



Querying Probabilistic XML Databases

Sept. 21st 2012

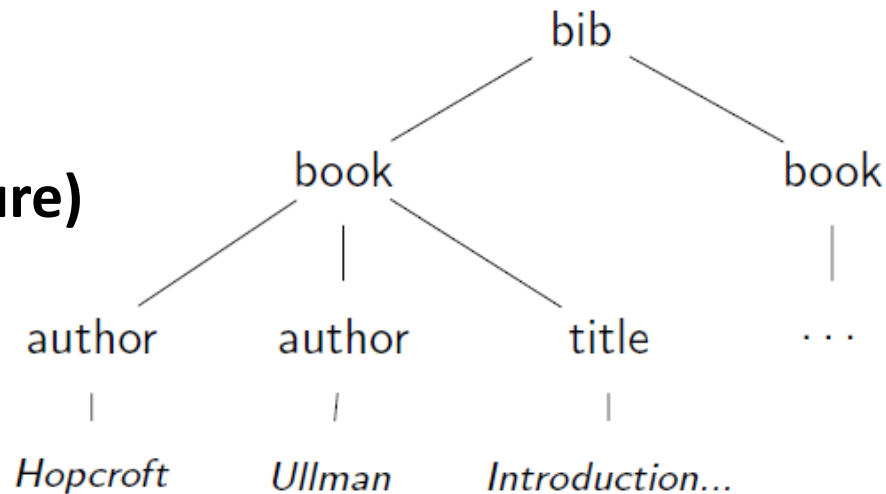
Asma Souihli

Network and Computer Science
Department



XML

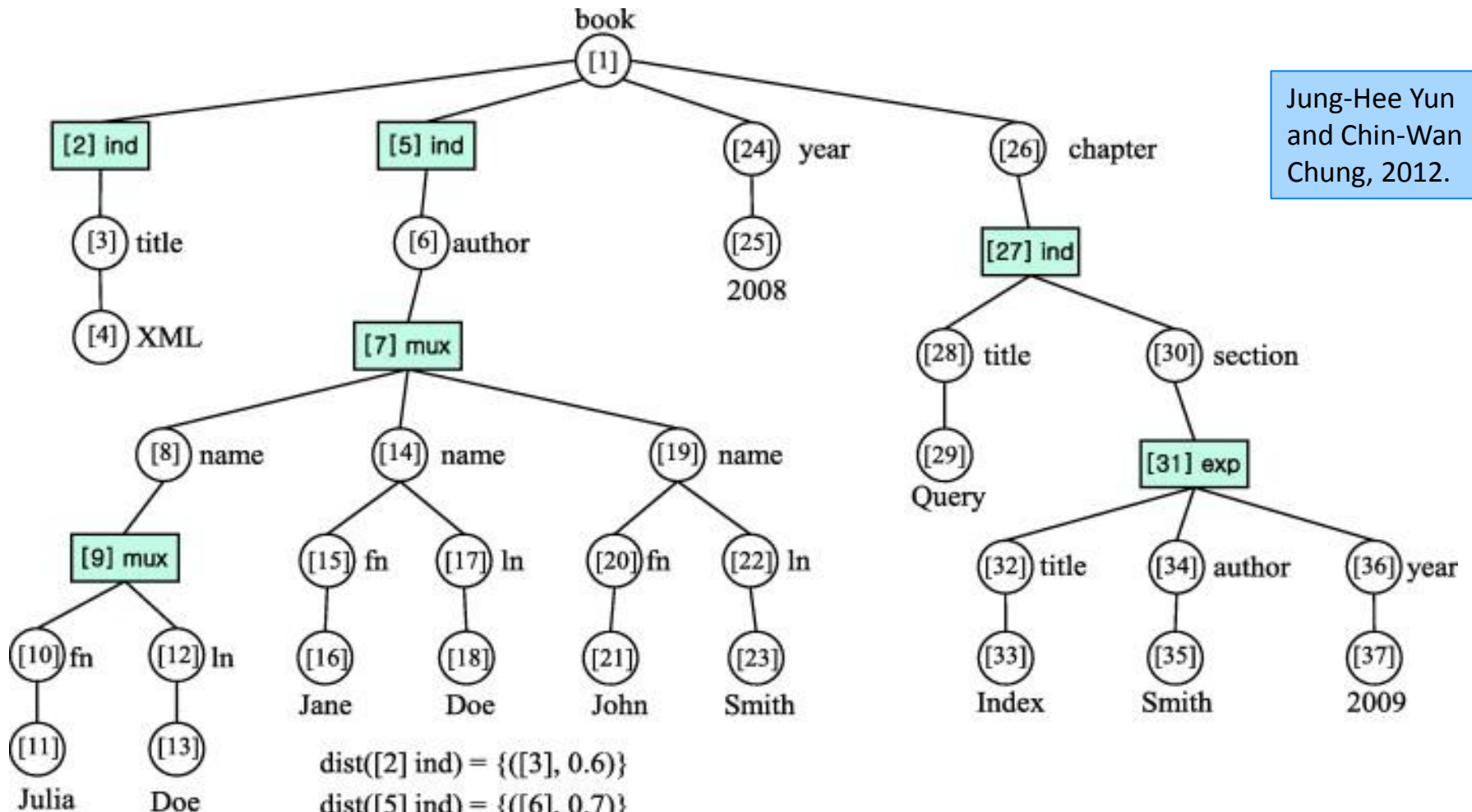
for semi-structured
data (*tree-like structure*)



```
<bib> <book year="1994">
  <title> TCP/IP Illustrated </title>
  <author> <last> Stevens </last> <first> W. </first> </author>
  <publisher> Addison-Wesley </publisher>
  <price> 65.95 </price>
</book>
... <book year="2000">
  <title> Data on the Web </title>
  <author> <last> Abiteboul</last> <first> Serge </first> </author>
  <author> <last> Buneman </last> <first> Peter </first> </author>
  <author> <last> Suciu </last> <first> Dan </first> </author>
  <publisher> Morgan Kaufmann Publishers </publisher>
  <price> 39.95 </price>
</book> ... </bib>
```

Probabilistic Data - PrXML

Jung-Hee Yun
and Chin-Wan
Chung, 2012.



Context

Uncertainty



Context

- In many of these tasks, information is described in a semi-structured manner
- Especially when the source (e.g., XML or HTML) is already in this form
- Representation by means of a hierarchy of nodes is natural



Outline

1. PrXML Models

Local Dependency

Long-distance Dependency

2. Querying P-documents

Types of Queries

Probabilistic Lineage

Complexity of Queries

3. The ProApproX System

Computation Algorithms

Lineage Decomposition Techniques

Evaluation Plans

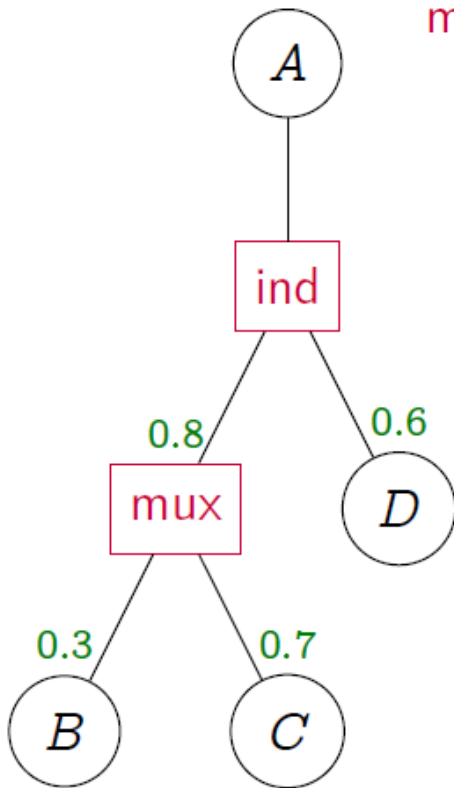
Experiments

4. Conclusions

PrXML Models – Local Dependency

ind children of the node are chosen **independently** of one another, according to their probabilities

