Repairing Dimension Hierarchies under Inconsistent Reclassification

Mónica Caniupán [1] Alejandro Vaisman [2]

[1] Universidad del Bío-Bío - Chile

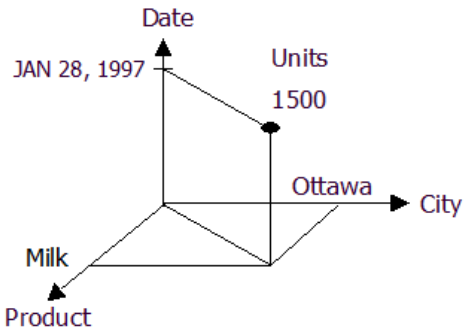
[2] Université Libre de Bruxelles - Belgium

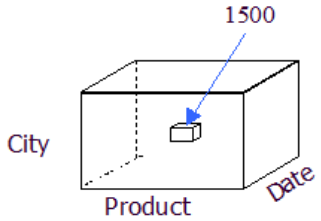
Brussels, Belgium, October 2011

Table of contents

Data Warehouses

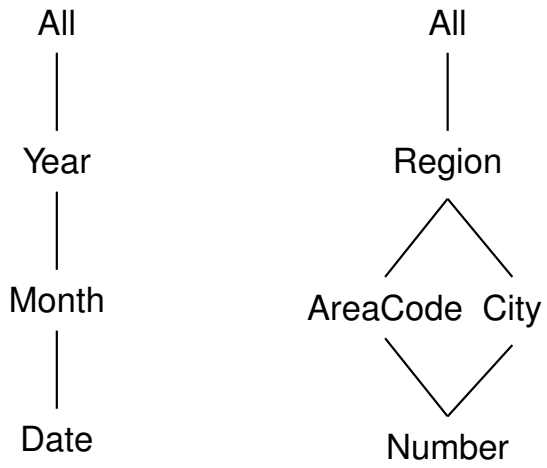
- Data Warehouses (DWs) are data repositories queried by OLAP systems, which require aggregation of data
- The DWs consist mainly of dimension and facts:
  - Dimensions: way in which data is organized
  - Facts: numerical data related with the dimensions





Dimension Schemas

- A dimension schema $\mathcal{S} = (\mathcal{C}, \nearrow)$ [Hurtado et. al.;ACM Trans.2005]
  - $\mathcal{C}$ is a set of *categories*,
  - $\nearrow$ is a child/parent relation between categories

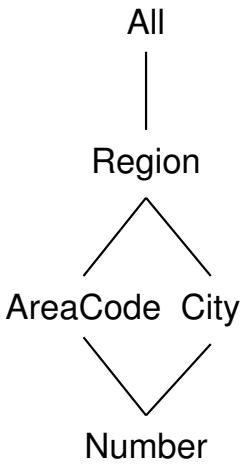


Time and Phone dimension schemas

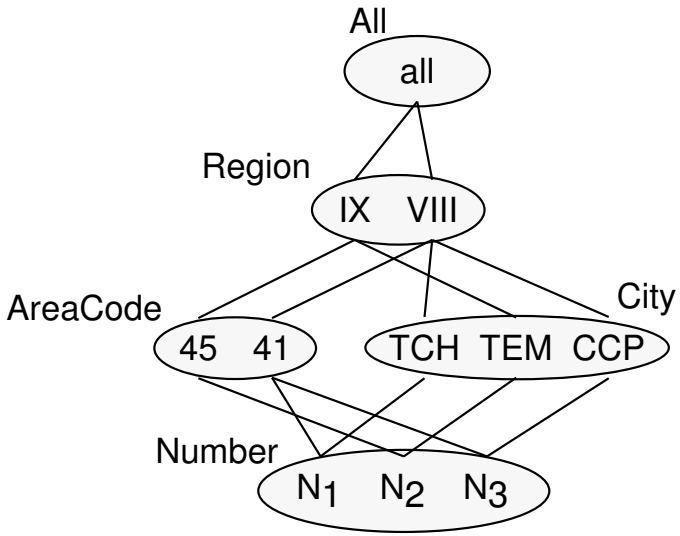
- For the Phone dimension schema:
  - $\mathcal{C} = \{\text{Number, AreaCode, City, Region, All}\}$
  - $\nearrow = \{\text{(Number, AreaCode), (Number, City), (AreaCode, Region), (City, Region), (Region, All)}\}$

