Probabilistic XML Data Management

Pierre Senellart

INFRES Seminar, 18 November 2010

Uncertain data

Numerous sources of uncertain data:

- Measurement errors
- Data integration from contradicting sources
- Imprecise mappings between heterogeneous schemata
- Imprecise automatic process (information extraction, natural language processing, etc.)
- Imperfect human judgment

Managing this imprecision

Objective

Not to pretend this imprecision does not exist, and manage it as rigorously as possible throughout a long, automatic and human, potentially complex, process.

Especially:

- Use probabilities to represent the confidence in the data
- Query data and retrieve probabilistic results
- Allow adding, deleting, modifying data in a probabilistic way
- (If possible) Keep throughout the process lineage/provenance information, so as to ensure traceability

Managing this imprecision

Objective

Not to pretend this imprecision does not exist, and manage it as rigorously as possible throughout a long, automatic and human, potentially complex, process.

Especially:

- Use probabilities to represent the confidence in the data
- Query data and retrieve probabilistic results
- Allow adding, deleting, modifying data in a probabilistic way
- (If possible) Keep throughout the process lineage/provenance information, so as to ensure traceability

Outline

Outline

The relational model

- Data stored into tables
- Every table has a precise schema (type of columns)
- Adapted when the information is very structured

Patient	Examin. 1	Examin. 2	Diagnosis
A	23	12	α
B	10	23	β
C	2	4	γ
D	15	15	α
E	15	17	β

Simple probabilistic annotations

Patient	Examin. 1	Examin. 2	Diagnosis	Probability
A	23	12	α	0.9
B	10	23	β	0.8
C	2	4	γ	0.2
C	2	14	γ	0.4
D	15	15	α	0.6
D	15	15	β	0.4
E	15	17	β	0.7
E	15	17	α	0.3

- Allow representation of the confidence in each row of the table
- Efficient algorithms to compute the probability of a query answer
- Impossible to express dependencies across rows

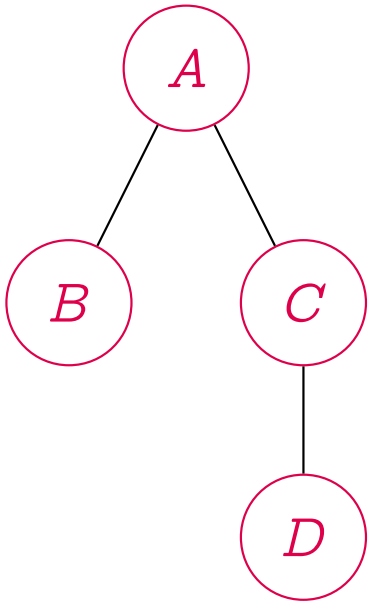
Annotations + constraints

Patient	Examin. 1	Examin. 2	Diagnosis	Probability
A	23	12	α	0.9
B	10	23	β	0.8
C	2	4	γ	0.2 $\}\oplus$
C	2	14	γ	0.4
D	15	15	β	0.6 $\}\oplus$
D	15	15	α	0.4
E	15	17	β	0.7 $\}\oplus$
E	15	17	α	0.3

- Still efficient algorithms for query answering
- Simple dependencies (exclusion) can be expressed, but not more complex ones

Outline

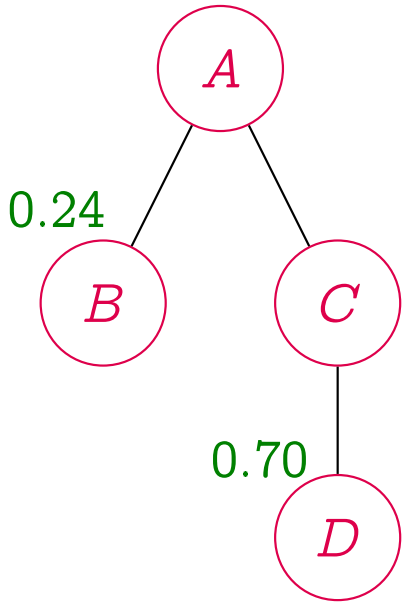
The semistructured model and XML




```
<a>
   <b>...</b>
   <c>
      <d>...</d>
   </c>
</a>
```

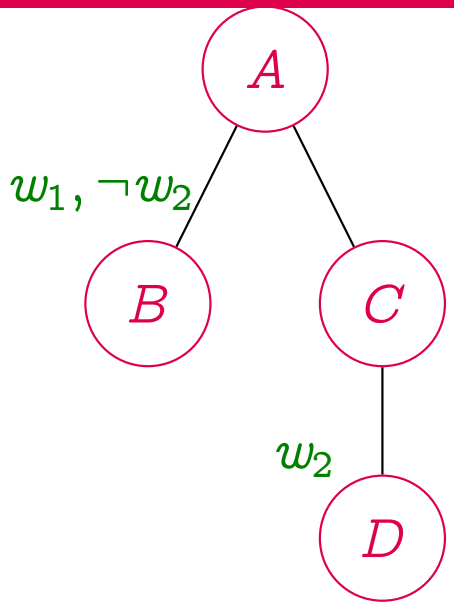
- Tree-like structuring of data
- No (or less) schema constraints
- Allow mixing tags (structured data) and text (unstructured content)
- Particularly adapted to tagged or heterogeneous content

Simple probabilistic annotations



- Probabilities associated to tree nodes
- Express parent/child dependencies
- Impossible to express more complex dependencies
- $\Rightarrow$ some sets of possible worlds are not expressible this way!