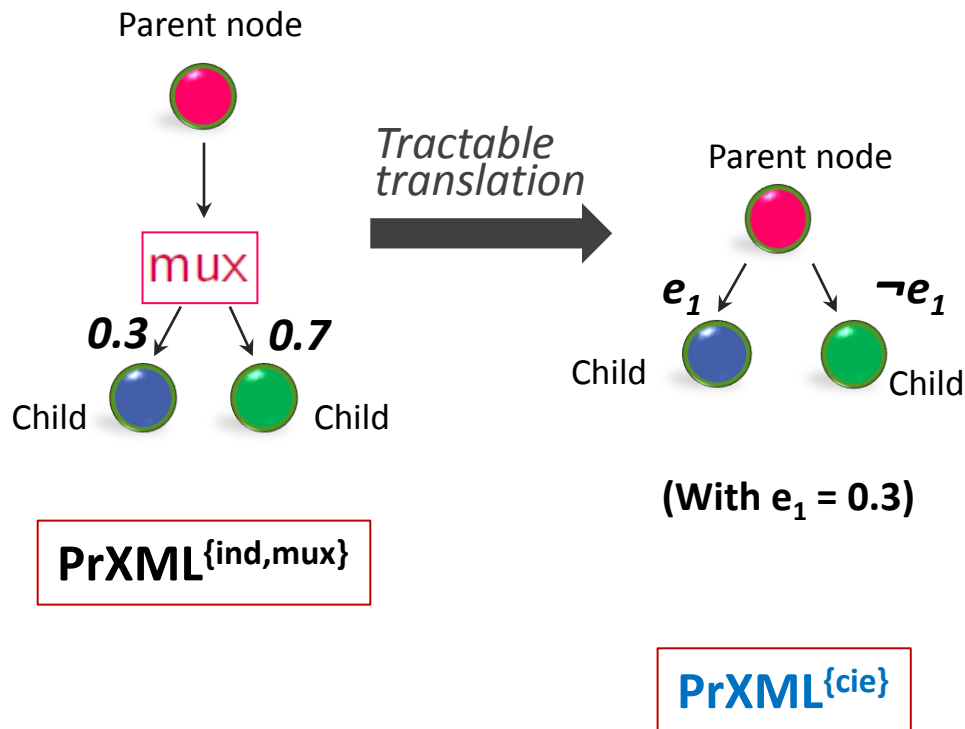
mux children of the node are chosen in a **mutually exclusive** way, depending of their probabilities, that must sum up to 1 or less



➡ Local dependency
(**mux** and **ind** nodes)

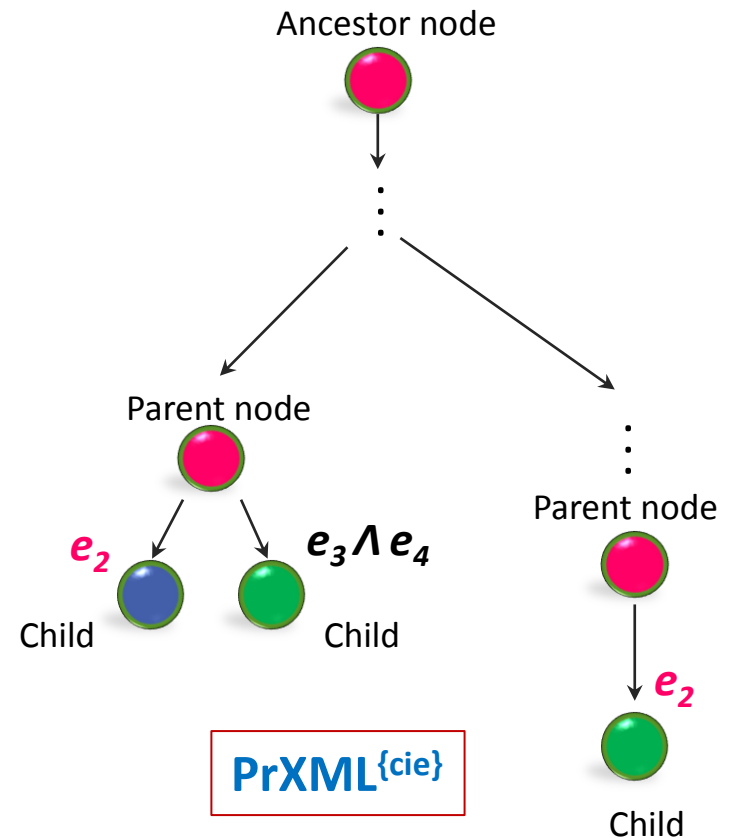
PrXML Models – Long-distance Dependency

Local dependency (*mux* and *ind* nodes)

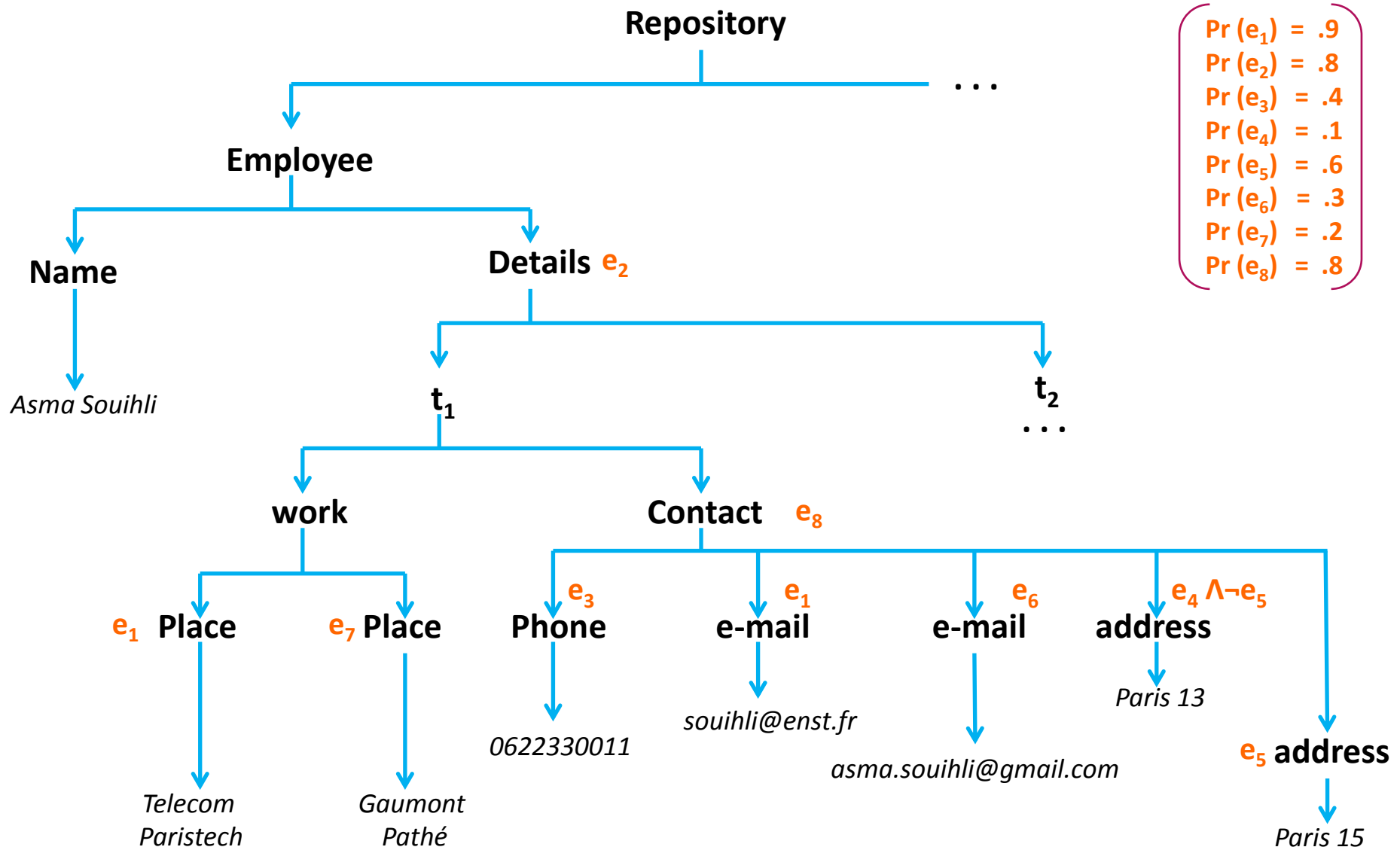


S. Abiteboul, B. Kimelfeld, Y. Sagiv,
and P. Senellart. 2009

Long-distance dependency (Conjunction of independent events \rightarrow *cie*)



