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

Z2 = {410**}

↑

Z1 = {4107*. 4109*}

↑

Z0 = {41075, 41076, 41095, 41099}

S1 = {Person}

↑

S0 = {Male, Female}

#	Zip	Age	Nationality	Condition
1	41076	< 40	*	Heart Disease
2	48202	< 40	*	Heart Disease
3	41076	< 40	*	Cancer
4	48202	< 40	*	Cancer

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

A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In PODS'04.

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)

K. LeFerve, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient Full-Domain K-Anonymity. In SIGMOD 05

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* ∪ *More* ∪ *Most*, k constraint; domain generalization hierarchies DGH_{Ai}, where i=1,…,n.

Output: MT containing a generalization of PT[QI]

Assumes: |PT| ≥ k

Method:

1. **freq** ← 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 $A_j \in$ QI based on *outliers* and if so, identifies the A_j 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

µ-Argus

Birth	ZIP	*occurs*	*sid*	*outliers*
1965	02141	2	{t1,t2}	{}
1965	02138	2	{t3,t4}	{}
1964	02138	3	{t5,t6,t7}	{}
1965	02139	1	{t8}	{}
1964	02139	2	{t9,t10}	{}
1967	02138	2	{t11,t12}	{}

V

Race	Birth	Sex	ZIP	*occurs*	*sid*	*outliers*
black	1965	male	02141	2	{t1,t2}	{}
black	1965	female	02138	2	{t3,t4}	{}
black	1964	female	02138	2	{t5,t6}	{}
white	1964	male	02138	1	{t7}	{}
white	1965	female	02139	1	{t8}	{{birth,zip}}
white	1964	male	02139	2	{t9,t10}	{}
white	1967	male	02138	2	{t11,t12}	{}

freq

Figure 13 *Most* × *More* combination test and resulting freq

Race	Birth	Sex	ZIP	*occurs*	*sid*	*outliers*
black	1965	male	02141	2	{t1,t2}	{}
black	1965	female	02138	2	{t3,t4}	{}
black	1964	female	02138	2	{t5,t6}	{}
white	1964	male	02138	1	{t7}	{{**birth**,sex,zip}, {race,**birth**,zip}}
white	1965	female	02139	1	{t8}	{{**birth**,zip}, {**sex**,zip}, {**birth**,**sex**,zip}, {race,**birth**,**sex**}, {race,**birth**,zip}, {race,**sex**}, {race,**birth**}}
white	1964	male	02139	2	{t9,t10}	{}
white	1967	male	02138	2	{t11,t12}	{}

Figure 14 freq before suppression

id	Race	BirthDate	Gender	ZIP
t1	black	1965	male	02141
t2	black	1965	male	02141
t3	black	1965	female	02138
t4	black	1965	female	02138
t5	black	1964	female	02138
t6	black	1964	female	02138
t7	white		male	02138
t8	white			02139
t9	white	1964	male	02139
t10	white	1964	male	02139
t11	white	1967	male	02138
t12	white	1967	male	02138

MT

id	Race	BirthDate	Gender	ZIP
t1	black	1965	male	02141
t2	black	1965	male	02141
t3	black	1965	female	02138
t4	black	1965	female	02138
t5	black	1964	female	02138
t6	black	1964	female	02138
t7	white	1964	male	02138
t8	white		female	02139
t9	white	1964	male	02139
t10	white	1964	male	02139
t11	white	1967	male	02138
t12	white	1967	male	02138

MT actual

Figure 15 Results from the µ-Argus algorithm and from the program

Problems With μ-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 QI = $(A_1, \ldots, A_n)$,
k constraint; hierarchies DGH_{Ai}, where i=1,…,n.
Output: MGT, a generalization of PT[QI] with respect to k
Assumes: |PT |≥ k
Method:

1. freq ← a frequency list contains distinct sequences of values of PT[QI], 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 ← generalize the values of A_j in freq
3. freq ← suppress sequences in freq occurring less than k times.
4. freq ← enforce k requirement on suppressed tuples in freq.
5. **Return** MGT ← construct table from freq

