also presented below each possible world.

In order to determine the probability values of  $p_2(a_i)$ , one has to resolve the set of equations stating that the probabilities of each possible world of  $\mathcal{D}_1, \mathcal{D}_2$  are the same. However, one can show that this set of equations does not admit any valid solution.

Thus, it is still open whether there are cases of constraints defined by simple queries, and where world-equivalent unconstrained documents are substantially larger than constrained ones.

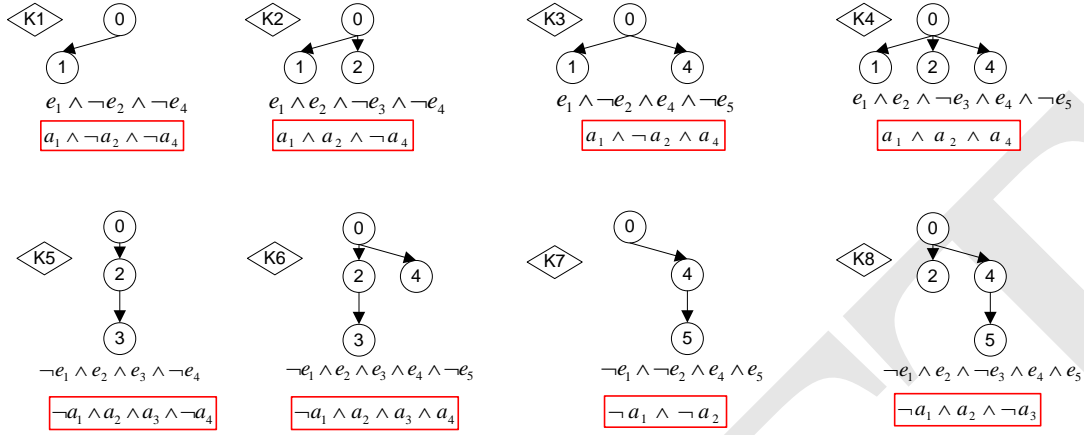


Figure 4: Eight possible worlds for probabilistic XML documents in Figure 3

## 6 mutually exclusive constraints

In this section, we introduce the kinds of constraints we further consider, as special cases for which we will then present polynomial-time algorithms in the following sections; we also introduce some important definitions. We first define four classes of mutually exclusive constraints considered in this paper and especially three particular semantics. One particular technical tool relates to the *relevant part* of a probabilistic XML document for a given constraint, that we study next. We then introduce the notion of *possible worlds according to the relevant part* of a probabilistic XML document for a constraint.

**Data model and Constraints** We restrict our study to mutually exclusive constraints over a set of nodes  $N = \{x_1, x_2, \dots, x_q\}$ . We consider two semantics: *with maybe semantics* and *without maybe semantics*. A mutually exclusive constraint with maybe semantics, called *WMB* (With-MayBe) means that at most one node exists. A mutually exclusive constraint without maybe semantics means that exactly one node exists (this is the variant presented in Section 4). Under this semantics, there are two sub-cases: (1) the first one that we denote by *WOMBA* (standing for, *WithOut-MayBe-Absolutely*) translates the fact that exactly one node exists; (2) the second means that exactly one node exists if the lowest common ancestor exists – we refer to this latter by *WOMBI*, i.e. *WithOut-MayBe-If*.

In addition, we consider a special class of probabilistic XML documents, namely *probabilistic XML documents with independent events* ( $\text{PrXML}^{\text{ind}}$ ) as input of the conditioning problem. In a probabilistic XML document  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  of this special class, for all two nodes  $x_i, x_j \in V(D)$  and independent events  $e_i, e_j$ , when  $f_1(x_i) = e_i$  then  $x_i \neq x_j \Rightarrow e_i \neq e_j$ . These two restrictions will allow us to propose algorithms despite the general intractability result.

We present below the four classes of mutually exclusive constraints we consider throughout the rest of this paper.

- *Mutually Exclusive Siblings (MES) Constraints*: all nodes in  $N$  are siblings, i.e.

$$\forall x_i, x_j \in N \quad \text{parent}(x_i) = \text{parent}(x_j).$$

- *Mutually Exclusive Ancestor-Descendant (MEAD) Constraints*: there is a node  $x$  in  $N$  such that  $x$  is the lowest common ancestor of any two nodes in  $N$ , i.e.

$$\exists x \in N \forall x_1, x_2 \in N \quad x = \text{LCA}(x_1, x_2)$$

- *Mutually Exclusive Descendance (MED) Constraints*: any two distinct pairs of couple of nodes in  $N$  have the same lowest common ancestor, i.e.

$$\forall x_i, x_j, x_a, x_b \in N \quad \text{LCA}(x_i, x_j) = \text{LCA}(x_a, x_b) \notin N.$$

- *MED&AD Mutually Exclusive Constraints*: combination of *MED* and *MEAD* constraints. The set of mutually exclusive nodes  $N$  can be divided into two disjoint sets as  $N = X \cup Y$  and  $X \cap Y = \emptyset$ . The set of nodes  $X = \{x_1 \dots x_g\}$  has the property that:

$$\forall x_h, x_l, x_r, x_s \in X \quad \text{LCA}(x_h, x_l) = \text{LCA}(x_r, x_s) \notin N.$$

$Y = \{y_1 \dots y_b\}$  can be divided into  $g$  disjoint subsets  $Y_1 \dots Y_g$ . The set of nodes  $x_i \cup Y_i$  has the property that  $\forall y_1, y_2 \in Y_i (x_i = \text{LCA}(y_1, y_2))$ , where  $i \in [1, g]$ . Note that  $q = g + b$ .

We will mostly focus on the WOMBA semantics in the following, details for the other semantics can be found in [25]. Conditioning on *MED&AD Mutually Exclusive Constraints* is omitted in the article due to space limit, and it is included in [25].