Example



Outline

1. PrXML Models

Local Dependency

Long-distance Dependency

2. Querying P-documents

Types of Queries

Probabilistic Lineage

Complexity of Queries

3. The ProApproX System

Computation Algorithms

Lineage Decomposition Techniques

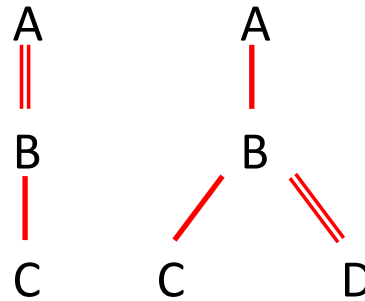
Evaluation Plans

Experiments

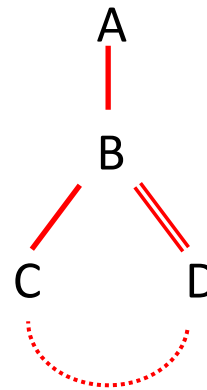
4. Conclusions

Querying P-documents – Types of Queries

- Tree Pattern Queries (TPQ)



- Tree Pattern Queries with joins (TPQJ)



Example

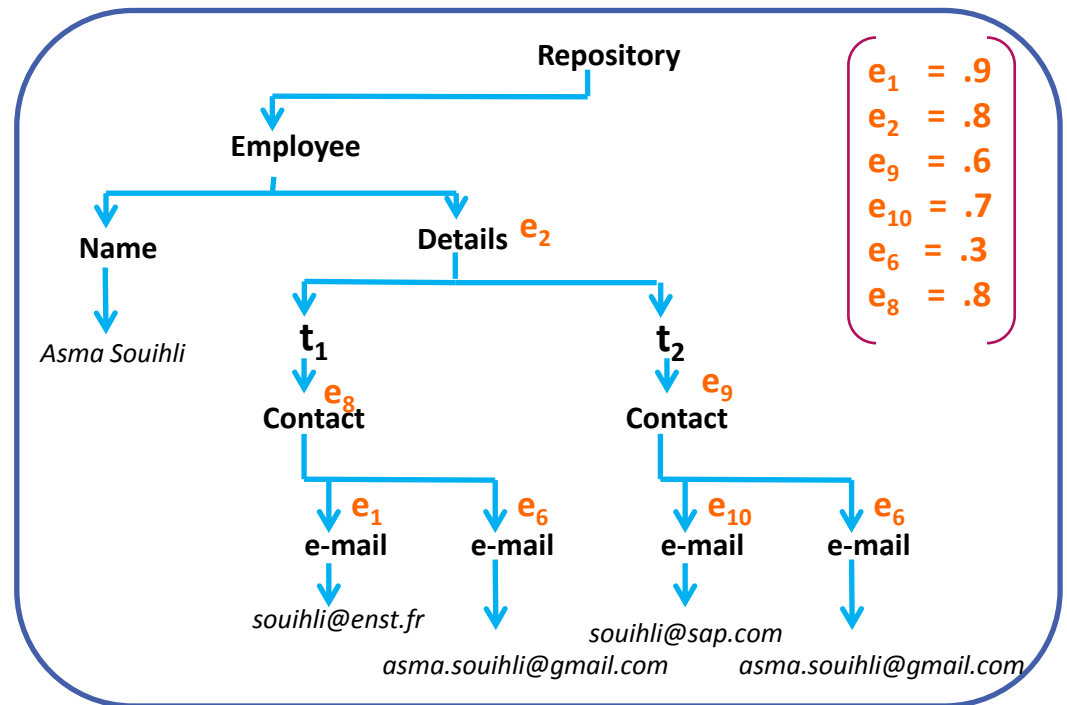
- **Q1:** / Employee [Name= "Asma Souihli"] // e-mail / text()

enst.fr: $e_2 \wedge e_8 \wedge e_1$ **C1**

gmail.com: $e_2 \wedge e_8 \wedge e_6$ **C2**

sap.com: $e_2 \wedge e_9 \wedge e_{10}$ **C3**

gmail.com : $e_2 \wedge e_9 \wedge e_6$ **C4**



Querying PrXML – Probabilistic Lineage

- Probability to find an e-mail:

$$\Pr(Q1) = \Pr(\mathbf{C1} \vee \mathbf{C2} \vee \mathbf{C3} \vee \mathbf{C4}) \longrightarrow \text{Probabilistic lineage (DNF shape)}$$

- Possible results:

$$\Pr(\textit{asma.souihli@gmail.com}) = \Pr(\mathbf{C2} \vee \mathbf{C4})$$

$$\Pr(\textit{souihli@enst.fr}) = \Pr(\mathbf{C1})$$

$$\Pr(\textit{souihli@sap.com}) = \Pr(\mathbf{C3})$$

Querying PrXML – Probabilistic Lineage

- When is a linear computation possible?

- if C_1 and C_2 are independent, then:

$$\Pr(\mathbf{C}_1 \wedge \mathbf{C}_2) = \Pr(\mathbf{C}_1) \times \Pr(\mathbf{C}_2) \quad \textcircled{\wedge}$$

$$\Pr(\mathbf{C}_1 \vee \mathbf{C}_2) = 1 - ((1 - \Pr(\mathbf{C}_1)) \times (1 - \Pr(\mathbf{C}_2))) \quad \textcircled{\vee}$$

- if C_1 and C_2 are inconsistent (mutually exclusive), then:

$$\Pr(\mathbf{C}_1 \vee \mathbf{C}_2) = \Pr(\mathbf{C}_1) + \Pr(\mathbf{C}_2) \quad \textcircled{+}$$

Back to the Example

e_1	=	.9
e_2	=	.8
e_9	=	.6
e_{10}	=	.7
e_6	=	.3
e_8	=	.8

$$\Pr(@\text{enst.fr}) = \Pr(\mathbf{C1}) = \Pr(e_2 \wedge e_8 \wedge e_1) = .8 \times .8 \times .9 \\ = 0.576$$

$$\Pr(@\text{sap.com}) = \Pr(\mathbf{C3}) = 0.336$$

$$\Pr(@\text{gmail.com}) = \Pr(\mathbf{C2} \vee \mathbf{C4}) = (e_2 \wedge e_8 \wedge e_6) \vee (e_2 \wedge e_9 \wedge e_6)$$

→ Factorization:

$$\Pr(@\text{gmail.com}) = (e_2 \wedge e_6) \wedge (e_8 \vee e_9) = .8 \times .3 \times (1 - (1-.8)(1-.6)) \\ = 0.2208$$

Querying PrXML – Probabilistic Lineage

e_1	=	.9
e_2	=	.8
e_9	=	.6
e_{10}	=	.7
e_6	=	.3
e_8	=	.8

$$\begin{aligned}\Pr(Q1) &= \Pr(\mathbf{C1} \vee \mathbf{C2} \vee \mathbf{C3} \vee \mathbf{C4}) \\ &= \Pr [(e_2 \wedge e_8 \wedge e_1) \vee (e_2 \wedge e_8 \wedge e_6) \vee (e_2 \wedge e_9 \wedge e_{10}) \vee (e_2 \wedge e_9 \wedge e_6)]\end{aligned}$$

→ Factorization:

$$= \Pr [e_2 \wedge ((e_8 \wedge (e_1 \vee e_6)) \vee (e_9 \wedge (e_{10} \vee e_6)))]$$

→ Difficult to evaluate !

Solutions..

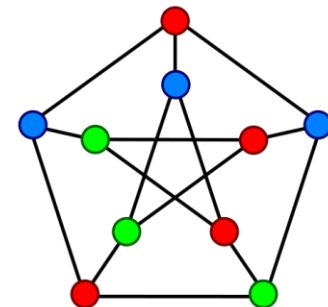
- One possible (naïve) way, is to find the truth value assignments that satisfy the propositional formula (probabilistic lineage)
(out of $2^{\text{\#literals}}$ possible assignments/worlds !)
- And sum the probabilities of these satisfying assignments to get the answer

e_1	e_2	e_6	e_8	e_9	e_{10}	Probability	$C1 \vee C2 \vee C3 \vee C4$
false	false	false	false	false	false	0.0845	false
false	false	false	false	false	true	0.3345	false
false	false	false	false	true	false	0.87	false
...

Querying PrXML – Complexity of Queries

- Probabilities of the satisfying assignments for the DNF (lineage formula) : **#P-Hard** problem
 - No polynomial time algorithm for the exact solution if $P \neq NP$
 - **#P** problems ask "how many" rather than "are there any"

How many graph coloring using k colors are there for a particular graph G ?