Core Datafly Algorithm

Datafly

Race	BirthDate	Gender	ZIP	#occurs	
black	9/20/65	male	02141	1	t1
black	2/14/65	male	02141	1	t2
black	10/23/65	female	02138	1	t3
black	8/24/65	female	02138	1	t4
black	11/7/64	female	02138	1	t5
black	12/1/64	female	02138	1	t6
white	10/23/64	male	02138	1	t7
white	3/15/65	female	02139	1	t8
white	8/13/64	male	02139	1	t9
white	5/5/64	male	02139	1	t10
white	2/13/67	male	02138	1	t11
white	3/21/67	male	02138	1	t12
2	12	2	3		

A

Race	BirthDate	Gender	ZIP	#occurs	
black	1965	male	02141	2	t1,t2
black	1965	female	02138	2	t3, t4
black	1964	female	02138	2	t5, t6
white	1964	male	02138	1	t7
white	1965	female	02139	1	t8
white	1964	male	02139	2	t9, t10
white	1967	male	02138	2	t11, t12
2	3	2	3		

B

Figure 9 Intermediate stages of the core Datafly algorithm

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
black	1965	female	02138	painful eye
black	1965	female	02138	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1964	male	02139	obesity
white	1964	male	02139	fever
white	1967	male	02138	vomiting
white	1967	male	02138	back pain

MGT resulting from Datafly, *k*=2, QI={*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

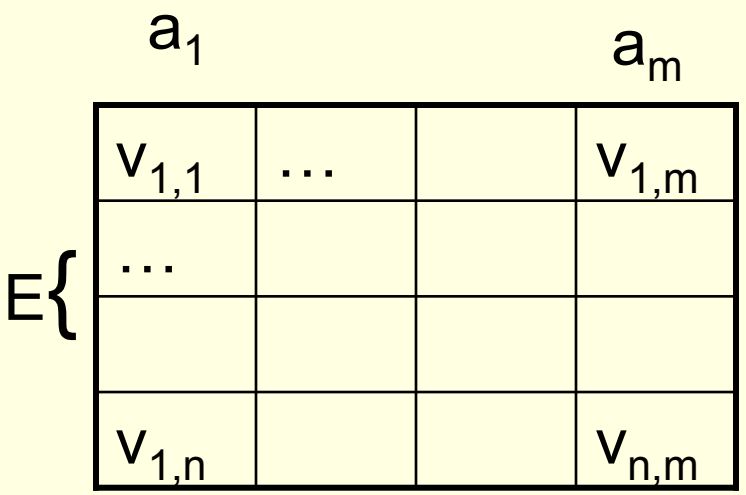
- Practical solution to guarantee optimality
- Main techniques
  - Framing the problem into a set-enumeration search problem
  - Tree-search strategy with cost-based pruning and dynamic search rearrangement
  - Data management strategies

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, …, a_m}
  - Equivalent classes: E



Finding Optimal Anonymization

- Optimal anonymization determined by a cost metric
- Cost metrics
  - Discernibility metric: penalty for non-suppressed tuples and suppressed tuples

$$C_{\mathrm{DM}}(g,k) = \sum_{\forall E \text{ s.t. } |E| \geq k} |E|^2 + \sum_{\forall E \text{ s.t. } |E| < k} |D||E|$$

