Tractable Query Answering
Under Probabilistic Constraints

Antoine Amarilli\textsuperscript{1}, Pierre Bourhis\textsuperscript{2}, Pierre Senellart\textsuperscript{1,3}

\textsuperscript{1}Télécom ParisTech
\textsuperscript{2}CNRS-LIFL
\textsuperscript{3}National University of Singapore

September 4th, 2014
Tractable Query Evaluation
On Probabilistic Instances

Antoine Amarilli\textsuperscript{1}, Pierre Bourhis\textsuperscript{2}, Pierre Senellart\textsuperscript{1,3}

\textsuperscript{1}Télécom ParisTech
\textsuperscript{2}CNRS-LIFL
\textsuperscript{3}National University of Singapore

September 4th, 2014
Instances and queries

- Given a relational instance (\(=\) set of facts, hypergraph)
  \[ I = \{ R(a, b), R(b, c), S(c) \} \]

- Given a conjunctive query (CQ) (existentially quantified)
  \[ q : \exists xy R(x, y) \land S(y) \]
Instances and queries

- Given a relational instance (\(=\) set of facts, hypergraph)
  \(I = \{ R(a, b), R(b, c), S(c) \}\)

- Given a conjunctive query (CQ) (existentially quantified)
  \(q : \exists xy R(x, y) \land S(y)\)

\(\rightarrow\) Query evaluation (model checking) of \(q\) on \(I\)
Instances and queries

- Given a relational instance (= set of facts, hypergraph)
  \[ I = \{ R(a, b), R(b, c), S(c) \} \]

- Given a conjunctive query (CQ) (existentially quantified)
  \[ q : \exists xy \ R(x, y) \land S(y) \]

  → Query evaluation (model checking) of \( q \) on \( I \)
Instances and queries

- Given a relational instance (\(=\) set of facts, hypergraph)
  \(I = \{R(a, b), R(b, c), S(c)\}\)

- Given a conjunctive query (CQ) (existentially quantified)
  \(q : \exists xy \, R(x, y) \land S(y)\)

  \(\rightarrow\) Query evaluation (model checking) of \(q\) on \(I\)
Instances and queries

- Given a relational instance (\(=\) set of facts, hypergraph)
  \( I = \{R(a, b), R(b, c), S(c)\} \)

- Given a conjunctive query (CQ) (existentially quantified)
  \( q : \exists xy \, R(x, y) \land S(y) \)

  \[ \rightarrow \text{Query evaluation (model checking) of } q \text{ on } I \]

  \[ \rightarrow \text{Data complexity: } q \text{ is fixed} \]
Uncertain and probabilistic instances

- Set of uncertain events
  - $e_{flight}$: CDG $\rightarrow$ VIE flight AF1756 takes place
  - $e_{bus}$: Vienna $\rightarrow$ Bratislava buses are running
Uncertain and probabilistic instances

- **Set of uncertain events**
  
  - $e_{\text{flight}}$: CDG $\rightarrow$ VIE flight AF1756 takes place
  - $e_{\text{bus}}$: Vienna $\rightarrow$ Bratislava buses are running

- **Annotate instance facts with formulae on the events**

<table>
<thead>
<tr>
<th>Instance</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsIn(jdoe, Paris)</td>
<td>$\neg e_{\text{flight}}$</td>
</tr>
<tr>
<td>IsIn(jdoe, Vienna)</td>
<td>$e_{\text{flight}} \land \neg e_{\text{bus}}$</td>
</tr>
<tr>
<td>IsIn(jdoe, Bratislava)</td>
<td>$e_{\text{flight}} \land e_{\text{bus}}$</td>
</tr>
</tbody>
</table>
Uncertain and probabilistic instances

