THE NATIONAL UNIVERSITY
of SINGAPORE

School of Computing
Computing 1, 13 Computing Drive, Singapore 117417

TR10/13

Conditioning Probabilistic XML (Extended Version)

Ruiming Tang, Dongxu Shao, M. Lamine Ba, Pierre Senellart and Stéphane Bressan

October 2013

T e c h n i c a l R e p o r t

Foreword

This technical report contains a research paper, development or tutorial article, which has been submitted for publication in a journal or for consideration by the commissioning organization. The report represents the ideas of its author, and should not be taken as the official views of the School or the University. Any discussion of the content of the report should be sent to the author, at the address shown on the cover.

David ROSENBLUM
Dean of School

Conditioning Probabilistic XML (Extended Version)

Ruiming Tang Dongxu Shao M. Lamine Ba
Pierre Senellart Stéphane Bressan

October 22, 2013

A probabilistic database instance denotes a set of possible non-probabilistic database instances called possible worlds, each of which has a probability. 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. Adding such constraints to the set of possible worlds with their probabilities is *conditioning* in the probabilistic sense of the term. The problem is to find a new probabilistic database instance that denotes the subset of possible worlds with their new probabilities corresponding to the conditional 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 algorithms for conditioning probabilistic XML data in tractable cases.

1 Introduction

Uncertainty of data, in its various forms, naturally arises from such applications as information extraction [8], information integration [24, 10] and version control [5], for instance.

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.

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. Adding such constraints to the set of possible worlds with their probabilities is conditioning in the probabilistic sense of the term. The question is to find a new probabilistic database instance that denotes the subset of possible worlds with their new probabilities corresponding to the conditional probabilities.

Probabilistic relational databases [7, 6, 22, 11] represent uncertainties at value or tuple level, while schema is constrained. The conditioning problem in probabilistic relational databases has been studied in [18] and [23], respectively. Koch and Olteanu [18] 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. [23] 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 [20].

In contrast, probabilistic XML databases [17], 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, 17] is a general framework encompassing various probabilistic XML models from the literature [19, 24, 2, 15]. 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 [15]) family of p-documents, namely $\mathsf{PrXML}^{\mathsf{fie}}$ [15], 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 5.3).

Let us illustrate the conditioning problem for probabilistic XML with a simple student database and mutual exclusiveness constraints. Consider the $\mathsf{PrXML}^{\mathsf{fie}}$ 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 $\mathsf{PrXML}^{\mathsf{ind}}$. 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.

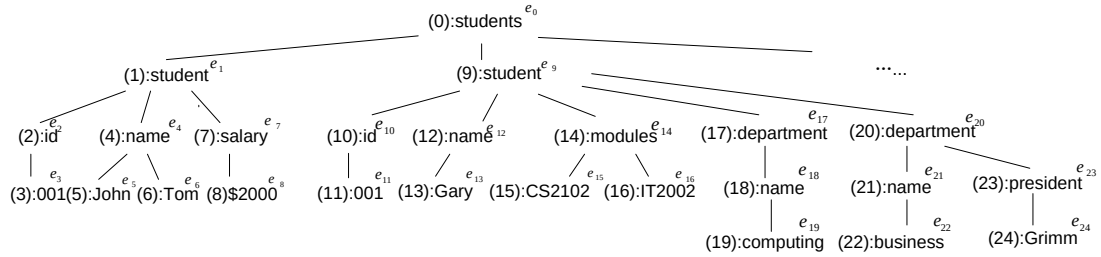


Figure 1: Example PrXML$^{\text{fie}}$ p-document: data about students

- *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$^{\text{ind,mux}}$ from [17]).

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$^{\text{fie}}$ 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, and 10 respectively.

2 Related Work

This section reviews state-of-the-art probabilistic XML representation systems and conditioning 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 worlds, distributional nodes enable to define the process of obtaining random documents. As surveyed in [17, 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 [17, 1] we denote a given probabilistic XML data model by $\mathsf{PrXML}^{\mathscr{X}}$, where $\mathscr{X} \subseteq \{\mathsf{ind}, \mathsf{mux}, \mathsf{det}, \mathsf{exp}, \mathsf{cie}, \mathsf{fie}\}$ contains distributional nodes considered in this model. We mostly focus in this paper on $\mathsf{PrXML}^{\mathsf{fie}}$ documents. For completeness, we review most commonly studied probabilistic XML models.

$\mathsf{PrXML}^{\mathsf{mux}}$, $\mathsf{PrXML}^{\mathsf{ind}}$, $\mathsf{PrXML}^{\{\mathsf{mux},\mathsf{ind}\}}$ [19, 16, 9, 24], and $\mathsf{PrXML}^{\mathsf{exp}}$ [13, 12] 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. Given a p-document of a local dependencies model, the random process which generates an ordinary document, along with its probability, from this p-document runs in two steps. The first step is to choose one subset of children for each distributional node according to its type. All unchosen children and their descendants nodes are removed The second step is to delete all distributional nodes, and treat the children of distributional nodes as the children of the lowest ancestor of the distributional nodes. The outcome of these two steps is a possible world (an XML document), and its probability is the multiplication of the probabilities of selected subsets of children of distributional nodes. Multiple outcomes may lead to the same possible world, therefore the probability of a possible XML document is the sum of probabilities of all outcomes leading to it.

$\mathsf{PrXML}^{\mathsf{cie}}$ [2] and $\mathsf{PrXML}^{\mathsf{fie}}$ [15] are global dependency models handling in addition long-distance dependencies between nodes – 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, $\mathsf{PrXML}^{\mathsf{cie}}$ and $\mathsf{PrXML}^{\mathsf{fie}}$ 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. The process of generating a possible world from a p-document using fie or cie nodes is to draw independently the truth value of each Boolean event according to their probabilities, and select the children of distributional nodes for which the Boolean formulae is evaluated to be true. The probability of this outcome is the probability of the corresponding assignment of events. The probability of an XML document is the sum of probabilities of all outcomes leading to it. Note that $\mathsf{PrXML}^{\mathsf{cie}}$ p-documents belong also to $\mathsf{PrXML}^{\mathsf{fie}}$ but there is no efficient translations from the latter to the former [15].

$\mathsf{PrXML}^{\mathsf{fie}}$ has been shown in [15] to be the most *expressive* and *succinct* probabilistic XML representation system in the literature. In this paper, we propose an extended $\mathsf{PrXML}^{\mathsf{fie}}$ 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 [18] and [23]. The authors of [18] propose an approach to do conditioning probabilistic relations in an attribute-level model. They adapt algorithms and heuristics for Boolean validity checking and simplification to solve the general NP-hard conditioning problem. In [23], some of the authors of the present work adapt the algorithm in [18] to a proposed tuple-level model. Special practical families of constraints are identified (i.e., *observation* and *X-tuple* constraints) for which efficient algorithms are presented.

The problem of the evaluation of constraints in probabilistic XML has only been investigated in [9]. Given a probabilistic XML document and a specific constraint in a pre-defined language, Cohen et al. study in [9] 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., $\mathsf{PrXML}^{\{\mathsf{mux},\mathsf{ind}\}}$. 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 [15], the authors define the semantics of two elementary kinds of updates, *insertions* and *deletions*, using a *locator query*[1]. The main result of this study is that $\mathsf{PrXML}^{\mathsf{fie}}$ 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.

Trees and XML documents Given an unordered, directed tree t, we consider $\mathsf{V}(t)$ and $\mathsf{E}(t)$ as, respectively, the set of nodes and edges of t – the special node $\mathsf{root}(t)$ 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 $\mathsf{label}(n_i)$. Any two nodes $n_1, n_2 \in \mathsf{V}(t)$ such that $(n_1,n_2) \in \mathsf{E}(t)$ 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 $\mathsf{parent}(n)$ to represent the parent of node n. Two nodes n_2, n_3 are *siblings* if $(n_1,n_2),(n_1,n_3) \in \mathsf{E}(t)$. 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

[1] A locator query is a tree-pattern query specifying the nodes where the update is to be performed.

relationship is a special case of ancestor-descendant relationship. We use $\mathsf{path\_node}(n_i, n_j)$ 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 $\mathsf{LCA}(N)$ 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 [23] to a probabilistic XML data model.

Definition 3.1. *(Events and complex events) Let E be a set of symbols called **events** (e). A **complex event** (ce) 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 $\mathsf{C}(E)$ the set of complex events formed with the events in E.*

Definition 3.2. *(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 $\mathsf{V}(D)$ to $\mathsf{C}(E)$ (associating each node in the document with a complex event), C – the constraint – is an element of $\mathsf{C}(E)$, 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 [23] 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 $\mathsf{parent}(n)$ exists and $f(n)$ is true.

Definition 3.3. *(Interpretation, model) An **interpretation** of a complex event ce is an assignment of each event in ce to $\{\mathsf{true}, \mathsf{false}\}$. A **model** of a complex event ce is an interpretation of ce that makes ce true. The set of models of ce is denoted as $\mathscr{M}(ce)$.*

Definition 3.4. *(Probability of complex event) Given a probability function p over a set E of events, The **probability** of a complex event $ce \in \mathsf{C}(E)$, denoted as $p(ce)$ is*

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

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

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

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

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

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

Lemma 3.7. *Let* $\mathscr{D}=\langle D,E,f,C,p\rangle$ *be a probabilistic XML document.* $\mathscr{D}$ *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 3.8. *(Probability of a node) Given a probabilistic XML document* $\mathscr{D}=\langle D,E,f,C,p\rangle$*, the* ***probability*** *of a node* $n \in \mathsf{V}(D)$*, denoted by* $p_{\mathscr{D}}(n)$*, is defined as:*

$$p_{\mathscr{D}}(n) = \sum_{\substack{D' \in \mathfrak{P}(\mathscr{D}) \\ n \in \mathsf{V}(D')}} p_{\mathscr{D}}(D').$$

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

Definition 3.9. *(World equivalence) Given two probabilistic XML documents* $\mathscr{D}_1=\langle D,E_1,f_1,C_1,p_1\rangle$ *and* $\mathscr{D}_2=\langle D,E_2,f_2,C_2,p_2\rangle$*, we say that* $\mathscr{D}_1$ *and* $\mathscr{D}_2$ *are* ***world-equivalent****, denoted by* $\mathscr{D}_1 \equiv_w \mathscr{D}_2$*, if and only if*

$$D' \in \mathfrak{P}(\mathscr{D}_1) \Longleftrightarrow D' \in \mathfrak{P}(\mathscr{D}_2) \quad \textit{and} \quad p_{\mathscr{D}_1}(D') = p_{\mathscr{D}_2}(D').$$

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

Theorem 3.10. *Let* $\mathscr{D}_1 = \langle D,E_1,f_1,C,p_1\rangle$ *be a probabilistic XML document. If* $\mathscr{D}_1$ *is consistent, then there exists an* unconstrained *probabilistic XML document* $\mathscr{D}_2 = \langle D,E_2,f_2,\emptyset,p_2\rangle$ *such that* $\mathscr{D}_1 \equiv_w \mathscr{D}_2$*.*

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 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 $pwd(K_i)$. For each node $n \in \mathsf{V}(D)$, 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 $\bigwedge_{n' \in \mathsf{path\_node}(\mathsf{root}(D),n)} f_1(n')$.