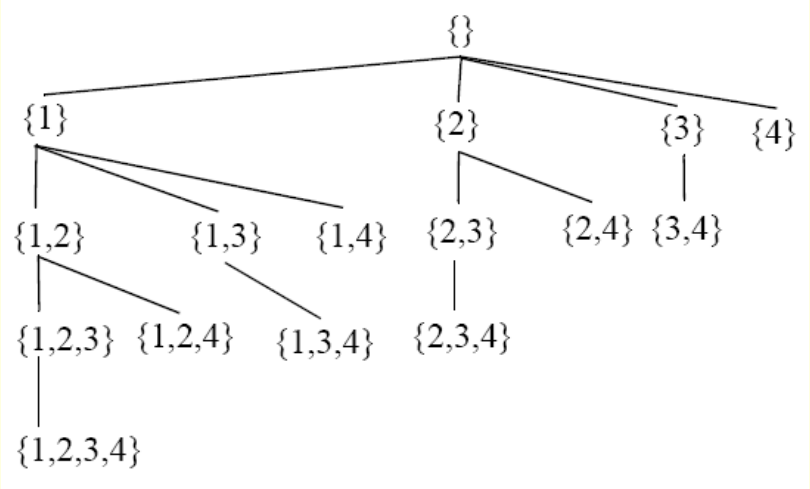
  - Classification metric

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 or F]>, Marital Status: <[Married], [Widowed or Divorced], [Never Married]>
  - {1, 2, 4, 6, 7, 9} -> {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

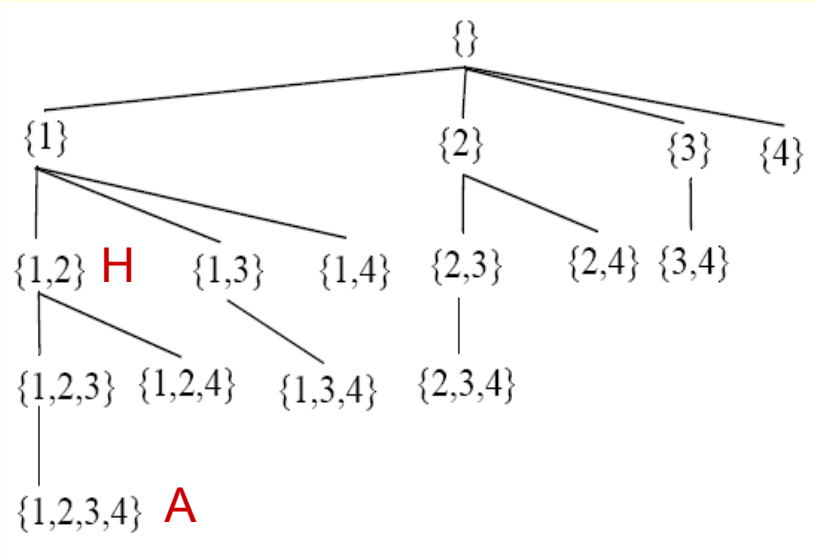
- Goal
  - Find the best anonymization in the powerset with lowest cost
- Algorithm
  - set enumeration search through tree expansion - size 2^n
  - Top-down depth first search
- Heuristics
  - Cost-based pruning
  - Dynamic tree rearrangement



Set enumeration tree over powerset of {1,2,3,4}

Node Pruning through Cost Bounding

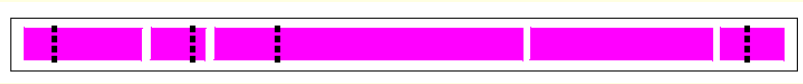
- Intuitive idea
  - prune a node H if none of its descendents can be optimal
- Cost lower-bound of subtree of H
  - Cost of suppressed tuples bounded by H
  - Cost of non-suppressed tuples bounded by A

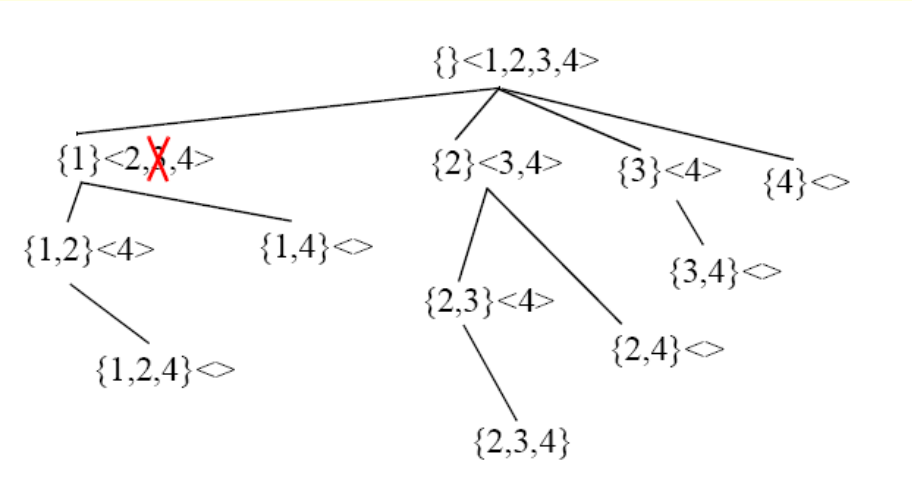


$$LB_{\mathrm{DM}}(H, A) = \sum_{\forall t \in D} \begin{cases} |D| & \text{when } t \text{ is suppressed by } H, \\ \max(|E_{A,t}|, k) & \text{otherwise.} \end{cases}$$

