# Indexing Uncertain Categorical Data

Sarvjeet Singh etc.
Presented by: Yi Zhang

## The problem

- Uncertainty in discrete-domain data
  - E.g. Nurse 10 in Room {3,5} at 10:05
  - Results from text classifiers
- Existing relational DB doesn't handle well
  - Allowing multiple values? app use own complex model to manage uncertainty
  - Just keep most likely value? lossy
  - Need proper index structure for categorical uncertain data



## Contributions

- Data models for uncertainty
- Semantics of queries
- Index structures
- Evaluation of index structures



## Data Model

- UDA: uncertain discrete attribute
  - A distribution over its discrete domain
  - Example

| Employee | Department |
|---|---|
| Jim | {(Shoes, 0.5),(Sales, 0.5)} |
| Tom | {(Sales, 0.4), (Clothes, 0.6)} |
| Lin | {(Hardware, 0.6), (Sales, 0.4) } |
| Nancy | {(HR, 1.0)} |

  - UDA u can be represented as prob. vector <p1,...,pN> s.t. Pr(u=di)=pi
  - For UDAs u and v $Pr(u = v) = \sum_{i=1}^{N} u.p_i \times v.p_i$



## Distribution Similarity Metrics

- Given two UDAs, how similar are they?
- Useful for clustering in index structures
- Manhattan distance

  $L_1$: $L_1(u, v) = \sum_{i=1}^{N} |u.p_i - v.p_i|$
- Euclidean distance

  $L_2$: $L_2(u, v) = \sqrt{\sum_{i=1}^{N} (u.p_i - v.p_i)^2}$
- Kullback-Leibler divergence

  $KL(u,v)$: $KL(u, v) = \sum_{i=1}^{N} u.p_i \log(u.p_i / v.p_i)$

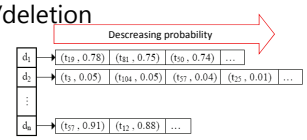


## Queries

- Equality queries
  - PEQ: given UDA q, return all tuples t from R, s.t. Pr(q=t.a)≥0
  - PETQ: same as PEQ, except Pr(q=t.a) ≥τ (constant)
  - PEQ-top-k: return k tuples w/ highest equality prob.
- Distributional queries
  - DSTQ: given UDA q, divergence function F, return all tuples t from R s.t. F(q,t.a)≤ τ
  - DSQ-top-k
- Equality join queries
  - PETJ: $R \bowtie_{R_a = S_b, \tau} S$ consists all pairs of tuples r, s from R, S s.t. Pr(r.a=s.b) ≥τ
  - PEJ-top-k

## Inverted Index

- Outer list: for each domain value di
  - Inner list: list of <t,p> where t.di=p
- Rank aggregation…
- Simple insertion/deletion
- What about searching?






## Search on Inverted Index

- PETQ queries: given UDA q and const τ
- Inv-index-search
  - I/O significant
- Highest-prob-first
- Row pruning
- Column pruning
  - Needs random I/O

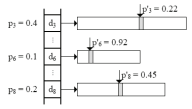
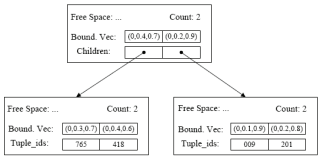



**Figure 2. Highest-prob-first Search for** $q = \langle (d_3, 0.4), (d_8, 0.2), (d_6, 0.1) \rangle$.



## PDR-tree

- Consider UDA as point in high-dim space
- Similar UDAs stored in a page
- Page described by MBR
- Parent's MBR covers all its children






## Operations on PDR-trees

- Insert(u)
  - To insert into current page, update MBR; then choose best child page and insert.
    - Min area increase
    - Most similar MBR
  - Start from root and recurse
- Split()
  - Top-down: pick two children MBRs as cluster centers and all other UDAs go to a closer cluster
  - Bottom-up: each UDA forms an independent cluster and proceed by merging closest pair of clusters until two remains



## Queries on PDR-trees

- PETQ(q, τ)
  - Depth-first search
  - Enters an node c if qualifies c.mbr·q ≥τ; prune otherwise
  - At leaf level, check each UDA and output
  - For top-k queries, update threshold dynamically during search
  - Doesn't work with distribution similarity queries…



## Experimental Setup

- Datasets
  - Real
    - CRM databases, 100k text doc, 50 categories
      - CRM1: prob values generated by auto categorization
      - CRM2: unsupervised fuzzy clustering
  - Synthetic
    - Uniform: 5 categories, prob. random chosen, 10k tuples
    - Pairwise: 5 categories, each tuple has 2 non-zero categories w/ roughly equal prob., 10k tuples
    - Gen3: for each tuple, size of non-zero categories follows geometric dist.
- Page size 8KB, 100 blocks for buffering
- PETQ and PEQ-top-k queries
- Measure # of I/O operations

## Divergence Measures

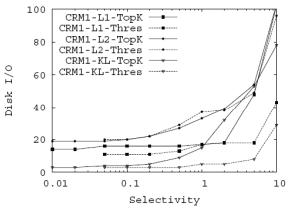



Figure 4. L1 vs L2 vs KL (PDR-tree)

- KL: if hit an MBR, most UDAs will qualify cause they are similar in dist.
- Top-k: explore more tuples



## Inverted Index vs. PDR-tree

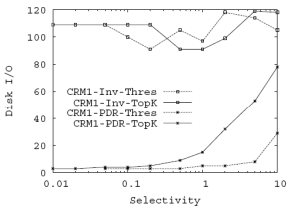



Figure 6. Inverted Index vs PDR-tree (CRM1)

- Non-zero probability in many categories
- Large number of lists accessed in inverted index



## Data/Domain Size

- Inverted Index
  - Reduction in avg length of each list as # of lists increase
- PDR-tree
  - Data generation related
  - Non-zero entries at both ends of experimental space smaller than in the middle

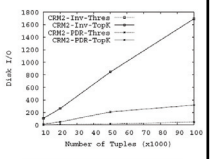



Figure 8. Scalability with Size of Data

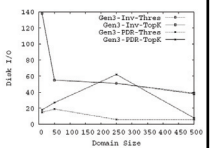



Figure 9. Scalability with Domain Size



## Discussion

- How to use the mentioned criteria to insert a UDA into a PDR-tree
- No motivation for MBR-based approach used in PDR-Tree
- Not obvious how to extend PDR-Tree to work for Distributional Similarity Queries