- Set of uncertain events
  
  $e_{\text{flight}}$: CDG $\rightarrow$ VIE flight AF1756 takes place
  
  $e_{\text{bus}}$: Vienna $\rightarrow$ Bratislava buses are running

- Annotate instance facts with formulae on the events
  
  $\text{IsIn}(\text{jdoe, Paris})$  \hspace{1cm} $\neg e_{\text{flight}}$
  
  $\text{IsIn}(\text{jdoe, Vienna})$  \hspace{1cm} $e_{\text{flight}} \land \neg e_{\text{bus}}$
  
  $\text{IsIn}(\text{jdoe, Bratislava})$  \hspace{1cm} $e_{\text{flight}} \land e_{\text{bus}}$

  $\rightarrow$ Semantics: a set of instances (possible worlds).
Uncertain and probabilistic instances

- Set of uncertain events
  - $e_{flight}$: CDG $\rightarrow$ VIE flight AF1756 takes place
  - $e_{bus}$: Vienna $\rightarrow$ Bratislava buses are running
- Annotate instance facts with formulae on the events

  \[
  \begin{align*}
  \text{IsIn}(\text{jdoe, Paris}) & \quad \neg e_{flight} \\
  \text{IsIn}(\text{jdoe, Vienna}) & \quad e_{flight} \land \neg e_{bus} \\
  \text{IsIn}(\text{jdoe, Bratislava}) & \quad e_{flight} \land e_{bus}
  \end{align*}
  \]

  → Semantics: a set of instances (possible worlds).
- Add a probability distribution on each event
  - each event has probability $0 < p < 1$ of being true
  - all events are assumed to be independent
Uncertain and probabilistic instances

- Set of uncertain events
  - $e_{\text{flight}}$: CDG $\rightarrow$ VIE flight AF1756 takes place
  - $e_{\text{bus}}$: Vienna $\rightarrow$ Bratislava buses are running

- Annotate instance facts with formulae on the events

<table>
<thead>
<tr>
<th>Instance</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsIn(jdoe, Paris)</td>
<td>$\neg e_{\text{flight}}$</td>
</tr>
<tr>
<td>IsIn(jdoe, Vienna)</td>
<td>$e_{\text{flight}} \land \neg e_{\text{bus}}$</td>
</tr>
<tr>
<td>IsIn(jdoe, Bratislava)</td>
<td>$e_{\text{flight}} \land e_{\text{bus}}$</td>
</tr>
</tbody>
</table>

→ Semantics: a set of instances (possible worlds).

- Add a probability distribution on each event
  - each event has probability $0 < p < 1$ of being true
  - all events are assumed to be independent

→ Semantics: a probability distribution on instances.
Uncertain and probabilistic instances

- **Set of uncertain events**
  
  \[ e_{\text{flight}} \quad \text{CDG} \rightarrow \text{VIE} \quad \text{flight AF1756 takes place} \]
  
  \[ e_{\text{bus}} \quad \text{Vienna} \rightarrow \text{Bratislava} \quad \text{buses are running} \]

- **Annotate instance facts with formulae** on the events

  | IsIn(jdoe, Paris) | \( \neg e_{\text{flight}} \) |
  | IsIn(jdoe, Vienna) | \( e_{\text{flight}} \land \neg e_{\text{bus}} \) |
  | IsIn(jdoe, Bratislava) | \( e_{\text{flight}} \land e_{\text{bus}} \) |

  \[ \rightarrow \text{Semantics: a set of instances (possible worlds).} \]

  - **Add a probability distribution** on each event
    - each event has probability \( 0 < p < 1 \) of being true
    - all events are assumed to be **independent**

  \[ \rightarrow \text{Semantics: a probability distribution on instances.} \]

  \[ \rightarrow \text{Query evaluation: determine the probability of } q \text{ on } \hat{I}. \]
Hardness and tractability

- **With arbitrary annotations**
  - Query evaluation is \#P-hard even with a single fact
    (Immediate reduction from \#SAT)

- **With simple annotations** (one unique event per tuple)
  - Query evaluation is \#P-hard on arbitrary instances
    (Use the instance to do the reduction)
Hardness and tractability