We set $f_2(n) = \bigvee_{n \in pwd(K_i)} K_i$. This makes sure that $\mathscr{D}_2$ has the same set of possible worlds as $\mathscr{D}_1$. The probability of K_i is computed as $p_2(K_i) = \frac{p_1(K_i)}{p(C)}$. This equation guarantees that $\mathscr{D}_2$ and $\mathscr{D}_1$ 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: $K_1 = a_1, K_i = \neg a_1 \wedge ... \wedge \neg a_{i-1} \wedge a_i (i = 2, ..., q-1), K_q = \neg a_1 \wedge ... \wedge \neg a_{q-1}$. 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 $\mathscr{D}_2$ such that $\mathscr{D}_1 \equiv_w \mathscr{D}_2$. □

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

This conditioning problem is solved by finding a world-equivalent probabilistic XML document with empty constraint, given the probabilistic XML document with a constraint. The existence of such a probabilistic XML document with no constraint is a direct consequence of Theorem 3.10.

Corollary 3.11. *Let $\mathscr{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$ be a* consistent *probabilistic XML document. The conditioning problem over $\mathscr{D}_1$ always has a solution.*

4 From Query to Constraint

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; we also discuss a simpler class of constraints, namely node (non-)existence constraint.

The query language we consider is *twig* queries [4] (or *tree pattern* queries), also known as the *downward navigational* fragment of XPath. We occasionally reuse XPath notation.

(Non-)existence constraints A constraint indicating that a node i exists (resp. does not exist) corresponds to a *positive* (resp., *negative*) form of *single-path* twig query, which navigates a path from the root to the node i and specifies the node i as the output.

In the case of a node existence constraint, constraining by a query results in a constraint formula that is the conjunct of formulae of all nodes from the root to the node i, i.e., $\bigwedge_{n \in \mathsf{path\_node}(root(D), i)} f(n)$. The opposite constraint, i.e., a node non-existence constraint results in the negation of the twig query for the existence of the node i. The formula of such a constraint is then $\neg(\bigwedge_{n \in \mathsf{path\_node}(root(D), i)} f(n))$.

Example 4.1. *Consider the probabilistic XML document in Figure 2. Let us illustrate existence and non-existence constraints over nodes in this* PrXML$^{\mathsf{fie}}$ *p-document. First, we consider the constraint that node* 3 *exists. The query is* /R/B/C *with node* 3 *as the output. The formula of the constraint is* $\bigwedge_{n \in \mathsf{path\_node}(0,3)} f(n) = f(0) \wedge f(2) \wedge f(3) = e_0 \wedge e_2 \wedge e_3$. *Second, we consider the constraint that node* 4 *does not exist. The query is* $\neg$(/R/B/D)*, with node* 4 *as the output. The formula of the constraint is* $\neg(\bigwedge_{n \in \mathsf{path\_node}(0,4)} f(n)) = \neg(f(0) \wedge f(2) \wedge f(4)) = \neg(e_0 \wedge e_2 \wedge e_4)$.

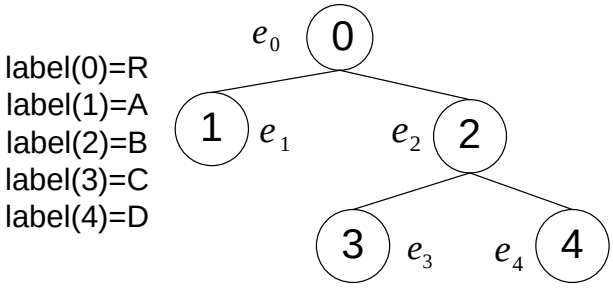


Figure 2: p-Document with independent choices

Mutually exclusive constraints A *mutually exclusive constraint* (or *mutex* constraint for short) over a set $\{x_1, x_2, \ldots, x_q\}$ 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 4.2. *Follow Example 4.1. 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 complex events associated to nodes are general elements of $\mathsf{C}(E)$, 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, relies on the notion of *ws-sets* and *ws-trees* introduced in [18], is presented as Algorithm 1.

The exponential-time running time is a direct consequence of the running time of the conditioning algorithm of [18]. We now show the correction of this algorithm:

Proposition 5.1. *Algorithm 1 solves the conditioning problem, as long as the input document is consistent.*

Algorithm 1: General conditioning algorithm for probabilistic XML data

Data: $\mathscr{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$
Result: $\mathscr{D}_2 = \langle D, E_2, f_2, \emptyset, p_2 \rangle$ such that $\mathscr{D}_1 \equiv_w \mathscr{D}_2$

1 **foreach** *node n* in *breadth-first-traversal of D* **do**
2 $\quad F_1(n) \leftarrow \bigwedge_{n' \in \mathsf{path\_node}(\mathsf{root}(D), n)} f_1(n')$;
3 $F_2 \leftarrow$result of the conditioning algorithm in [18] applied with $F_1(n)$ the constraint for each node n;
4 **foreach** *node n* in *breadth-first-traversal of D* **do**
5 $\quad$ Let α_n such that $F_2(n) = f_2(\mathsf{parent}(n)) \wedge \alpha_n$;
6 $\quad f_2(n) \leftarrow \alpha_n$;

Proof. For each $D' \in \mathfrak{P}(\mathscr{D}_2)$, according to Definition 3.5, the probability of D' is:

$$p_{\mathscr{D}_2}(D') = p\left(\bigwedge_{n \in D'} \bigwedge_{n_i \in \mathsf{path\_node}(\mathsf{root}(D'), n)} f_2(n_i) \right.$$
$$\left. \wedge \bigwedge_{n \notin D'} \neg \left(\bigwedge_{n_i \in \mathsf{path\_node}(\mathsf{root}(D'), n)} f_2(n_i) \right) \right).$$

Because of line 4 to 6 in Algorithm 1,

$$p_{\mathscr{D}_2}(D') = \bigwedge_{n \in D'} F_2(n) \wedge \bigwedge_{n \notin D'} \neg F_2(n).$$

Before conditioning, suppose there is a probabilistic relation with only one attribute which stores the identifiers of nodes. The formula associated with the tuple representing node n is $F_1(n) = \bigwedge_{n' \in \mathsf{path\_node}(\mathsf{root}(D), n)} f_1(n')$ (line 2 in Algorithm 1). This probabilistic relation gives the probabilistic conditions of all nodes of the probabilistic XML document $\mathscr{D}_1$. The parent-child relationship in $\mathscr{D}_1$ is reflected in the formula of the probabilistic relation. We use the conditioning algorithm in [18] (line 3 in Algorithm 1) in this probabilistic relational setting. The formula of each tuple is updated to $F_2(n)$. According to Theorem 5.3 of [18],

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

But then, according to Definition 3.5,

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

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

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 $\mathsf{PrXML}^{\mathsf{fie}}$ case).

Proposition 5.2. *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 ϕ 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 ϕ and all other nodes are annotated with true. Then the global constraint is ϕ and thanks to Lemma 3.7, checking consistency amounts to checking satisfiability of ϕ. □

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 5.3. *For all $k \geq 1$ there is a probabilistic XML document $\mathscr{D}_1 = \langle D, E_1, f_1, C_1, p_1 \rangle$ with $|D| = O(k)$, $|E| = O(k)$, and for all $n \in D$, $|f_1(n)| = O(1)$ such that every unconstrained probabilistic XML document $\langle D, E_2, f_2, \emptyset, p_2 \rangle$ such that $\sum_{n \in D} |f_2(n)| = \Omega(2^k)$.*

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 $2^{2^k} - 1$ 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. □

Assuming the constraint is not part of the input, or 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 5.4. *If there exists an* NP*-definable query Q such that, for all $k \geq 1$ there exists a probabilistic XML document $\mathscr{D}_1$ of depth 1 with independent events on all nodes constrained by Q such that no unconstrained document $\mathscr{D}_2$ having the same set of possible worlds has representation size in $O(n)$, then there exists an* NP *problem for which all circuits are supra-linear.*

Proof. This is immediate: if a linear-size circuit (with inputs the variables of E_2 and output the nodes of D) existed, it could be linearly translated to a set of formulae for each node, using standard tricks coding circuits to Boolean expressions. [21] □

The existence of an NP (or even PTIME) problem with a supra-linear circuit is a long-standing open problem. [14] that has potential applications to the P/NP problem [3].