Querying PrXML – Complexity of Queries

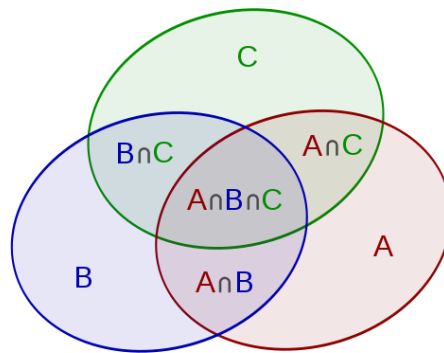
- A union of sets (clauses) problem: **#P-Hard** problem

$$\Pr(C_1) = \Pr(e_1) \cdot \Pr(e_2)$$

$$\Pr(C_2) = \Pr(e_2) \cdot \Pr(e_3)$$

$$\Pr(C_1 \cap C_2) = \Pr(e_1) \cdot \Pr(e_2) \cdot \Pr(e_3)$$

$$\Pr(C_1 \cup C_2) = \Pr(C_1) + \Pr(C_2) - \Pr(C_1 \cap C_2)$$



For dependent probabilistic clauses $C_1 \dots C_n$
the inclusion-exclusion principle becomes:

$$\Pr\left(\bigcup_{i=1}^n C_i\right) = \sum_{k=1}^n (-1)^{k-1} \sum_{\substack{J \subseteq \{1, \dots, n\} \\ |J|=k}} \Pr(C_J)$$

where:

$$C_J = \bigcap_{j \in J} C_j$$

Outline

1. PrXML Models

Local Dependency

Long-distance Dependency

2. Querying P-documents

Types of Queries

Probabilistic Lineage

Complexity of Queries

3. The ProApproX System

Computation Algorithms

Lineage Decomposition Techniques

Evaluation Plans

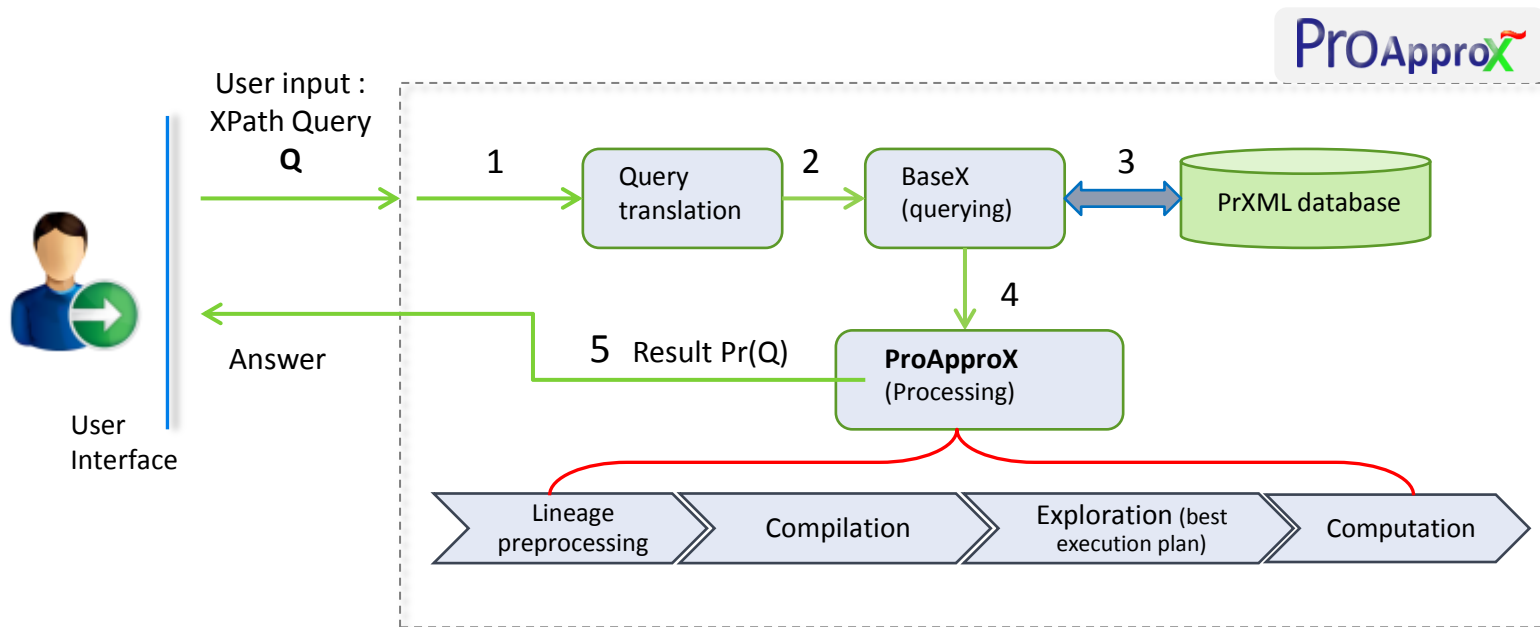
Experiments

4. Conclusions

The ProApproX System

[CIKM 2012, SIGMOD 2011]

- Translates into a probabilistic database with only *cie* nodes
- Translates the user query into a lineage query



Back to the Example

Q1: / Employee [Name= "Asma Souihli"] // e-mail / text()

- To get the lineage for the boolean projection :


```
for $x1 in /employee
for $x2 in $x1/name[.="Asma Souihli"]
for $x3 in $x1//email/text()
let $leaves:=(($x2,$x3))
let $atts:=(for $i in $leaves return $i/ancestor-or-self::* /attribute(event))
return text{distinct-values(for $att in $atts return string($att))}
```

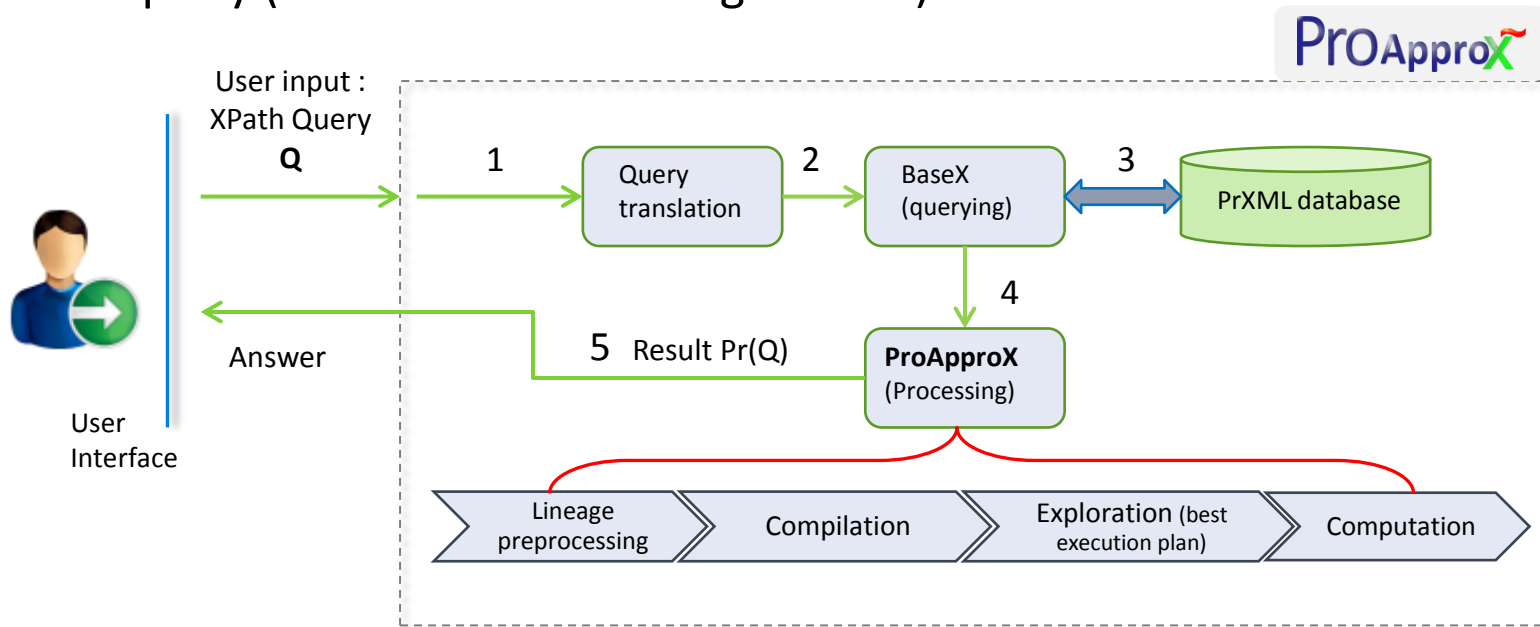
- To get lineages of answers:

```
for $val in distinct-values(/employee [name="Asma Souihli"]//email/text())
order by $val
return <match>
{$val}{
for $x1 in /employee
for $x2 in $x1/name[.="Asma Souihli "]
for $x3 in $x1//email/text()
let $leaves:=(($x2,$x3))
let $atts:=(for $i in $leaves return $i/ancestor-or-self::* /attribute(event))
where $x3=$val
return <clause>{distinct-values(for $att in $atts return string($att))}</clause>
}</match>
```