**Local Tree and Local Possible Worlds** We introduce the concept of *local tree* as being the relevant part of a given probabilistic XML document under constraints. We formally define this local tree, as well as the set of corresponding local possible worlds, as follows.

**Theorem 2.** Let  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  be a consistent probabilistic XML document in  $\text{PrXML}^{\text{ind}}$ . If  $f_1(n_i)$  is independent from  $C$ , after conditioning, it is possible to condition so that its formula  $f_2(n_i)$  is a unique independent event, and  $p_1(f_1(n_i)) = p_2(f_2(n_i))$ .

*Proof.* Since  $f_1(n_i)$  is a unique independent event  $e_i$  and it is independent from  $C$ , we can choose  $f_2(n_i)$  to be also a unique independent event, i.e.,  $f_2(n_i) = a_i$ . Obviously, we have  $p_2(a_i) = p_1(e_i)$  as proven below.

$$p_2(a_i) = p_1(e_i|C) = \frac{p_1(e_i \wedge C)}{p_1(C)} = \frac{p_1(e_i) \cdot p_1(C)}{p_1(C)} = p_1(e_i).$$

□

**Corollary 2.** Let  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  be a consistent probabilistic XML document in  $\text{PrXML}^{\text{ind}}$  and  $C$  is a mutually exclusive constraint over a set of nodes  $N$ . After conditioning (1) the formulae of all the nodes in the paths from  $\text{root}(D)$  to  $x \in N$  must be updated and the probabilities of new events must be defined; (2) for other nodes (which are not in those paths), the formulae and probabilities can be left unmodified.

Corollary 2 is a direct consequence of Theorem 2, because if a node  $n_i$  is not in the paths from root( $D$ ) to  $x \in N$ , then its formula  $f_1(n_i)$  is independent from  $C$ . **We can exclude such irrelevant nodes for the conditioning operation.** Under Corollary 2 we give the formal definition of the local tree in next.

**Definition 10.** (Local tree) Assume  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  is in  $\text{PrXML}^{\text{ind}}$  and  $C$  is a mutually exclusive constraint over a set of nodes  $N$ . The **local tree**, that we denote by  $\text{LT}(C, D)$ , of  $D$  with respect to the constraint  $C$  is a tree obtained by considering only the paths from root of  $D$  to the nodes in  $N$  and by excluding the rest of nodes.

**Example 3.** Consider the document in Figure 2 and the constraint in Example 1. The local tree is the tree in Figure 2, excluding node 4 and the edge from node 2 to node 4.

**Theorem 3.** Let us give two probabilistic XML documents  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  ( $\mathcal{D}_1$  is in  $\text{PrXML}^{\text{ind}}$  and  $C$  is a mutually exclusive constraint) and  $\mathcal{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$ . We claim that  $\mathcal{D}_1 \equiv_w \mathcal{D}_2$  iff

$$\langle \text{LT}(C, D), E_1, f_1, C, p_1 \rangle \equiv_w \langle \text{LT}(C, D), E_2, f_2, \emptyset, p_2 \rangle.$$

To end this section, we deduce the local possible worlds of the local tree as follows.

**Definition 11.** (Local possible worlds) The **local possible worlds** of  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$  (in  $\text{PrXML}^{\text{ind}}$ ) correspond to possible worlds of  $\langle \text{LT}(C, D), E_1, f_1, C, p_1 \rangle$ .

Now, let us study the number of local possible worlds for each kind of mutually exclusive constraint sketched above.

**Number of Local Possible Worlds** Assume there are  $m + 1$  nodes on the path from the root node to the lowest common ancestor of all the nodes in  $N$ . Figure 5(a), Figure 5(b) and Figure 5(c) present local trees of *MES*, *MEAD* and *MED* constraints, respectively. In the given tree examples, the set of mutually exclusive nodes is  $N = \{x_1 \dots x_q\}$ . Figure 5(d) depicts the local tree of *MED&AD* constraint, and  $q$  mutually exclusive nodes are shaded in the figure.

In Figure 5(b) and 5(c) there are  $k$  nodes in the paths from node  $m$  to node  $x_i$  (excluding  $m$  and including  $x_i$ ). In Figure 5(d), there are  $g$  shaded nodes ( $x_1, x_2, \dots, x_g$ ) in the higher level, each of which there are  $h_i$  shaded descendant nodes (for simplicity we assume  $h_i = h$  for  $i \in [1, g]$ ). Assume there are  $k_1$  nodes in the paths from node  $m$  to node  $x_i$ 's for  $i \in [1, g]$  (excluding  $m$  and including  $x_i$ ) and there are  $k_2$  nodes in the path from node  $x_i$ 's to node  $y_{(i,j)}$  ( $j \in [1, h_i]$ ) (excluding  $x_i$  and including  $y_{(i,j)}$ ). We have  $q = g + gh$ .

Table 1 shows the number of local possible worlds for presented mutually exclusive semantics. The number of local possible worlds of *MES* constraint is linear to  $q$ , while this size for *MEAD*, *MED*, and *MED&AD* mutually exclusive constraints is exponential in  $q$ .