There is therefore little hope to find reasonable queries that would force a non-linear blowup 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:

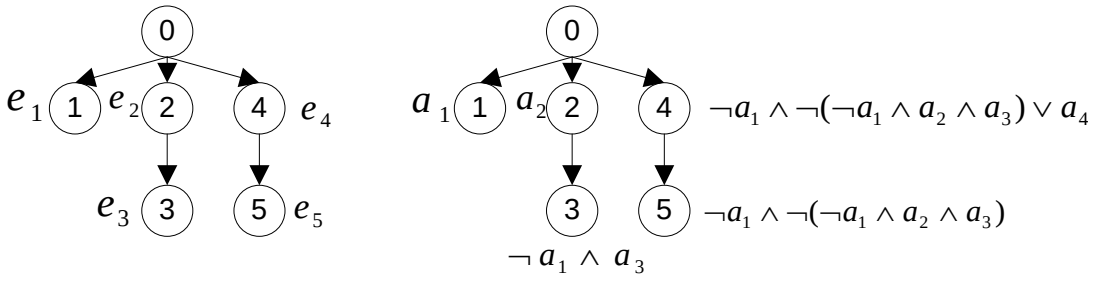


Figure 3: Two probabilistic XML documents in Example 5.5

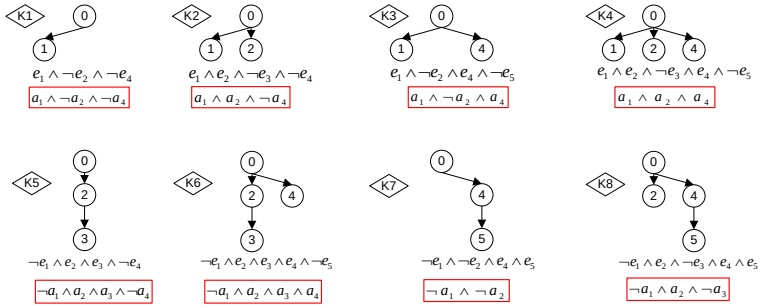


Figure 4: Eight possible worlds for probabilistic XML documents in Figure 3

Example 5.5. *Consider the probabilistic XML document $\mathscr{D}_1$ 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 $\mathscr{D}_2$ is on the right-hand-side in Figure 3.*

$\mathscr{D}_1$ and $\mathscr{D}_2$ have the same set of possible worlds as presented in Figure 4. The corresponding formulae of possible worlds in $\mathscr{D}_1$ and $\mathscr{D}_2$ 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 $\mathscr{D}_1, \mathscr{D}_2$ 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.

6 mutually exclusive constraints

In this section, we introduce the kind of constraints we further consider, as special cases for which we will then present 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 $N = \{x_1, x_2, ..., x_q\}$. 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* ($\mathsf{PrXML}^{\mathsf{ind}}$) as input of the conditioning problem. In a probabilistic XML document $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$ 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.

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

- *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.

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

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

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

- *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:

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

  $Y = \{y_1 \dots y_b\}$ 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 $\forall\, y_1, y_2 \in Y_i \; (x_i = \mathsf{LCA}(y_1, y_2))$, where $i \in [1, g]$. Note that $q = g + b$.

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 6.1. *Let $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1\rangle$ be a consistent probabilistic XML document with independent events. If $f_1(x_i)$ is independent from C, after conditioning, it is possible to condition so that its formula $f_2(x_i)$ is a unique independent event, and $p_1(f_1(x_i)) = p_2(f_2(x_i))$.*

Proof. Since $f_1(x_i)$ is an unique independent event e_i and it is independent from C, we can choose $f_2(x_i)$ to be also an unique independent event, i.e., $f_2(x_i) = a_i$. Obviously, we have $p_2(a_i) = p_1(e_1)$ as proven below.

$$\begin{aligned} & p_2(a_i) = p_2(x_i|\mathsf{parent}(x_i)) = p_1((x_i|\mathsf{parent}(x_i))|C) \\ & = \frac{p_1((x_i|\mathsf{parent}(x_i)) \wedge C)}{p_1(C)} = \frac{p_1(x_i|\mathsf{parent}(x_i)) \cdot p_1(C)}{p_1(C)} \\ & = \frac{p_1(e_i) \cdot p_1(C)}{p_1(C)} = p_1(e_i). \quad \square \end{aligned}$$

□

Corollary 6.2. *Let $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1\rangle$ be a consistent probabilistic XML document with independent events and C a mutually exclusive constraint over the set of nodes N. After conditioning (1) the formulae of all the nodes in the paths from $\mathsf{root}(D)$ to $x \in N$ must be updated and the probabilities of new events must be defined; (2) for other nodes n_i (which are not in those paths), the formulae and probabilities can be left unmodified.*

The formulae of mutually exclusive constraints are presented in Section 4. Corollary 6.2 is a direct consequence of Theorem 6.1, because if a node n_i is not in the paths from $\mathsf{root}(D)$ to $n \in N$, then its formula $f_1(n_i)$ is independent from C. We can exclude such irrelevant nodes for the conditioning operation. Under Corollary 6.2 we give the formal definition of the local tree in next.

Definition 6.3. *(Local tree) Assume $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1\rangle$ as a probabilistic XML document with independent events and C a mutually exclusive constraint over the set of nodes N. The **local tree**, that we denote by $\mathsf{LT}(C, D)$, 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 6.4. *Consider the probabilistic XML document in Figure 2 and the constraint in Example 4.2. The local tree is the tree in Figure 2, excluding node 4 and the edge from node 2 to node 4.*

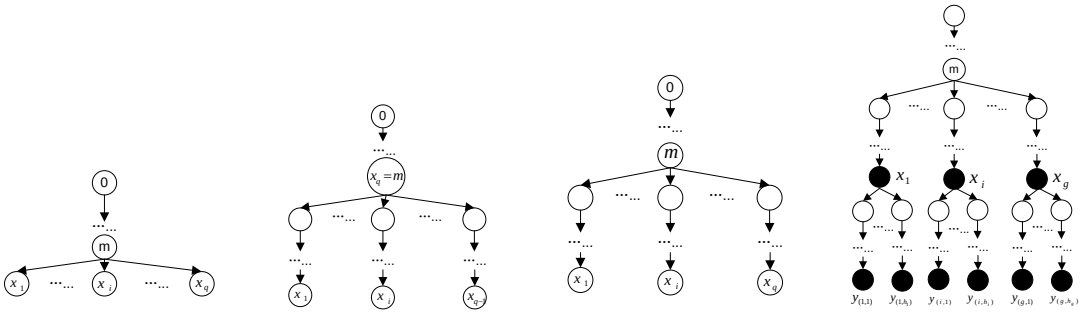
Theorem 6.5. *Let us give two probabilistic XML documents $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1\rangle$ and $\mathscr{D}_2 = \langle D, E_2, f_2, \emptyset, p_2\rangle$ with C as a mutually exclusive constraint. We claim that $\mathscr{D}_1 \equiv_w \mathscr{D}_2$ iff*

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

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

Definition 6.6. *(Local possible worlds) The **local possible worlds** of $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1\rangle$ correspond to possible worlds of $\langle LT(C, D), E_1, f_1, C, p_1\rangle$.*

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



(a) local tree of *MES* constraints (b) local tree of *MEAD* constraints (c) local tree of *MED* constraints (d) local tree of *MED&AD* constraints

Figure 5: Local trees under considered mutually exclusive constraints

Table 1: Number of local possible worlds of different constraints under different semantics

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	$m+1+g(k_1^{g-1}k_2^h)+k_1^g$	$g(k_1^{g-1}k_2^h)$	$m+1+g(k_1^{g-1}k_2^h)$

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 $N=\{x_1\ldots x_q\}$. 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 $(x_1,x_2,...,x_g)$ 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.

7 MutEx Siblings Constraints

In this section, we consider the *mutually exclusive siblings (MES)* constraint, i.e., all the nodes in N 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 $\mathsf{path\_node}(0,m)$. Each node is associated with

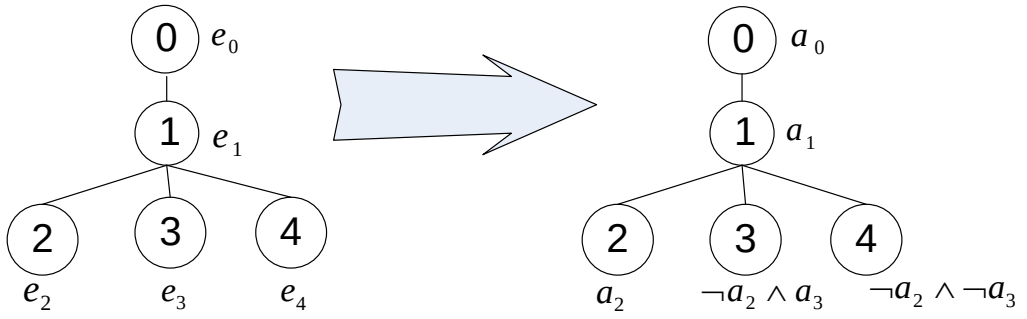


Figure 6: Probabilistic XML documents in Example 7.1

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

7.1 Without Maybe Semantics

7.1.1 WOMBA Semantics

In this section, we consider the *MES* constraint under WOMBA semantics. The constraint C tells that exactly only one of node m's children exists. It can be formulated as follows:

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

The probability of the constraint is then:

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

Example 7.1. *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 6.5, there is no need to input the entire document D of $\mathscr{D}_1 = \langle D, E_1, f_1, C, p_1 \rangle$, therefore we only consider the local tree $LT(C,D)$. Line 1 to line 2 assign new formulae to the nodes in $\mathsf{path\_node}(0,m)$. 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 $\mathsf{path\_node}(0,m)$ must exist, otherwise none of the nodes in N exists. Hence the probabilities of the events associated with the nodes in $\mathsf{path\_node}(0,m)$ are 1 (in Equation (1)). Equations (2), (3), (4) enumerate all local possible worlds and state that their probabilities are unchanged after conditioning. 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 $O(m+q)$.

Example 7.2. *Follow Example 7.1 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:*

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

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.