The ProApproX System

[CIKM 2012, SIGMOD 2011]

- Translates into a probabilistic database with only *cie* nodes
- Translates the user query into a lineage query
- Is built on top of a native XML DBMS 
- Processes the lineage formula to get the probability of the query (and of each matching answer)



The ProApprox System – Computation Algorithms

■ Additive approximation:

- For a fixed error ε and a DNF F , $A(F)$ is an additive ε -approximation of $\Pr(F)$ with a probability of at least δ (a fixed reliability factor) if:

$$\Pr(F) - \varepsilon \leq A(F) \leq \Pr(F) + \varepsilon$$

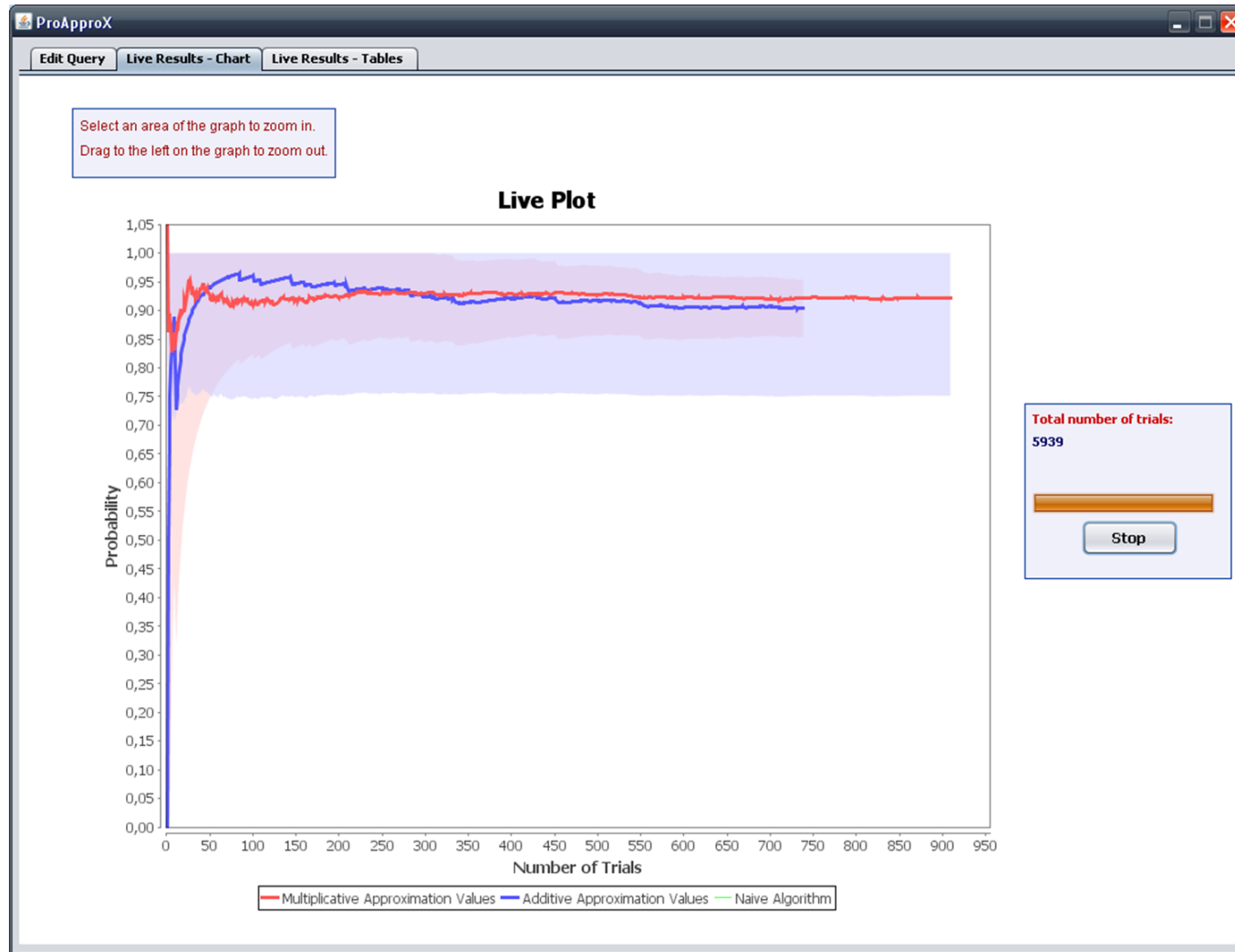
■ Multiplicative Approximation

- For a fixed error ε , a DNF F , $A(F)$ is an multiplicative ε -approximation of $\Pr(F)$ with a probability of at least δ if:

$$(1 - \varepsilon) \Pr(F) \leq A(F) \leq (1 + \varepsilon) \Pr(F)$$

DEMO 1

[SIGMOD 2011]



The ProApproX System – Computation Algorithms

■ Exact Computations:

- The naïve algorithm – Possible worlds

Finding the satisfying assignments out of $2^{\text{\#variables}}$ possible truth value assignments

$$O(2^n)$$

- The sieve algorithm – The inclusion-exclusion principle

Exponential in the number of clauses m

$$O(2^m)$$

The ProApprox System – Computation Algorithms

■ Approximations:

- Naïve Monte Carlo sampling for additive app. :

Linear but could take exponentially many samples
to converge to a good approximation for low probabilities

- Biased Monte Carlo sampling for multiplicative app. :

Running time grows in $O(n^3 \ln n)$
in the number of clauses

Kimelfeld,
Kosharovsky,
and Sagiv.
2009

- Self-Adjusting Coverage Algorithm for the DNF probability problem: M. Karp, M. Luby, and N. Madras. 1989