Dimension Instances

- A dimension instance $D = (\mathcal{M}, <)$ over $\mathcal{S} = (\mathcal{C}, \nearrow)$
  - $\mathcal{M}$ contains the elements of each category
  - $<$ roll-up relation between elements



Phone dimension schema

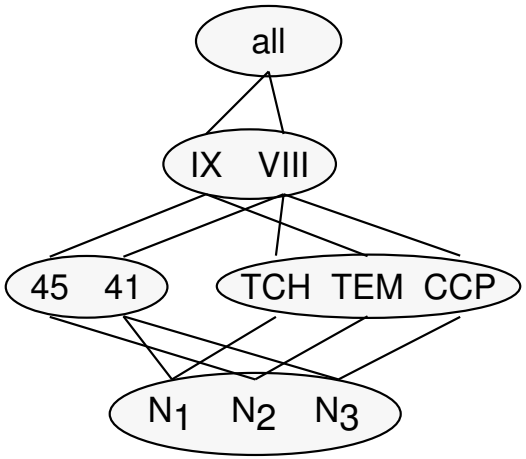


Phone dimension instance D

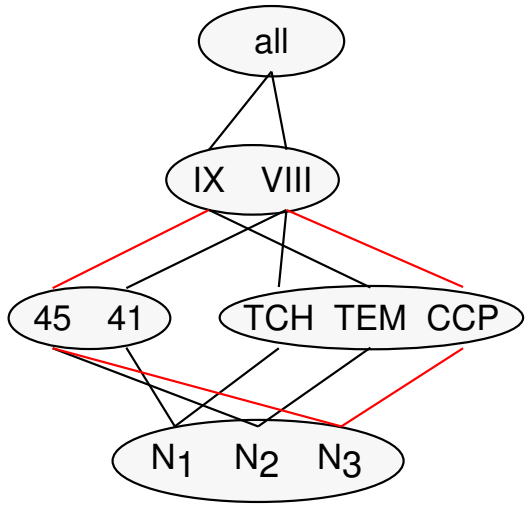
- $\mathcal{M} = \{\text{Number}(N_1), \text{Number}(N_2), \text{Number}(N_3), \text{AreaCode}(45), \ldots\}$
- $< \; = \{(N_1,41), (N_2,45), (N_3,41), (45,IX), \ldots\} \Rightarrow \mathcal{R}_D(c_i, c_j)$

Dimensions Constraints

- Strictness constraints: $c_i \rightarrow c_j$
  - the rollup relation $\mathcal{R}_D(c_i, c_j)$ is a function
- Covering constraints: $c_i \Rightarrow c_j$
  - every element in c_i rolls-up to at least one element in category c_j