Theorem 7.3. *Algorithm 2 is correct.*

Proof. The set of equations in line 7 of Algorithm 2 is developed based on Definition 3.9. 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: 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* $i \in \mathsf{path\_node}(0,m)$ **do**
2 $\quad f_2(i) \leftarrow a_i$;
3 $f_2(m+1) \leftarrow a_{m+1}$;
4 **for** $i \in [2, q-1]$ **do**
5 $\quad f_2(m+i) \leftarrow \neg a_{m+1} \wedge \neg a_{m+2} \wedge ... \wedge a_{m+i}$;
6 $f_2(m+q) \leftarrow \neg a_{m+1} \wedge \neg a_{m+2} \wedge ... \wedge \neg a_{m+q-1}$;
7 The probabilities of the new events are computed by solving the following set of equations:

$$p_2(a_0) = p_2(a_1) = ... = p_2(a_m) = 1 \tag{1}$$

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

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

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

7.1.2 WOMBI Semantics

In this section, we consider the *MES* constraint under WOMBI semantics. The constraint tells that if the path from node 0 to node m exists, only one of node m's children exists. Under this semantics, nodes in $\mathsf{path\_node}(0,m)$ may not exist. The constraint can be formulated as:

$$\begin{aligned} C &= (\bigwedge_{i=0}^{m} e_i) \Rightarrow (\bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{u=1}^{q} (e_{m+u} \wedge \bigwedge_{i=1, i\neq u}^{q} \neg e_{m+i}))) \\ &= \neg(\bigwedge_{i=0}^{m} e_i) \vee (\bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{u=1}^{q} (e_{m+u} \wedge \bigwedge_{i=1, i\neq u}^{q} \neg e_{m+i}))) \end{aligned}$$

The probability of the constraint is

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

This value $p_1(C)$ can be computed in linear time to the size of constraint. The conditioning algorithm is similar to Algorithm 2, except that the set of equations in line 7 is different:

$$\begin{cases} 1-p_2(a_0) = \frac{1-p_1(e_0)}{p(C)} & (1) \\ k \in [1,m] & \\ \prod_{i=0}^{k-1} p_2(a_i)\cdot(1-p_2(a_k)) = \frac{\prod_{i=0}^{k-1} p_1(e_i)\cdot(1-p_1(e_k))}{p(C)} & (2) \\ \prod_{i=0}^{m} p_2(a_i)\cdot p_2(a_{m+1}) = & \\ \frac{\prod_{i=0}^{m} p_1(e_i)\cdot p_1(e_{m+1})\cdot\prod_{i=1,i\neq 1}^{q}(1-p_1(e_{m+i}))}{p_1(C)} & (3) \\ k \in [2,q-1] & \\ \prod_{i=0}^{m} p_2(a_i)\prod_{i=1}^{k-1}(1-p_2(a_{m+i}))\cdot p_2(a_{m+k}) & \\ = \frac{\prod_{i=0}^{m} p_1(e_i)\cdot p_1(e_{m+k})\cdot\prod_{i=1,i\neq k}^{q}(1-p_1(e_{m+i}))}{p_1(C)} & (4) \\ \prod_{i=0}^{m} p_2(a_i)\cdot\prod_{i=1}^{q-1}(1-p_2(a_{m+i})) & \\ = \frac{\prod_{i=0}^{m} p_1(e_i)\cdot p_1(e_{m+q})\cdot\prod_{i=1,i\neq q}^{q}(1-p_1(e_{m+i}))}{p_1(C)} & (5) \end{cases}$$

Equation (1)(2) define the probabilities of a_i ($i \in [0,m]$). Their probabilities are not 1 (as in Algorithm 2) because these nodes may not be actual according to the constraint. On the left-hand-side of Equation (3)(4)(5), a factor $\prod_{i=0}^{m} p_2(a_i)$ is added for the same reason. The complexity of the conditioning algorithm remains the same as Algorithm 2.

7.2 WMB Semantics

In this section, we consider the *MES* constraint under WMB semantics. The constraint says that if the path from node 0 to node m exists, at most one of node m's children exists. Under this semantics, nodes in $\mathsf{path\_node}(0,m)$ may not exist and there may be none of m's children existing even if the path from node 0 to node m exists. It can be formulated as:

$$\begin{aligned} C &= (\bigwedge_{i=0}^{m} e_i) \Rightarrow (\bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{u=1}^{q} (e_{m+u} \wedge \bigwedge_{i=1,i\neq u}^{q} \neg e_{m+i}) \vee \bigwedge_{i=1}^{q} \neg e_{m+i})) \\ &= \neg(\bigwedge_{i=0}^{m} e_i) \vee (\bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{u=1}^{q} (e_{m+u} \wedge \bigwedge_{i=1,i\neq u}^{q} \neg e_{m+i}) \vee \bigwedge_{i=1}^{q} \neg e_{m+i})) \end{aligned}$$

The probability of the constraint is given by the following expression:

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

This value $p_1(C)$ can be computed in linear time to the size of constraint. The conditioning algorithm is similar to the conditioning algorithm of WOMBI semantics, but with two differences. First, different from line 6 of Algorithm 2, the q^{th} child is associated with $\neg a_{m+1} \wedge \neg a_{m+2} \wedge \ldots \wedge a_{m+q}$. The Equation (5) of the conditioning algorithm of WOMBI semantics is replaced by the two equations below:

$$\begin{cases} \prod_{i=0}^{m} p_2(a_i)\prod_{i=1}^{q-1}(1-p_2(a_{m+i}))\cdot p_2(a_{m+q}) & \\ = \frac{\prod_{i=0}^{m} p_1(e_i)\cdot p_1(e_{m+q})\cdot\prod_{i=1,i\neq q}^{q}(1-p_1(e_{m+i}))}{p_1(C)} & \\ \prod_{i=0}^{m} p_2(a_i)\prod_{i=1}^{q}(1-p_2(a_{m+i})) = \frac{\prod_{i=0}^{m} p_1(e_i)\cdot\prod_{i=1}^{q}(1-p_1(e_{m+i}))}{p_1(C)} & \end{cases}$$

A new event a_{m+q} is introduced to represent the case that none of the nodes in N exists by assigning $a_{m+q} = \mathsf{false}$ when $a_i = \mathsf{false}$ $(i \in [m+1, m+q-1])$. The complexity of the conditioning algorithm remains the same as Algorithm 2.

8 MutEx AD Constraints

Let us now turn to *mutually exclusive Ancestor-Descendant (MEAD)* constraints, i.e., all nodes in N are in an ancestor-descendant relationship. In particular, 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 $(1 \leq i \leq m)$. In such a constraint, none of the $q-1$ nodes are in an AD relationship, so this restriction can be raised. However, we will relax this constraint at the end of this part in order to investigate constraints with nested AD relationships.

The ids of the remaining $q-1$ nodes in N are $x_1, ..., x_{q-1}$ and their associated events are $e_{x_1}, ..., e_{x_{q-1}}$. 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^{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 6.2, the local tree of the input probabilistic XML document is the part including nodes in $\mathsf{path\_node}(0, x_i)$ $(i \in [1, q-1])$. We will not discuss the formulae of other nodes.

8.1 Without Maybe Semantics

We details here *MEAD* constraints under different *without maybe semantics*.

8.1.1 WOMBA Semantics

We start with the *MEAD* constraint under 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:

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

Its probability is

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

Example 8.1. *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:*

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

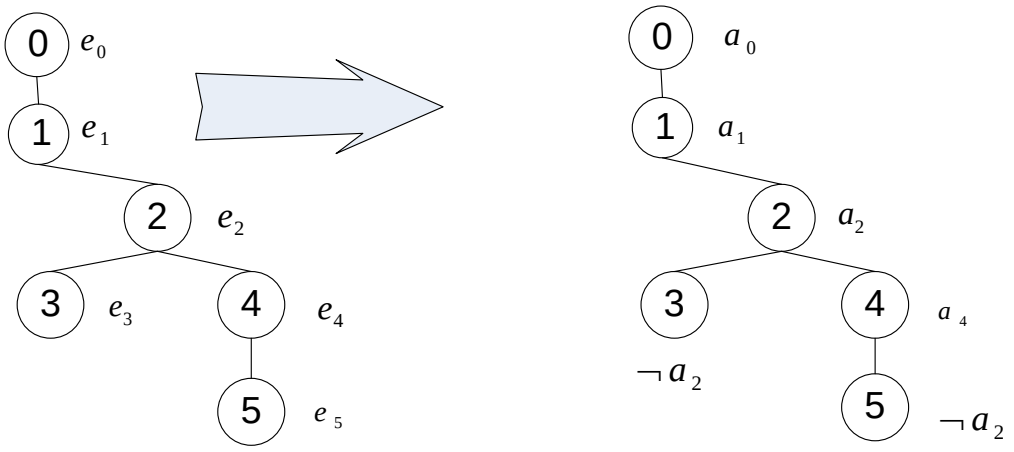


Figure 7: Probabilistic XML documents in Example 8.1

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. Lines 1 to 2 assign new formulae to the nodes in $\mathsf{path\_node}(0,m)$. 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. Lines 7–8 presents the set of equations to compute the probabilities of the new events. The probabilities of the events associated with the nodes in $\mathsf{path\_node}(0,m)$ are 1 (in Equation (5)). Equation (6) defines the probabilities of other new events. 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).

Example 8.2. *Follow Example 8.1 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 8.3. *Algorithm 3 is correct.*

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

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 $\langle LT(C,D), E_2, f_2, \emptyset, p_2 \rangle$

foreach *node* $i \in \mathsf{path\_node}(0,m)$ **do**
  $f_2(i) = a_i$;
for $i \in [1, q-1]$ **do**
  $f_2(x_i) \leftarrow \neg a_m$;
  for $j \in [1, k_i - 1]$ **do**
    $f_2((i,j)) \leftarrow a_{(i,j)}$;

The probabilities of the new events are computed by solving the following set of equations:

$$p_2(a_0) = p_2(a_1) = ... = p_2(a_m) = 1 \tag{5}$$

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

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

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

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

Therefore the probability of this possible world is

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

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

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. □

In the following section, we deal with *MEAD* constraints under *WOMBI semantics*.

8.1.2 WOMBI Semantics

The semantics of the *MEAD* constraint under WOMBI semantics means that if the path from node 0 to node $m-1$ exists, node m exists while the other nodes in N do not. Under this semantics, nodes in $\mathsf{path\_node}(0, m-1)$ may not exist. The constraint can be formulated as:

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

The probability of the constraint is given by

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

The obtained value $p_1(C)$ can be computed in linear time to the size of constraint. The conditioning algorithm is similar to Algorithm 3. However, probabilities of $a_i (i \in [0, m-1])$ are not 1 (in Equation (0)), because these nodes may not be actual. Their probabilities are defined as:

$$\begin{cases} 1 - p_2(a_0) = \frac{1 - p_1(e_0)}{p_1(C)} \\ k \in [1, m-1] \\ \prod_{i=0}^{k-1} p_2(a_i)(1 - p_2(a_k)) = \frac{\prod_{i=0}^{k-1} p_1(e_i) \cdot (1 - p_1(e_k))}{p_1(C)} \end{cases}$$

The complexity of the conditioning algorithm is the same as Algorithm 3.

