CQA: Query Rewriting

Jan Chomicki

University at Buffalo and Warsaw University

March 22, 2007
Query rewriting

Constraints in **clausal form** (disjunctions of literals).

**Residues**
- Associated with single literals \( p(\bar{x}) \) or \( \neg p(\bar{x}) \) (one of each for every database relation \( p \))
- For each literal and each constraint that contains a complementary literal (after renaming), the **local residue** is obtained by removing the complementary literal and the quantifiers for its associated variables
- For each literal, the **global residue** = conjunction of local residues.

**Functional dependencies**
\[
(\forall x, y, z, y', z')(\neg E(x, y, z) \\
\lor \neg E(x, y', z') \lor y = y')
\]
\[
(\forall x, y, z, y', z')(\neg E(x, y, z) \\
\lor \neg E(x, y', z') \lor z = z')
\]

**Query**
\( E(x, y, z) \)

**Local residues**
\[
(\forall y', z')(\neg E(x, y', z') \lor y = y')
\]
\[
(\forall y', z')(\neg E(x, y', z') \lor z = z')
\]
Constructing the rewritten query

**Literal expansion**
For every literal in the original query, construct the expanded version as the conjunction of this literal and its global residue.

**Iteration**
The expansion step is iterated by replacing the literals in the residue by their expanded versions, until no changes occur.

**Query expansion**
Replace the literals in the query by their final expanded versions.

**Functional dependencies**

\[ (\forall x, y, z, y', z')(\neg E(x, y, z) \lor \neg E(x, y', z') \lor y = y') \]

\[ (\forall x, y, z, y', z')(\neg E(x, y, z) \lor \neg E(x, y', z') \lor z = z') \]

**Query**

\[ E(x, y, z) \]

**Rewritten query**

\[ E(x, y, z) \land (\forall y', z')(\neg E(x, y', z') \lor y = y') \land (\forall y', z')(\neg E(x, y', z') \lor z = z') \]
Integrity constraints

\[(\forall x)(\neg P(x) \lor R(x)) \land (\forall x)(\neg R(x) \lor S(x)))\]

<table>
<thead>
<tr>
<th>Literal</th>
<th>Residue</th>
<th>First expansion</th>
<th>Second (final) expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R(x))</td>
<td>(S(x))</td>
<td>(R(x) \land S(x))</td>
<td>(R(x) \land S(x))</td>
</tr>
<tr>
<td>(P(x))</td>
<td>(R(x))</td>
<td>(P(x) \land R(x))</td>
<td>(P(x) \land R(x) \land S(x))</td>
</tr>
<tr>
<td>(\neg R(x))</td>
<td>(\neg P(x))</td>
<td>(\neg R(x) \land \neg P(x))</td>
<td>(\neg R(x) \land \neg P(x))</td>
</tr>
<tr>
<td>(\neg S(x))</td>
<td>(\neg R(x))</td>
<td>(\neg S(x) \land \neg R(x))</td>
<td>(\neg S(x) \land \neg R(x) \land \neg P(x))</td>
</tr>
</tbody>
</table>

Scope of query rewriting

- queries involving conjunctions of literals (relational algebra: \(\sigma, \times, \neg\)) and binary universal integrity constraints [ABC99]
- existentially-quantified conjunctions (\(\pi, \sigma, \times\)) and single-key dependencies (under certain syntactic restrictions) [FM05]
M. Arenas, L. Bertossi, and J. Chomicki.
Consistent Query Answers in Inconsistent Databases.

A. Fuxman and R. J. Miller.
First-Order Query Rewriting for Inconsistent Databases.
Full version to appear in JCSS.