Strict and covering

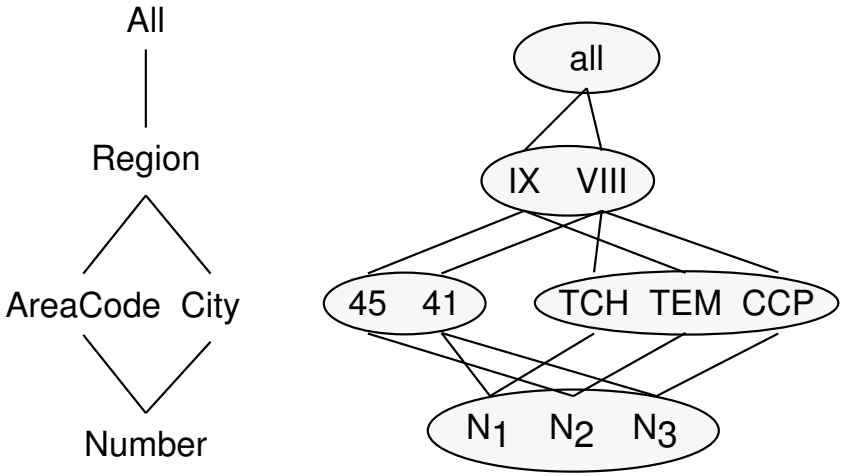


Non-strict and covering

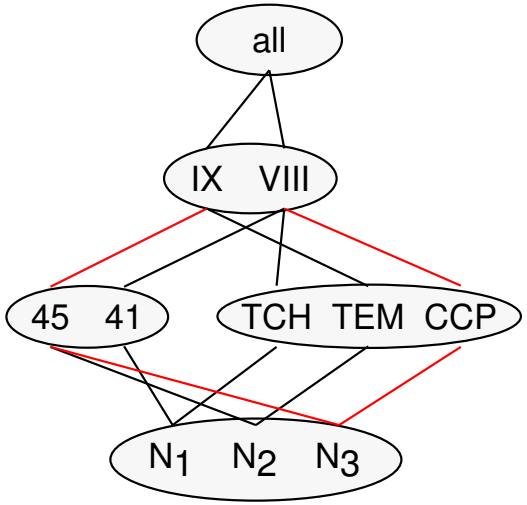
- A dimension D is *strict* (*covering*) if all of its rollup relations are strict (covering)

Inconsistent Reclassifications

- Most research and industrial applications assume strict and covering dimensions
  - Summarization operations between two connected categories are always correct
- Dimensions containing one *conflicting level* may become non-strict and non-covering after reclassifications and this can lead to incorrect results when summarizing data [Hurtado, Mendelzon, Vaisman; DOLAP, ICDE'99]



Dimension schema and instance D



Reclassify$(D, \text{Number}, \text{AreaCode}, \text{N}_3, 45)$

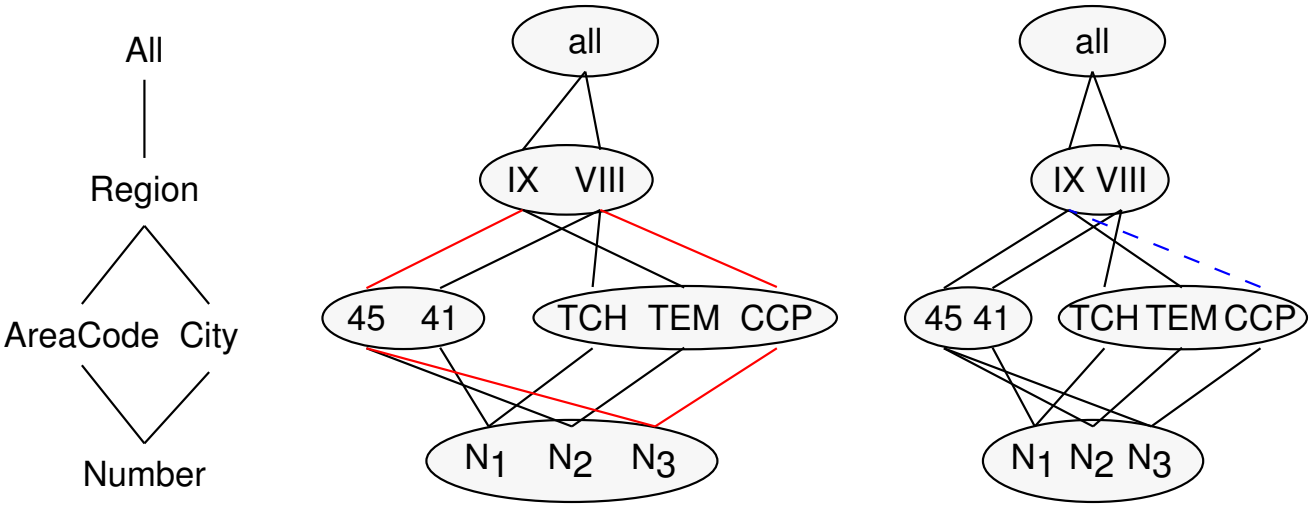
- We must *repair* the non-strict dimension

