Similarity Query Processing for Probabilistic Sets

Ming Gao 1, Cheqing Jin 1, Wei Wang 2, Xuemin Lin 1,2, Aoying Zhou 1*

1Shanghai Key Laboratory on trustworthy computing, Software Engineering Institute, East China Normal University, Shanghai, China
2The University of New South Wales, Sydney, Australia
omega.mgao@gmail.com, {cqjin, ayzhou}@sei.ecnu.edu.cn
{weiw, lxue}@cse.unsw.edu.au
Motivation

• Evaluate similarity between uncertain sets

• Existing work
  – Huge model size
  – Significant similarity evaluation cost

• This paper
  – Comprehensive study for probabilistic set may have thousands of elements
Solution

- Similarities based on dynamic programming
  - Expected Similarity (ES)
  - Confidence-based Similarity (CS)
- Exact query processing based on pruning
  - Individual pruning
  - Batch pruning
- Approximate query processing based on sampling
Agenda

• Introduction
• Related work
• Problem definition and data normalization
• Exact similarity computation
• Pruning techniques
• Approximate solution
• Experiments
Introduction

• Applications
  – Personalization systems
  – Multi-label classification

• Contribution
  – Handle large p-sets efficiently
  – Similarity measure based on dynamic programming
  – Pruning techniques and approximate methods
  – Experiments upon synthetic and real datasets
Related work

• Uncertain Data Management
  – Information extraction and integration, multimedia retrieval, optical character recognition
  – MayBMS, MystiQ, Trio

• Similarity Search
  – Top-k, k-NN, reverse k-NN, range queries

• Similarity Join
  – Batch similarity queries
Related work

• Efficient processing of probabilistic set-containment queries on uncertain set-valued data.
  • Same
    • Probabilistic set model, one of the similarity measure
  • Different
    • Pruning methods, approximate methods

• Probabilistic string similarity joins.
  • Different
    • Non-neglectible correlations
    • Involving aggregated probabilities
Related work

- Set similarity join on probabilistic data. [VLDB 2010]

<table>
<thead>
<tr>
<th>Model</th>
<th>Expressive Power</th>
<th>Exact Similarity Computation</th>
<th>Upper Bound Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set-level [27]</td>
<td>Most general</td>
<td>$O(N^2)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Element-level [27]</td>
<td>Can model exclusion</td>
<td>$\Omega(2^n)$</td>
<td>$O(n^2)$ (online) or $O(n)$ (offline)</td>
</tr>
<tr>
<td>Our p-set model</td>
<td>A special case of Element-level model</td>
<td>$O(n^3)$</td>
<td>$O(n)$</td>
</tr>
</tbody>
</table>

- Models and Similarity Evaluation
  - Set-level
  - Element-level

- Pruning Rules
  - Jaccard Distance pruning
  - Probability upper bound pruning
Problem definition and data normalization

• Probabilistic set model

\[ A = \{ a_i : p_{a_i} | a_i \in \mathcal{D}, \forall i \in [1, n] \} \]

• Possible world semantics

\[
\begin{align*}
\mathcal{W}(A, B) &= \mathcal{W}(A) \times \mathcal{W}(B) \\
\text{Pr}[w] &= \prod_{t \in w} p_t \prod_{t \not\in w} (1 - p_t) \\
(w_a, w_b) \in \mathcal{W}(A, B) \text{ is } \text{Pr}[w_a] \cdot \text{Pr}[w_b].
\end{align*}
\]

• Jaccard coefficient

\[
\text{jac}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}
\]
Problem definition and data normalization

• Example
  – P-sets

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${1 : 0.7, 2 : 1.0}$</td>
<td>${1 : 1.0, 2 : 0.5, 3 : 0.8}$</td>
</tr>
</tbody>
</table>

