An Interactive Clustering-based Approach to Integrating Source Query Interfaces on the Deep Web

Presented by Yingying Tao

Motivation

- Large number of data sources on web are hidden behind query interfaces
  - User has to access each source individually
  - A unified query interface is required for integration

- Limitations of current solutions
  - Flat schema
  - 1:1 mapping
  - Black-box fashion
  - Laborious parameter tuning

Hierarchical Modeling

- Query interface in HTML forms is consisted by fields
  - Text input box, selection lists, check box, etc.

- Each field contains three properties:
  - Name: id of the field
  - Label: description of the field
  - Domain: set of valid values the field may take
Hierarchical Modeling

- Hierarchical schema – ordered tree
  - Leaf element: field in the interface
  - Internal element: group/super-group of fields
  - Sibling elements: elements with same parent

Interface Matching

- Interface matching – identify semantically similar fields over different query interfaces
  - 1:1 mapping vs. 1:m mapping

Challenges and solutions

- 1:1 mapping
  - Label mismatch problem
    - Bridging approach
  - Label & b → x → c

- 1:m mapping
  - More complex
    - Field matching via clustering

Interface Matching via Clustering

- Field similarity function
  
  For two field e and f in different interface, their similarity

  \[ S(e,f) = \lambda_1 \cdot \text{linguistic}_\text{Sim}(e,f) + \lambda_2 \cdot \text{domain}_\text{Sim}(e,f) \]

  \[ \lambda_1 = \text{name}_\text{Sim}(e,f) + \lambda_2 \cdot \text{value}_\text{Sim}(e,f) \]

  \[ \lambda_2 = \text{value}_\text{Sim}(e,f) \]

  \[ d \text{ and } d' \Rightarrow \text{domain of field } e \text{ and } f \]
Finding 1:1 mappings – Greedy matching

1. Place each field in \( D \) in a cluster by itself.
2. While there are two clusters with similarity \( \geq \tau \),
   - choose two clusters, \( c_1 \) and \( c_2 \), whose similarity
     is the largest over all pairs of clusters.
   - merge the two if necessary.
   - merge \( c_1 \) and \( c_2 \) into a new cluster \( c_{12} \) and
     remove clusters \( c_1 \) and \( c_2 \).
   - remove all rows and columns associated with \( c_1 \) and
     \( c_2 \) in \( D \), and add a new row and column for \( c_{12} \).
   - compute similarities of \( c_{12} \) with other clusters
     using Formula 1.
3. Return the clusters of fields.

Interface Matching via Clustering

Breaking tie – more than one pair with same max similarity

- Select the first best choice

Question: how to determine the order of fields?

Finding 1:m mappings – three phases

- Preliminary 1-m matching phase
- Clustering phase
- Final 1-m matching phase
Interface Matching via Clustering

- Identify preliminary 1:m mappings
  Aggregate type
  Is-a type
  Infinite domains

- Obtain final 1:m mappings
  Bridging approach:
  $a \{b_1, b_2\} \& b_1 \sim c_1, b_2 \sim c_2$

User Interactions and Parameter Learning

- Parameter learning – learning the threshold
  
  **Observation:**
  - Matching fields typically have at least one large component similarity
  - Non-matching fields normally have small similarities in both components

  **Approach:** Finding the gap

User Interactions and Parameter Learning

- User Interaction – resolving uncertainties
  - Determine possible homonyms
    - High linguistic similarity but low domain similarity
  - Determine possible synonyms
  - Check Ask-Merge procedure
  - Determine Possible 1:m mappings
Experiments

- Data set
  - 5 domains, 20 query interfaces for each
  - Manually transformed into schema trees
  - All weight coefficients based on observation

- Performance Measurement
  - Precision (P)
  - Recall (R)
  - F-measure (F)

Experiments

- Experimental results
  - Automatic field matching accuracy
    - Threshold set to zero
      - Average P = 88.2%, R = 91.1%, F = 89.5%
    - Threshold learning results
      - Average P = 95.2%, R = 88.0%, F = 91.3%
  - User interaction results
    - Average P = 96.0%, R = 94.0%, F = 94.8%

Experiments

- Component contribution
  - 1:m mappings
  - Instance information
  - Tie resolution

Question: Under what circumstances 1:m mapping may have a worse performance?
**Conclusions and Future Work**

- Conclusions
  - Flat schema vs. schema tree
  - 1:1 mapping vs. 1:m mapping
  - Blackbox vs. user interaction
  - Threshold tuning vs. threshold learning

- Future work
  - Automatically generating schema trees
  - Better solutions for breaking the tie
  - Self-learning on weight coefficients

**Discussions**

- Effectiveness vs. efficiency?
- Depth of the schema tree: what’s the purpose?
- Transitivity of the bridging approach?
- How to handle dynamic query interfaces?
- How to determine the weight coefficients?
- How to define the order of fields for breaking the tie?
- When will the 1:m mapping approach has a worse performance?