Annotations with event variables



semantics

$p_1 = 0.06$

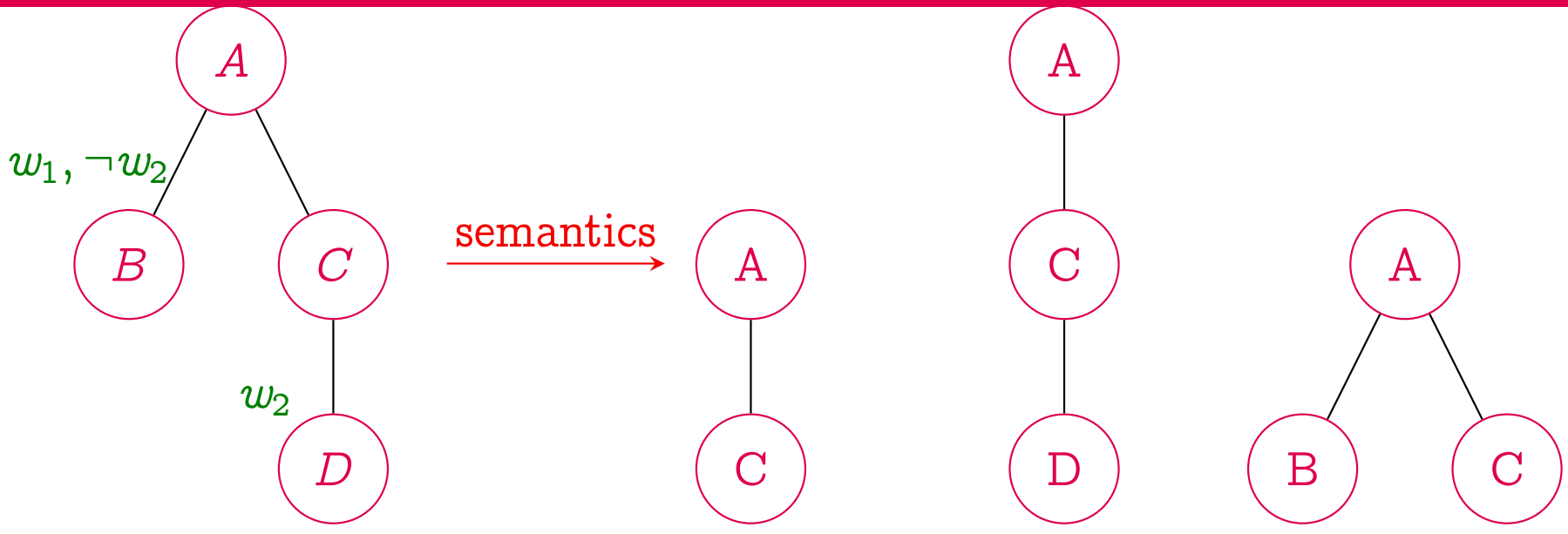
$p_2 = 0.70$

$p_3 = 0.24$

Event	Prob.
w_1	0.8
w_2	0.7

- Expresses arbitrarily complex dependencies
- Obviously, has an equivalent in the relational case

Annotations with event variables



Event	Prob.
w_1	0.8
w_2	0.7

$p_1 = 0.06$ $p_2 = 0.70$ $p_3 = 0.24$

- Expresses arbitrarily complex dependencies
- Obviously, has an equivalent in the relational case

Outline

Outline

Why XML?