Table 1:

Constraint	WMB	WOMBA	WOMBI
MES	$m + q + 2$	$q$	$m + q + 1$
MEAD	$m + 2 + k^{q-1}$	$k^{q-1}$	$m + 1 + k^{q-1}$
MED	$m + 1 + q \cdot k^{q-1} + k^q$	$q \cdot k^{q-1}$	$m + 1 + q \cdot k^{q-1}$
MED&AD	$m + 1 + g(k_1^{g-1} k_2^h) + k_1^g$	$g(k_1^{g-1} k_2^h)$	$m + 1 + g(k_1^{g-1} k_2^h)$

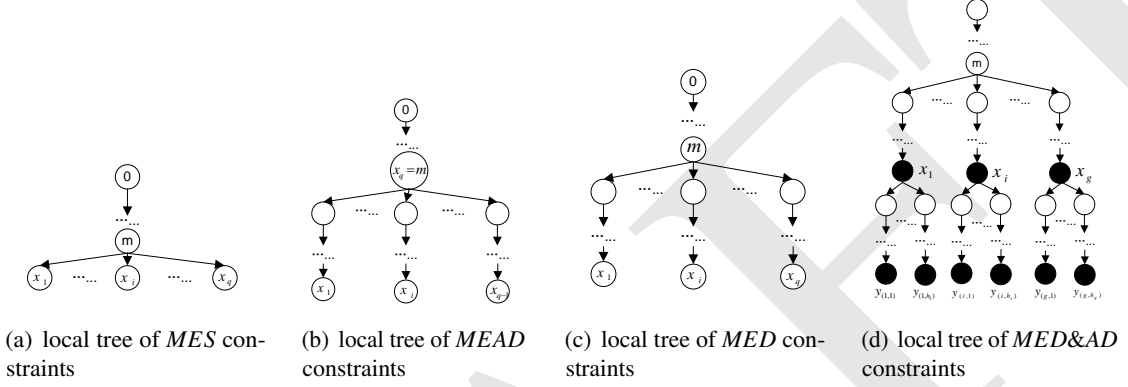


Figure 5: Local trees under considered mutually exclusive constraints

## 7 MutEx Siblings Constraints

In this section, we consider the *mutually exclusive siblings (MES)* constraint (Constraint 1 in the example in Section 1), i.e., the nodes in  $N = \{x_1, x_2, \dots, x_q\}$  are mutually exclusive and they are siblings. We assume the node id of the root of  $D$  is 0, and the node id of the parent node of all the nodes in  $N$  is  $m$ . There are  $m + 1$  nodes in  $\text{path\_node}(0, m)$ . Each node is associated with an independent event  $e_i$  ( $i \in [0, m]$ ). There are  $q$  nodes in  $N$ , i.e., node  $m$  has  $q$  children (with id  $m + i$ ,  $i \in [1, q]$ ), each of which is associated with an independent event  $e_i$  ( $i \in [m + 1, m + q]$ ).

According to Corollary 2, the local tree of the input probabilistic XML document is the part including nodes in  $\text{path\_node}(0, m + i)$  ( $i \in [1, q]$ ). Therefore we will not discuss the formulae of the other nodes.

As previously mentioned, we restrict in this present paper to WOMBA semantics (see [25] for other kinds of constraints). The constraint  $C$  tells that exactly only one of node  $m$ 's children exists. It can be formulated as follows:

$$C = \bigwedge_{i=0}^m e_i \wedge \left( \bigvee_{u=1}^q (e_{m+u} \wedge \bigwedge_{i=1, i \neq u}^q \neg e_{m+i}) \right)$$

The probability of the constraint is then:

$$p_1(C) = \prod_{i=0}^m p_1(e_i) \sum_{u=1}^q (p_1(e_{m+u}) \prod_{i=1, i \neq u}^q (1 - p_1(e_{m+i}))) = \prod_{i=0}^m p_1(e_i) \prod_{i=1}^q (1 - p_1(e_{m+i})) \sum_{i=1}^q \frac{p_1(e_{m+i})}{1 - p_1(e_{m+i})}$$



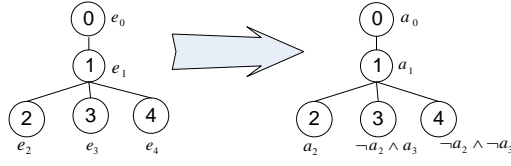


Figure 6: Probabilistic XML documents in Example 4

**Example 4.** Consider the local tree of a probabilistic XML document shown on the left-hand-side in Figure 6. The constraint is that one of nodes 2, 3, 4 exists. It is formulated as

$$C = e_0 \wedge e_1 \wedge [(e_2 \wedge \neg e_3 \wedge \neg e_4) \vee (\neg e_2 \wedge e_3 \wedge \neg e_4) \vee (\neg e_2 \wedge \neg e_3 \wedge e_4)]$$

Thanks to independence of the  $e_i$ 's, the probability of the constraint can be computed as follows.

$$\begin{aligned} p_1(C) &= p_1(e_0) \times p_1(e_1) \times [p_1(e_2)(1 - p_1(e_3))(1 - p_1(e_4)) + \\ & (1 - p_1(e_2))p_1(e_3)(1 - p_1(e_4)) + (1 - p_1(e_2))(1 - p_1(e_3))p_1(e_4)] \\ &= p_1(e_0)p_1(e_1)(1 - p_1(e_2))(1 - p_1(e_3))(1 - p_1(e_4)) \left( \frac{p_1(e_2)}{1 - p_1(e_2)} + \frac{p_1(e_3)}{1 - p_1(e_3)} + \frac{p_1(e_4)}{1 - p_1(e_4)} \right) \end{aligned}$$

This probability  $p_1(C)$  can be computed in linear time to the size of constraint. Thanks to that, and to the fact that there are linearly many local possible worlds (see Table 1), it is possible to condition the tree in a very simple manner. Algorithm 2 is the conditioning algorithm for an *MES* constraint under *WOMBA* semantics. Due to Theorem 3, there is no need to input the entire document  $D$  of  $\mathcal{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$ , therefore we only consider the local tree  $LT(C, D)$ . Line 1 to line 2 assign new formulae to the nodes in  $\text{path\_node}(0, m)$ . Line 3 to line 6 assign new formulae to the mutually exclusive nodes in  $N$ , which guarantees that only one of them exists. Line 7 presents the set of equations to compute the probabilities of the new events. Since one of the nodes in  $N$  exists, all the nodes in  $\text{path\_node}(0, m)$  must exist, otherwise none of the nodes in  $N$  exists. Hence the probabilities of the events associated with the nodes in  $\text{path\_node}(0, m)$  are 1 (in Equation (1)). Equations (2), (3), (4) enumerate all local possible worlds and state that their probabilities are unchanged after conditioning.