– All the joint possible worlds

<table>
<thead>
<tr>
<th>$w_a$</th>
<th>$w_b$</th>
<th>$\Pr[(w_a, w_b)]$</th>
<th>Jaccard</th>
</tr>
</thead>
<tbody>
<tr>
<td>${2^A}$</td>
<td>${1^B}$</td>
<td>0.03</td>
<td>0</td>
</tr>
<tr>
<td>${2^A}$</td>
<td>${1^B, 2^B}$</td>
<td>0.03</td>
<td>0.5</td>
</tr>
<tr>
<td>${2^A}$</td>
<td>${1^B, 3^B}$</td>
<td>0.12</td>
<td>0</td>
</tr>
<tr>
<td>${2^A}$</td>
<td>${1^B, 2^B, 3^B}$</td>
<td>0.12</td>
<td>0.333</td>
</tr>
<tr>
<td>${2^A, 1^A}$</td>
<td>${1^B}$</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td>${2^A, 1^A}$</td>
<td>${1^B, 2^B}$</td>
<td>0.07</td>
<td>1</td>
</tr>
<tr>
<td>${2^A, 1^A}$</td>
<td>${1^B, 3^B}$</td>
<td>0.28</td>
<td>0.333</td>
</tr>
<tr>
<td>${2^A, 1^A}$</td>
<td>${1^B, 2^B, 3^B}$</td>
<td>0.28</td>
<td>0.666</td>
</tr>
</tbody>
</table>

Yifu Huang
Problem definition and data normalization

• Expected Similarity (ES)

\[
ES(A, B) = \sum_{(w_a, w_b) \in W(A, B)} \text{sim}(w_a, w_b) \cdot \Pr[(w_a, w_b)] \\
= \sum_{w_a \in W(A) \land w_b \in W(B)} \text{sim}(w_a, w_b) \cdot \Pr[w_a] \cdot \Pr[w_b]
\]

• Confidence-based Similarity (CS)

\[
CS(A, B, \text{minconf}) = \max\{ x \mid \text{CPr}(x, A, B) \geq \text{minconf} \}
\]

– conditioned cumulative probability \( \text{CPr}(x, A, B) \)

\[
\text{CPr}(x, A, B) = \sum_{(w_a, w_b) \in W(A, B) \land \text{sim}(w_a, w_b) \geq x} \Pr[(w_a, w_b)]
\]
Problem definition and data normalization

• Example

\[
\begin{array}{c|c|c|c}
A & B & Pr[(w_a, w_b)] & Jaccard \\
\hline
\{1 : 0.7, 2 : 1.0\} & \{1 : 1.0, 2 : 0.5, 3 : 0.8\} & & \\
\{2^A\} & \{1^B\} & 0.03 & 0 \\
\{2^A\} & \{1^B, 2^B\} & 0.03 & 0.5 \\
\{2^A\} & \{1^B, 3^B\} & 0.12 & 0 \\
\{2^A\} & \{1^B, 2^B, 3^B\} & 0.12 & 0.333 \\
\{2^A, 1^A\} & \{1^B\} & 0.07 & 0.5 \\
\{2^A, 1^A\} & \{1^B, 2^B\} & 0.07 & 1 \\
\{2^A, 1^A\} & \{1^B, 3^B\} & 0.28 & 0.333 \\
\{2^A, 1^A\} & \{1^B, 2^B, 3^B\} & 0.28 & 0.666 \\
\end{array}
\]

\[
\begin{array}{c|c}
& Jaccard \\
ES(A, B) & 0.44 \\
CS(A, B, minconf = 0.3) & 0.666 \\
CS(A, B, minconf = 0.5) & 0.333 \\
\end{array}
\]
Problem definition and data normalization

• Normalization of two p-sets

\[ A = \{ c_1 : p_{c_1}^A, \ldots, c_k : p_{c_k}^A, d_1 : p_{d_1}, \ldots, d_{n-k} : p_{d_{n-k}} \} \]
\[ B = \{ c_1 : p_{c_1}^B, \ldots, c_k : p_{c_k}^B, d_{n-k+1} : p_{d_{n-k+1}}, \ldots, \\
\quad d_{n+m-2k} : p_{d_{n+m-2k}} \} \]

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1 : 0.7, 2 : 1.0}</td>
<td>{1 : 1.0, 2 : 0.5, 3 : 0.8}</td>
</tr>
</tbody>
</table>