- Extensive literature about probabilistic relational databases

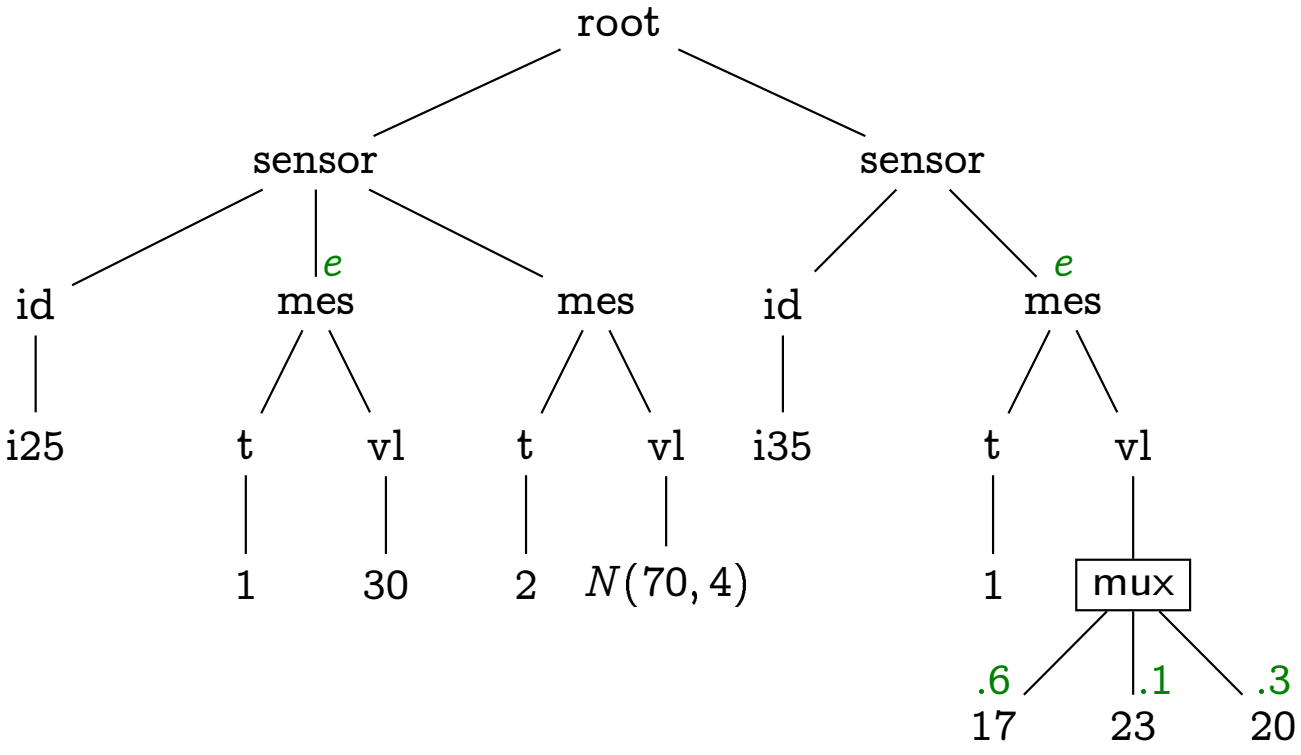
  [Dalvi et al., 2009, Widom, 2005, Koch, 2009]
- Cases where a tree-like model might be appropriate:
  - No schema or few constraints on the schema (Web data!)
  - Independent modules annotating freely a content warehouse
  - Inherently tree-like data (e.g., mailing lists, parse trees)
- Different typical querying languages: SQL vs XPath/XQuery, conjunctive queries vs tree-pattern queries

Remark

Some results can be transferred from one model to the other. In other cases, connection much trickier! (See later.)

A general probabilistic XML model

[Abiteboul et al., 2009]



- e: event "it did not rain" at time 1
- mux: mutually exclusive options
- $N(70, 4)$: normal distribution

- Compact representation of a set of possible worlds
- Two kinds of dependencies: global (e) and local (mux)
- Generalizes all previously proposed models of the literature

Events and lineage

- Event variables: can represent the provenance of data
- Typically:
  1. At each (probabilistic) update, a new event variable is introduced
  2. Query results are given with probabilities, but also with the lineage of the query [Foster et al., 2008]
- Allow to keep track, with no additional cost, of the provenance of data!

Outline

Querying probabilistic XML

Semantics of a (Boolean) query = probability:

1. Generate all possible worlds of a given probabilistic document

2. In each world, evaluate the query
3. Add up the probabilities of the worlds that make the query true

EXPTIME algorithm! Can we do better, i.e., can we apply directly the algorithm on the probabilistic document?

We shall talk about data complexity of query answering.

Querying probabilistic XML

Semantics of a (Boolean) query = probability:

1. Generate all possible worlds of a given probabilistic document (possibly exponentially many)
2. In each world, evaluate the query
3. Add up the probabilities of the worlds that make the query true

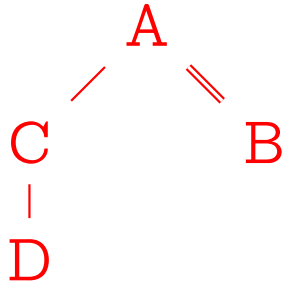
EXPTIME algorithm! Can we do better, i.e., can we apply directly the algorithm on the probabilistic document?

We shall talk about data complexity of query answering.

