Πgora: An Integration System for Probabilistic Data

Dan Olteanu and Lampros Papageorgiou and Sebastiaan J. van Schaik
Department of Computer Science, University of Oxford, Oxford, OX1 3QD, UK

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF OXFORD

Probabilistic Data Formalisms

Real-world applications model probabilistic data using a plethora of different formalisms, e.g.,

- **Bayesian networks** are a natural fit for managing expert knowledge, where the probabilistic relationship between input random variables, which are observable quantities, unknown parameters, or hypotheses, exhibits conditional independence.
  - Examples from the **UCI machine learning repository** at `http://archive.ics.uci.edu/ml/datasets.html`
- The **pc-tables** are relations extended with a special column that encodes the uncertainty of the records using probabilistic events.
  - **NELL tables** at `http://rtw.ml.cmu.edu/rtw/` consist of records extracted from hundreds of millions of web pages.
  - **Google Squared tables** aggregate unstructured, possibly contradictory information representing answers to keyword search queries.
- **Finite State Transducers (FSTs)** are stochastic automata used by optical character recognition programs, such as those powering **Google Books**, to capture probability distributions over all possible strings that could be represented in a given image.
  - Examples at `http://hazy.cs.wisc.edu/hazy/staccato/`

They support probabilistic processing to varying degrees:

- The **pc-tables formalism supports select-project-join queries** whose answers can be represented as pc-tables.
  - as implemented by the MayBMS/SPROUT query engine
- **Bayesian networks support inference queries** that ask for the conditional probability of an event given another event.
  - as implemented by the SMILE Bayesian inference engine
- **FSTs support selection queries** that ask for the probability that a certain string occurs in their possible runs.
  - as implemented by the Staccato system

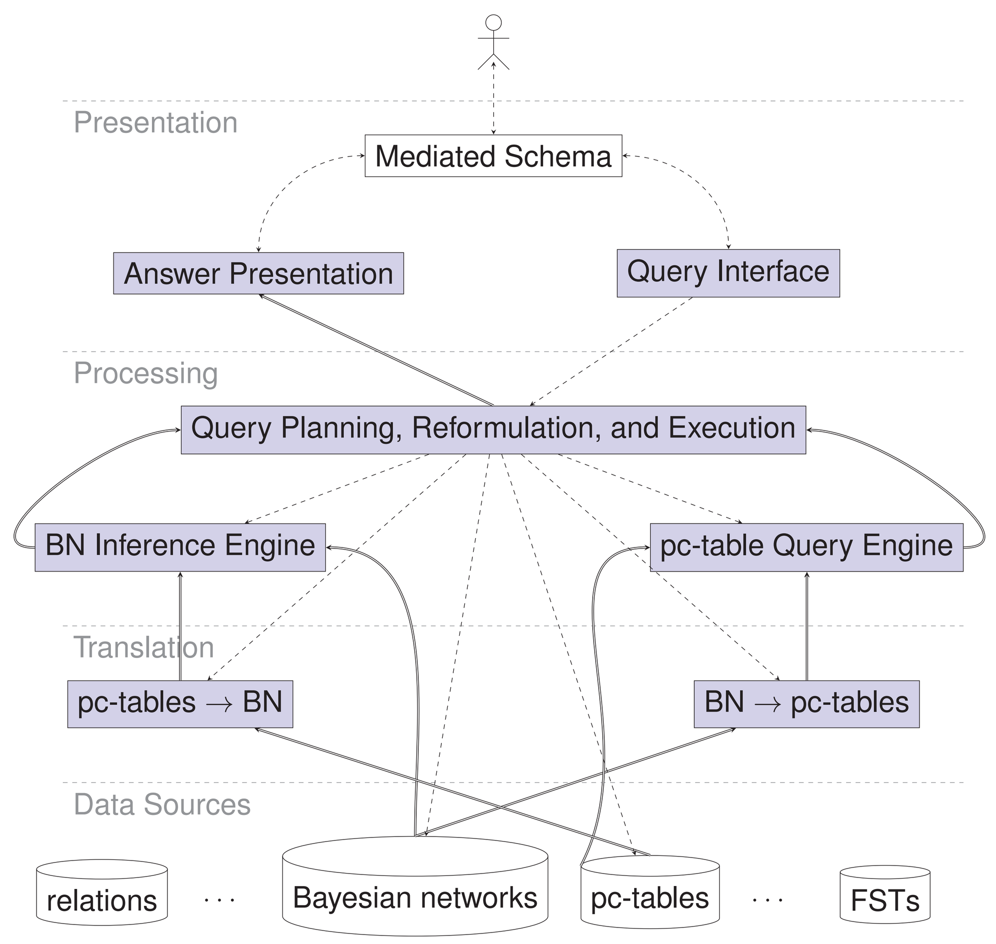
They admit a common interpretation via the possible worlds semantics:

- pc-tables represent finite probability distributions over sets of possible tables.
- Bayesian networks represent finite probability distributions over sets of correlated observations.
- FSTs represent finite probability distributions over sets of possible strings represented in an image.