- **With arbitrary annotations**
  - Query evaluation is \#P-hard even with a single fact
    (Immediate reduction from \#SAT)

- **With simple annotations** (one unique event per tuple)
  - Query evaluation is \#P-hard on arbitrary instances
    (Use the instance to do the reduction)

- **Existing work:**
  - Fix a simple annotation scheme
  - Show dichotomy between \#P-hard and PTIME queries
Hardness and tractability

- **With arbitrary annotations**
  - Query evaluation is \#P-hard even with a single fact
    (Immediate reduction from \#SAT)

- **With simple annotations** (one unique event per tuple)
  - Query evaluation is \#P-hard on arbitrary instances
    (Use the instance to do the reduction)

- **Existing work:**
  - Fix a simple annotation scheme
  - Show dichotomy between \#P-hard and PTIME queries

- **Our approach:**
  - Find a restriction on the instance and annotations
  - Show that many queries are tractable in this case
Bounded treewidth

An idea from instances without probabilities...

- If an instance has low treewidth then it is almost a tree
- Assume that the instance treewidth is constant...
Bounded treewidth

An idea from instances without probabilities...

- If an instance has **low treewidth** then it is almost a tree
- Assume that the instance treewidth is **constant**...

instance \( I \)

\[ R(a, b) \ R(b, c) \ S(c) \]
Bounded treewidth

An idea from instances without probabilities...

- If an instance has **low treewidth** then it is almost a tree
- Assume that the instance treewidth is **constant**...

\[
\text{instance } I \xrightarrow{} \text{tree encoding } T_i
\]

\[
R(a, b) \ R(b, c) \ S(c)
\]

**tree decomposition**

\[
O(|I|) \text{ for fixed width}
\]
**Bounded treewidth**

An idea from instances without probabilities...

- If an instance has **low treewidth** then it is almost a tree.
- Assume that the instance treewidth is **constant**...

\[
\begin{align*}
\text{instance } I & \quad \rightarrow \quad \text{tree encoding } T_I \\
R(a, b) \quad R(b, c) \quad S(c) & \quad \text{tree decomposition} \quad \text{tree encoding} \\
& \quad O(|I|) \text{ for fixed width}
\end{align*}
\]

\[
\exists xy \ R(x, y) \land S(y)
\]

query \( q \)
Bounded treewidth

An idea from instances without probabilities...

- If an instance has low treewidth then it is almost a tree
- Assume that the instance treewidth is constant...

\[
\text{instance } I \xrightarrow{\text{tree encoding }} T_I
\]

\[
R(a, b) R(b, c) S(c)
\]

\[
\text{tree decomposition}
\]

\[
O(|I|) \text{ for fixed width}
\]

\[
\text{rewriting}
\]

\[
O(1) \text{ data complexity}
\]

\[
\exists xy \ R(x, y) \land S(y)
\]

\[
\text{query } q \xrightarrow{\text{tree automaton }} A_q
\]
Bounded treewidth

An idea from instances without probabilities...

- If an instance has low treewidth then it is almost a tree
- Assume that the instance treewidth is constant...

\begin{itemize}
  \item instance \( I \) \( R(a, b) R(b, c) S(c) \)
  \item tree encoding \( T_I \)
  \item tree decomposition \( O(|I|) \) for fixed width
  \item rewriting \( O(1) \) data complexity
  \item query \( \exists xy R(x, y) \land S(y) \) \( A_q \)
  \item deterministic tree automaton \( A_q \)
  \item evaluation linear time
  \item query answer
\end{itemize}
Bounded treewidth

An idea from instances without probabilities...