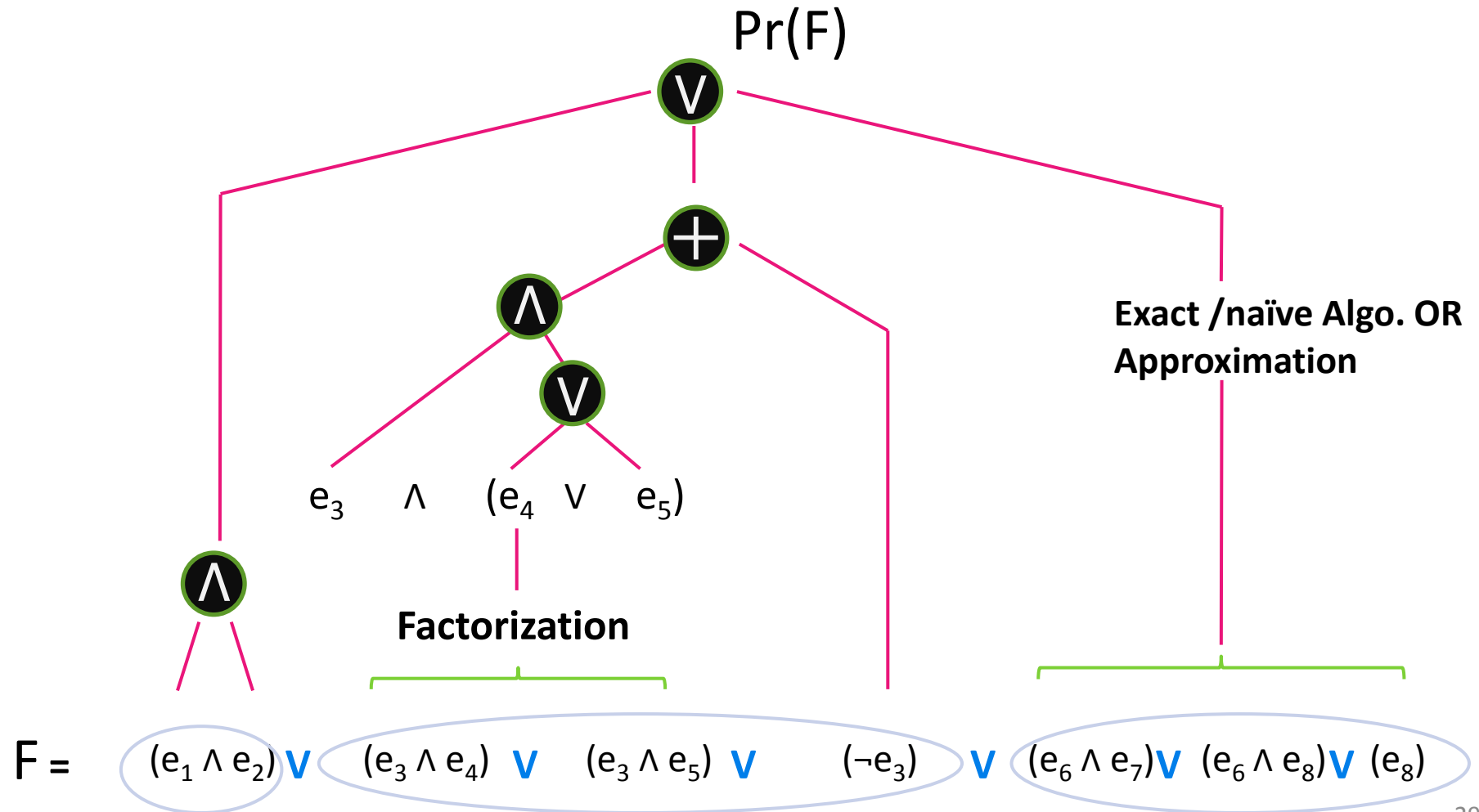
Linear in the length of F times $\ln(1/\delta) / \epsilon^2$

The ProApproX System – Computation Algorithms

- Possibility to derive a multiplicative approximation from an additive approximation (*and vice versa*)
- Cost models and cost constants:

Algorithm alg	cost_{alg}	C_{alg} (ms)
naïve	$C_{\text{naïve}} \times 2^N \times L$	$4 \cdot 10^{-5}$
sieve	$C_{\text{sieve}} \times 2^m \times \frac{L}{m}$	$5 \cdot 10^{-5}$
AddMC	$C_{\text{AddMC}} \times \ln \frac{2}{\delta} \times \frac{L}{\varepsilon^2}$	$4 \cdot 10^{-5}$
MulMC	$C_{\text{AddMC}} \times \ln \frac{2}{\delta} \times \frac{L}{\ell^2 \varepsilon^2}$	$4 \cdot 10^{-5}$
coverage	$C_{\text{coverage}} \times \ln \frac{2}{\delta} \times \frac{(1+\varepsilon) \times L}{\varepsilon^2}$	10^{-3}

The ProApproX System – Lineage Decomposition Techniques



DEMO 2

[CIKM 2012]

ProApproX2.0.

Query Editor

```
//movie[actors/actor/name="Brooke Smith"]/title
```

Approximation Settings

Error: 0.5 Reliability factor: 0.95 Precision: Multiplicative

Run

Results

Query Info

Timing:	Probability:
Total Time: 67.6208009 ms	Probability: 0.1681246173
Timing:	number of items: 13
XPath Parsing: 0.0268749 ms	
Query Translations: 0.0045257 ms	
Lineage Extraction (XQuery Time): 25.9647246 ms	
Optimization: 0.2822425 ms	

Items extraction:

for \$val in distinct-values(//movie[actors/actor/name="Brooke Smith"]/title) order by \$val return <match>{\$val}{ for \$x1 in //movie for \$x2 in \$x1/actors/actor/name[.="Brooke Smith"] for \$x3 in \$x1/title let \$leaves:=((\$x2,\$x3) let \$atts:=(for \$i in \$leaves return \$i/ancestor-or-self::*/@attribute(event)) where \$x3=\$val return <clause>{distinct-values(for \$att in \$atts return string(\$att))}) </clause> } </match>

Items

Trials	Probability
The Namesake	0.0915268913
Passage, The	0.0235956359
Caretaker, The	0.0188765087
Paleface, The	0.0094382543
Nemesis, The	0.0047191272
Homemaker, The	0.0047191272
Kot v meshke	0.0047191272
Yardsale, The	0.0047191272
She Hate Me	0.0042696605
Games, The	0.003735926

Evaluation Tree

Node Info

w1500)) V ([notw1502, notw1513, notw1503, notw1510, notw1506, notw1517, notw1504, w1512, notw1506, notw1511, notw1507, notw1504, notw1513, notw1505, notw1514, notw1497, notw1498, notw1499, notw1510, w1645, notw1644, notw1509, notw1508]) V ([notw1515, notw1516, notw1517, notw1511, notw1512, notw1513, notw1514, w1564, notw1497, notw1498, notw1499, notw1633, notw1565, notw1502, notw1503, notw1500, notw1501, notw1506, w1510, notw1507, notw1504, notw1505, w1632, notw1509, notw1508]) V ([notw1502, notw1515, notw1503, notw1516, notw1500, notw1517, notw1554, notw1501, notw1506, notw1511, notw1512, notw1504, notw1513, notw1505, notw1514, notw1497, notw1498, notw1499, notw1510, w1555, notw1509, notw1508, w1507])

DNF lineage size: 41 clauses

Algorithm: coverage

The ProApprox System – Evaluation Plans

- Propagation of ϵ (and δ) :

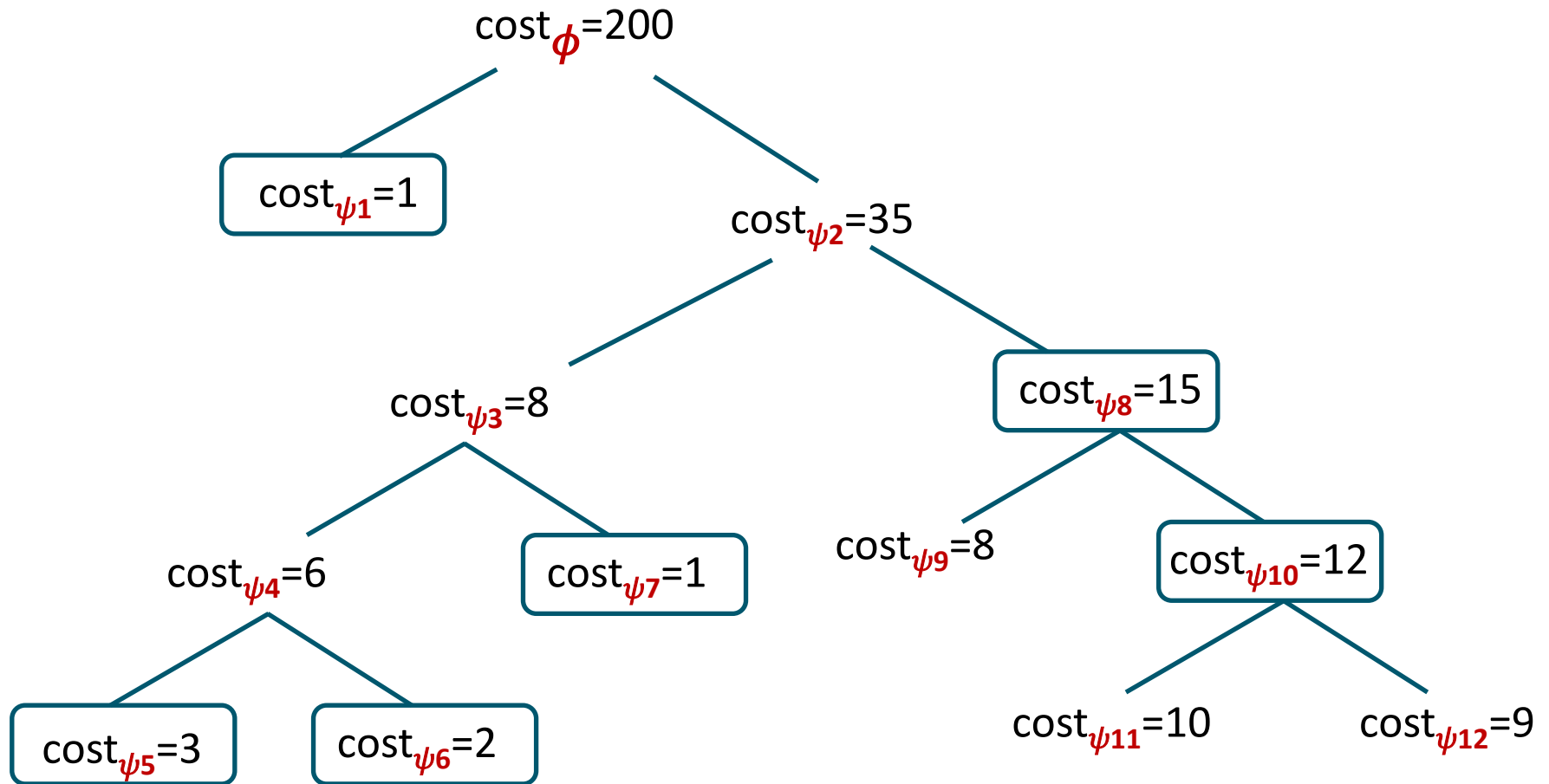
Proposition1. Let $\phi = \psi_1 \vee \psi_2$, and assume \tilde{p}_1 and \tilde{p}_2 are additive approximations of $\Pr(\psi_1)$ and $\Pr(\psi_2)$, to a factor of ϵ_1 and ϵ_2 , respectively. Then $1 - (1 - \tilde{p}_1)(1 - \tilde{p}_2)$ is an additive approximation of $\Pr(\phi)$ to a factor of ϵ if:

$$\epsilon_1 + \epsilon_2 + \epsilon_1 \epsilon_2 \leq \epsilon$$

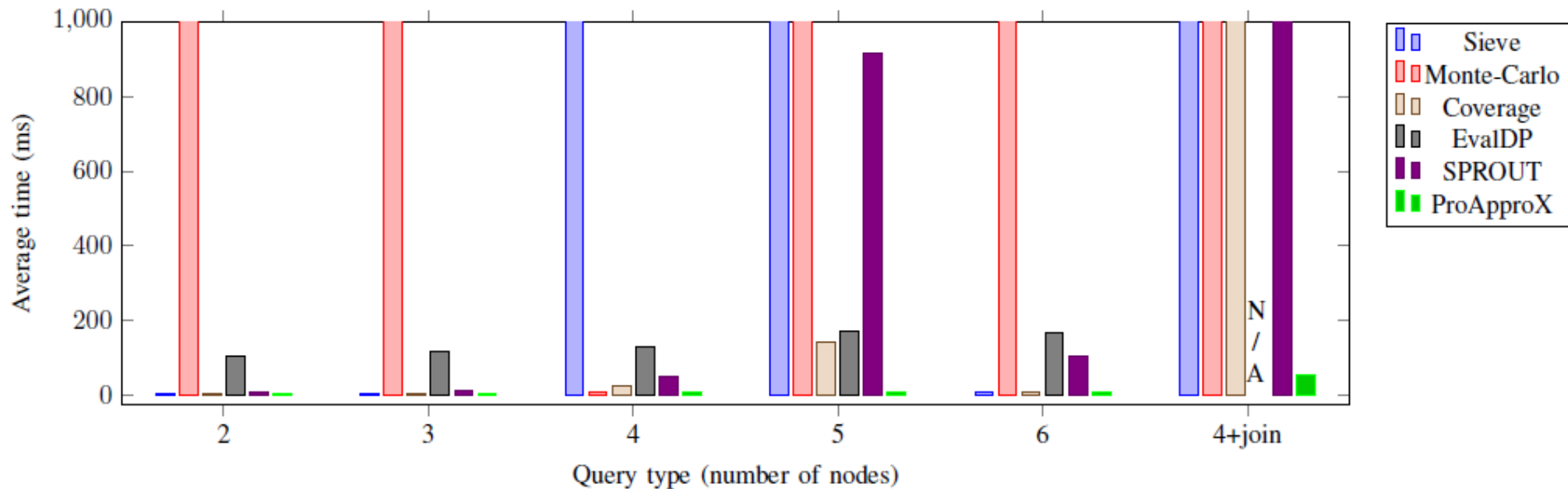
- Many possible values for ϵ_1 and ϵ_2 can be found
- Best assignments are not always obvious

The ProApproX System – Possible Evaluation Plans

Deterministic exploration:



The ProApproX System – Experiments

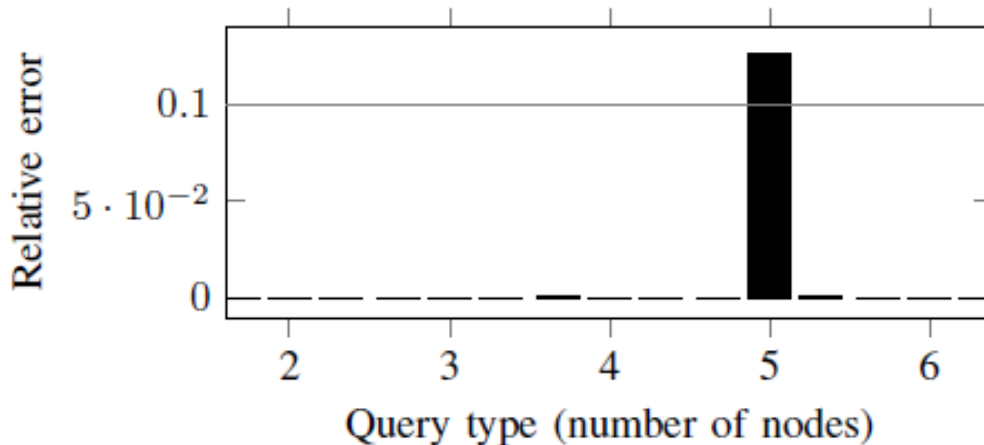


Running time of the different algorithms on the MondialDB dataset

The ProApproX System – Experiments

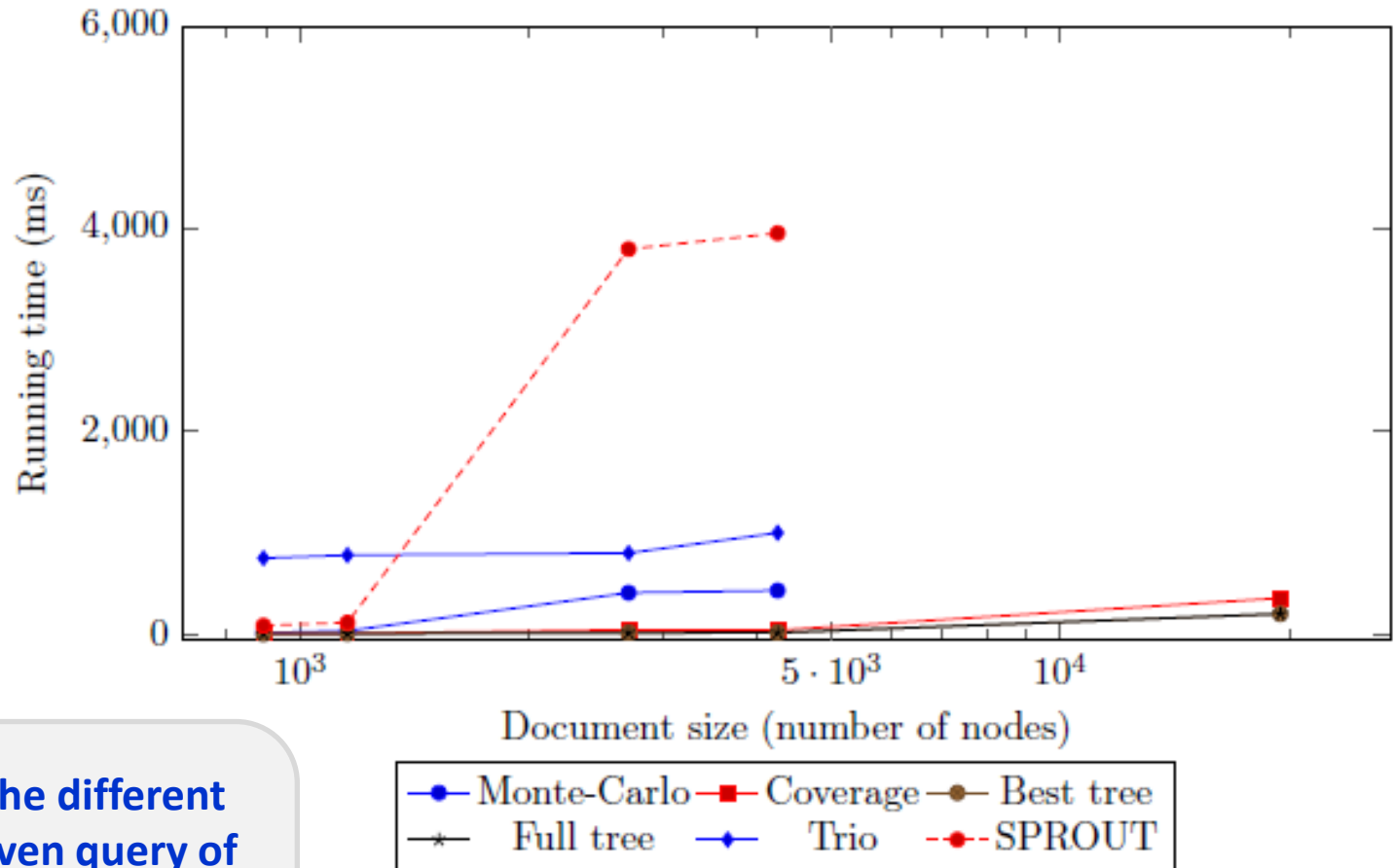
Query type	Avg DNF size	XQuery	Comp	Exp+Eval
2	6	96.82%	2.47%	0.71%
3	5	98.19%	1.38%	0.42%
4	43	97.41%	2.01%	0.58%
5	252	64.47%	33.76%	1.78%
6	6	99.69%	0.29%	0.03%
4+join	3656	61.49%	36.12%	2.39%

