What is MRDM?

- Problem: Data in multiple tables
  - Want rules/patterns/etc. across tables
- Solution: Represent as single table
  - Join the data
  - Construct a single view
  - Use standard data mining techniques
- Example: “Customer” and “Married-to”
  - Easy single-table representation
- Bad Example: *Ancestor of*
Basis of Solutions:
Inductive Logic Programming

• ILP Rule:
  – customer(CID,Name,Age,yes) ⇐
    Age > 30 ∧ purchase(CID,PID,D,Value,PM) ∧
    PM = credit card ∧ Value > 100

• Learning methods:
  – Database represented as clauses (rules)
  – Unification: Given rule (function/clause),
    discover values for which it holds

Example

• How do we learn the “daughter” relationship?
  – Is this classification? Association?
• Covering Algorithm: “guess” at rule explaining only
  positive examples
  – Remove positive examples explained by rule
  – Iterate

<table>
<thead>
<tr>
<th>Training examples</th>
<th>Background knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>daughter(mary,ann).</td>
<td>parent(ann,mary). female(ann).</td>
</tr>
<tr>
<td>daughter(eve,tom).</td>
<td>parent(ann,tom). female(mary).</td>
</tr>
<tr>
<td>daughter(tom,ann).</td>
<td>parent(tom,eve). female(eve).</td>
</tr>
<tr>
<td>daughter(eve,ann).</td>
<td>parent(tom,ian).</td>
</tr>
</tbody>
</table>
How to make a good “guess”

• Clause subsumption:
  Generalize
  – More general clause
    (daughter(mary,Y)
    subsumes
    daughter(mary,ann)
• Start with general
  hypotheses and move
  to more specific

Issues

• Search space – efficiency
• Noisy data
  – positive examples labeled as negative
  – Missing data (e.g., a daughter with no parents
    in the database)
• What else might we want to learn?
**WARMR: Multi-relational association rules**

Algorithm WARMR (r, L, key, minfreq, Q)
Input: Database r; Declarative language bias L and key;
threshold minfreq
Output: All queries $Q \in L$ with frequency $\geq$ minfreq
1. Initialize level $d = 1$
2. Initialize the set of candidate queries $Q_1 := \{ r \vdash key \}$
3. Initialize the set of (in) frequent queries $F := \emptyset$; $I := \emptyset$
4. While $Q_1$ not empty
5. Find frequency of all queries $Q \in Q_1$
6. Move those with frequency below minfreq to $I$
7. Update $F := F \cup Q_1$
8. Compute new candidates:
   $Q_{d+1} = \text{WARMR}(r, L, F; Q_d)$
9. Increment $d$
10. Return $F$

Function WARMRGen ($L; I; F; Q_d$):
1. Initialize $Q_{d+1} := \emptyset$
2. For each $Q_j \in Q_d$, and for each refinement $Q'_j \in L$ of $Q_j$:
   Add $Q'_j$ to $Q_{d+1}$, unless:
   (i) $Q'_j$ is more specific than some query $r \in I$; or
   (ii) $Q'_j$ is equivalent to some query $r \in Q_{d+1} \cup F$
3. Return $Q_{d+1}$

---

**Multi-Relational Decision Trees**

```
procedure IterativeConquer(TreeOnTranslationSynt, DeclarativeBias, Examples)
  if TERMINATION(Criterion(Examples))
    then
      NoLeaf = DirectNextLast(Examples)
      return NoLeaf
  else
    PossibleTranslations = CONTENTRTree(TreeOnTranslationSynt, DeclarativeBias)
    BestTree = EmbedRTree(PossibleTranslations, Examples)
    (Split, Split) = SplitTree(Examples, TreeOnTranslationSynt, BestTree)
    LeftSubtree = IterativeConquer(TreeOnTranslationSynt, DeclarativeBias, Split)
    RightSubtree = IterativeConquer(TreeOnTranslationSynt, DeclarativeBias, Split)
    return [BestTree, LeftSubtree, RightSubtree]
```