8.2 WMB Semantics

Now, let us inroduce the *MEAD* constraint under *WMB semantics* in our setting. The constraint says that if the path from node 0 to node $m-1$ exists, at most one node in N exists. Under this semantics, the path may not exist and none of the nodes may exist even when the path exists. The constraint can be formulated as:

$$C = \bigwedge_{i=0}^{m-1} e_i \Rightarrow (\bigwedge_{i=0}^{m-1} e_i \wedge \neg e_m \vee \bigwedge_{i=0}^{m} e_i \wedge \bigwedge_{i=1}^{q-1} \neg (\bigwedge_{j=1}^{k_i} e_{(i,j)})$$

The probability of the constraint is

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

The conditioning algorithm is similar to the conditioning algorithm of WOMBI semantics. However, the probability of a_m is not 1 but defined as $\prod_{i=0}^{m-1} p_2(a_i)(1-p_2(a_m)) = \frac{\prod_{i=0}^{m-1} p_1(e_i)\cdot(1-p_1(e_m))}{p_1(C)}$. The reason is that even if the path from node 0 to node $m-1$ exists, node m still may not be actual, according to WMB semantics. The complexity of the conditioning algorithm is the same as Algorithm 3.

8.3 Relaxation: nested AD is allowed

Lemma 8.4. *If there is a pair of nodes (x_i, x_j) in the $q-1$ nodes being in ancestor-descendant relationship, we can remove x_j from N without affecting C and $p_1(C)$. x_j can be treated as other irrelevant nodes of the constraint C.*

Proof. We only prove the case in WOMBA semantics. The proofs for other two semantics are similar.

For the ease of representation, we use

$$F_1(n) = \bigwedge_{n' \in \mathsf{path\_node}(\mathsf{root}(D), n)} f_1(n')$$

$C = F_1(m) \wedge \bigwedge_{i=1}^{q-1} \neg(F_1(x_i))$. Assume x_j is a descendant of x_i, then it is always true to say that if $F_1(x_i)$ is false, then $F_1(x_j)$ is false. If C is true, $F_1(x_i)$ has to be false. Therefore we can safely remove $\neg(F_1(x_j))$ from C. The probability of C is unchanged.

We do not need to consider the formula of x_j and the probability of its event. Because $f_2(x_i) = \neg a_m$ and $p_2(a_m) = 1$, which means x_i is always not actual. Therefore we do not need to care about the descendant of x_i, including x_j. □

9 Mutex Descendance Constraints

In this section, we consider the *mutually exclusive descendance (MED)* constraint, i.e., every pair of the nodes in N shares the same lowest common ancestor – node m. The ids of the nodes in N are $\{x_1, ..., x_q\}$. 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^{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 6.2, the local tree of the input probabilistic XML document is the part including nodes in $\mathsf{path\_node}(0, x_i)$ $(i \in [1, q])$. We will not discuss the formulae of the other nodes.

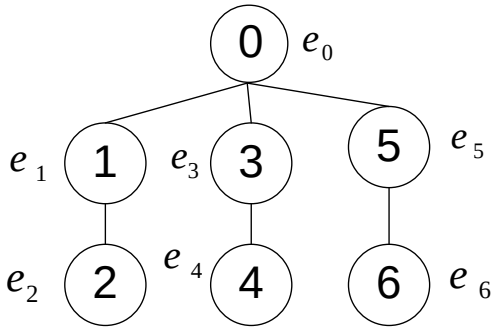


Figure 8: Probabilistic XML documents in Example 9.1

9.1 Without Maybe Semantics

9.1.1 WOMBA Semantics

We consider the *MED* constraint under WOMBA semantics.

The constraint is exactly one node in N exists. The constraint C says that (1) nodes in $\mathsf{path\_node}(0,m)$ 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:

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

The probability of the constraint is

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

$$= \prod_{i=0}^{m} p_1(e_i) \prod_{i=1}^{q} (1 - \prod_{j=1}^{k_i} p_1(e_{(i,j)})) (\sum_{i=1}^{q} \frac{\prod_{j=1}^{k_i} p_1(e_{(i,j)})}{1 - \prod_{j=1}^{k_i} p_1(e_{(i,j)})})$$

Example 9.1. *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:*

$$C = e_0 \wedge (e_1 \wedge e_2 \wedge \neg(e_3 \wedge e_4) \wedge \neg(e_5 \wedge e_6) \vee e_3 \wedge e_4 \wedge$$
$$\neg(e_1 \wedge e_2) \wedge \neg(e_5 \wedge e_6) \vee e_5 \wedge e_6 \wedge \neg(e_1 \wedge e_2) \wedge \neg(e_3 \wedge e_4))$$

Its probability is

$$p_1(C) = p_1(e_0)(p_1(e_1)p_1(e_2)(1 - p_1(e_3)p_1(e_4))(1 - p_1(e_5)p_1(e_6))$$
$$+ p_1(e_3)p_1(e_4)(1 - p_1(e_1)p_1(e_2))(1 - p_1(e_5)p_1(e_6)) +$$
$$p_1(e_5)p_1(e_6)(1 - p_1(e_1)p_1(e_2))(1 - p_1(e_3)p_1(e_4)))$$
$$= p_1(e_0)(1 - p_1(e_1)p_1(e_2))(1 - p_1(e_3)p_1(e_4))(1 - p_1(e_5)p_1(e_6))$$
$$(\frac{p_1(e_1)p_1(e_2)}{1 - p_1(e_1)p_1(e_2)} + \frac{p_1(e_3)p_1(e_4)}{1 - p_1(e_3)p_1(e_4)} + \frac{p_1(e_5)p_1(e_6)}{1 - p_1(e_5)p_1(e_6)})$$

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

Algorithm 4 is the conditioning algorithm for an *MED* 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 $\mathsf{path\_node}(0,m)$. Line 3 to line 4 assign new formulae to nodes in $\mathsf{path\_node}(m,x_1)$. Line 5 to line 8 assign formulae to the nodes in $\mathsf{path\_node}(m,x_i)$ $(i \in (1,q))$. Line 9 to line 11 assign new formulae to nodes in $\mathsf{path\_node}(m,x_q)$. Line 12 presents the set of equations to compute the probabilities of the new events. The probabilities of the events associated with the nodes in $\mathsf{path\_node}(0,m)$ are 1 (in Equation (1)).

In Algorithm 4, Equation (2) ensures that the probability of the possible world, in which node x_1 exists and all paths from m to $x_i (i \in [2,q])$ do not exist, remains the same after conditioning. For $i \in [2,q-1]$, Equation (3) ensures that the probability of the possible world, in which node x_i exists and all paths from m to $x_u (u \in [1,q], u \neq i)$ do not exist, is unchanged after conditioning. Equation (4) makes sure that the probability of the possible world, in which node x_q exists and all paths from m to $x_i (i \in [1,q-1])$ do not exist, is the same after conditioning.

Applying Equation (5) with Equation (3)(4) results in extending the set of possible worlds of which the probabilities remain the same after conditioning, by including the possible worlds in which there are some nodes in the path m to x_1. Applying Equation (6) or (7) with Equation (2)(3)(4) includes the possible worlds in which there are some nodes in the path m to $x_i (i \in [2,q])$. Applying Equation (8) with Equation (2), (3) includes the possible worlds in which there are some nodes in the path m to x_q.

Applying a combination of Equation (6)(7)(8) with Equation (2)(3)(4) extends the set of possible worlds of which the probabilities are the same in similar ways. We prove that these equations makes sure that the probabilities of all the possible worlds remain unchanged after conditioning.

Algorithm 4 introduces $m + k_1 + k_q - q + 1 + \sum_{i=2}^{q-1} 2k_i$ 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 Algorithm 4 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.

Example 9.2. *We illustrate the algorithm on Example 9.1. After the conditioning, the formula of all 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 8 newly 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 the probabilities are the same after the conditioning.

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

Algorithm 4: Conditioning algorithm for *MED* constraint under WOMBA semantics

Data: $\langle LT(C,D), E_1, f_1, C, p_1 \rangle$
Result: A world equivalent $\langle LT(C,D), E_2, f_2, \emptyset, p_2 \rangle$

foreach *node* $i \in \mathsf{path\_node}(0,m)$ **do**
  $f_2(i) = a_i$;
foreach *node* $(1,j) \in \mathsf{path\_node}(m,x_1)$, *excluding node m* **do**
  $f_2((1,j)) = a_{(1,j)}$
for $i \in (1,q)$ **do**
  for $j \in [1,k_i)$ **do**
    $f_2((i,j)) = \bigwedge_{u=1}^{i-1}(\neg(\bigwedge_{v=1}^{k_u} f_2((u,v)))) \wedge \theta_{(i,j)} \vee \eta_{(i,j)}$
  $f_2((i,k_i)) = (\bigwedge_{u=1}^{i-1}(\neg(\bigwedge_{v=1}^{k_u} f_2((u,v)))) \vee \neg(\bigwedge_{j=1}^{k_i-1} f_2((i,j)))) \wedge \eta_{(i,j)}$//node x_i
foreach *node* $(q,j) \in \mathsf{path\_node}(m,x_q)$, *excluding node m and node* x_q **do**
  $f_2((q,j)) = \bigwedge_{i=1}^{q-1}(\neg(\bigwedge_{v=1}^{k_i} f_2((i,v)))) \vee \alpha_j$
$f_2((q,k_q)) = \bigwedge_{i=1}^{q-1}(\neg(\bigwedge_{v=1}^{k_i} f_2((i,v))))$//node x_q;
The probabilities of the new events are computed by resolving the set of equations as