**Example 5.** Follow Example 4 and perform conditioning according to Algorithm 2. The result of conditioning is presented on the right-hand-side of Figure 6. In order to compute the probabilities of new events, the set of equations is:

$$\begin{cases} p_2(a_0) = p_2(a_1) = 1 \\ p_2(a_2) = \frac{p_1(e_0) \cdot p_1(e_1) \cdot p_1(e_2) \cdot (1 - p_1(e_3)) \cdot (1 - p_1(e_4))}{p_1(C)} \\ (1 - p_2(a_2)) \cdot p_2(a_3) = \frac{p_1(e_0) \cdot p_1(e_1) \cdot (1 - p_1(e_2)) \cdot p_1(e_3) \cdot (1 - p_1(e_4))}{p_1(C)} \\ (1 - p_2(a_2)) \cdot (1 - p_2(a_3)) = \frac{p_1(e_0) \cdot p_1(e_1) \cdot (1 - p_1(e_2)) \cdot (1 - p_1(e_3)) \cdot p_1(e_4)}{p_1(C)} \end{cases}$$

The set of equations is easy to solve.  $p_2(a_3)$  can be computed using the third and fourth equations.  $p_2(a_2)$  can be computed using the second equation.

---

**Algorithm 2:** Conditioning algorithm for *MES* constraint under WOMBA semantics
 

---

**Data:**  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$

**Result:** A world equivalent  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$

- 1 **foreach** node  $i \in \text{path\_node}(0, m)$  **do**
- 2   |  $f_2(i) \leftarrow a_i$ ;
- 3  $f_2(m+1) \leftarrow a_{m+1}$ ;
- 4 **for**  $i \in [2, q-1]$  **do**
- 5   |  $f_2(m+i) \leftarrow \neg a_{m+1} \wedge \neg a_{m+2} \wedge \dots \wedge a_{m+i}$ ;
- 6  $f_2(m+q) \leftarrow \neg a_{m+1} \wedge \neg a_{m+2} \wedge \dots \wedge \neg a_{m+q-1}$ ;
- 7 The probabilities of the new events are computed by solving the following set of equations:

$$p_2(a_0) = p_2(a_1) = \dots = p_2(a_m) = 1 \quad (1)$$

$$p_2(a_{m+1}) = \frac{\prod_{i=0}^m p_1(e_i) \cdot p_1(e_{m+1}) \cdot \prod_{i=1, i \neq 1}^q (1 - p_1(e_{m+i}))}{p_1(C)} \quad (2)$$

$$\begin{aligned} \forall k \in [2, q-1], \prod_{i=1}^{k-1} (1 - p_2(a_{m+i})) \cdot p_2(a_{m+k}) \\ = \frac{\prod_{i=0}^m p_1(e_i) \cdot p_1(e_{m+k}) \cdot \prod_{i=1, i \neq k}^q (1 - p_1(e_{m+i}))}{p_1(C)} \end{aligned} \quad (3)$$

$$\prod_{i=1}^{q-1} (1 - p_2(a_{m+i})) = \frac{\prod_{i=0}^m p_1(e_i) \cdot p_1(e_{m+q}) \cdot \prod_{i=1, i \neq q}^q (1 - p_1(e_{m+i}))}{p_1(C)} \quad (4)$$


---

**Theorem 4.** Given  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$ , Algorithm 2 outputs a world equivalent  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$ . Algorithm 2 performs in linear time to the size of the local tree.

*Proof.* The set of equations in line 7 of Algorithm 2 is developed based on Definition 8. The left-hand-side of each equation is the probability of a possible world in  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$ , while the right-hand-side of the equation is the probability of the same possible world in  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$ . These equations guarantee the input and output are world-equivalent.

Algorithm 2 introduces  $m+q$  new events and  $p_1(C)$  can be computed in linear time. The set of equations can be solved in linear time because each variable can be determined in turn by simple operations on the equations as addition and division. Hence the complexity of Algorithm 2 is  $O(m+q)$ .  $\square$

## 8 MutEx AD Constraints

Let us now turn to *mutually exclusive Ancestor-Descendant (MEAD)* constraints (Constraint 2 in the example in Section 1), i.e., the nodes in  $N = \{x_1, x_2, \dots, x_q\}$  are mutually exclusive and **there is one node  $x_q = m$  which is the lowest common ancestor of every pair of nodes in  $N$** . The node id of the root of  $D$  is 0. There are  $m+1$  nodes in the path from node 0 to node  $m$ . Each node is associated with an independent event  $e_i$  ( $1 \leq i \leq m$ ). Note that any two of the other  $q-1$

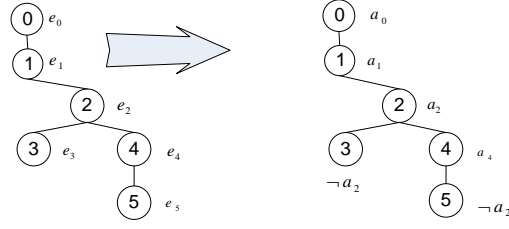


Figure 7: Probabilistic XML documents in Example 6

nodes in  $N$  (except node  $m$ ) cannot be in an AD relationship; otherwise node  $m$  is not the lowest common ancestor of the two nodes which are in an AD relationship.

The ids of the remaining  $q - 1$  nodes in  $N$  are  $x_1, \dots, x_{q-1}$  and their associated events are  $e_{x_1}, \dots, e_{x_{q-1}}$ . There are  $k_i$  nodes in the path from node  $m$  to node  $x_i$  (by excluding  $m$  while including  $x_i$ ). We use  $(i, j)$  to represent the  $j^{\text{th}}$  node in the path from node  $m$  to node  $x_i$ . The associated event of node  $(i, j)$  is  $e_{(i,j)}$ . Node  $x_i$  is the node  $(i, k_i)$ .