Useless Value Pruning

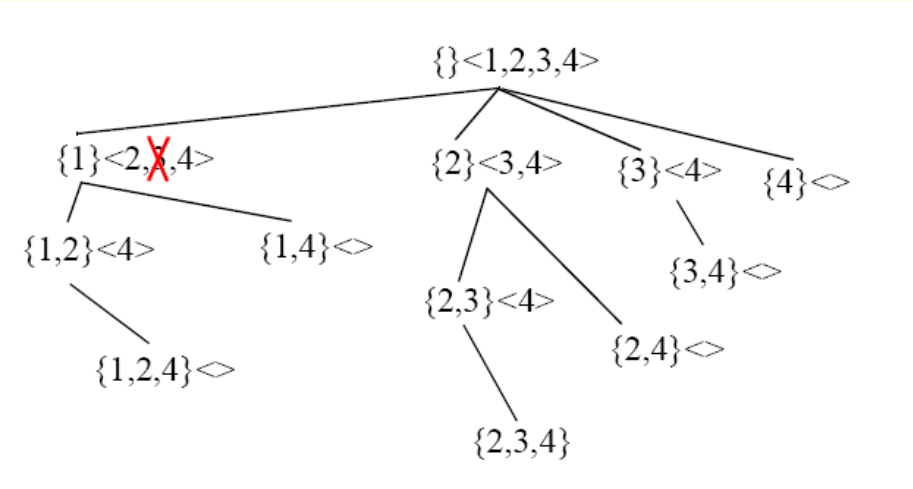
- Intuitive idea
  - Prune useless values that have no hope of improving cost
- Useless values
  - Only split equivalence classes into suppressed equivalence classes (size < k)





Tree Rearrangement

- Intuitive idea
  - Dynamically reorder tree to increase pruning opportunities
- Heuristics
  - sort the values based on the number of equivalence classes induced