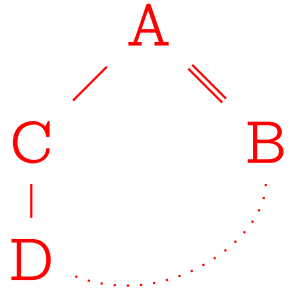
Boolean query languages on trees

Single-path queries (SP) (no branching)

Tree-pattern queries (TP)



Tree-pattern queries with joins (TPJ)



Monadic second-order queries (MSO) generalization of TP, do not cover TPJ unless the size of the alphabet is bounded

The #P and $FP^{\#P}$ complexity classes

- A (counting) problem is in #P if there is a PTIME non-deterministic Turing machine whose number of accepting paths, given as input the input of the problem, is the output of the problem.
- A problem is #P-hard if any #P problem can be PTIME-reduced to it (via a Turing reduction). #2DNF, the problem of counting the number of assignments satisfying a formula in 2-DNF, is #P-complete.

- A (computation) problem is in $FP^{\#P}$ if it is computable by a PTIME Turing machine with access to a $\#P$ oracle.
- A problem is $FP^{\#P}$-hard if any $FP^{\#P}$ problem can be PTIME-reduced to it (via a Turing reduction). Equivalently, a computation problem is $FP^{\#P}$-hard if it is #P-hard.

The #P and $FP^{\#P}$ complexity classes

- A (counting) problem is in #P if there is a PTIME non-deterministic Turing machine whose number of accepting paths, given as input the input of the problem, is the output of the problem.
- A problem is #P-hard if any #P problem can be PTIME-reduced to it (via a Turing reduction). #2DNF, the problem of counting the number of assignments satisfying a formula in 2-DNF, is #P-complete.

- A (computation) problem is in $FP^{\#P}$ if it is computable by a PTIME Turing machine with access to a $\#P$ oracle.
- A problem is $FP^{\#P}$-hard if any $FP^{\#P}$ problem can be PTIME-reduced to it (via a Turing reduction). Equivalently, a computation problem is $FP^{\#P}$-hard if it is #P-hard.

Complexity of query evaluation

	Local dependencies	Arbitrary dependencies
SP	PTIME	$FP^{\#P}$-complete [Kimelfeld et al., 2008]
TP	PTIME [Kimelfeld and Sagiv, 2007, Kimelfeld et al., 2008, 2009]	$FP^{\#P}$-complete
TPJ	$FP^{\#P}$-complete [Abiteboul et al., 2010]	$FP^{\#P}$-complete
MSO	PTIME [Cohen et al., 2009]	$FP^{\#P}$-complete

Remark

Project-free queries are tractable with arbitrary dependencies. [Senellart and Abiteboul, 2007]

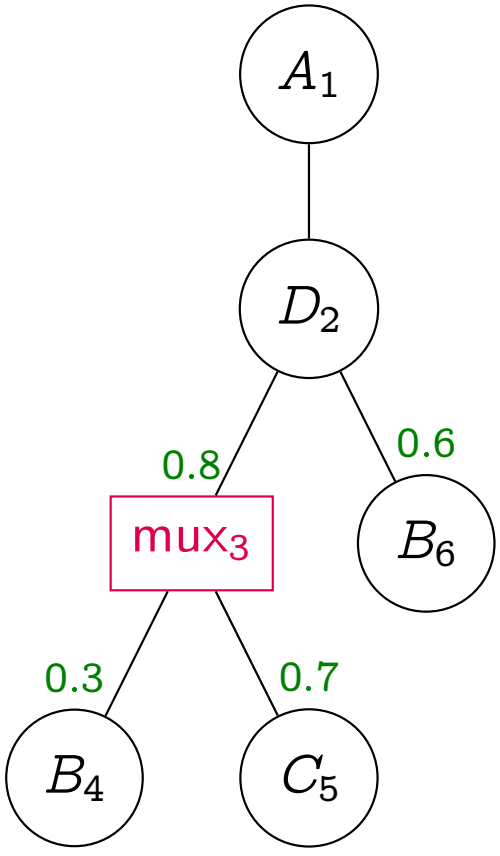
Complexity of query evaluation

	Local dependencies	Arbitrary dependencies
TP	PTIME [Kimelfeld and Sagiv, 2007, Kimelfeld et al., 2008, 2009]	
TPJ	$FP^{\#P}$-complete [Abiteboul et al., 2010]	

TP PTIME for local dependencies

[Kimelfeld and Sagiv, 2007]