• Size and expected size

<table>
<thead>
<tr>
<th>( w_a )</th>
<th>( w_b )</th>
<th>( \Pr[(w_a, w_b)] )</th>
<th>Jaccard</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2}</td>
<td>{1}</td>
<td>0.03</td>
<td>0</td>
</tr>
<tr>
<td>{2}</td>
<td>{1, 2}</td>
<td>0.03</td>
<td>0.5</td>
</tr>
<tr>
<td>{2}</td>
<td>{1, 3}</td>
<td>0.12</td>
<td>0</td>
</tr>
<tr>
<td>{2}</td>
<td>{1, 2, 3}</td>
<td>0.12</td>
<td>0.333</td>
</tr>
<tr>
<td>{2, 1}</td>
<td>{1}</td>
<td>0.07</td>
<td>0.5</td>
</tr>
<tr>
<td>{2, 1}</td>
<td>{1, 2}</td>
<td>0.07</td>
<td>1</td>
</tr>
<tr>
<td>{2, 1}</td>
<td>{1, 3}</td>
<td>0.28</td>
<td>0.333</td>
</tr>
<tr>
<td>{2, 1}</td>
<td>{1, 2, 3}</td>
<td>0.28</td>
<td>0.666</td>
</tr>
</tbody>
</table>

2013/7/28
Yifu Huang
Exact similarity computation

- Equivalent classes
- Example

\[ H[i, j] = \sum_{(w_a, w_b) \in \mathcal{W}(A, B) \land |w_a \cap w_b| = i \land |w_a \cup w_b| = j} \Pr[w_a] \cdot \Pr[w_b] \]

<table>
<thead>
<tr>
<th>( w_a )</th>
<th>( w_b )</th>
<th>( \Pr[(w_a, w_b)] )</th>
<th>( i )</th>
<th>( j )</th>
<th>Jaccard</th>
</tr>
</thead>
<tbody>
<tr>
<td>{2^A}</td>
<td>{1^B}</td>
<td>0.03</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>{2^A}</td>
<td>{1^B, 3^B}</td>
<td>0.12</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>{2^A}</td>
<td>{1^B, 2^B, 3^B}</td>
<td>0.12</td>
<td>1</td>
<td>3</td>
<td>0.333</td>
</tr>
<tr>
<td>{2^A, 1^A}</td>
<td>{1^B, 3^B}</td>
<td>0.28</td>
<td>1</td>
<td>3</td>
<td>0.333</td>
</tr>
<tr>
<td>{2^A}</td>
<td>{1^B, 2^B}</td>
<td>0.03</td>
<td>1</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>{2^A, 1^A}</td>
<td>{1^B}</td>
<td>0.07</td>
<td>1</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>{2^A, 1^A}</td>
<td>{1^B, 2^B, 3^B}</td>
<td>0.28</td>
<td>2</td>
<td>3</td>
<td>0.666</td>
</tr>
<tr>
<td>{2^A, 1^A}</td>
<td>{1^B, 2^B}</td>
<td>0.07</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( i = 0 )</th>
<th>( j = 0 )</th>
<th>( j = 1 )</th>
<th>( j = 2 )</th>
<th>( j = 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( j = 0 )</td>
<td>0</td>
<td>0</td>
<td>0.03</td>
<td>0.12</td>
</tr>
<tr>
<td>( j = 1 )</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>( j = 2 )</td>
<td>0.07</td>
<td>0.28</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Exact similarity computation

- Calculate ES

\[
ES = \sum_{i=1}^{k} \sum_{j=i}^{m+n-k} H[i,j] \cdot (i/j)
\]

- Calculate CS

---

**Algorithm 1: Calculate CS from \( H[i,j] \)**

| Input: \( H[i,j] \), \( \text{minconf} \)  
| Data: heap is a max-heap on the similarity values. |

1. for \( i = 1 \) to \( k \) do  heap.push(1.0, \( i \), \( i \));  
2. \( CPr \leftarrow 0; \ sim \leftarrow 0; \)  
3. while heap.empty = false do  
   4. \( (\text{sim}, \ i, \ j) \leftarrow \text{heap.pop}; \)  
   5. \( CPr \leftarrow CPr + H[i,j]; \)  
   6. if \( CPr \geq \text{minconf} \) then break;  
   7. if \( j < m+n-k \) then heap.push(\( \frac{i}{j+1}, i, j + 1 \));  
8. return \( \text{sim} \)

---

\[ H[i,j] \]

