Uncertain Data

Background

- Databases: “Data is certain”
  - Bad!
  - What if you know something with 80, 99% confidence?
  - Some information is better than no information

Examples

- Basic: 4 v 9
- Bing/Google Translate
- Information Extraction
- CURE: “Ship ID”

Getting it wrong

- ICE Databases
- Credit Reports
- Zillow

Examples in Practice

- Image Classifier
- Bing Translate
- GitHub-CSV
- Calendar
- maybe-screen

Layers of Abstraction

Layer 1: Possible Worlds

- Question: What does it mean for data to be “Uncertain”? 
Question: What does it mean to run a query on “Uncertain Data”?

General approach: Not just 1 database, N databases

- Each database is a “Possible World” (like Schroedinger’s Cat: In one world the cat is alive, and in the other it isn’t)

- Extend deterministic query semantics to possible worlds:
  - \( Q(D) := \{ \text{Query}(D) \mid D \in \mathcal{D} \} \)
  - The query is evaluated in all possible worlds simultaneously.
  - All results that *could* occur, do occur

Possible Worlds semantics has a number of benefits:

- Agnostic to the database/data representation (works on Graph, JSON, Relational, etc...)

- Agnostic to the query semantics

- Even agnostic to the number of possible worlds (may even be infinite)

- If we can define what it means for a query to be correct in one world, we can define what it means for a query to be correct in all possible worlds.
  - … we just may not be able to run it efficiently

Possible Worlds also works with probabilities

- Probabilistic Database: \( \langle D, P \rangle \)
  - \( P : D \rightarrow [0,1] \); A probability measure over each world

- We can talk about the probability of a particular query result: \( R = Q(D) \)
  - \( P[R = Q(D)] = \sum(D \in \mathcal{D} \text{ where } Q(D) = R) \text{ of } P(D = D) \)
  - Sum up the probability of all worlds where \( Q \) has that result.

Aside: What Can You Do by Querying PDBs

- Figure out the probability of a specific outcome
  - compute \( P[R] \)
- Figure out the (k) most likely outcome(s)
  - compute \( \text{Argmax}[P[R]](Q(D)) \)

- Figure out which outcomes are possible
  - compute the set \( Q(D) \)

- Obtain a randomly selected sample from \( Q(D) \)
  - Typically sampled according to \( P(D) \)

- Figure out which outcomes are certain
  - compute the intersection of all relations in the set \( Q(D) \)
  - refine this somewhat… more shortly

- Visualize any of the above
  - e.g., Compute a histogram for the set of all possible outcomes
  - e.g., Compute a CDF
  - e.g., Visualize areas on a map
  - e.g., Graphs with error-bars

- Layer 2: Factorizing Worlds

  - Factorizing on Tuples

    - Idea 1: Give each tuple a probability
      - \( R(A, B, p) \rightarrow p \) defines the probability that any given \(<A,B>\) is in \( R \)
      - Often called the Tuple-Independent Model

    - Idea 2: Give each tuple a distribution of possible values
      - \( R(A, B, v) \rightarrow v \) is a tuple identifier. Only one tuple with a given identifier can be in \( R \). Can also assign a probability for each tuple set
      - Often called X-Tuples

    - Idea 3:
      - \( R(A, B, \phi) \rightarrow \phi \) is a boolean expression that determines whether a given \(<A,B>\) is in \( R \) (condition column)
• Often called C-Tables (though just a simplified form of them)

▼ Factorizing on Attributes
• Extended Null-Value Semantics: Labeled Nulls

▼ Observations
▼ Conflicts: What happens when...
• Tuple Independent + Self-Join?
• X-Tuple + Aggregate?
• C-Table + Multiple instances of the same variable?

▼ General Approach:
• D is a database with Labeled Nulls + Condition Columns (= Full C-Tables)
• v is a valuation or assignment of values to labeled nulls / condition column variables

▼ D = D[v]
• A (full) valuation defines one possible world of the database

▼ Computing Probabilities

▼ Lineage Formulas
▼ p[(A and B) or (A and C)] != 1 - ( 1 - (p[A] * p[B]) ) * ( 1 - (p[A] * p[C]) )
▼ pA * (1 - (1-pB)(1-pC) )
• pA * (pC + pB - pBpC)
• Not the same unless pApA = pA -> pA = 0 or 1

▼ Problem: Computing (A or B) is only possible if:
• A, B are mutually exclusive: pA + pB
• A, B are independent: 1 - (1-pA)(1-pB)
Naive Approach 1: MC methods:
- Pick A, B, C according to their probabilities
- Repeat enough times, you get a distribution of T/F similar to the overall probability

Naive Approach 2: Shannon Expansion
- Pick a variable (e.g., A) from the formula F
- Rewrite the formula:
  - \((A \land F[A \ \text{true}]) \lor ((\neg A) \land F[A \ \text{false}]\))
  - Now you have 2 mutually exclusive formulas:
  - \(p(F) = p_A \cdot p(F[A \ \text{true}]) + p_{\neg A} \cdot p(F[A \ \text{false}])\)
  - Other techniques as well

Cheating: What if most of the results are certain?
- Demo: Mimir
- Trick: Annotations