According to Corollary 2, the local tree of the input probabilistic XML document is the part including nodes in  $\text{path\_node}(0, x_i)$  ( $i \in [1, q - 1]$ ). We will not discuss the formulae of other nodes.

Again, we focus on the WOMBA semantics. The constraint is exactly one node in  $N$  exists. Node  $m$  is the ancestor of all the other nodes in  $N$ , therefore if node  $m$  does not exist, all the other nodes in  $N$  cannot be actual. Hence, the constraint tells that node  $m$  exists while the other nodes in  $N$  do not. An *MEAD* constraint under WOMBA semantics is formulated as follows:

$$C = \bigwedge_{i=0}^m e_i \wedge \bigwedge_{i=1}^{q-1} \neg \left( \bigwedge_{j=1}^{k_i} e_{(i,j)} \right).$$

Its probability is

$$p_1(C) = \prod_{i=0}^m p_1(e_i) \prod_{i=1}^{q-1} \left( 1 - \prod_{j=1}^{k_i} p_1(e_{(i,j)}) \right).$$

**Example 6.** Consider the local tree of a probabilistic XML document on the left-hand-side in Figure 7. The constraint is one of node 2, 3, 5 exists. The constraint can be formulated as:

$$C = e_0 \wedge e_1 \wedge e_2 \wedge \neg e_3 \wedge \neg(e_4 \wedge e_5).$$

Its probability is

$$p_1(C) = p_1(e_0)p_1(e_1)p_1(e_2)(1 - p_1(e_3))(1 - p_1(e_4)p_1(e_5)).$$

This probability  $p_1(C)$  can be computed in *linear time* to the size of the constraint. Algorithm 3 is the conditioning algorithm for an *MEAD* constraint under WOMBA semantics. The input is the local tree of the constraint and the document. Line 1 to line 2 assign new formulae to the nodes in  $\text{path\_node}(0, m)$ . Line 4 assigns new formulae to nodes  $x_i$  ( $i \in [1, q - 1]$ ), which

guarantees that only one of node  $m$  and  $x_i$ 's exists. Line 5 to line 6 assign formulae to the nodes between  $m$  and  $x_i$ 's. As can be observed, their formulae are independent. Line 7 presents the set of equations to compute the probabilities of the new events. The probabilities of the events associated with the nodes in  $\text{path\_node}(0, m)$  are 1 (in Equation (5)). Equation (6) defines the probabilities of other new events.

---

**Algorithm 3:** Conditioning algorithm for the *MEAD* constraint under WOMBA semantics

---

**Data:**  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$

**Result:** A world equivalent  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$

```

1 foreach node  $i \in \text{path\_node}(0, m)$  do
2   |  $f_2(i) = a_i$ ;
3 for  $i \in [1, q - 1]$  do
4   |  $f_2(x_i) \leftarrow \neg a_m$ ;
5   | for  $j \in [1, k_i - 1]$  do
6   |   |  $f_2((i, j)) \leftarrow a_{(i, j)}$ ;
7 The probabilities of the new events are computed by solving the following set of
  equations:

```

$$p_2(a_0) = p_2(a_1) = \dots = p_2(a_m) = 1 \quad (5)$$

$$\frac{p_2(a_{(i, j)})}{1 - p_2(a_{(i, j)})} = \frac{p_1(e_{(i, j)})(1 - \prod_{u=j+1}^{k_i} p_1(e_{(i, u)}))}{1 - p_1(e_{(i, j)})} \quad (\text{if } p_1(e_{(i, j)}) \neq 1) \quad (6)$$

$$p_2(a_{(i, j)}) = 1 \quad (\text{if } p_1(e_{(i, j)}) = 1) \quad (7)$$


---

**Example 7.** Follow Example 6 and do conditioning using Algorithm 3. The result of conditioning is presented on the right-hand-side of Figure 7. In order to compute the probabilities of new events, solve the equations:

$$\begin{cases} p_2(a_0) = p_2(a_1) = p_2(a_2) = 1 \\ \frac{p_2(a_4)}{1 - p_2(a_4)} = \frac{p_1(e_4) \cdot (1 - p_1(e_5))}{1 - p_1(e_4)} \end{cases}$$

**Theorem 5.** Given  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$ , Algorithm 3 outputs a world equivalent  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$ . Algorithm 3 performs in linear time to the size of the local tree.

*Proof.* From Equation (1) in Algorithm 3, we have:

If  $p_1(e_{(i, j)}) \neq 1$ , then  $p_2(a_{(i, j)}) = \frac{p_1(e_{(i, j)})(1 - \prod_{u=j+1}^{k_i} p_1(e_{(i, u)}))}{1 - \prod_{u=j}^{k_i} p_1(e_{(i, u)})}$ ; if  $p_1(e_{(i, j)}) = 1$ , then  $p_2(a_{(i, j)}) = 1$ .

Pick any possible world in  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$ . Let us assume that in this possible world, in the path from node  $m$  to node  $x_i$  ( $i \in [1, q - 1]$ ), node  $(i, z_i)$  exists while its child does not. We assume  $z_i < k_i - 1$ . The proof is almost the same when  $z_i = k_i - 1$  therefore we omit it.