<table>
<thead>
<tr>
<th></th>
<th>( j = 0 )</th>
<th>( j = 1 )</th>
<th>( j = 2 )</th>
<th>( j = 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i = 0 )</td>
<td>0</td>
<td>0</td>
<td>0.03</td>
<td>0.12</td>
</tr>
<tr>
<td>( i = 1 )</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>( i = 2 )</td>
<td></td>
<td>0.07</td>
<td>0.28</td>
<td></td>
</tr>
</tbody>
</table>
Exact similarity computation

• Computing $H$
  – Common element
    \[
    H^l[i, j] = H^{l-1}[i, j](1 - p_i^A)(1 - p_i^B) \\
    + H^{l-1}[i, j-1](p_i^A(1 - p_i^B) + (1 - p_i^A)p_i^B) \\
    + H^{l-1}[i-1, j-1]p_i^A p_i^B
    \]
  – Distinct element
    \[
    H^l[i, j] = H^{l-1}[i, j](1 - p_i) + H^{l-1}[i, j-1]p_i
    \]
  – Time complexity $O(n^3)$
  – Space complexity $O(n^2)$
Pruning Techniques

Algorithm 2: Answer Queries with Pruning \((Q, \{O_i\}, \tau, \minconf)\)

\begin{itemize}
  \item \(C \leftarrow \) candidates that survive the batch pruning (c.f., Sec. V-D);
  \item \textbf{foreach} \(p\)-set in \(C\) \textbf{do}
    \begin{itemize}
      \item \(\text{pruned} \leftarrow \text{false};\)
      \item \textbf{if} the query type is ESQ then
        \begin{itemize}
          \item \(ub \leftarrow \text{calcESUpperBound}(Q, O_i)\) (c.f., Sec. V-B);
          \item \textbf{if} \(ub < \tau\) \textbf{then} \(\text{pruned} \leftarrow \text{true}\)
        \end{itemize}
      \item \textbf{if} the query type is CSQ then
        \begin{itemize}
          \item \(ub \leftarrow \text{calcCSUpperBound}(Q, O_i, \tau)\) (c.f., Sec. V-C);
          \item \textbf{if} \(ub < \minconf\) \textbf{then} \(\text{pruned} \leftarrow \text{true}\)
        \end{itemize}
      \item \textbf{if} \(\text{pruned} = \text{false}\) \textbf{then}
        \begin{itemize}
          \item \(\text{sim} \leftarrow \) the similarity value between \(Q\) and \(O_i;\)
          \item \textbf{if} \(\text{sim} \geq \tau\) \textbf{then}
            \begin{itemize}
              \item output \(O_i;\)
            \end{itemize}
        \end{itemize}
    \end{itemize}
\end{itemize}

\[E[|A|], \text{ is } \sum_{w \in W(A)} |w| \cdot Pr[w] = \sum_{i=1}^{n} p_i^A\]

\[E[|A \cap B|], \text{ is } \sum_{(w_a, w_b) \in W(A, B)} |w_a \cap w_b| \cdot Pr[(w_a, w_b)] = \sum_{i=1}^{k} p_i^A \cdot p_i^B\]

\[E[|A \cup B|], \text{ is } E[|A|] + E[|B|] - E[|A \cap B|] = \sum_{i=1}^{k} (p_i^A + p_i^B - p_i^A \cdot p_i^B) + \sum_{i=k+1}^{n+m-k} p_i\]
Pruning Techniques

• Pruning Rule for ESQ

\[ E[X/Y] < UB_1(E[X], E[Y]) \]

\[ UB_1(u, v) = \min_{\exp(-u/3) \leq \epsilon \leq 1} \left( 2\epsilon + \frac{u + \sqrt{-3u \ln \epsilon}}{v - \sqrt{-2v \ln \epsilon}} \right) \]

\[ UB_1(E[|Q \cap O|], E[|Q \cup O|]) \leq \tau \]

• Pruning Rule for CSQ

\[ \Pr[X \geq \alpha Y] < UB_2(E[X], E[Y], \alpha) \]

\[ UB_2(u, v, \alpha) = \min_{u \leq \xi \leq \min(\alpha v, 2u)} \left( e^{-(\alpha v - \xi)^2 / 2\alpha^2 v} + e^{-\xi-u)^2 / 3u} \right) \]

\[ E[|Q \cap O|] \leq \tau \cdot E[|Q \cup O|] \]

\[ UB_2(E[|Q \cap O|], E[|Q \cup O|], \tau) \leq \text{minconf} \]
Pruning Techniques

