Interactive Join Query Inference with JIM

Angela Bonifati, Radu Ciucanu, Sławek Staworko
University of Lille & INRIA, France

Motivations
Large amount of data
- Coming from disparate data sources
- Shaped as denormalized tables
- Carrying little knowledge of metadata

Unfamiliar users
- Willing to query the data
- Unable to formally specify a query

We study the problem of query inference via simple tuple labeling as positive or negative examples.

Query inference via simple tuple labeling
Example: A user working for a travel agency wants to build a list of “flight+hotel” packages

<table>
<thead>
<tr>
<th>Flight</th>
<th>Hotel</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>To</td>
</tr>
<tr>
<td>Paris</td>
<td>Lille</td>
</tr>
<tr>
<td>Paris</td>
<td>Lille</td>
</tr>
<tr>
<td>Paris</td>
<td>Lille</td>
</tr>
<tr>
<td>Lille</td>
<td>NYC</td>
</tr>
<tr>
<td>Lille</td>
<td>NYC</td>
</tr>
<tr>
<td>Lille</td>
<td>NYC</td>
</tr>
<tr>
<td>NYC</td>
<td>Paris</td>
</tr>
<tr>
<td>NYC</td>
<td>Paris</td>
</tr>
<tr>
<td>Paris</td>
<td>NYC</td>
</tr>
<tr>
<td>Paris</td>
<td>NYC</td>
</tr>
<tr>
<td>Paris</td>
<td>NYC</td>
</tr>
</tbody>
</table>

Consistent queries:
- Airline = Discount
- To = City ∧ Airline = Discount

Join Inference Machine
Four types of interactions:
1. All tuples – The user can label any tuple in any order
2. Gray uninformative – After each new label we gray out the tuples that become uninformative
3. Top informative – We preprocess the database and show only the top informative tuples
4. Use a strategy – We use an interactive strategy of proposing tuples

Two types of datasets:
1. Relational tables
2. Sets of pictures varying in 4 features: number, symbol, shading, color

Interactive scenario
Input: a set of tuples

Is there any informative tuple left?
Yes
- Take an informative tuple t
- Ask label for t
- label + or -

No
- Output: inferred query q

Contributions
- We characterize the informative tuples (that contribute to the inference process)
- We prune uninformative tuples based on the lattice of join predicates (see rhs)
- We propose different strategies of interactively presenting tuples to the user while minimizing the number of user interactions*

1. Random – take a random informative tuple
2. Local – take an informative tuple according to a fixed order on the lattice
3. Lookahead – take the most informative tuple according to its “entropy”