Data Anonymization -
Generalization Algorithms

Li Xiong

CS573 Data Privacy and Anonymity
Generalization and Suppression

- **Generalization**
  - Replace the value with a less specific but semantically consistent value

- **Suppression**
  - Do not release a value at all

### Z0 = {41075, 41076, 41095, 41099}

### Z1 = {4107*, 4109*}

### Z2 = {410**}

### S0 = {Male, Female}

### S1 = {Person}

<table>
<thead>
<tr>
<th>#</th>
<th>Zip</th>
<th>Age</th>
<th>Nationality</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>41076</td>
<td>&lt; 40</td>
<td>*</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>2</td>
<td>48202</td>
<td>&lt; 40</td>
<td>*</td>
<td>Heart Disease</td>
</tr>
<tr>
<td>3</td>
<td>41076</td>
<td>&lt; 40</td>
<td>*</td>
<td>Cancer</td>
</tr>
<tr>
<td>4</td>
<td>48202</td>
<td>&lt; 40</td>
<td>*</td>
<td>Cancer</td>
</tr>
</tbody>
</table>
Complexity

Search Space:

• Number of generalizations = \( \prod_{\text{attrib } i} (\text{Max level of generalization for attribute } i + 1) \)

If we allow generalization to a different level for each value of an attribute:

• Number of generalizations = \( \prod_{\text{attrib } i} (\text{Max level of generalization for attribute } i + 1) \)
Hardness result

- Given some data set $R$ and a QI $Q$, does $R$ satisfy k-anonymity over $Q$?
  - Easy to tell in polynomial time, NP!

- Finding an *optimal* anonymization is not easy
  - NP-hard: reduction from k-dimensional perfect matching
  - A polynomial solution implies $P = NP$

Taxonomy of Generalization Algorithms

- Top-down specialization vs. bottom-up generalization
- Global (single dimensional) vs. local (multi-dimensional)
- Complete (optimal) vs. greedy (approximate)
- Hierarchy-based (user defined) vs. partition-based (automatic)

Generalization algorithms

- Early systems
  - μ-Argus, Hundpool, 1996 - Global, bottom-up, greedy
  - Datafly, Sweeney, 1997 - Global, bottom-up, greedy

- k-anonymity algorithms
  - AllMin, Samarati, 2001 - Global, bottom-up, complete, impractical
  - MinGen, Sweeney, 2002 - Global, bottom-up, complete, impractical
  - Bottom-up generalization, Wang, 2004 – Global, bottom-up, greedy
  - TDS (Top-Down Specialization), Fung, 2005 - Global, top-down, greedy
  - K-OPTIMIZE, Bayardo, 2005 – Global, top-down, partition-based, complete
  - Incognito, LeFevre, 2005 – Global, bottom-up, hierarchy-based, complete
  - Mondrian, LeFevre, 2006 – Local, top-down, partition-based, greedy
μ-Argus

- Hundpool and Willenborg, 1996
- Greedy approach
- Global generalization with tuple suppression
- Not guaranteeing k-anonymity
μ-Argus

Input: Private Table $PT$; quasi-identifier $QI = (A_1, \ldots, A_n)$, disjoint subsets of $QI$ known as $Identifying$, $More$, and $Most$ where $QI = Identifying \cup More \cup Most$, $k$ constraint; domain generalization hierarchies $DGH_{Ai}$, where $i=1,\ldots,n$.

Output: $MT$ containing a generalization of $PT[QI]$

Assumes: $|PT| \geq k$

Method:
1. $freq \leftarrow$ a frequency list containing distinct sequences of values of $PT[QI]$, along with the number of occurrences of each sequence.
2. Generalize each $A_i \in QI$ in $freq$ until its assigned values satisfy $k$.
3. Test 2- and 3- combinations of $Identifying$, $More$ and $Most$ and let outliers store those cell combinations not having $k$ occurrences.
4. Data holder decides whether to generalize an $A_i \in QI$ based on outliers and if so, identifies the $A_i$ to generalize. $freq$ contains the generalized result.
5. Repeat steps 3 and 4 until the data holder no longer elects to generalize.
6. Automatically suppress a value having a combination in outliers, where precedence is given to the value occurring in the most number of combinations of outliers.