• Batch Pruning
  – Discard many p-sets in the database without even evaluating their similarity upper bounds
    • Index all the p-sets in the database by their expected sizes
    • Compute a lower bound $S_L$ and an upper bound $S_U$ of the expected size for the appropriate query type
    • Only consider p-sets in the database whose expected sizes fall within $[S_L, S_U]$. 
Pruning Techniques

• Batch Pruning
  – How to decide $S_L$ and $S_U$
  – Batch Pruning for ESQ

$$x + \sqrt{-3x \ln \epsilon^*} = (\tau - 2\epsilon^*)(E[|Q|] - \sqrt{-2E[|Q|] \ln \epsilon^*})$$
$$x - \sqrt{-2x \ln \epsilon^*} = \left( E[|Q|] + \sqrt{-3E[|Q|] \ln \epsilon^*} \right) / (\tau - 2\epsilon^*)$$

– Batch Pruning for CSQ

$$\exp \left( \frac{-(\xi_1^* - x)^2}{3x} \right) = \text{minconf}/2$$
$$\exp \left( \frac{-(\tau \cdot x - \xi_2^*)^2}{2\tau^2 \cdot x} \right) = \text{minconf}/2$$
Approximate solution

• Sampling-based method
  – Approximate algorithm for ES
    \[ \left\lceil \frac{(\ln \frac{2}{\delta})/(2\varepsilon^2)}{\varepsilon} \right\rceil \quad \Pr \left[ \left| \hat{ES} - ES \right| \leq \varepsilon \right] \geq 1 - \delta \]
  – Approximate algorithm for CS
    \[ G = 24 \cdot \left\lfloor \ln \frac{1}{\delta} \right\rfloor, \quad M = \left\lceil 2\varepsilon^{-2} \right\rceil \quad \Pr \left[ CS^- \leq \hat{CS} \leq CS^+ \right] \geq 1 - \delta \]
  – O(n)
Experiments

• Implementation
  – Java
  – Intel Pentium IV 2.8GHz CPU
  – 4GB memory
• Synthetic datasets
  – SYNα-U
    • a uniform distribution within the range of \([v, 0.9]\) with a default \(v\) value of 0.2.
  – SYNα-G
    • a Gaussian distribution \(N(u, o)\) capped to the range of \((0, 1]\). By default, \(u = 0.8\) and \(o = 0.2\).
Experiments

• Real-world datasets
  – pDBLP
    • a fairly simple yet effective method based on topical terms used in authors’ DBLP entries
  – pDeli
    • the social bookmarking dataset which was crawled from the Del.icio.us web site during 2006 and 2007
  – Sigmoid function
    \[ p(e) = \frac{2}{1+\exp(-e(e))} - 1 \]
Experiments

• Default parameters

\[ minconf = 0.5 \text{ (for CS)}, \tau = 0.5, \alpha = 1000, \gamma = 10\%, \]
\[ \epsilon = 0.06, \delta = 0.06, \nu = 0.2, \sigma = 0.2, \text{ and } \mu = 0.8. \]

• Measures
  
  – Memory Usage
  – Computation Time
  – Query Time, Pruning time
  – Candidate size, result size
  – Pruning rate
  – Average precision
Experiments

• Computing Similarities Exactly

(a) Space consumption
(b) Computation time
Experiments

• Computing Similarities Approximately
Experiments

- Evaluating Pruning Efficiency on SYN
Experiments

- Performance on the pDBLP Dataset
Experiments

- Performance on the pDeli Dataset

<table>
<thead>
<tr>
<th>AP@k</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES</td>
<td>0.7000</td>
<td>0.6675</td>
<td>0.6250</td>
<td>0.5875</td>
<td>0.5825</td>
<td>0.5500</td>
</tr>
<tr>
<td>CS</td>
<td>0.7000</td>
<td>0.6785</td>
<td>0.6280</td>
<td>0.5825</td>
<td>0.575</td>
<td>0.5500</td>
</tr>
</tbody>
</table>

(a) Pruning Rate
(b) Pruning Rate
(c) Query Time
(d) Query Time
Thank You!