- If an instance has low treewidth then it is almost a tree
- Assume that the instance treewidth is constant...

\[
\text{instance } I \xrightarrow{R(a, b) \ R(b, c) \ S(c)} \text{ tree encoding } T_I
\]

- tree decomposition
- \(O(|I|)\) for fixed width
- evaluation linear time
- query answer

\[
\text{rewriting } O(1) \text{ data complexity }
\]

\[
\exists xy \ R(x, y) \land S(y) \xrightarrow{\text{query } q} \text{ tree automaton } A_q
\]

\[
\text{evaluation linear time}
\]

\[
\text{query answer}
\]
Bounded treewidth

An idea from instances without probabilities...

- If an instance has low treewidth then it is almost a tree
- Assume that the instance treewidth is constant...

instance $I$ → tree encoding $T_I$

R$(a, b)$ $R(b, c)$ $S(c)$

tree decomposition

$O(|I|)$ for fixed width

evaluation linear time

query answer

rewriting

$O(1)$ data complexity

$\exists xy \ R(x, y) \land S(y)$

query $q$

deterministic tree automaton $A_q$

→ Linear time data complexity

instance $I$

R$(a, b)$ $R(b, c)$ $S(c)$

tree encoding $T_I$

tree decomposition

$O(|I|)$ for fixed width

evaluation linear time

query answer

rewriting

$O(1)$ data complexity

$\exists xy \ R(x, y) \land S(y)$

query $q$

deterministic tree automaton $A_q$

→ Linear time data complexity
An idea from probabilities without instances...

- Represent a propositional formula $F$ as a **Boolean circuit**
- Assume the circuit has **constant treewidth**

→ Probability of $F$ can be computed in **linear time**
  (using **junction tree** algorithm for Bayesian networks)
  (assuming constant-time **arithmetic operations**)

**Tractable inference**
cc-tables

- **Boolean circuit** for the annotations
**cc-tables**

- **Boolean circuit** for the annotations

\[
\begin{array}{c}
1/2 \quad 1/2 \quad 1/2 \\
\land \quad \land \\
R(a, b) \quad R(b, c) \quad R(c, d)
\end{array}
\]
cc-tables

- **Boolean circuit** for the annotations

\[
\begin{array}{c}
1/2 & 1/2 & 1/2 \\
\text{∧} & & \text{R}(a, b) \\
\text{∧} & \text{R}(b, c) & \text{R}(c, d)
\end{array}
\]

- **Circuit** must have low treewidth
- **Instance** must have low treewidth
  \[\rightarrow\] **Need simultaneous decomposition**
**cc-tables**

- **Boolean circuit** for the annotations
  
  \[
  \begin{array}{ccc}
  1/2 & 1/2 & 1/2 \\
  \land \\
  \land \\
  \end{array}
  \]

  \[
  R(a, b) \\
  R(b, c) \\
  R(c, d) \\
  \]

- **Circuit** must have low treewidth
- **Instance** must have low treewidth
  
  → Need **simultaneous** decomposition

- **Circuit** must have low treewidth

  ![Boolean circuit diagram](image)
Main result

instance $I$

\[
\begin{align*}
1/2 & 1/2 1/2 \\
\land & \land \\
R(a, b) & R(b, c) \\
R(c, d) &
\end{align*}
\]
Main result

instance $I$

\[ R(a, b) \land R(b, c) \land R(c, d) \]

tree encoding $T_i$

\[ 1/2 \quad 1/2 \quad 1/2 \]
\[ \land \]
\[ R(a, b) \]
\[ R(b, c) \]
\[ R(c, d) \]

\[ 1/2 \quad 1/2 \quad 1/2 \]
\[ \land \]
\[ R(a, b) \]
\[ R(b, c) \]
\[ R(c, d) \]

tree decomposition $O(|I|)$ for fixed width
Main result

\[ \exists xy \ R(x, y) \land S(y) \]