Outline

r-repairs for Dimension Instances

A minimal r-repair for a dimension D and a set of reclassifications $\mathcal{R}$ is a new dimension D' that:

- is consistent wrt $\Sigma = \Sigma_s(\mathcal{S}) \cup \Sigma_c(\mathcal{S})$
- contains (a, b) in $<'$, for every $\mathit{Reclassify}(D, c_i, c_j, a, b) \in \mathcal{R}$
- is obtained by applying a minimum number of insertions and deletions to the original rollup relations



$\mathit{Reclassify}(D, \text{Number}, \text{AreaCode}, N_3, 45)$

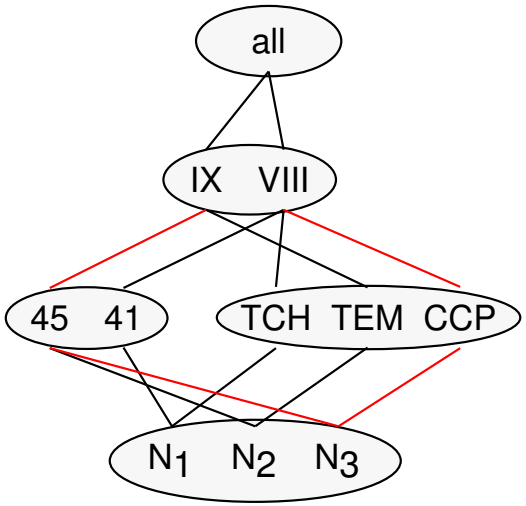
D_1

$\Delta(D, D_1) = 2$

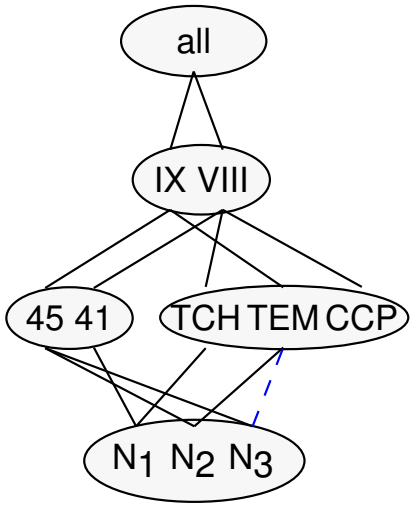
r-repairs for Dimension Instances

A minimal r-repair for a dimension D and a set of reclassifications $\mathcal{R}$ is a new dimension D' that:

- is consistent wrt $\Sigma = \Sigma_s(\mathcal{S}) \cup \Sigma_c(\mathcal{S})$
- contains (a, b) in $<'$, for every $\mathit{Reclassify}(D, c_i, c_j, a, b) \in \mathcal{R}$
- is obtained by applying a minimum number of insertions and deletions to the original rollup relations

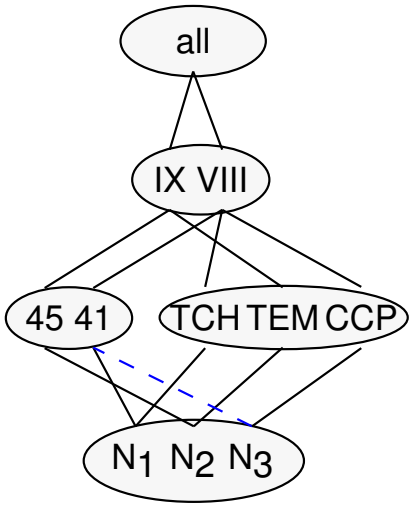


$\mathit{Reclassify}(D, \text{Number}, \text{AreaCode}, N_3, 45)$



D_2

$\Delta(D, D_2) = 2$



D_3

$\Delta(D, D_3) = 2$ X

Results

- Given a dimension D over a schema $\mathcal{S}$ with $\Sigma = \Sigma_s(\mathcal{S}) \cup \Sigma_c(\mathcal{S})$, and a set of reclassifications $\mathcal{R}$ applied over D, producing dimension D_u