**Proportion of time
(MondialDB - Best Tree)**



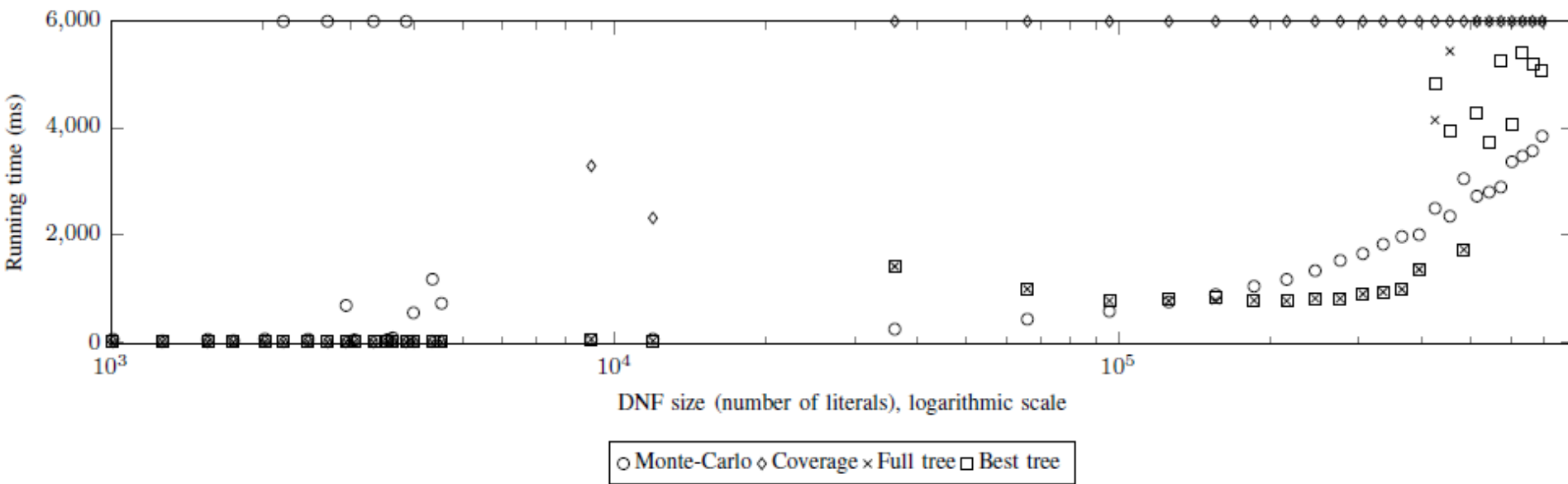
**Relative error on the probabilities
computed by the algorithm on
the MondialDB over each non
join query with respect to the
exact probability values
($\varepsilon = 0.1$, $\delta = 95\%$)**

The ProApproX System – Experiments



Running time of the different algorithms on a given query of the movie dataset.
(times greater than 5s are not shown)

The ProApproX System – Experiments



**Running time of the different algorithms
on the synthetic dataset**

Outline

1. PrXML Models

Local Dependency

Long-distance Dependency

2. Querying P-documents

Types of Queries

Probabilistic Lineage

Complexity of Queries

3. The ProApproX System

Lineage Decomposition Techniques

Computation Algorithms

Demo

Evaluation Plans

Experiments

4. Conclusions

Contributions

- We have introduced an original optimizer-like approach to evaluating query results over probabilistic XML
- Over a more expressive PrXML model
- Positive tree-pattern queries, possibly with joins

[Submitted **ICDE 2013**]

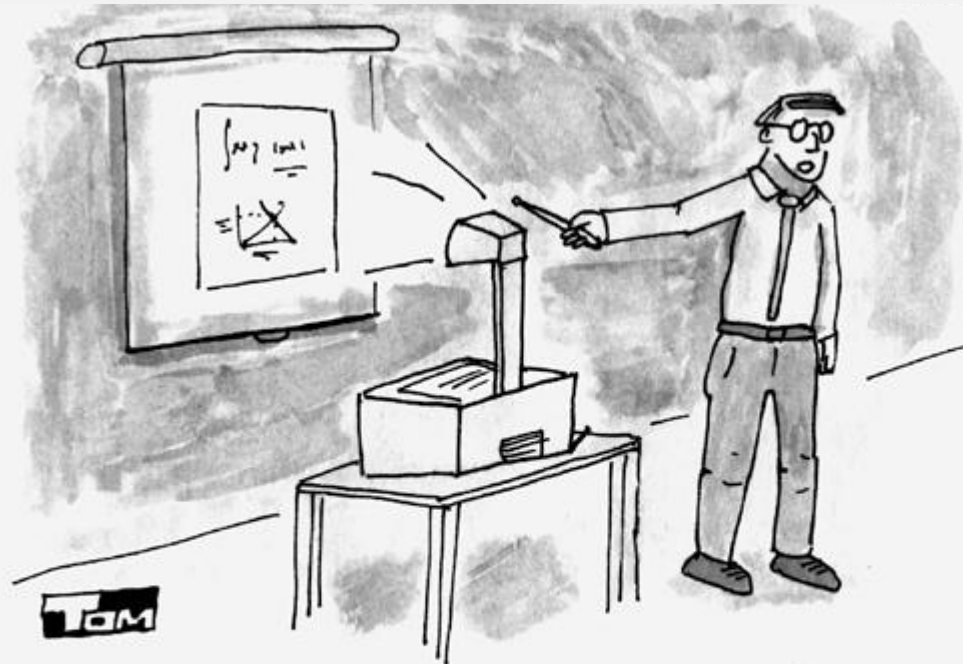
Contributions

- Main observation - optimal probability evaluation algorithm to use depends on the characteristics of the formula:
 - Few variables naïve algorithm
 - Few clauses sieve algorithm
 - Monte-Carlo is very good at approximating high probabilities
 - Sometimes the structure of a query makes the probability of a query easy to evaluate (EvalDP)
 - Refined approximation methods best when everything else fails (coverage)

Perspectives

- Exploiting the structure of the query to obtain factorized lineage
- Most evaluation algorithms scale effortlessly (with the exception of the self-adjusting coverage algorithm, which requires synchronization)
 - distribute the probability computation over multi-core or distributed architectures
- Processing DNFs, but the technique could probably be extended to arbitrary formulas
- Define the range of negated TPQ queries having a DNF lineage

Thank you.



ACTUALLY, THAT ASSUMPTION ISN'T REALLY NECESSARY. WE CAN SEE HERE THAT THE POINT-COW APPROXIMATION WORKS EQUALLY WELL.