Bottom-up dynamic programming algorithm. Query: /A//B



	A_1	D_2	mux_3	B_4	C_5	B_6
/B				1	0	1
//B				1	0	1
/A//B				0	0	0

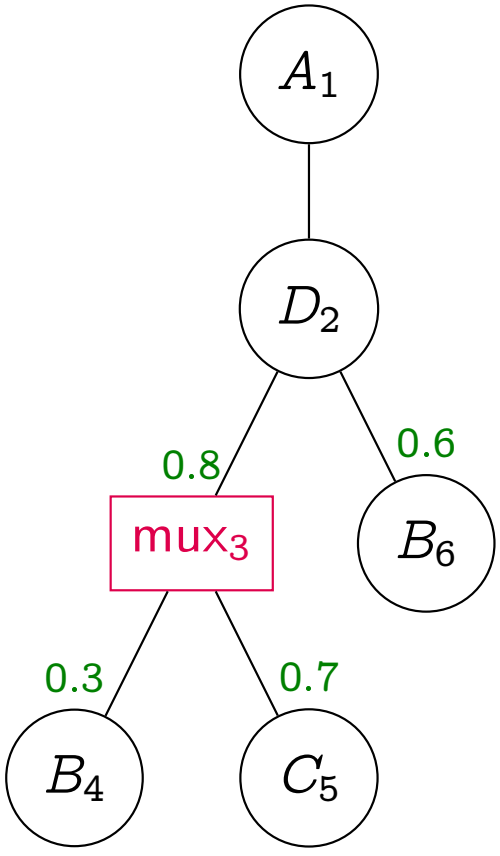
mux convex sum

ordinary inclusion-exclusion

TP PTIME for local dependencies

[Kimelfeld and Sagiv, 2007]

Bottom-up dynamic programming algorithm. Query: /A//B



	A_1	D_2	mux_3	B_4	C_5	B_6
/B			0.3	1	0	1
//B			0.3	1	0	1
/A//B			0	0	0	0

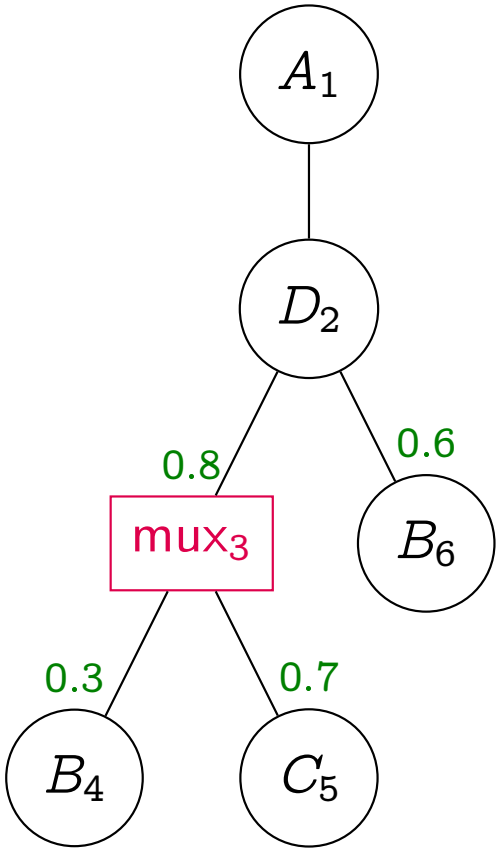
mux convex sum

ordinary inclusion-exclusion

TP PTIME for local dependencies

[Kimelfeld and Sagiv, 2007]

Bottom-up dynamic programming algorithm. Query: /A//B



	A_1	D_2	mux_3	B_4	C_5	B_6
/B		0	0.3	1	0	1
//B		0.696	0.3	1	0	1
/A//B		0	0	0	0	0

mux convex sum

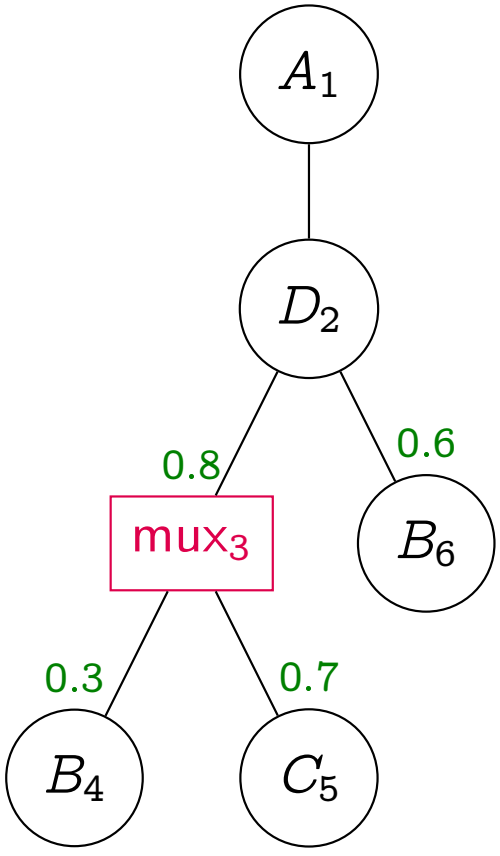
ordinary inclusion-exclusion

$$\Pr(D_2 \models //B) = 1 - (1 - 0.8 \times \Pr(\text{mux}_3 \models //B)) \times (1 - 0.6 \times \Pr(B_6 \models //B))$$
$$= 1 - (1 - 0.8 \times 0.3) \times (1 - 0.6) = 0.696$$