$$
\begin{cases}
p_2(a_0) = p_2(a_1) = ... = p_2(a_m) = 1 \qquad (1) \\
\prod_{j=1}^{k_1} p_2(a_{(1,j)}) \prod_{i=2}^{q-1}(1-p_2(\eta_{(i,1)})) \cdot (1-p_2(\alpha_1)) = \\
\frac{\prod_{j=1}^{k_1} p_1(e_{(1,j)}) \prod_{i=2}^{q}(1-p_1(e_{(i,1)}))}{p_1(C)} \qquad (2) \\
for \;\; i \in [2,q-1] \\
(1-p_2(a_{(1,1)})) \prod_{j=1}^{k_i-1}((p_2(\theta_{(i,j)}) + p_2(\eta_{(i,j)}) - p_2(\theta_{(i,j)}) \\
p_2(\eta_{(i,j)})))\eta_{(i,k_i)} \prod_{u=2,u\neq i}^{q-1}(1-p_2(\eta_{(u,1)}))(1-p_2(\alpha_1)) = \\
\frac{\prod_{j=1}^{k_i} p_1(e_{(i,j)}) \prod_{u=1,u\neq i}^{q}(1-p_1(e_{(u,1)}))}{p_1(C)} \qquad (3) \\
(1-p_2(a_{(1,1)}) \prod_{i=2}^{q-1}((1-p_2(\eta_{(i,1)})) \cdot (1-p_2(\theta_{(i,1)}))) = \\
\frac{\prod_{j=1}^{k_q} p_1(e_{(q,j)}) \prod_{i=1}^{q-1}(1-p_1(e_{(i,1)}))}{p_1(C)} \qquad (4) \\
for \;\; j \in [1,k_1-1] \\
\frac{1-p_2(a_{(1,j)})}{p_2(a_{(1,j)})(1-p_2(a_{(1,j+1)}))} = \frac{1-p_1(e_{(1,j)})}{p_1(e_{(1,j)})(1-p_1(e_{(1,j+1)}))} \qquad (5) \\
for \;\; i \in [2,q-1] \\
for \;\; j \in [1,k_i-2] \\
\frac{1-p_2(\eta_{(i,j)})}{p_2(\eta_{(i,j)})(1-p_2(\eta_{(i,j+1)}))} = \frac{1-p_1(e_{(i,j)})}{p_1(e_{(i,j)})(1-p_1(e_{(i,j+1)}))} \\
= \frac{(1-p_2(\eta_{(i,j)}))\cdot(1-p_2(\theta_{(i,j)}))}{(p_2(\eta_{(i,j)})+p_2(\theta_{(i,j)})-p_2(\eta_{(i,j)})\cdot p_2(\theta_{(i,j)}))\cdot(1-p_2(\eta_{(i,j+1)}))(1-p_2(\theta_{(i,j+1)}))} \qquad (6) \\
\frac{1-p_2(\eta_{(i,k_i-1)})}{p_2(\eta_{(i,k_i-1)})} = \frac{1-p_1(e_{(i,k_i-1)})}{p_1(e_{(i,k_i-1)})(1-p_1(e_{(i,k_i)}))} \\
= \frac{(1-p_2(\theta_{(i,k_i-1)}))\cdot(1-p_2(\eta_{(i,k_i-1)}))}{(p_2(\theta_{(i,k_i-1)})+p_2(\eta_{(i,k_i-1)})-p_2(\theta_{(i,k_i-1)})\cdot p_2(\eta_{(i,k_i-1)}))(1-p_2(\eta_{(i,k_i)}))} \qquad (7) \\
for \;\; j \in [1,k_q-2] \\
\frac{1-p_2(\alpha_j)}{p_2(\alpha_j)(1-p_2(\alpha_{j+1}))} = \frac{1-p_1(e_{(q,j)})}{p_1(e_{(q,j)})(1-p_1(e_{(q,j+1)}))} \qquad (8)
\end{cases}
$$

$$
\begin{cases}
p_2(a_0) = 1 \\
p_2(a_1)p_2(a_2)(1-p_2(\eta_1))(1-p_2(\alpha_1)) = \dfrac{p_1(e_1)p_1(e_2)(1-p_1(e_3))(1-p_1(e_5))}{p_1(C)} \\
(1-p_2(a_1))(p_2(\theta_1)+p_2(\eta_1)-p_2(\theta_1)p_2(\eta_1))\cdot p_2(\eta_2)(1-p_2(\alpha_1)) = \\
\qquad \dfrac{(1-p_1(e_1))p_1(e_3)p_1(e_4)(1-p_1(e_5))}{p_1(C)} \\
(1-p_2(a_1))(1-p_2(\eta_1))(1-p_2(\theta_1)) = \dfrac{(1-p_1(e_1))(1-p_1(e_3))p_1(e_5)p_1(e_6)}{p_1(C)} \\
\dfrac{1-p_2(a_1)}{p_2(a_1)(1-p_2(a_2))} = \dfrac{1-p_1(e_1)}{p_1(e_1)(1-p_1(e_2))} \\
\dfrac{1-p_2(\eta_1)}{p_2(\eta_1)} = \dfrac{1-p_1(e_3)}{p_1(e_3)(1-p_1(e_4))} \\
\dfrac{(1-p_2(\eta_1))(1-p_2(\theta_1))}{(p_2(\theta_1)+p_2(\eta_1)-p_2(\theta_1)p_2(\eta_1))(1-p_2(\eta_2))} = \dfrac{1-p_1(e_3)}{p_1(e_3)(1-p_1(e_4))}
\end{cases}
$$

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* $p_2(a_0)=1, p_2(a_1)=\frac{143}{189}, p_2(a_2)=\frac{120}{143}, p_2(\theta_1)=\frac{15}{23}, p_2(\eta_1)=\frac{1}{4}, p_2(\eta_2)=\frac{15}{17}, p_2(\alpha_1)=\frac{1}{5}$.

The correctness of the equation system is based on a few technical definitions and results:

Definition 9.3. *(primitive possible worlds) A possible world is* ***primitive*** *if all true nodes are in the same path.*

Lemma 9.4. *The probabilities of all primitive possible worlds are preserved by Algorithm 4.*

Proof. It is easy to check that Equation $(2), (3)$ and (4) preserve the probabilities of all primitive possible worlds. □

Definition 9.5. *(adjacent possible world) Let* W *and* W^* *be two possible worlds where the only difference is that one node is false in* W *and true in* W^*. *Then we say* W *is* ***adjacent*** *to* W^* $(W \prec_1 W^*)$.

Lemma 9.6. *If* $W \prec_1 W^*$, *then Algorithm 4 guarantees* $\frac{p_2(W)}{p_2(W^*)} = \frac{p_1(W)}{p_1(W^*)}$.

Proof. Assume that the different node is $(2, j)$ and $1 < j < k_j - 1$ for simplicity. The proof is similar for other pairs of adjacent possible worlds.

Case 1 All nodes in the first branch exist.

Assume in W, each branch ends at (i, t_i) for $i = 3, \ldots, q$.

Then

$$
\begin{aligned}
&W = a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)} \wedge f_2(2,1) \wedge \ldots \wedge f_2(2,j-1) \wedge (\neg f_2(2,j)) \\
&\bigwedge_{i=3,\ldots,q} \Big(\bigwedge_{l=1,\ldots,t_i} f_2(i,l) \wedge (\neg f_2(i,t_i)) \Big) \\
&= a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)} \wedge \eta_{(2,1)} \wedge \ldots \wedge \eta_{(2,i-1)} \wedge (\neg \eta_{(2,i)}) \bigwedge S
\end{aligned}
$$

where S is the tail part for W.

And

$$W^* = a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)} \wedge f_2(2,1) \wedge \ldots \wedge f_2(2,j) \wedge (\neg f_2(2,j+1))$$
$$\bigwedge_{i=3,\ldots,q} \Big(\bigwedge_{l=1,\ldots,t_i} f_2(i,l) \wedge (\neg f_2(i,t_i)) \Big)$$
$$= a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)} \wedge \eta_{(2,1)} \wedge \ldots \wedge \eta_{(2,i)} \wedge (\neg \eta_{(2,i+1)}) \bigwedge S^*$$

where S^* is the tail part for W^*.

Since all nodes in the first branch are true for W and W^*, so $S = S^*$. Hence $\frac{p_2(W)}{p_2(W^*)} = \frac{1-p_2(\eta_{(2,j)})}{p_2(\eta_{(2,j)})(1-p_2(\eta_{(2,j+1)}))}$.

By Equation (6),
$\frac{1-p_2(\eta_{(2,j)})}{p_2(\eta_{(2,j)})(1-p_2(\eta_{(2,j+1)}))} = \frac{1-p_1(e_{(2,j)})}{p_1(e_{(2,j)})(1-p_1(e_{(2,j+1)}))} = \frac{p_1(W)}{p_1(W^*)}$.

Case 2 Not all nodes in the first branch exist.

Assume in W, each branch ends at (i,t_i) for $i \neq 2$.

Then

$$W = a_{(1,1)} \wedge \ldots \wedge a_{(1,t_1-1)} \wedge (\neg a_{(1,t_1)}) \wedge f_2(2,1) \wedge \ldots \wedge f_2(2,j-1)$$
$$\wedge (\neg f_2(2,j)) \bigwedge_{i=3,\ldots,q} \Big(\bigwedge_{l=1,\ldots,t_i} f_2(i,l) \wedge (\neg f_2(i,t_i)) \Big)$$
$$= a_{(1,1)} \wedge \ldots \wedge a_{(1,t_1-1)} \wedge (\neg a_{(1,t_1)}) \wedge (\theta_{(2,1)} \vee \eta_{(2,1)}) \wedge \ldots$$
$$\wedge (\theta_{(2,i-1)} \vee \eta_{(2,i-1)}) \wedge (\neg \theta_{(2,i)} \wedge \neg \eta_{(2,i)}) \bigwedge S$$

where S is the tail part for W.

And

$$W^* = a_{(1,1)} \wedge \ldots \wedge a_{(1,t_1-1)} \wedge (\neg a_{(1,t_1)}) \wedge f_2(2,1) \wedge \ldots \wedge f_2(2,j)$$
$$\wedge (\neg f_2(2,j+1)) \bigwedge_{i=3,\ldots,q} \Big(\bigwedge_{l=1,\ldots,t_i} f_2(i,l) \wedge (\neg f_2(i,t_i)) \Big)$$
$$= a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)} \wedge (\theta_{(2,1)} \vee \eta_{(2,1)}) \wedge \ldots \wedge (\theta_{(2,i)} \vee \eta_{(2,i)}) \wedge$$
$$(\neg \theta_{(2,i+1)} \wedge \neg \eta_{(2,i+1)}) \bigwedge S^*$$

where S^* is the tail part for W^*.