The probability of the path from node  $m$  to node  $x_i$  is

$$\begin{aligned}
& p_2(a_{(i,1)})p_2(a_{(i,2)})\dots p_2(a_{(i,z_i)})(1 - p_2(a_{(i,z_i+1)})) \\
&= \frac{p_1(e_{(i,1)})(1 - \prod_{u=2}^{k_i} p_1(e_{(i,u)}))}{1 - \prod_{u=1}^{k_i} p_1(e_{(i,u)})} \frac{p_1(e_{(i,2)})(1 - \prod_{u=3}^{k_i} p_1(e_{(i,u)}))}{1 - \prod_{u=2}^{k_i} p_1(e_{(i,u)})} \\
&\dots \frac{p_1(e_{(i,z_i)})(1 - \prod_{u=z_i+1}^{k_i} p_1(e_{(i,u)}))}{1 - \prod_{u=z_i}^{k_i} p_1(e_{(i,u)})} \frac{1 - p_1(e_{(i,z_i+1)})}{1 - \prod_{u=z_i+1}^{k_i} p_1(e_{(i,u)})} \\
&= \frac{\prod_{j=1}^{z_i} p_1(e_{(i,j)})(1 - p_1(e_{(i,z_i+1)}))}{1 - \prod_{u=1}^{k_i} p_1(e_{(i,u)})}
\end{aligned}$$

Therefore the probability of this possible world is

$$\begin{aligned}
\prod_{i=1}^{q-1} \left( \prod_{j=1}^{z_i} p_2(a_{(i,j)})(1 - p_2(a_{(i,z_i+1)})) \right) &= \frac{\prod_{i=1}^{q-1} \left( \prod_{j=1}^{z_i} p_1(e_{(i,j)})(1 - p_1(e_{(i,z_i+1)})) \right)}{\prod_{i=1}^{q-1} \left( 1 - \prod_{u=1}^{k_i} p_1(e_{(i,u)}) \right)} \\
&= \frac{\prod_{i=0}^m p_1(e_i) \prod_{i=1}^{q-1} \left( \prod_{j=1}^{z_i} p_1(e_{(i,j)})(1 - p_1(e_{(i,z_i+1)})) \right)}{\prod_{i=0}^m p_1(e_i) \prod_{i=1}^{q-1} \left( 1 - \prod_{u=1}^{k_i} p_1(e_{(i,u)}) \right)} \\
&= \frac{\prod_{i=0}^m p_1(e_i) \prod_{i=1}^{q-1} \left( \prod_{j=1}^{z_i} p_1(e_{(i,j)})(1 - p_1(e_{(i,z_i+1)})) \right)}{p_1(C)}
\end{aligned}$$

We deduce that this value is the probability of the same possible world in  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$ . Therefore Algorithm 3 is correct.

Algorithm 2 introduces  $m + \sum_{i=1}^q (k_i - 1)$  new events. The set of equations can be solved in linear time because every variable only occurs in one equation. Therefore the complexity of Algorithm 3 is linear (even though there are exponentially many local possible worlds).  $\square$

## 9 MutEx Descendance Constraints

In this section, we consider the *mutually exclusive descendance (MED)* constraint (Constraint 3 in the example in Section 1), i.e., the nodes in  $N = \{x_1, x_2, \dots, x_q\}$  are mutually exclusive and every pair of the nodes in  $N$  shares the same lowest common ancestor – node  $m$  (node  $m$  is not in  $N$ ). The id of the root of  $D$  is 0. There are  $m + 1$  nodes in the path from node 0 to node  $m$ .

There are  $k_i$  nodes along the path from node  $m$  to node  $x_i$  (excluding node  $m$  and including node  $x_i$ ). We use  $(i, j)$  to represent the  $j^{\text{th}}$  node along the path from node  $m$  to node  $x_i$ . The associated event of node  $(i, j)$  is  $e_{(i,j)}$ . Note that the node  $x_i$  is the node  $(i, k_i)$ .

According to Corollary 2, the local tree of the input probabilistic XML document is the part including nodes in  $\text{path\_node}(0, x_i)$  ( $i \in [1, q]$ ). We will not discuss the formulae of the other nodes.

We consider the *MED* constraint under WOMBA semantics.

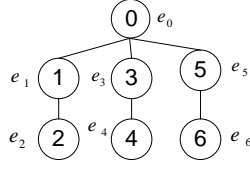


Figure 8: Probabilistic XML documents in Example 8

The constraint is exactly one node in  $N$  exists. The constraint  $C$  says that (1) nodes in  $\text{path\_node}(0, m)$  exist; (2) when  $x_i$  exists, any  $x_k (k \neq i)$  cannot exist, however, the nodes between node  $m$  and node  $x_k$  may exist. The constraint can be formulated as follows:

$$C = \bigwedge_{i=0}^m e_i \wedge \bigvee_{i=1}^q \left( \bigwedge_{j=1}^{k_i} e_{(i,j)} \wedge \bigwedge_{u=1, u \neq i}^q \left( \neg \bigwedge_{j=1}^{k_u} e_{(u,j)} \right) \right)$$

The probability of the constraint is

$$\begin{aligned} p_1(C) &= \prod_{i=0}^m p_1(e_i) \sum_{i=1}^q \left( \prod_{j=1}^{k_i} p_1(e_{(i,j)}) \prod_{u=1, u \neq i}^q \left( 1 - \prod_{j=1}^{k_u} p_1(e_{(u,j)}) \right) \right) \\ &= \prod_{i=0}^m p_1(e_i) \prod_{i=1}^q \left( 1 - \prod_{j=1}^{k_i} p_1(e_{(i,j)}) \right) \left( \sum_{i=1}^q \frac{\prod_{j=1}^{k_i} p_1(e_{(i,j)})}{1 - \prod_{j=1}^{k_i} p_1(e_{(i,j)})} \right) \end{aligned}$$

**Example 8.** Consider a local tree of a probabilistic XML document in Figure 8. The constraint is one of nodes 2, 4, 6 exists. The constraint can be formulated as:

$$\begin{aligned} C &= e_0 \wedge (e_1 \wedge e_2 \wedge \neg(e_3 \wedge e_4) \wedge \neg(e_5 \wedge e_6) \vee e_3 \wedge e_4 \wedge \\ &\quad \neg(e_1 \wedge e_2) \wedge \neg(e_5 \wedge e_6) \vee e_5 \wedge e_6 \wedge \neg(e_1 \wedge e_2) \wedge \neg(e_3 \wedge e_4)) \end{aligned}$$