μ-Argus algorithm
Figure 13 Most x More combination test and resulting freq

Figure 14 freq before suppression

Figure 15 Results from the μ-Argus algorithm and from the program
Problems With $\mu$-Argus

1. Only 2- and 3- combinations are examined, there may exist 4 combinations that are unique – may not always satisfy k-anonymity

2. Enforce generalization at the attribute level (global) – may over generalize
The Datafly System

- Sweeney, 1997
- Greedy approach
- Global generalization with tuple suppression
Datafly Algorithm

Input: Private Table $PT$; quasi-identifier $Ql = (A_1, \ldots, A_n)$, $k$ constraint; hierarchies $DGH_{Ai}$, where $i=1,\ldots,n$.
Output: MGT, a generalization of $PT[Ql]$ with respect to $k$
Assumes: $|PT| \geq k$
Method:
1. $freq \leftarrow$ a frequency list contains distinct sequences of values of $PT[Ql]$, along with the number of occurrences of each sequence.
2. while there exists sequences in $freq$ occurring less than $k$ times that account for more than $k$ tuples do
   2.1. let $A_j$ be attribute in $freq$ having the most number of distinct values
   2.2. $freq \leftarrow$ generalize the values of $A_j$ in $freq$
3. $freq \leftarrow$ suppress sequences in $freq$ occurring less than $k$ times.
4. $freq \leftarrow$ enforce $k$ requirement on suppressed tuples in $freq$.
5. Return $MGT \leftarrow$ construct table from $freq$
Datafly

Figure 9 Intermediate stages of the core Datafly algorithm

MGT resulting from Datafly, $k=2$, $QI=\{\text{Race, Birthdate, Gender, ZIP}\}$
Problems With Datafly

1. Generalizing all values associated with an attribute (global)
2. Suppressing all values within a tuple (global)
3. Selecting the attribute with the greatest number of distinct values as the one to generalize first – computationally efficient but may over generalize
Generalization algorithms

- Early systems
  - µ-Argus, Hundpool, 1996 - Global, bottom-up, greedy
  - Datafly, Sweeney, 1997 - Global, bottom-up, greedy

- k-anonymity algorithms
  - AllMin, Samarati, 2001 - Global, bottom-up, complete, impractical
  - MinGen, Sweeney, 2002 - Global, bottom-up, complete, impractical
  - Bottom-up generalization, Wang, 2004 – Global, bottom-up, greedy
  - TDS (Top-Down Specialization), Fung, 2005 - Global, top-down, greedy
  - K-OPTIMIZE, Bayardo, 2005 – Global, top-down, partition-based, complete
  - Incognito, LeFevre, 2005 – Global, bottom-up, hierarchy-based, complete
  - Mondrian, LeFevre, 2006 – Local, top-down, partition-based, greedy
K-OPTIMIZE

- Practical solution to guarantee optimality
- Main techniques
  - Framing the problem into a set-enumeration search problem
  - Tree-search strategy with cost-based pruning and dynamic search rearrangement
  - Data management strategies
Anonymization Strategies

- **Local suppression**
  - Delete individual attribute values
  - E.g. <Age=50, Gender=M, State=CA>

- **Global attribute generalization**
  - Replace specific values with more general ones for an attribute
  - Numeric data: partitioning of the attribute domain into intervals. E.g. Age={[1-10],...,[91-100]}
  - Categorical data: generalization hierarchy supplied by users. E.g. Gender = [M or F]
K-Anonymization with Suppression

- K-anonymization with suppression
  - Global attribute generalization with local suppression of outlier tuples.

- Terminologies
  - Dataset: D
  - Anonymization: \{a_1, \ldots, a_m\}

Equivalent classes: \(E \{ v_{1,1}, \ldots, v_{1,m}, \ldots, v_{1,n}, \ldots, v_{n,m} \} \)
Finding Optimal Anonymization

- Optimal anonymization determined by a cost metric
- Cost metrics
  - Discernibility metric: penalty for non-suppressed tuples and suppressed tuples
    \[ C_{DM}(g, k) = \sum_{E \text{ s.t. } |E| \geq k} |E|^2 + \sum_{E \text{ s.t. } |E| < k} |D||E| \]
  - Classification metric
Modeling Anonymizations

- Assume a total order over the set of all attribute domain

- Set representation for anonymization
  - E.g. Age: \([10-29],[30-49]\), Gender: \([M\text{ or } F]\), Marital Status: \([\text{Married}],[\text{Widowed or Divorced}],[\text{Never Married}]\)
  - \([1,2,4,6,7,9] \rightarrow [2,7,9]\)