TP PTIME for local dependencies

[Kimelfeld and Sagiv, 2007]

Bottom-up dynamic programming algorithm. Query: /A//B



	A_1	D_2	mux_3	B_4	C_5	B_6
/B	0	0	0.3	1	0	1
//B	0.696	0.696	0.3	1	0	1
/A//B	0.696	0	0	0	0	0

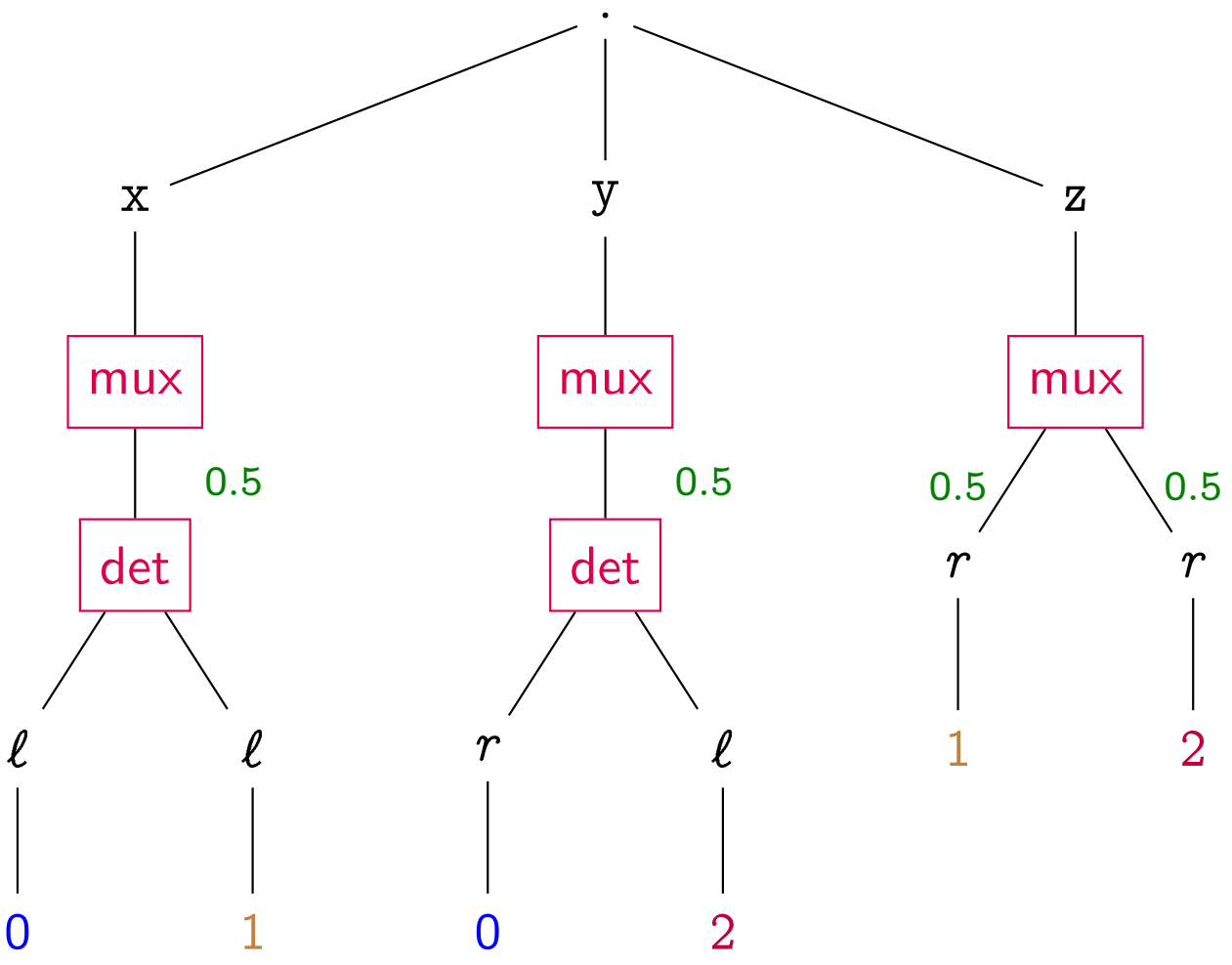
mux convex sum

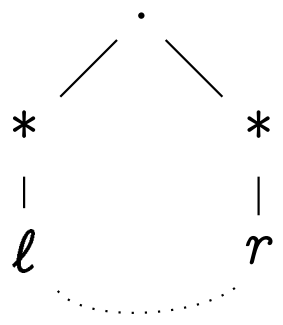
ordinary inclusion-exclusion

TPJ $FP^{\#P}$-hard for local dependencies

[Abiteboul et al., 2010]

Reduction from #2DNF. Example: $\varphi = xy \vee x\neg z \vee yz$.