Since $\neg(a_{(1,1)} \wedge \ldots \wedge a_{(1,k_1)})$ and $\neg(f_2(2,1) \wedge \ldots \wedge f_2(2,k_2))$, so $S = S^*$. Hence
$\frac{p_2(W)}{p_2(W^*)} =$
$\frac{(1-p_2(\eta_{(2,j)}))(1-p_2(\theta_{(2,j)}))}{(p_2(\eta_{(2,j)})+p_2(\theta_{(2,j)})-p_2(\eta_{(2,j)})p_2(\theta_{(2,j)}))(1-p_2(\eta_{(2,j+1)}))(1-p_2(\theta_{(2,j+1)}))}$
$= \frac{p_1(W)}{p_1(W^*)}$ by Equation (6). □

Combining Lemma 9.4 and Lemma 9.6, the probabilities of all possible worlds are preserved.

Theorem 9.7. *Algorithm 4 is correct, assuming that the equation system has a solution.*

Proof. It is easy to see that this algorithm does not introduce new possible worlds. Then it suffices to verify that the algorithm preserves the probability of each possible world.

Let W be an arbitrary possible world. Then there are a sequence of possible worlds $W_1, \ldots, W_n = W$ and a primitive possible world W_0 such that $1 < i \leq n$, $W_{i-1} \prec_1 W_i$.

By Lemma 9.4, $p_1(W_0) = p_2(W_0)$.

By Lemma 9.6, $1 < i \leq n$, $\frac{p_1(W_i)}{p_1(W_{i-1})} = \frac{p_2(W_i)}{p_2(W_{i-1})}$.

Hence

$p_2(W) = (\prod_{i=1}^{n} \frac{p_2(W_i)}{p_2(W_{i-1})}) p_2(W_0) = (\prod_{i=1}^{n} \frac{p_1(W_i)}{p_1(W_{i-1})}) p_1(W_0) = p_1(W)$.

□

9.1.2 WOMBI Semantics

In this section, we consider the *MED* constraint under WOMBI semantics. The constraint tells that the path from node 0 to node m may not exist and if this path exists, one of node $x_i (i \in [1,q])$ exists. It can be formulated as:

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

The probability of the constraint is

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

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

$p_1(C)$ can be computed in linear time to the size of constraint. The conditioning algorithm is similar to Algorithm 4, except for two differences. First, the probabilities of $a_i (i \in [0,m])$ (in Equation (0)) are not 1. Because nodes $i \in [0,m]$ may not be actual. Instead, the probabilities of $a_i's$ are defined by

$$\begin{cases} 1 - p_2(a_0) = \frac{1 - p_1(e_0)}{p_1(C)} \\ k \in [1,m] \\ \prod_{i=0}^{k-1} p_2(a_i) \cdot (1 - p_2(a_k)) = \frac{\prod_{i=0}^{k-1} p_1(e_i) \cdot (1 - p_1(e_k))}{p_1(C)} \end{cases}$$

Second, Equation (1)(2)(3) have to be modified slightly, by adding the factor $\prod_{i=0}^{m} p_2(a_i)$ to the left-hand-side of these equations, and adding the factor $\prod_{i=0}^{m} p_1(e_i)$ to the right-hand-side of the equations. The complexity of this conditioning algorithm is the same as Algorithm 4.

9.2 WMB semantics

In this section, we consider *MED* constraint under WMB semantics. The constraint tells that the path from node 0 to node m may not exist and if this path exists, at most one node in N exists. It can be formulated as:

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

The probability of the constraint is

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

This probability $p_1(C)$ can be computed in linear time to the size of constraint. The conditioning algorithm is similar to Algorithm 4. However, the formulae of nodes in $\mathsf{path\_node}(m, x_1)$ and $\mathsf{path\_node}(m, x_q)$ are different from the formulae in Algorithm 4. The formulae of nodes in $\mathsf{path\_node}(m, x_1)$ are $f_2((1, j)) = a \vee x_{(1,j)}$, where $j \in [1, x_1]$. The formulae of nodes in $\mathsf{path\_node}(m, x_q)$ follow the formulae of nodes in $\mathsf{path\_node}(m, x_i)$ where $i \in (1, q)$ in line 5 to line 8 in Algorithm 4. This conditioning algorithm introduces $\sum_{i=1}^{q} 2k_i$ new events. The probabilities of the new events are computed by resolving the set of equations as

$$
\begin{cases}
1-p_2(a_0)=\frac{1-p_1(e_0)}{p_1(C)} & (1)\\
k\in[1,m] & \\
\prod_{i=0}^{k-1}p_2(a_i)\cdot(1-p_2(a_k))=\frac{\prod_{i=0}^{k-1}p_1(e_i)\cdot(1-p_1(e_k))}{p_1(C)} & (2)\\
\frac{(1-p_2(a_m))}{p_2(a_m)(1-p_2(a)p_2(x_{(1,1)}))\prod_{i=2}^{q}(1-\theta_{(i,1)})(1-\eta_{(i,1)})}= & \\
\frac{1-p_1(e_m)}{p_1(e_m)\prod_{i=1}^{q}(1-p_1(e_{(i,1)}))} & (3)\\
\frac{(1-p_2(a))p_2(x_{(1,1)})(1-p_2(x_{(1,2)}))}{(1-p_2(a))(1-p_2(x_{(1,1)}))}=\frac{p_1(e_1)(1-p_1(e_2))}{1-p_1(e_1)} & (4)\\
for\ j\in(1,k_1-1] & \\
\frac{p_2(x_{(1,j)})(1-p_2(x_{(1,j+1)}))}{1-p_2(x_{(1,j)})}=\frac{p_1(e_{(1,j)})(1-p_1(e_{(1,j+1)}))}{1-p_1(e_{(1,j)})} & (5)\\
\frac{(p_2(a)+\sum_{j=1}^{k_1}p_2(x_{(1,j)})-p_2(a)\sum_{j=1}^{k_1}p_2(x_{(1,j)}))\prod_{i=2}^{q}(1-p_2(\theta_{(i,1)}))}{(1-p_2(a))\prod_{j=1}^{k_i-1}p_2(x_{(1,j)})(1-p_2(x_{(1,k_i)}))}= & \\
\frac{1-p_1(e_{(1,k_1)})}{p_1(e_{(1,k_1)})} & (6)\\
for\ i\in[2,q] & \\
for\ j\in[1,k_i-1) & \\
\frac{(p_2(\theta_{(i,j)})+p_2(\eta_{(i,j)})-p_2(\theta_{(i,j)})p_2(\eta_{(i,j)}))(1-p_2(\theta_{(i,j+1)})(1-p_2(\eta_{(i,j+1)})}{(1-p_2(\theta_{(i,j)}))(1-\eta_{(i,j)})} & \\
=\frac{p_2(\eta_{(i,j))(1-p_2(\eta_{(i,j+1)}))}}{1-p_2(\eta_{(i,j)})}=\frac{p_1(e_{(i,j)})(1-p_1(e_{(i,j+1)}))}{1-p_1(e_{(i,j)})} & (7)\\
\frac{(p_2(\theta_{(i,k_i-1)})+p_2(\eta_{(i,k_i-1)})-p_2(\theta_{(i,k_i-1)})p_2(\eta_{(i,k_i-1)}))(1-p_2(\eta_{(i,k_i)}))}{(1-p_2(\theta_{(i,k_i-1)}))(1-p_2(\eta_{(i,k_i-1)}))} & \\
=\frac{p_2(\eta_{(i,k_i-1)})}{1-p_2(\eta_{(i,k_i-1)})}=\frac{p_1(e_{(i,k_i-1)})(1-p_1(e_{(i,k_i)}))}{1-p_1(e_{(i,k_i-1)})} & (8)\\
for\ i\in[2,q) & \\
\frac{p_2(\eta_{(i,k_i)})}{(1-p_2(\eta_{(i,k_i)}))\prod_{u=i+1}^{q}(1-p_2(\theta_{(u,1)}))}=\frac{p_1(e_{(i,k_i)})}{1-p_1(e_{(i,k_i)})} & (9)\\
\frac{p_2(\eta_{(q,k_q)})}{1-p_2(\eta_{(i,k_i)})}=\frac{p_1(e_{(q,k_q)})}{1-p_1(e_{(q,k_q)})} & (10)
\end{cases}
$$

Equation (1)(2) model the possible worlds that some nodes in the path from node 0 to node m do not exist and make sure the probabilities of these possible worlds unchanged after conditioning. Equation (3) models the ratio between the probabilities of the possible world of node m does not exist and the possible world of node m exists but none of its children exists. Equation (4)(5)(6) compute the probabilities of events in the path from m to x_1. Equation (7)(8)(9) compute the probabilities of events in the path from m to x_i where $i\in[2,q)$. Equation (7)(8)(10) compute the probabilities of events in the path from m to x_q. Similar to Algorithm 4, by combing these equations, the probabilities of all the possible worlds remain the same. For example, the possible world, in which only the path from node 0 to node x_1 exists, while all other children of node m do not exist, is modeled by Equation (3)(4)(5)(6) and the its probability value is the same because of these four equations. The complexity of this conditioning algorithm is the same as Algorithm 4. The proof of its correctness is similar to the proof of Theorem 9.7.

10 MED&AD Mutex Constraints

In this section, we consider the mutually exclusive constraints that combine the *MEAD* and *MED* constraints. The id of the root of D is 0. As introduced in Section 6, the set of mutually

exclusive nodes $N = X \cup Y$. As shown in Figure 5(d), $X = \{x_1, ..., x_g\}$ is the set of shaded nodes in the higher level and Y is the set of shaded nodes in the lower level. Node m is the lowest common ancestor of nodes in X. Assume for each x_i, there are h_i nodes in Y as its descendant: $y_{(i,1)}, ..., y_{(i,h_i)}$. The formal definition of this class of constraints refers to Section 6. For each node with id n, the associated event before conditioning is e_n.