- Power set representation for entire anonymization space
  - Power set of \([2,3,5,7,8,9]\) - order of \(2^n!\)
  - \(\{\}\) – most general anonymization
  - \([2,3,5,7,8,9]\) – most specific anonymization
Optimal Anonymization Problem

Goal
- Find the best anonymization in the powerset with lowest cost

Algorithm
- set enumeration search through tree expansion - size $2^n$
- Top-down depth first search

Heuristics
- Cost-based pruning
- Dynamic tree rearrangement

Set enumeration tree over powerset of \{1,2,3,4\}
Node Pruning through Cost Bounding

- **Intuitive idea**
  - prune a node $H$ if none of its descendents can be optimal

- **Cost lower-bound of subtree of $H$**
  - Cost of suppressed tuples bounded by $H$
  - Cost of non-suppressed tuples bounded by $A$

$$L_{BM}(H, A) = \sum_{t \in D} \begin{cases} |D| & \text{when } t \text{ is suppressed by } H, \\ \max(|A_t|, k) & \text{otherwise.} \end{cases}$$
Useless Value Pruning

- Intuitive idea
  - Prune useless values that have no hope of improving cost

- Useless values
  - Only split equivalence classes into suppressed equivalence classes (size < k)
Tree Rearrangement

- Intuitive idea
  - Dynamically reorder tree to increase pruning opportunities

- Heuristics
  - sort the values based on the number of equivalence classes induced
Experiments

- Adult census dataset
  - 30k records and 9 attributes
  - Fine: powerset of size $2^{160}$
- Evaluations of performance and optimal cost
- Comparison with greedy/stochastic method
  - 2-phase greedy generalization/specialization
  - Repeated process
Results – Comparison

- None of the other optimal algorithms can handle the census data
- Greedy approaches, while executing quickly, produce highly sub-optimal anonymizations
- Comparison with 2-phase method (greedy + stochastic)
Comments

- Interesting things to think about
  - Domains without hierarchy or total order restrictions
  - Other cost metrics
  - Global generalization vs. local generalization
Generalization algorithms

- Early systems
  - μ-Argus, Hundpool, 1996 - Global, bottom-up, greedy
  - Datafly, Sweeney, 1997 - Global, bottom-up, greedy

- k-anonymity algorithms
  - AllMin, Samarati, 2001 - Global, bottom-up, complete, impractical
  - MinGen, Sweeney, 2002 - Global, bottom-up, complete, impractical
  - Bottom-up generalization, Wang, 2004 – Global, bottom-up, greedy
  - TDS (Top-Down Specialization), Fung, 2005 - Global, top-down, greedy
  - K-OPTIMIZE, Bayardo, 2005 – Global, top-down, partition-based, complete
  - Incognito, LeFevre, 2005 – Global, bottom-up, hierarchy-based, complete
  - Mondrian, LeFevre, 2006 – Local, top-down, partition-based, greedy
Mondrian

- Top-down partitioning
- Greedy
- Local (multidimensional) – tuple/cell level
Global Recoding

- Mapping domains of quasi-identifiers to generalized or altered values using a *single* function

**Notation**
- $D_x$ is the domain of attribute $X_i$ in table $T$

**Single Dimensional**
- $\phi_i : D_{x_i} \rightarrow D'$ for each attribute $X_i$ of the quasi-identifier
- $\phi_i$ applied to values of $X_i$ in tuple of $T$
Local Recoding

- Multi-Dimensional
  - Recode domain of value vectors from a set of quasi-identifier attributes
  - \( \varphi : D_{x_1} \times \ldots \times D_{x_n} \rightarrow D' \)
  - \( \varphi \) applied to vector of quasi-identifier attributes in each tuple in \( T \)
Partitioning

- Single Dimensional
  - For each $X_i$, define non-overlapping single dimensional intervals that covers $D_{x_i}$
  - Use $\phi_i$ to map $x \in D_x$ to a summary stat

- Strict Multi-Dimensional
  - Define non-overlapping multi-dimensional intervals that covers $D_{x_1} \ldots D_{x_d}$
  - Use $\phi$ to map $(x_{x_1} \ldots x_{x_d}) \in D_{x_1} \ldots D_{x_d}$ to a summary stat for its region
Global Recoding Example

### k = 2

**Quasi Identifiers**

- Age, Sex, Zipcode

### Patient Data

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>25</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitis</td>
</tr>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>27</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>

### Single Dimensional

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>[25-28]</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>[25-28]</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
</tbody>
</table>