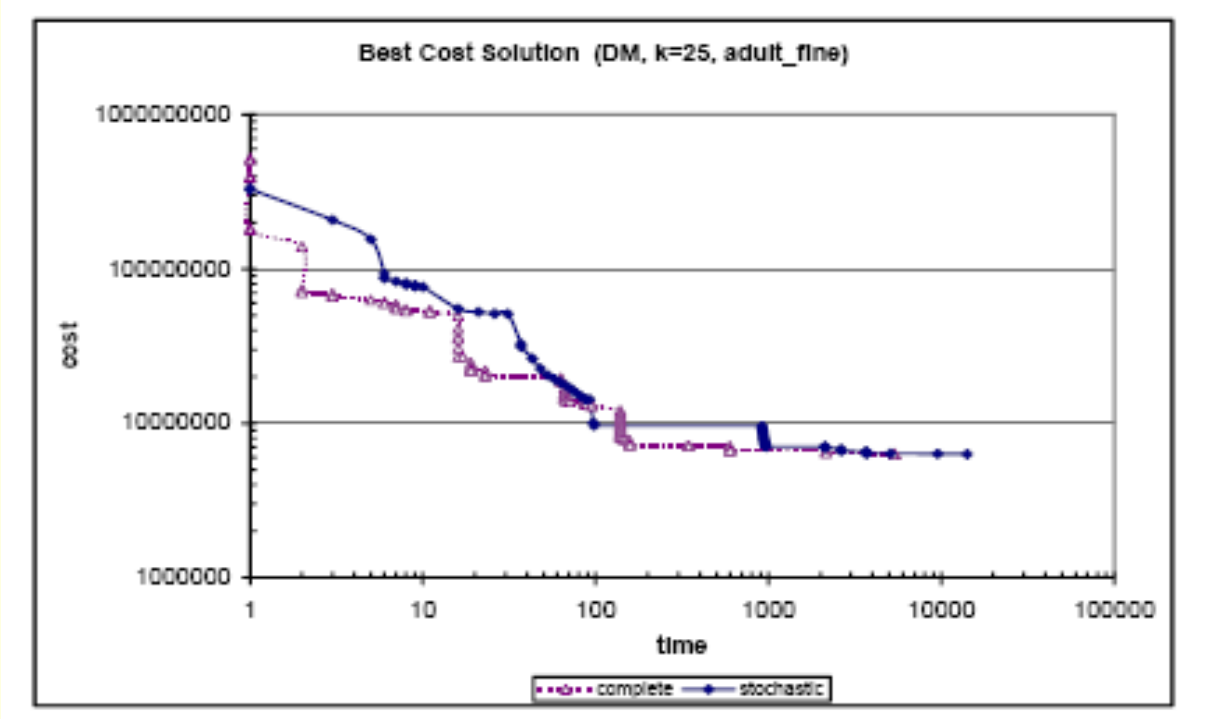


Experiments

- 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
  - $\varphi_i : D_{xi} \rightarrow D'$ for each attribute X_i of the quasi-id
  - φ_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
  - $\varphi : D_{x1} \times \ldots \times D_{xn} \rightarrow D'$
  - φ 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_{xi}
  - Use φ_i to map $x \varepsilon D_x$ to a summary stat
- Strict Multi-Dimensional
  - Define non-overlapping multi-dimensional intervals that covers $D_{x1} \ldots D_{xd}$
  - Use φ to map $(x_{x1} \ldots x_{xd}) \varepsilon D_{x1} \ldots D_{xd}$ to a summary stat for its region

Global Recoding Example

Patient Data

Age	Sex	Zipcode	Disease
25	Male	53711	Flu
25	Female	53712	Hepatitis
26	Male	53711	Brochitis
27	Male	53710	Broken Arm
27	Female	53712	AIDS
28	Male	53711	Hang Nail

k = 2

Quasi Identifiers

Age, Sex, Zipcode

Single Dimensional

Age	Sex	Zipcode	Disease
[25-28]	Male	[53710-53711]	Flu
[25-28]	Female	53712	Hepatitis
[25-28]	Male	[53710-53711]	Brochitis
[25-28]	Male	[53710-53711]	Broken Arm
[25-28]	Female	53712	AIDS
[25-28]	Male	[53710-53711]	Hang Nail

Partitions

Age : {[25-28]}

Sex: {Male, Female}

Zip : {[53710-53711], 53712}

Multi-Dimensional

Age	Sex	Zipcode	Disease
[25-26]	Male	53711	Flu
[25-27]	Female	53712	Hepatitis
[25-26]	Male	53711	Brochitis
[27-28]	Male	[53710-53711]	Broken Arm
[25-27]	Female	53712	AIDS
[27-28]	Male	[53710-53711]	Hang Nail