Its probability is

$$\begin{aligned} p_1(C) &= p_1(e_0) (p_1(e_1) p_1(e_2) (1 - p_1(e_3) p_1(e_4)) (1 - p_1(e_5) p_1(e_6)) \\ &\quad + p_1(e_3) p_1(e_4) (1 - p_1(e_1) p_1(e_2)) (1 - p_1(e_5) p_1(e_6)) + \\ &\quad p_1(e_5) p_1(e_6) (1 - p_1(e_1) p_1(e_2)) (1 - p_1(e_3) p_1(e_4))) \\ &= p_1(e_0) (1 - p_1(e_1) p_1(e_2)) (1 - p_1(e_3) p_1(e_4)) (1 - p_1(e_5) p_1(e_6)) \\ &\quad \left( \frac{p_1(e_1) p_1(e_2)}{1 - p_1(e_1) p_1(e_2)} + \frac{p_1(e_3) p_1(e_4)}{1 - p_1(e_3) p_1(e_4)} + \frac{p_1(e_5) p_1(e_6)}{1 - p_1(e_5) p_1(e_6)} \right) \end{aligned}$$

This probability  $p_1(C)$  can be computed in linear time to the size of the constraint.

Similarly as in the other cases, we can devise an algorithm to condition probabilistic XML documents under this kind of constraint (such an algorithm is described in [25]). This algorithm introduces  $m + k_1 + k_q - q + 2 + \sum_{i=2}^{q-1} 2k_i$  new events and  $p_1(C)$  can be computed in linear time. However, we do not have a proof that shows the set of equations in the algorithm always has a solution, though the set of equations did have such a solutions in all cases we experimented with. When there is a solution, the algorithm runs in polynomial-time to provide an approximate solution to the equations.

**Example 9.** We illustrate the algorithm on Example 8. After the conditioning, the formulae of all the nodes are:

$$\begin{cases} f_2(0) = a_0, f_2(1) = a_1, f_2(2) = a_2 \\ f_2(3) = (\neg(f_2(1) \wedge f_2(2)) \wedge \theta_1) \vee \eta_1 \\ f_2(4) = \neg(f_2(1) \wedge f_2(2) \wedge f_2(3)) \wedge \eta_2 \\ f_2(5) = \neg(f_2(1) \wedge f_2(2)) \wedge \neg(f_2(3) \wedge f_2(4)) \vee \alpha_1 \\ f_2(6) = \neg(f_2(1) \wedge f_2(2)) \wedge \neg(f_2(3) \wedge f_2(4)) \end{cases}$$

There are 7 new created events. In order to compute the probabilities of these events, a straightforward way is to enumerate all the possible worlds and set equations to make sure that their probabilities are the same after conditioning.

There are 12 possible worlds, 4 for each of the following cases: when node 2 exists while node 4, 6 do not; when node 4 exists while node 2, 6 do not; and when node 6 exists while node 2, 4 do not.

The algorithm gives us a set of 7 equations to solve to get the probabilities of the new events. They are:

$$\left\{ \begin{array}{l} p_2(a_0) = 1 \\ p_2(a_1)p_2(a_2)(1-p_2(\eta_1))(1-p_2(\alpha_1)) = \frac{p_1(e_0)p_1(e_1)p_1(e_2)(1-p_1(e_3))(1-p_1(e_5))}{p_1(C)} \\ (1-p_2(a_1))(p_2(\theta_1) + p_2(\eta_1) - p_2(\theta_1)p_2(\eta_1)) \cdot p_2(\eta_2)(1-p_2(\alpha_1)) = \\ \frac{p_1(e_0)(1-p_1(e_1))p_1(e_3)p_1(e_4)(1-p_1(e_5))}{p_1(C)} \\ (1-p_2(a_1))(1-p_2(\eta_1))(1-p_2(\theta_1)) = \frac{p_1(e_0)(1-p_1(e_1))(1-p_1(e_3))p_1(e_5)p_1(e_6)}{p_1(C)} \\ \frac{1-p_2(a_1)}{p_2(a_1)(1-p_2(a_2))} = \frac{1-p_1(e_1)}{p_1(e_1)(1-p_1(e_2))} \\ \frac{1-p_2(\eta_1)}{p_2(\eta_1)} = \frac{1-p_1(e_3)}{p_1(e_3)(1-p_1(e_4))} \\ \frac{(1-p_2(\eta_1))(1-p_2(\theta_1))}{(p_2(\theta_1) + p_2(\eta_1) - p_2(\theta_1)p_2(\eta_1))(1-p_2(\eta_2))} = \frac{1-p_1(e_3)}{p_1(e_3)(1-p_1(e_4))} \end{array} \right.$$

Assume  $p_1(e_0) = 1, p_1(e_1) = p_1(e_2) = \frac{1}{2}, p_1(e_3) = p_1(e_4) = \frac{1}{3}, p_1(e_5) = p_1(e_6) = \frac{1}{4}$ . Resolving the equations, the probabilities of the new events are  $p_2(a_0) = 1, p_2(a_1) = \frac{143}{189}, p_2(a_2) = \frac{120}{143}, p_2(\theta_1) = \frac{15}{23}, p_2(\eta_1) = \frac{1}{4}, p_2(\eta_2) = \frac{15}{17}, p_2(\alpha_1) = \frac{1}{5}$ .

**Theorem 6.** The algorithm for MED constraints presented in [25] outputs a world equivalent  $\langle LT(C, D), E_2, f_2, \emptyset, p_2 \rangle$  in polynomial-time, given  $\langle LT(C, D), E_1, f_1, C, p_1 \rangle$ , assuming that the equation system has a solution.