Πgora's Capabilities

Queryable uniform interface to heterogeneous probabilistic data

- provides uniform interface = **mediated relational schema**
  - Each local source is registered to the system with a relational schema that becomes part of the mediated schema.
  - pc-tables export a relational schema without the events column.
  - Bayesian networks export a relational schema consisting of one attribute per node.
  - FSTs export schemas with one attribute.
- integrates data available in different probabilistic formalisms
- enables expressive querying across them
  - select-project-join queries
  - exact/approximate probability computation aggregate
  - a new `GIVEN` clause that allows to formulate conditionals.
- provides a query evaluation mechanism over the mediated schema
  - **Strategy 1**: Use different engines to evaluate subqueries natively supported by the formalisms of the input data sources. Translate intermediate results to pc-tables and complete the evaluation.
  - **Strategy 2**: Offline translation of all input data sources into either the pc-tables or Bayesian networks, followed by evaluation using either a query or an inference engine.
- provides transformations of sources to existing formalisms followed by evaluation using a single query/inference engine.
  - pc-tables, Bayesian networks, and FSTs are complete representation systems but of incomparable succinctness.
  - exponential-time translations between the formalisms
  - polynomial-time translations of Bayesian networks and FSTs into *pc-tables with event definitions*



Legend: Double/dashed arrows denote data/control flow.

Demonstration Scenario: Medical Data

Query: probability of a pregnant woman suffering from a left breast tumour, given that she also suffers from hypothyroidism.

```
SELECT conf() FROM Hypothyroid H, Breast_cancer B
WHERE B.tumour='true' AND B.breast='left' AND H.tumour='true' AND H.pregnant='true'
GIVEN B.age=H.age AND H.hypothyroid='primary'
```

Data sources: Bayesian networks `Hypothyroid` and `Breast_cancer`.

Strategy 1: purely Bayesian evaluation.

- Phrase the SQL query as a sum of inference queries:

$$\sum_{\text{B.age}} (P(\text{B.tumor} = \text{true} \wedge \text{B.breast} = \text{left} \wedge \text{H.tumor} = \text{true} \wedge \text{H.pregnant} = \text{true} \mid (\text{B.age} = \text{H.age} \wedge \text{H.hypothyroid} = \text{primary})))$$

- For a given value x for `age`, we have the inference query:

$$P(\text{B.tumor} = \text{true} \wedge \text{B.breast} = \text{left} \wedge \text{H.tumor} = \text{true} \wedge \text{H.pregnant} = \text{true} \mid (\text{B.age} = x \wedge \text{H.age} = x \wedge \text{H.hypothyroid} = \text{primary}))$$

- Since the two Bayesian networks are independent, we can regroup as follows:

$$P(\text{B.tumor} = \text{true} \wedge \text{B.breast} = \text{left} \mid \text{B.age} = x) * P(\text{H.tumor} = \text{true} \wedge \text{H.pregnant} = \text{true} \mid (\text{H.age} = x \wedge \text{H.hypothyroid} = \text{primary}))$$

Strategy 2: evaluation using pc-tables translations of Bayesian networks.

- Resolve the `GIVEN` clause using the conditional probability formula:

$$P(A \mid B) = \frac{P(A \wedge B)}{P(B)}$$

Strategy 3: hybrid evaluation assuming `Breast_cancer` is a pc-table.

- Split the query into the subqueries over each of `Hypothyroid` and `Breast_cancer`.
- For each value of x for `age` we have the inference query over `Hypothyroid`:

$$\forall x: P_H(x) = P(\text{H.tumor} = \text{true} \wedge \text{H.pregnant} = \text{true} \mid \text{H.age} = x \wedge \text{H.hypothyroid} = \text{primary})$$

- Rewrite the subquery over `Breast_cancer` by resolving the `GIVEN` clause:

```
CREATE TABLE T1 AS SELECT B.age, conf() AS p1 FROM Breast_cancer B
WHERE B.tumor='true' AND B.breast='left' GROUP BY B.age

CREATE TABLE T2 AS SELECT B.age, conf() as p2 FROM Breast_cancer B GROUP BY B.age

CREATE TABLE T3 AS SELECT T1.age, p1/p2 AS PB FROM T1, T2 WHERE T1.age = T2.age
```

- The query answer is obtained by joining the independent intermediate results:

$$\sum_{age} P_B(age) * P_H(age)$$

where $P_B(age)$ denotes P_B for the tuple (age, P_B) in T_3.