Partitions

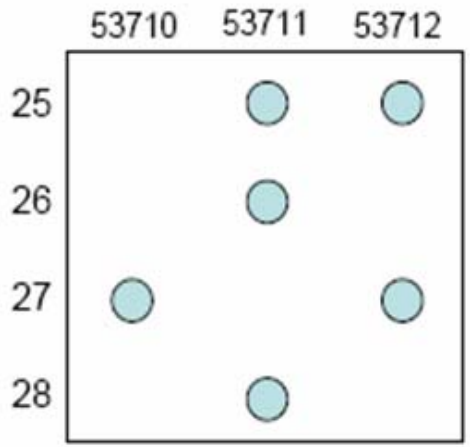
{Age: [25-26],Sex: Male, Zip: 53711}

{Age: [25-27],Sex: Female, Zip: 53712}

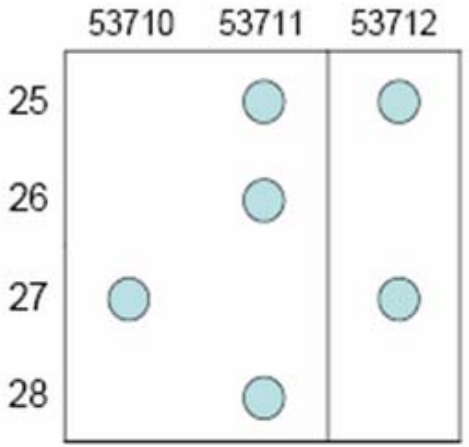
{Age: [27-28],Sex: Male, Zip: [53710-53711]}

Global Recoding Example 2

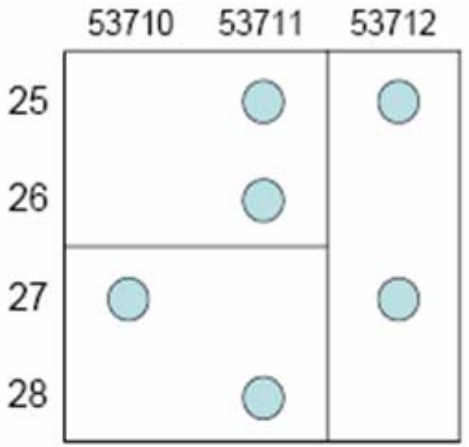
k = 2
Quasi Identifiers
    Age, Zipcode



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(*nlogn*)

Greedy Partitioning Algorithm

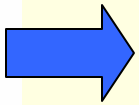
Anonymize($partition$)
  if (no allowable multidimensional cut for $partition$)
    return $\phi : partition \rightarrow summary$
  else
    $dim \leftarrow$ choose_dimension()
    $fs \leftarrow$ frequency_set($partition$, dim)
    $splitVal \leftarrow$ find_median(fs)
    $lhs \leftarrow \{t \in partition : t.dim \leq splitVal\}$
    $rhs \leftarrow \{t \in partition : t.dim > splitVal\}$
    return Anonymize(rhs) $\cup$ Anonymize(lhs)

Algorithm Example

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

Age	Sex	Zipcode	Disease
25	Male	53711	Flu
25	Female	53712	Hepatitis
26	Male	53711	Brochitis
27	Male	53710	Broken Arm
27	Female	53712	AIDS
28	Male	53711	Hang Nail

Patient Data



Age	Sex	Zipcode	Disease
[25-26]	Male	53711	Flu
[25-27]	Female	53712	Hepatitis
[25-26]	Male	53711	Brochitis
[27-28]	Male	[53710-53711]	Broken Arm
[25-27]	Female	53712	AIDS
[27-28]	Male	[53710-53711]	Hang Nail

Anonymized Data

Algorithm Example

Iteration # 1 (full table)

partition

Age	Sex	ZipCode	Disease
25	Male	53711	Flu
25	Female	53712	Hepatitus
26	Male	53711	Bronchitus
27	Male	53710	Broken Arm
27	Female	53712	AIDS
28	Male	53711	Hang Nail

dim = Zipcode

fs

Zipcode	Count
53710	1
53711	3
53712	2

splitVal = 53711

LHS

Age	Sex	ZipCode	Disease
25	Male	53711	Flu
26	Male	53711	Bronchitus
27	Male	53710	Broken Arm
28	Male	53711	Hang Nail

RHS

Age	Sex	ZipCode	Disease
25	Female	53712	Hepatitus
27	Female	53712	AIDS

Algorithm Example continued

Iteration # 2 (LHS from iteration # 1)

partition

Age	Sex	ZipCode	Disease
25	Male	53711	Flu
26	Male	53711	Bronchitus
27	Male	53710	Broken Arm
28	Male	53711	Hang Nail

dim = Age

fs

Age	Count
25	1
26	1
27	1
28	1

splitVal = 26

LHS

Age	Sex	ZipCode	Disease
25	Male	53711	Flu
26	Male	53711	Bronchitus

RHS

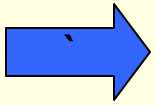
Age	Sex	ZipCode	Disease
27	Male	53710	Broken Arm
28	Male	53711	Hang Nail

