Uncertainty and Incompleteness

Antoine Amarilli

Télécom ParisTech; Institut Mines–Télécom; CNRS LTCI

May 19th, 2015
Motivation

- Traditional data management: data is correct and complete
- How realistic is this?
Motivation

- Traditional data management: data is **correct** and **complete**
- How **realistic** is this?
  - **Noisy** extractors
  - **Untrustworthy** contributors
  - **Crappy** crowd answers
  - **Non-exhaustive** sources
Motivation

- Traditional data management: data is correct and complete
- How realistic is this?
  - Noisy extractors
  - Untrustworthy contributors
  - Crappy crowd answers
  - Non-exhaustive sources

→ How to adapt to uncertain and incomplete data?
Uncertainty and Incompleteness

Uncertain data (aka. closed world)

- The truth is a **subset** of our database
- Find out in which **cases** our query holds
Uncertain data (aka. closed world)

- The truth is a \textit{subset} of our database
- Find out in which \textit{cases} our query holds
  \rightarrow Provenance!
Uncertain data (aka. closed world)

- The truth is a subset of our database
- Find out in which cases our query holds
  → Provenance!
- Quantitative models: probabilities on the database
Uncertain data (aka. closed world)

- The truth is a **subset** of our database
- Find out in which **cases** our query holds
  → Provenance!
- **Quantitative** models: probabilities on the database
  → #P-hardness lurks
Uncertain data (aka. closed world)

- The truth is a **subset** of our database
- Find out in which **cases** our query holds
  → Provenance!
- **Quantitative** models: probabilities on the database
  → #P-hardness lurks
→ Our idea: show tractability for restricted databases
Uncertainty and Incompleteness

Uncertain data (aka. closed world)

- The truth is a **subset** of our database
- Find out in which **cases** our query holds
  → Provenance!
- **Quantitative** models: probabilities on the database
  → #P-hardness lurks
→ **Our idea**: show tractability for restricted databases

---

**Provenance Circuits for Trees and Treelike Instances**

Antoine Amarilli\(^1\), Pierre Bourhis\(^2\), and Pierre Senellart\(^{1,3}\)

\(^1\) Institut Mines-Télécom; Télécom ParisTech; CNRS LTCI
first.last@telecom-paristech.fr

\(^2\) CNRS CRIStAL; Université Lille 1; INRIA Lille
pierre.bourhis@univ-lille1.fr

\(^3\) National University of Singapore; CNRS IPAL

**Abstract.** Query evaluation in monadic second-order logic is known to be tractable on trees and treelike instances, even though it is hard for arbitrary instances. This article presents a provenance framework for trees and treelike instances, by describing a linear-time construction of a circuit provenance representation for MSO queries. We show how this provenance can be connected to the usual definitions of semiring provenance on relational instances, even though we compute it in an unusual way, using tree automata; we do so via intrinsic definitions of provenance for general semirings, that are independent from the operational details of query evaluation. We show applications of this provenance to capture existing counting and probabilistic results on trees and treelike instances, and give novel consequences for probability evaluation.
Incomplete data (aka. open world)

- The truth is a superset of our database
- Harder to formalize...
Incomplete data (aka. open world)

- The truth is a superset of our database
- Harder to formalize...
  - Impose logical constraints
  - Ask whether the query is certain
Incomplete data (aka. open world)

- The truth is a **superset** of our database
- Harder to formalize...
  - → Impose **logical constraints**
  - → Ask whether the query is **certain**
- → **Idea 1**: combine existing approaches
  (description logics, existential rules)
Incomplete data (aka. open world)

- The truth is a superset of our database
- Harder to formalize...
  - Impose logical constraints
  - Ask whether the query is certain
- Idea 1: combine existing approaches
  (description logics, existential rules)
- Idea 2: what about assuming finiteness?
The truth is a superset of our database
Harder to formalize...

→ Impose logical constraints
→ Ask whether the query is certain

→ Idea 1: combine existing approaches (description logics, existential rules)
→ Idea 2: what about assuming finiteness?

Incomplete data (aka. open world)