Theorem

Problem: deciding if there exists an *r-repair* D' of D_u with respect to Σ, such that $dist(D_u, D') \leq k$
Complexity: NP-complete

- The proof uses a reduction from the set covering problem which is NP-complete

Theorem

Problem: deciding if D' is a minimal *r-repair* of D with respect to Σ
Complexity: co-NP-complete

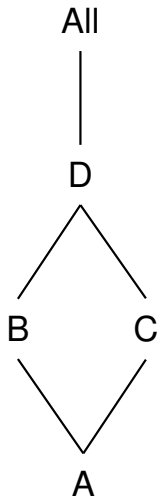
Outline

Algorithms for r-repairs

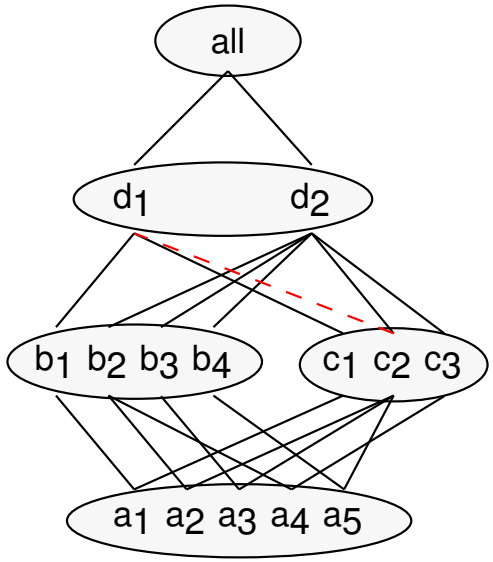
- We study *r-repairs* for dimensions containing *at most one conflicting level*
- The algorithms find an *r-repair* for a dimension under a ser of reclassifications $|\mathcal{R}| = 1$ in polynomial time
- The algorithms repair dimensions with respect to strictness
- The heuristics are:
  - To choose a repair operation *that do not generate new non-strict paths*
  - To choose a repair operation *that requires the least number of changes*

Heuristics: No Generation of New non-strict Paths

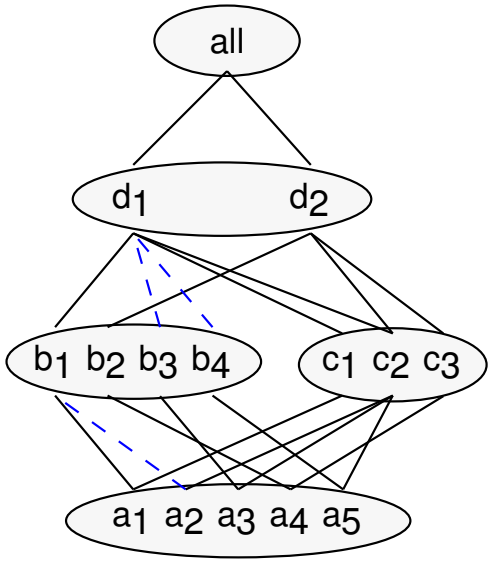
- Elements incident to c_2: a_2, a_3, a_5