TELECOM ParisTech

Approximating the probability of a query

In some sense, all variants of probabilistic XML are tractable: [Kimelfeld et al., 2009]

- Additive approximation: simple Monte Carlo sampling
- . . . but additive approximation is not good enough for low probabilities
- (Slower) multiplicative approximation for TP queries: minor rewriting then biased Monte Carlo
- In practice, possible to automatically choose an appropriate approximation method [Senellart and Souihli, 2010].

Aggregate Queries [Cohen et al., 2008, Abiteboul et al., 2010]

Aggregate Queries: sum, count, avg, countd, min, max, etc.
Distributions? Possible values? Expected value?

- Computing HAVING queries is tractable for count, max, min, ratio.
- Computing expected values of sum and count tractable with arbitrary dependencies. Everything else intractable.
- Computing expected values of every of these aggregate functions is tractable with local dependencies.
- Computing distributions and possible values is tractable for count, min, max, intractable for the others.

Continuous distributions do not add any more complexity!

What about updates? [Senellart and Abiteboul, 2007, Abiteboul et al., 2009, Kharlamov et al., 2010]

Updates defined by a query (cf. XUpdate, XQuery Update).
Semantics: for all matches of a query, insert or delete a node in the tree at a place located by the query.

Results

- Most updates are intractable with local dependencies: the result of an update can require an exponentially larger representation size
- Insertions with a for-all-match semantics are tractable with arbitrary dependencies; deletions are intractable.
- Some insert-if-there-is-a-match operations tractable for local dependencies but not for arbitrary dependencies.

Outline

Tractable extensions of the local dependency model

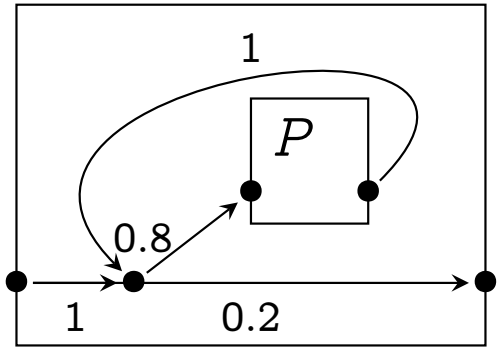
- Arbitrary dependencies: not tractable
- Local dependencies: not practical
- Somewhere in between?
  - What makes the arbitrary dependency model hard?
  - How can the local dependency model be generalized, while remaining tractable?
- And can we go further? cf. XML schemas
  - Trees of unbounded depth
  - Trees of unbounded width
  - Infinite trees?

Prob. DTDs via rec. Markov chains

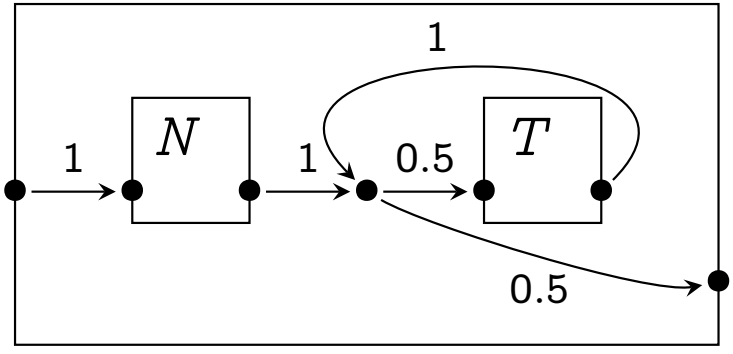
[Benedikt et al., 2010]

```
<!ELEMENT directory (person*)>
<!ELEMENT person (name,phone*)>
```

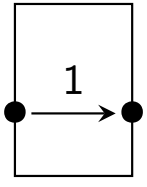
D: directory



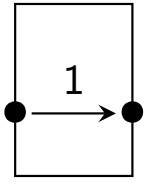
P: person



N: name



T: phone



Such simple tree-like RMCs generalize probabilistic XML, and MSO queries are tractable!

Outline

Link with probabilistic relational models

Relational case

(Block-independent disjoint model, [Dalvi and Suciu, 2007])

- Some conjunctive queries are PTIME
- Others are #P-hard
- Complex conditions to separate the two

XML case (Local dependencies)

- Tree pattern queries are PTIME
- Tree pattern queries with (non-trivial) joins are #P-hard

- Why does the XML case seem simpler?
- Is there some insight to be gained from one case to the other?
- Translating XML data and queries to the relational case yields queries with self-joins, a less well-understood setting

Link with probabilistic relational models

Relational case

(Block-independent disjoint model, [Dalvi and Suciu, 2007])

- Some conjunctive queries are PTIME
- Others are #P-hard
- Complex conditions to separate the two

XML case (Local dependencies)

- Tree pattern queries are PTIME
- Tree pattern queries with (non-trivial) joins are #P-hard

- Why does the XML case seem simpler?
- Is there some insight to be gained from one case to the other?
- Translating XML data and queries to the relational case yields queries with self-joins, a less well-understood setting

But where do probabilities come from?!