Algorithm Example continued

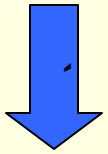
Iteration # 3 (LHS from iteration # 2)

partition

Age	Sex	ZipCode	Disease
25	Male	53711	Flu
26	Male	53711	Bronchitus



No Allowable Cut

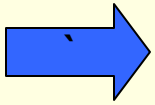


Summary: Age = [25-26] Zip= [53711]

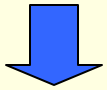
Iteration # 4 (RHS from iteration # 2)

partition

Age	Sex	ZipCode	Disease
27	Male	53710	Broken Arm
28	Male	53711	Hang Nail



No Allowable Cut



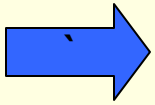
Summary: Age = [27-28] Zip= [53710 - 53711]

Algorithm Example continued

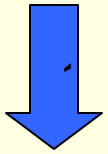
Iteration # 5 (RHS from iteration # 1)

partition

Age	Sex	ZipCode	Disease
25	Female	53712	Hepatitus
27	Female	53712	AIDS



No Allowable Cut



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

Experiment

- Adult dataset
- Data quality metric (cost metric)
  - Discernability Metric (C_{DM})
    - $C_{DM} = \Sigma_{EquivalentClasses\ E} |E|^2$
    - Assign a penalty to each tuple
  - Normalized Avg. Eqiv. Class Size Metric (C_{AVG})
    - C_{AVG} = (total_records/total_equiv_classes)/k