According to Corollary 6.2, the local tree of the input probabilistic XML document is the part including nodes in $\mathsf{path\_node}(0, y_{(i,j)})$ $(i \in [1,g], j \in [1,h_i])$. We will not discuss the formulae of the other nodes.

10.1 Without Maybe Semantics

10.1.1 WOMBA Semantics

The *MED&AD* constraint under WOMBA semantics means that (1) nodes in $\mathsf{path\_node}(0,m)$ exist; (2) when node x_i exists, any $x_k (k \neq i)$ cannot exist, and any shaded descendant nodes of x_i (i.e. $y_{(i,j)} (j \in [1,h_i])$) cannot exist. It can be formulated as

$$C = \bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{i=1}^{g} (\bigwedge_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} e_n \wedge \bigwedge_{u=1, u \neq i}^{g} \neg (\bigwedge_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} e_n) \wedge \bigwedge_{j=1}^{h_i} \neg (\bigwedge_{n \in \mathsf{path\_node}(x_i, y_{(i,j)})}^{n \neq x_i} e_n)))$$

The probability of the constraint is

$$p_1(C) = \prod_{i=0}^{m} p_1(e_i) (\sum_{i=1}^{g} (\prod_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} p_1(e_n) \prod_{u=1, u \neq i}^{q} (1 - \prod_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} p_1(e_n)) \prod_{j=1}^{h_i} (1 - \prod_{n \in \mathsf{path\_node}(x_i, y_{(i,j)})}^{n \neq x_i} p_1(e_n))))$$

$p_1(C)$ can be computed in linear time to the size of the constraint. As we can see, the constraint considered in this section is the combination of the constraints considered in Section 8 and Section 9. More specifically, the part of the local tree (in Figure 5(d)) between root and $X = \{x_1, ..., x_g\}$ is same as the local tree of *MED* constraint (in Figure 5(c)), while the part of the local tree (in Figure 5(d)) between x_i $(i \in [1,g])$ and $\{y_{(i,1)}, ..., y_{(i,h_i)}\}$ is same as the local tree of the *MEAD* constraint (in Figure 5(b)).

We devise our conditioning algorithm, namely Algorithm 5, based on Algorithm 3 and Algorithm 4. In Algorithm 5, the latter algorithm is applied to the part of the local tree between root and $X = \{x_1, ..., x_g\}$ (line 1), and Algorithm 3 is applied to the parts of the local tree between x_i $(i \in [1,g])$ and $\{y_{(i,1)}, ..., y_{(i,h_i)}\}$ (line 3). The sets of new events have to be disjointed every time algorithms are applied.

The complexity of Algorithm 5 is polynomial-time. Its correctness can be proved by adapting the proofs of Theorem 8.3 and Theorem 9.7.

Algorithm 5: Conditioning algorithm for *MED&AD* mutually exclusive 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 For the tree consisting in the paths from node 0 to node $x_i (i \in [1,g])$, apply Algorithm 4. ;
2 **for** $i \in [1,g]$ **do**
3 for the tree consisting in the paths from node x_i to node $y_{(i,j)} (j \in [1,h_i])$ (excluding node x_i), apply Algorithm 3;

10.1.2 WOMBI Semantics

In this section, we consider the *MED&AD* mutually exclusive constraint under WOMBI semantics. Under this semantics, nodes in $\mathsf{path\_node}(0,m)$ may not exist and when this path exists only one node in N exists. The constraint can be formulated as

$$C = \bigwedge_{i=0}^{m} e_i \Rightarrow \bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{i=1}^{g} (\bigwedge_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} e_n \wedge \bigwedge_{u=1,u\neq i}^{g} \neg (\bigwedge_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} e_n) \wedge \bigwedge_{j=1}^{h_i} \neg (\bigwedge_{n \in \mathsf{path\_node}(x_i,y_{(i,j)})}^{n \neq x_i} e_n)))$$

The probability of the constraint is

$$p_1(C) = 1 - \prod_{i=0}^{m} p_1(e_i) + \prod_{i=0}^{m} p_1(e_i) (\sum_{i=1}^{g} (\prod_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} p_1(e_n) \prod_{u=1,u\neq i}^{q} (1 - \prod_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} p_1(e_n)) \prod_{j=1}^{h_i} (1 - \prod_{n \in \mathsf{path\_node}(x_i,y_{(i,j)})}^{n \neq x_i} p_1(e_n))))$$

The conditioning algorithm is similar to Algorithm 5, but by combing the algorithms in Section 8.1.2 and Section 9.1.2.

10.2 With Maybe semantics

In this section, we consider the *MED&AD* mutually exclusive constraint under WMB semantics. Under this semantics, the nodes in $\mathsf{path\_node}(0,m)$ may not exist and at most one node in N exists when this path exists. The constraint can be formulated as

$$C = \bigwedge_{i=0}^{m} e_i \Rightarrow \bigwedge_{i=0}^{m} e_i \wedge (\bigvee_{i=1}^{g} (\bigwedge_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} e_n \wedge \bigwedge_{u=1,u\neq i}^{g} \neg (\bigwedge_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} e_n) \wedge \bigwedge_{j=1}^{h_i} \neg (\bigwedge_{n \in \mathsf{path\_node}(x_i,y_{(i,j)})}^{n \neq x_i} e_n)) \vee \bigwedge_{i=1}^{g} \neg (\bigwedge_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} e_n))$$

The probability of the constraint is

$$p_1(C) = 1 - \prod_{i=0}^{m} p_1(e_i) + \prod_{i=0}^{m} p_1(e_i)(\sum_{i=1}^{g}(\prod_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} p_1(e_n) \prod_{u=1, u \neq i}^{q} (1 - \prod_{n \in \mathsf{path\_node}(m,x_u)}^{n \neq m} p_1(e_n)) \prod_{j=1}^{h_i} (1 - \prod_{n \in \mathsf{path\_node}(x_i, y_{(i,j)})}^{n \neq x_i} p_1(e_n))) + \prod_{i=1}^{q} (1 - \prod_{n \in \mathsf{path\_node}(m,x_i)}^{n \neq m} p_1(e_n)))$$

The conditioning algorithm is similar to Algorithm 5, but by combing the algorithms in Section 8.2 and Section 9.2.

11 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 $\mathsf{PrXML}^{\mathsf{ind}}$ document is more powerful than simply adding mux nodes as the mutually exclusive constraints considered span the whole document. Conditioning is also fundamental in uncertainty modeling frameworks in artificial intelligence, such as probabilistic logic [20]; in these settings, conditioning amounts to eliminating or revising the likelihood of some possible worlds.

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] S. Abiteboul, B. Kimelfeld, Y. Sagiv, and P. Senellart. On the expressiveness of probabilistic XML models. *VLDB Journal*, 18(5):1041–1064, 2009.

[2] S. Abiteboul and P. Senellart. Querying and updating probabilistic information in XML. In *Proc. EDBT*, 2006.

[3] E. Allender. Chipping away at P vs NP: How far are we from proving circuit size lower bounds? In *CATS*, page 3, 2008.

[4] S. Amer-Yahia, S. Cho, L. V. S. Lakshmanan, and D. Srivastava. Minimization of tree pattern queries. In *Proc. SIGMOD*, 2001.

[5] M. L. Ba, T. Abdessalem, and P. Senellart. Uncertain version control in open collaborative editing of tree-structured documents. In *Proc. DocEng*, Florence, Italy, 2013.

[6] D. Barbará, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *Proc. IEEE Trans. Knowl. Data Eng.*, 1992.

[7] R. Cavallo and M. Pittarelli. The theory of probabilistic databases. In *Proc. VLDB*, 1987.

[8] C.-H. Chang, M. Kayed, M. R. Girgis, and K. F. Shaalan. A survey of Web information extraction systems. *IEEE Trans. on Knowl. and Data Eng.*, 2006.

[9] S. Cohen, B. Kimelfeld, and Y. Sagiv. Incorporating constraints in probabilistic XML. In *Proc. PODS*, 2008.

[10] X. L. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. *VLDB Journal*, 2009.

[11] T. Eiter, T. Lukasiewicz, and M. Walter. A data model and algebra for probabilistic complex values. *Proc. Ann. Math. Artif. Intell.*, 2001.

[12] E. Hung, L. Getoor, and V. S. Subrahmanian. Probabilistic interval XML. In *Proc. ICDT*, 2003.

[13] E. Hung, L. Getoor, and V. S. Subrahmanian. PXML: A probabilistic semistructured data model and algebra. In *Proc. ICDE*, 2003.

[14] R. Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55(1-3):40–56, 1982.

[15] E. Kharlamov, W. Nutt, and P. Senellart. Updating probabilistic XML. In *Proc. EDBT/ICDT Workshops*, 2010.

[16] E. Kharlamov and P. Senellart. Modeling, Querying, and Mining Uncertain XML Data. In A. Tagarelli, editor, *XML Data Mining: Models, Methods, and Applications*. Proc. IGI Global, 2011.

[17] B. Kimelfeld and P. Senellart. Probabilistic XML: Models and complexity. In Z. Ma and L. Yan, editors, *Proc. Advances in Probabilistic Databases for Uncertain Information Management*. Springer-Verlag, 2013.

[18] C. Koch and D. Olteanu. Conditioning probabilistic databases. *Proc. PVLDB*, 2008.

[19] A. Nierman and H. V. Jagadish. ProTDB: Probabilistic data in XML. In *Proc. VLDB*, 2002.

[20] N. J. Nilsson. Probabilistic logic. *Artif. Intell.*, 28(1):71–87, 1986.

[21] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[22] D. Suciu, D. Olteanu, C. Ré, and C. Koch. *Probabilistic Databases*. Morgan & Claypool, 2011.

[23] R. Tang, R. Cheng, H. Wu, and S. Bressan. A framework for conditioning uncertain relational data. In *Proc. DEXA*, 2012.

[24] M. van Keulen, A. de Keijzer, and W. Alink. A probabilistic XML approach to data integration. In *Proc. ICDE*, 2005.