Dimension schema



Reclassify$(D, \mathrm{C}, \mathrm{D}, c_2, d_1)$

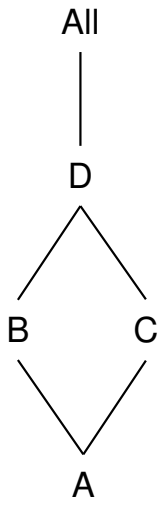


r-repair of D

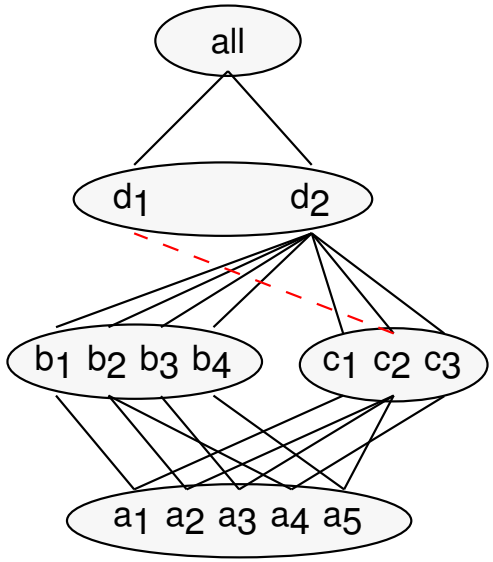
- Non-strict paths:
  - $a_2 \rightarrow b_2 \rightarrow d_2$, $a_2 \rightarrow c_2 \rightarrow d_1$
  - $a_3 \rightarrow b_3 \rightarrow d_2$, $a_3 \rightarrow c_2 \rightarrow d_1$
  - $a_5 \rightarrow b_4 \rightarrow d_2$, $a_5 \rightarrow c_2 \rightarrow d_1$
- a_2 must change parent in B or C
- Change parent of b_3
- Change parent of b_4

Heuristics: No Generation of New non-strict Paths

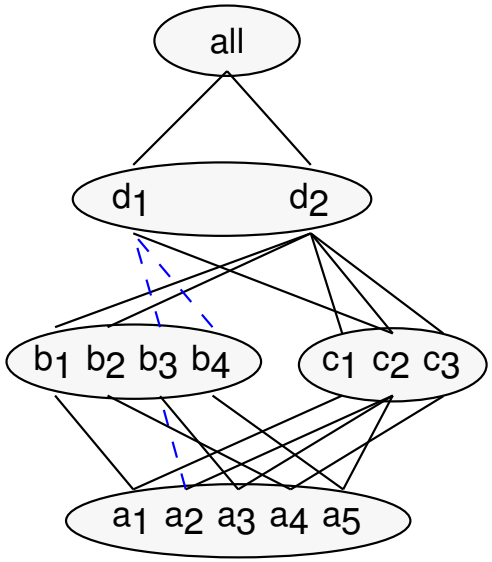
- Elements incident to c_2: a_2, a_3, a_5



Dimension schema



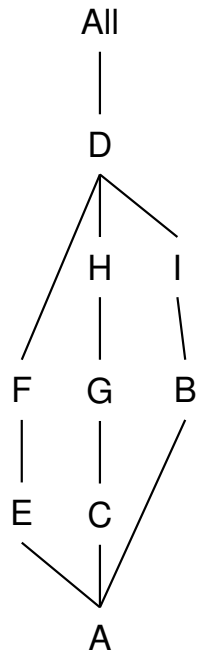
Reclassify(D, C, D, c_2, d_1)



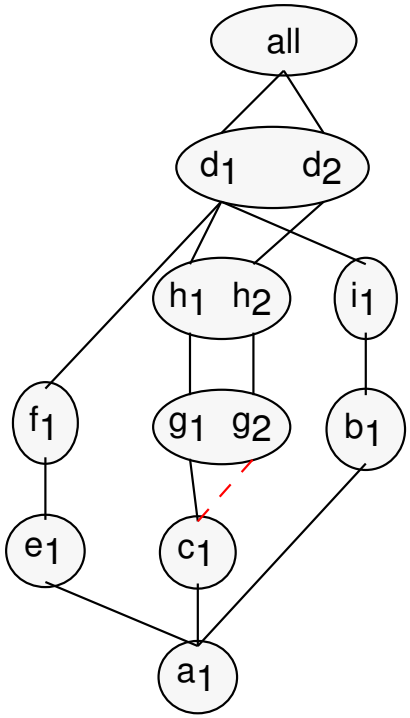
r-repair of *D*

- Non-strict paths:
  - $a_2 \rightarrow b_2 \rightarrow d_2$, $a_2 \rightarrow c_2 \rightarrow d_1$
  - $a_3 \rightarrow b_3 \rightarrow d_2$, $a_3 \rightarrow c_2 \rightarrow d_1$
  - $a_5 \rightarrow b_4 \rightarrow d_2$, $a_5 \rightarrow c_2 \rightarrow d_1$
- a_2 must change parent in B or C
- Change parent of b_3
- Change parent of b_4