Comparison results

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

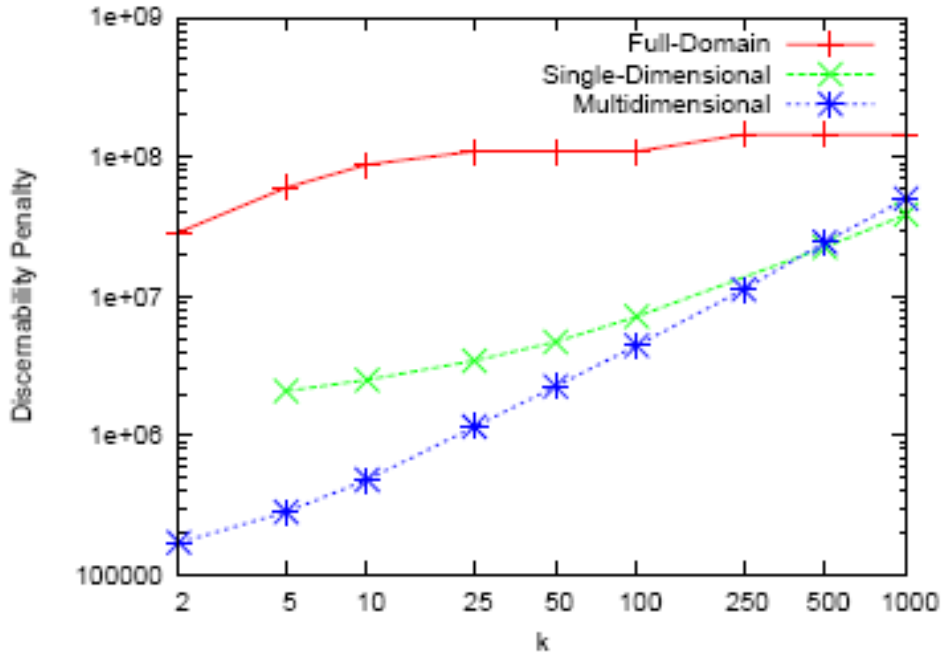
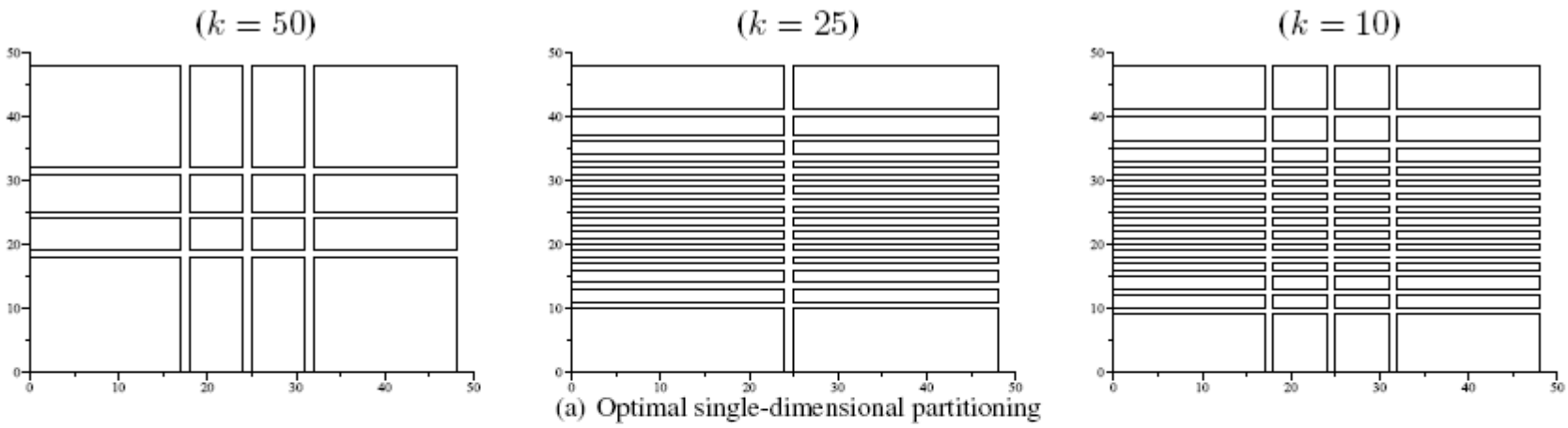
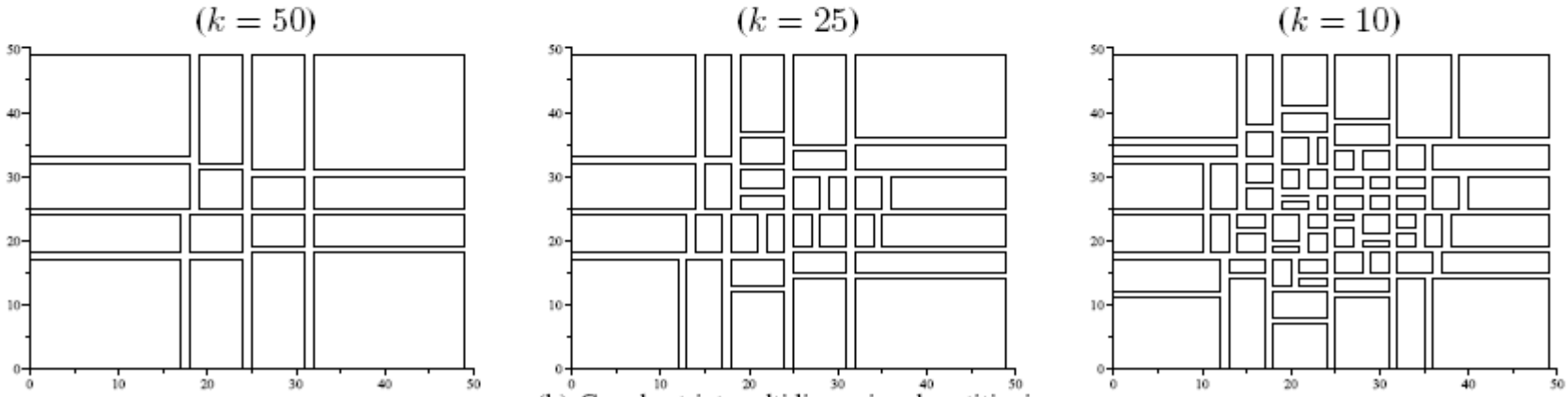


Figure 10. Quality comparison for Adults database using discernability metric

Data partitioning comparison



(a) Optimal single-dimensional partitioning



(b) Greedy strict multidimensional partitioning

Mondrian