The proof of this theorem is omitted and is available in [25].

## 10 Conclusion

In this paper, we have studied the problem of conditioning probabilistic XML data. In general, as observed in Section 5, conditioning is intractable and obtaining minimal representations relate to long-standing open problems in circuit complexity. We presented an exponential-time algorithm

for the general case. Then we focused on the special case of probabilistic XML with independent events and constraints as mutually exclusive constraints. We proposed a rather simplified but reasonable query language to represent the considered mutually exclusive constraints. We devised and presented tractable algorithms for four classes of mutually exclusive constraints. Note that incorporating this kind of mutually exclusive constraints in what is essentially a PrXML<sup>ind</sup> document is more powerful than simply adding mux nodes as the mutually exclusive constraints considered span the whole document.

We are currently investigating the following issues. We have not been able to establish or disprove the existence of a polynomial conditioning algorithm for general mutually exclusive constraints on the probabilistic XML model with independent events. One constructive way to do so is to establish under which conditions the existence of solutions for the equation systems yielded by the conditioning problems is assured; note that these equations are not necessarily linear, though it is in practice possible to numerically solve them if approximate solutions are enough. A further question is that of the minimality of the obtained conditioned document – does there exist another unconstrained document with simpler annotations that is world-equivalent to the original probabilistic document?

## References

- [1] ABITEBOUL, S., KIMELFELD, B., SAGIV, Y., AND SENELLART, P. On the expressiveness of probabilistic XML models. *VLDB Journal* 18, 5 (2009), 1041–1064.
- [2] ABITEBOUL, S., AND SENELLART, P. Querying and updating probabilistic information in XML. In *Proc. EDBT* (2006).
- [3] ALLENDER, E. Chipping away at P vs NP: How far are we from proving circuit size lower bounds? In *CATS* (2008), p. 3.
- [4] AMARILLI, A., AND SENELLART, P. On the connections between relational and XML probabilistic data models. In *BNCOD* (2013).
- [5] AMER-YAHIA, S., CHO, S., LAKSHMANAN, L. V. S., AND SRIVASTAVA, D. Minimization of tree pattern queries. In *Proc. SIGMOD* (2001).
- [6] BA, M. L., ABDESSALEM, T., AND SENELLART, P. Uncertain version control in open collaborative editing of tree-structured documents. In *Proc. DocEng* (Florence, Italy, 2013).
- [7] BARBARÁ, D., GARCIA-MOLINA, H., AND PORTER, D. The management of probabilistic data. *Proc. IEEE Trans. Knowl. Data Eng.* (1992).
- [8] CAVALLO, R., AND PITTARELLI, M. The theory of probabilistic databases. In *Proc. VLDB* (1987).
- [9] CHANG, C.-H., KAYED, M., GIRGIS, M. R., AND SHAALAN, K. F. A survey of Web information extraction systems. *IEEE Trans. on Knowl. and Data Eng.* (2006).



- [10] COHEN, S., KIMELFELD, B., AND SAGIV, Y. Incorporating constraints in probabilistic XML. In *Proc. PODS* (2008).
- [11] DONG, X. L., HALEVY, A., AND YU, C. Data integration with uncertainty. *VLDB Journal* (2009).
- [12] EITER, T., LUKASIEWICZ, T., AND WALTER, M. A data model and algebra for probabilistic complex values. *Proc. Ann. Math. Artif. Intell.* (2001).
- [13] HUNG, E., GETOOR, L., AND SUBRAHMANIAN, V. S. Probabilistic interval XML. In *Proc. ICDT* (2003).
- [14] HUNG, E., GETOOR, L., AND SUBRAHMANIAN, V. S. PXML: A probabilistic semistructured data model and algebra. In *Proc. ICDE* (2003).
- [15] KANNAN, R. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control* 55, 1-3 (1982), 40–56.
- [16] KHARLAMOV, E., NUTT, W., AND SENELLART, P. Updating probabilistic XML. In *Proc. EDBT/ICDT Workshops* (2010).
- [17] KHARLAMOV, E., AND SENELLART, P. Modeling, Querying, and Mining Uncertain XML Data. In *XML Data Mining: Models, Methods, and Applications*, A. Tagarelli, Ed. Proc. IGI Global, 2011.
- [18] KIMELFELD, B., AND SENELLART, P. Probabilistic XML: Models and complexity. In *Proc. Advances in Probabilistic Databases for Uncertain Information Management*, Z. Ma and L. Yan, Eds. Springer-Verlag, 2013.
- [19] KOCH, C., AND OLTEANU, D. Conditioning probabilistic databases. *Proc. PVLDB* (2008).
- [20] NIERMAN, A., AND JAGADISH, H. V. ProTDB: Probabilistic data in XML. In *Proc. VLDB* (2002).
- [21] NILSSON, N. J. Probabilistic logic. *Artif. Intell.* 28, 1 (1986), 71–87.
- [22] PAPANITRIOU, C. H. *Computational Complexity*. Addison-Wesley, 1994.
- [23] SUCIU, D., OLTEANU, D., RÉ, C., AND KOCH, C. *Probabilistic Databases*. Morgan & Claypool, 2011.
- [24] TANG, R., CHENG, R., WU, H., AND BRESSAN, S. A framework for conditioning uncertain relational data. In *Proc. DEXA* (2012).
- [25] TANG, R., SHAO, D., BA, M. L., SENELLART, P., AND BRESSAN, S. A framework for conditioning probabilistic XML data (extended version). Tech. rep., National University of Singapore, Singapore, 2014. Available at <http://www.comp.nus.edu.sg/~tang1987/tech-reports/conditioningXML.pdf>.
- [26] VAN KEULEN, M., DE KEIJZER, A., AND ALINK, W. A probabilistic XML approach to data integration. In *Proc. ICDE* (2005).