- Do the numbers assigned as probabilities in PDBMS really make sense?
- In some cases, sources of "good" probabilities:
  - Statistics
  - Conditional Random Fields [Lafferty et al., 2001]
  - Parse Trees of Stochastic Context-Free Grammars [Cohen and Kimelfeld, 2010]
  - Uncertain Schema Matching [Dong et al., 2009, Fagin et al., 2010]
  - Representing a Corpus with a Probabilistic Documents?
- What about the rest? Does it really make sense to model uncertainty with probabilities?

A system that just works

- Nothing else than toy systems exist for probabilistic XML
- What should it be based upon:
  - a probabilistic relational DBMS? [Hollander and van Keulen, 2010]
  - a native XML DBMS? [Senellart and Souihli, 2010]
- Systems issue: distribution, indexing, etc.
- And need for a killer application!
  - Probabilistic content warehouse?
  - Parse trees of natural language sentences?
  - Concise representation of a large corpus of XML documents?

Merci.

DataRing Project: P2P Data Sharing for Online Communities

Serge Abiteboul, Benny Kimelfeld, Yehoshua Sagiv, and Pierre Senellart. On the expressiveness of probabilistic XML models. *VLDB Journal*, 18(5):1041–1064, October 2009.

Serge Abiteboul, T-H. Hubert Chan, Evgeny Kharlamov, Werner Nutt, and Pierre Senellart. Aggregate queries for discrete and continuous probabilistic xml. In *Proc. ICDT*, Lausanne, Switzerland, March 2010.

Michael Benedikt, Evgeny Kharlamov, Dan Olteanu, and Pierre Senellart. Probabilistic XML via Markov chains. *Proceedings of the VLDB Endowment*, 3(1):770–781, September 2010. Presented at the VLDB 2010 conference, Singapore.

Sara Cohen and Benny Kimelfeld. Querying parse trees of stochastic context-free grammars. In *Proc. ICDT*, Lausanne, Switzerland, March 2010.

Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv. Incorporating constraints in probabilstic XML. In *Proc. PODS*, Vancouver, BC, Canada, June 2008.

Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv. Running tree automata on probabilistic XML. In *Proc. PODS*, Providence, RI, USA, June 2009.

Nilesh Dalvi, Chrisopher Ré, and Dan Suciu. Probabilistic databases: Diamonds in the dirt. *Communications of the ACM*, 52(7), 2009.

Nilesh N. Dalvi and Dan Suciu. Management of probabilistic data: foundations and challenges. In *Proc. PODS*, Beijing, China, June 2007.

Xin Luna Dong, Alon Y. Halevy, and Cong Yu. Data integration with uncertainty. *VLDB Journal*, 18(2):469–500, 2009.

Ronald Fagin, Benny Kimelfeld, and Phokion Kolaitis. Probabilistic data exchange. In *Proc. ICDT*, Lausanne, Switzerland, March 2010.

J. Nathan Foster, Todd J. Green, and Val Tannen. Annotated XML: queries and provenance. In *Proc. PODS*, Vancouver, BC, Canada, June 2008.

Emiel Hollander and Maurice van Keulen. Storing and querying probabilistic XML using a probabilistic relational dbms. In *Proc. MUD*, Singapore, September 2010.

Evgeny Kharlamov, Werner Nutt, and Pierre Senellart. Updating probabilistic XML. In *Proc. Updates in XML*, Lausanne, Switzerland, March 2010.

B. Kimelfeld and Y. Sagiv. Matching twigs in probabilistic XML. In *Proc. VLDB*, Vienna, Austria, September 2007.

Benny Kimelfeld, Yuri Kosharovsky, and Yehoshua Sagiv. Query efficiency in probabilistic XML models. In *Proc. SIGMOD*, Vancouver, BC, Canada, June 2008.

Benny Kimelfeld, Yuri Kosharovsky, and Yehoshua Sagiv. Query evaluation over probabilistic XML. *VLDB Journal*, 18(5): 1117–1140, October 2009.

Christoph Koch. MayBMS: A system for managing large uncertain and probabilistic databases. In Charu Aggarwal, editor, *Managing and Mining Uncertain Data*. Springer-Verlag, 2009.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, Williamstown, NJ, USA, June 2001.

Pierre Senellart and Serge Abiteboul. On the complexity of managing probabilistic XML data. In *Proc. PODS*, Beijing, China, June 2007.

Pierre Senellart and Asma Souihli. Un système de gestion de données XML probabilistes. In *Proc. BDA*, Toulouse, France, October 2010. Conference without formal proceedings. (Demonstration).

Jennifer Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *Proc. CIDR*, Asilomar, CA, USA, January 2005.