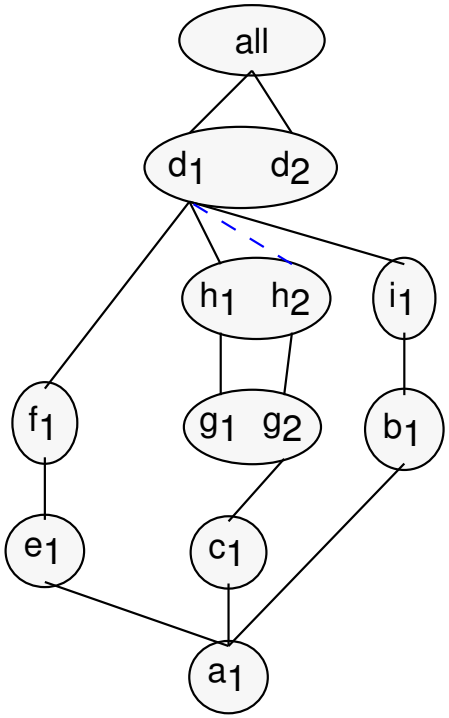
Heuristics: Least Number of Changes



Dimension schema



Reclassify$(D, \mathrm{C}, \mathrm{G}, c_1, g_2)$



r-repair of *D*

- Non-strict paths:
  - $a_1 \rightarrow e_1 \rightarrow f_1 \rightarrow d_1$
  - $a_1 \rightarrow c_1 \rightarrow g_2 \rightarrow h_2 \rightarrow d_2$
  - $a_1 \rightarrow b_1 \rightarrow i_1 \rightarrow d_1$
- $old\_CL = d_1$, $new\_CL = d_2$
- Paths reaching $d_1 = 2$, $d_2 = 1$
- Keep d_1 as ancestor of a_1

Complexity

- *search_path*() $\Longrightarrow$ captures the hierarchy schema
  - Runs in $O(k)$, k the number of paths reaching the CL and starting at BT
- *search_repair*() $\Longrightarrow$ captures inconsistencies and finds an *r-repair*
  - Worst case scenario: $O(n * m * k)$
    - n : number of elements at BT
    - m : longest path from BT to CL
    - k : number of alternative paths from BT to CL
  - Best case scenario: $O(n * \log\ m * k)$, when the hierarchy schema is a tree

Proposition

- *search_repair*() terminates in a finite number of steps
- *search_repair*() finds an *r-repair* for dimension D

Related Work

- Updating dimensions:
  - Hurtado, Mendelzon, Vaisman: *Updating Olap Dimensions*, DOLAP'99
  - Hurtado, Mendelzon, Vaisman: *Maintaining Data Cubes under Dimensions Updates*, ICDE'99
  - Letz, Henn, Vosse: *Consistency in Data Warehouses Dimensions*, IDEAS'02
- Repairing inconsistencies:
  - Pedersen, Jensen, Dyreson: *Extending Practical Pre-Aggregation in On-Line Analytical Processing*, VLDB'99
  - Bravo, Caniupan, Hurtado: *Logic Programs for Repairing Inconsistent Dimensions in Data Warehouses*, AMW'10

Conclusions

- Dimensions can be updated and they should be repaired if inconsistencies arise
- *r-repairs* are new dimension instances that satisfy the constraints, and keep the updates
- In general, finding *r-repairs* for dimension instances is NP-complete
- In practice, computing *r-repairs* can be done in polynomial time, for set of reclassifications $|\mathcal{R}| = 1$, and dimension schemas having at most one CL
- The algorithms may produce not necessarily minimal *r-repairs*
- Future work:
  - The experimentation of the algorithms in real-world DWs
  - The possibility of preventing rollups that could make no sense in practice. By prioritizing the rollup functions or defining some rollups to be fixed