query \( q \)
Main result

instance $I$

$R(a, b) \land R(b, c) \land R(c, d)$

tree encoding $T_I$

$1/2 \land R(a, b) \land 1/2 \land R(b, c) \land 1/2 \land R(c, d)$

tree decomposition $O(|I|)$ for fixed width

rewriting

$O(1)$ data complexity

$\exists xy \ R(x, y) \land S(y)$

query $q$

deterministic tree automaton $A_q$
Main result

instance $I$

$\exists xy R(x, y) \land S(y)$

query $q$

rewriting

$O(1)$ data complexity

tree encoding $T_I$

tree decomposition $O(|I|)$ for fixed width

bounded treewidth circuit $C$

instrumentation linear time

deterministic tree automaton $A_q$
Main result

instance $I$

1/2 1/2 1/2
∧
∧
R(a, b)
R(b, c)
R(c, d)

tree encoding $T_I$

bounded treewidth circuit $C$

instrumentation linear time

rewriting

O(1) data complexity

$\exists xy R(x, y) \land S(y)$

query $q$

deterministic tree automaton $A_q$

probabilistic inference

O($|C|$) for fixed width

0.42

probability $p$
Main result

instance $I$

$\exists xy R(x, y) \land S(y)$

query $q$

rewriting

$O(1)$ data complexity

tree decomposition $O(|I|)$ for fixed width

tree encoding $T_I$

bounded treewidth circuit $C$

instrumentation linear time

probabilistic inference $O(|C|)$ for fixed width

0.42 probability $p$

deterministic tree automaton $A_q$
Consequences

- For queries **representable as deterministic automata** ...
  - CQs
  - Monadic second-order
  - Guarded second-order
- ... on various **probabilistic models** ...
  - Tuple-independent tables
  - Block-independent disjoint tables
  - pc-tables (presented before)
  - Probabilistic XML
- ... assuming **bounded treewidth** (for reasonable definitions) ...
  - ... probability of fixed $q$ can be computed in $O(|I|)!$
Conclusion

- We can combine the following techniques:
  - Computing tree decompositions
  - Encoding problems to automata on tree encodings of instances
  - Evaluating probabilities on bounded-treewidth circuits
Conclusion

- We can combine the following techniques:
  - Computing tree decompositions
  - Encoding problems to automata on tree encodings of instances
  - Evaluating probabilities on bounded-treewidth circuits

- Applications:
  - Tractable probabilistic query evaluation in practice?
  - Reasoning under uncertain rules
    (hence the bait-and-switch on the title...)

Thanks for your attention!
Conclusion

- We can **combine** the following techniques:
  - Computing tree decompositions
  - Encoding problems to **automata** on tree encodings of instances
  - Evaluating **probabilities** on bounded-treewidth circuits

- **Applications:**
  - Tractable probabilistic query evaluation in **practice**?
  - Reasoning under **uncertain rules**
    (hence the bait-and-switch on the title...)

- **Questions:**
  - Other **semirings** than Boolean AND/OR?
  - Other tasks than **probabilistic inference**?
We can combine the following techniques:
- Computing tree decompositions
- Encoding problems to automata on tree encodings of instances
- Evaluating probabilities on bounded-treewidth circuits

Applications:
- Tractable probabilistic query evaluation in practice?
- Reasoning under uncertain rules
  (hence the bait-and-switch on the title...)

Questions:
- Other semirings than Boolean AND/OR?
- Other tasks than probabilistic inference?
**Conclusion**

- We can **combine** the following techniques:
  - Computing **tree decompositions**
  - Encoding problems to **automata** on tree encodings of instances
  - Evaluating **probabilities** on bounded-treewidth circuits

- **Applications:**
  - Tractable probabilistic query evaluation in **practice**?
  - Reasoning under **uncertain rules**
    (hence the bait-and-switch on the title...)

- **Questions:**
  - Other **semirings** than Boolean AND/OR?
  - Other tasks than **probabilistic inference**?

What are bounded-treewidth circuits good for?

5 votes 0 answers 48 views

http://cstheory.stackexchange.com/q/25624

modified aug 28 at 13:05 a3nm 1,432

Thanks for your attention!