### Multi-Dimensional

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>[25-26]</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>[25-27]</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>[25-26]</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitis</td>
</tr>
<tr>
<td>[25-27]</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
</tbody>
</table>
Global Recoding Example 2

$k = 2$
Quasi Identifiers
Age, Zipcode

Patient Data  Single Dimensional  Multi-Dimensional
Greedy Partitioning Algorithm

Problem
- Need an algorithm to find multi-dimensional partitions
- Optimal k-anonymous strict multi-dimensional partitioning is NP-hard

Solution
- Use a greedy algorithm
- Based on k-d trees
- Complexity $O(n \log n)$
Greedy Partitioning Algorithm

Anonymize(partition)
    if (no allowable multidimensional cut for partition)
        return \( \phi : \text{partition} \rightarrow \text{summary} \)
    else
        \( dim \leftarrow \text{choose dimension}() \)
        \( fs \leftarrow \text{frequency set}(\text{partition}, dim) \)
        \( \text{splitVal} \leftarrow \text{find median}(fs) \)
        \( \text{lhs} \leftarrow \{ t \in \text{partition} : t.\text{dim} \leq \text{splitVal} \} \)
        \( \text{rhs} \leftarrow \{ t \in \text{partition} : t.\text{dim} > \text{splitVal} \} \)
        return Anonymize(rhs) \cup \text{Anonymize(lhs)} \)
Algorithm Example

- $k = 2$
- Dimension determined heuristically
- Quasi-identifiers
  - Zipcode
  - Age

Patient Data

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>25</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Brochitis</td>
</tr>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>27</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>

Anonymized Data

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>Zipcode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>[25-26]</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>[25-26]</td>
<td>Male</td>
<td>53711</td>
<td>Brochitis</td>
</tr>
<tr>
<td>[25-27]</td>
<td>Female</td>
<td>53712</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>[25-27]</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
</tbody>
</table>
Algorithm Example

Iteration # 1 (full table)

partition

LHS

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>25</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitus</td>
</tr>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>27</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>

RHS

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>27</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
</tbody>
</table>

dim = Zipcode

fs

splitVal = 53711
Algorithm Example continued

Iteration # 2 (LHS from iteration # 1)

**partition**

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitis</td>
</tr>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>

**dim = Age**

**fs**

<table>
<thead>
<tr>
<th>Age</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>26</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
</tr>
</tbody>
</table>

**splitVal = 26**

**LHS**

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitis</td>
</tr>
</tbody>
</table>

**RHS**

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>
Algorithm Example  continued

Iteration # 3 (LHS from iteration # 2)

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Male</td>
<td>53711</td>
<td>Flu</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>53711</td>
<td>Bronchitus</td>
</tr>
</tbody>
</table>

Summary: Age = [25-26]    Zip= [53711]

Iteration # 4 (RHS from iteration # 2)

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>Male</td>
<td>53710</td>
<td>Broken Arm</td>
</tr>
<tr>
<td>28</td>
<td>Male</td>
<td>53711</td>
<td>Hang Nail</td>
</tr>
</tbody>
</table>

Summary: Age = [27-28]    Zip= [53710 - 53711]
Algorithm Example  continued

Iteration # 5 (RHS from iteration # 1)

partition

<table>
<thead>
<tr>
<th>Age</th>
<th>Sex</th>
<th>ZipCode</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Female</td>
<td>53712</td>
<td>Hepatitis</td>
</tr>
<tr>
<td>27</td>
<td>Female</td>
<td>53712</td>
<td>AIDS</td>
</tr>
</tbody>
</table>

No Allowable Cut

Summary: Age = [25-27]  Zip= [53712]
Experiment

- Adult dataset
- Data quality metric (cost metric)
  - Discernability Metric ($C_{DM}$)
    - $C_{DM} = \sum_{\text{EquivalentClasses } E} |E|^2$
    - Assign a penalty to each tuple
  - Normalized Avg. Eqiv. Class Size Metric ($C_{AVG}$)
    - $C_{AVG} = (\text{total}_\text{records}/\text{total}_\text{equiv_classes})/k$
Comparison results

- Full-domain method: Incognito
- Single-dimensional method: K-OPTIMIZE

Figure 10. Quality comparison for Adults database using discernability metric
Data partitioning comparison

(a) Optimal single-dimensional partitioning

(b) Greedy strict multidimensional partitioning
Mondrian