Introduction

- Generalization of OLAP Model to represent data ambiguity.
  - Imprecision
  - Uncertainty
- Criteria that must be satisfied by any approach to handle data ambiguity
  - Consistency
  - Faithfulness
- Possible world interpretation of data ambiguity.
- Allocation policies.
- Algorithms for evaluating aggregation queries AVERAGE, COUNT, and SUM
- An experimental evaluation.

Data Representation

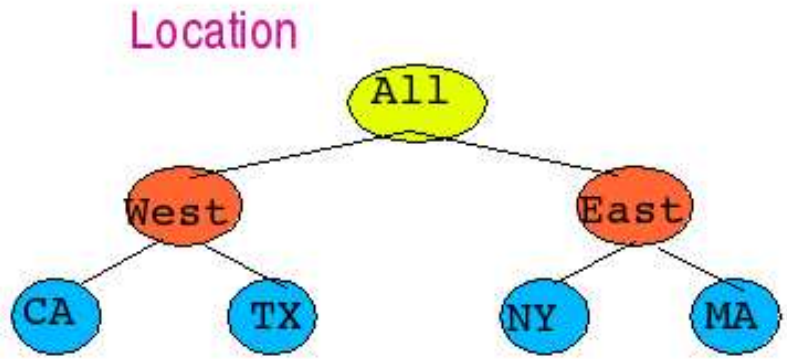
Extend measure and dimension attributes to support imprecision and uncertainty.

- Definition(**Uncertain Domain**)
  - An uncertain domain U over base domain O is the set of all possible probability distribution functions over O.
- Definition(**Imprecise Domains**)
  - An imprecise domain I over base domain B is a subset of the powerset of B with $\emptyset \notin I$; elements of I are called imprecise values

$$
\begin{aligned}
B \quad &= \{CA, TX, NY, MA\} \\
I \subseteq 2^B &= \{\{CA\}, \{TX\}, \{NY\}, \{MA\} \\
&\quad \{CA, TX\}, \{CA, NY\}, \{CA, MA\}, \{TX, NY\}... \\
&\quad \{CA, TX, NY\}, \{CA, TX, MA\}, ... \\
&\quad \{CA, TX, NY, MA\} \\
&\quad \} .
\end{aligned}
$$

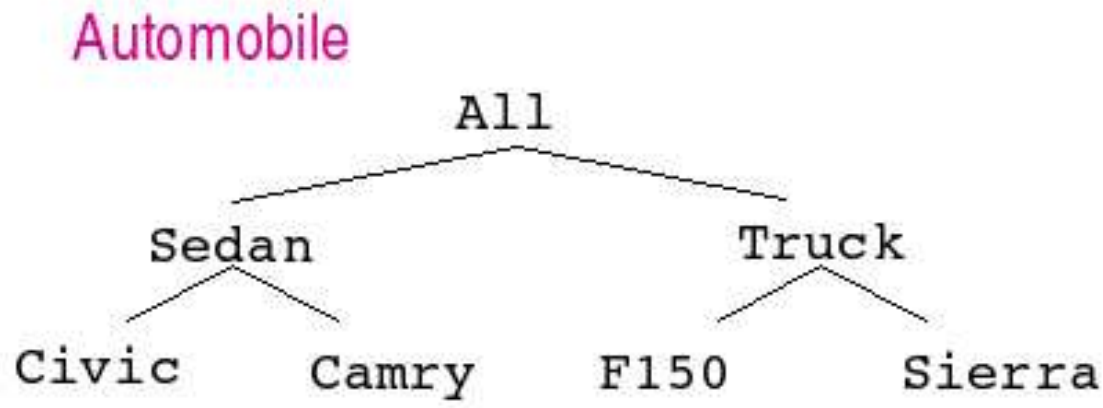
Data Represntation(contd..)

- Definition(**Hierarchical Domains**).
  - A hierarchical domain H over base domain B is defined to be an imprecise domain over B such that
    (1) H contains every singleton set and
    (2) for any pair of elements $h_1, h_2 \in H, h_1 \supseteq h_2$ or$h_1 \cap h_2 = \emptyset$



$\{\{CA\}, \{TX\}, \{NY\}, \{MA\}\ \{CA, TX\}, \{NY, MA\}, \{CA, TX, NY, MA\}\}$

Hierarchical Domains(contd..)



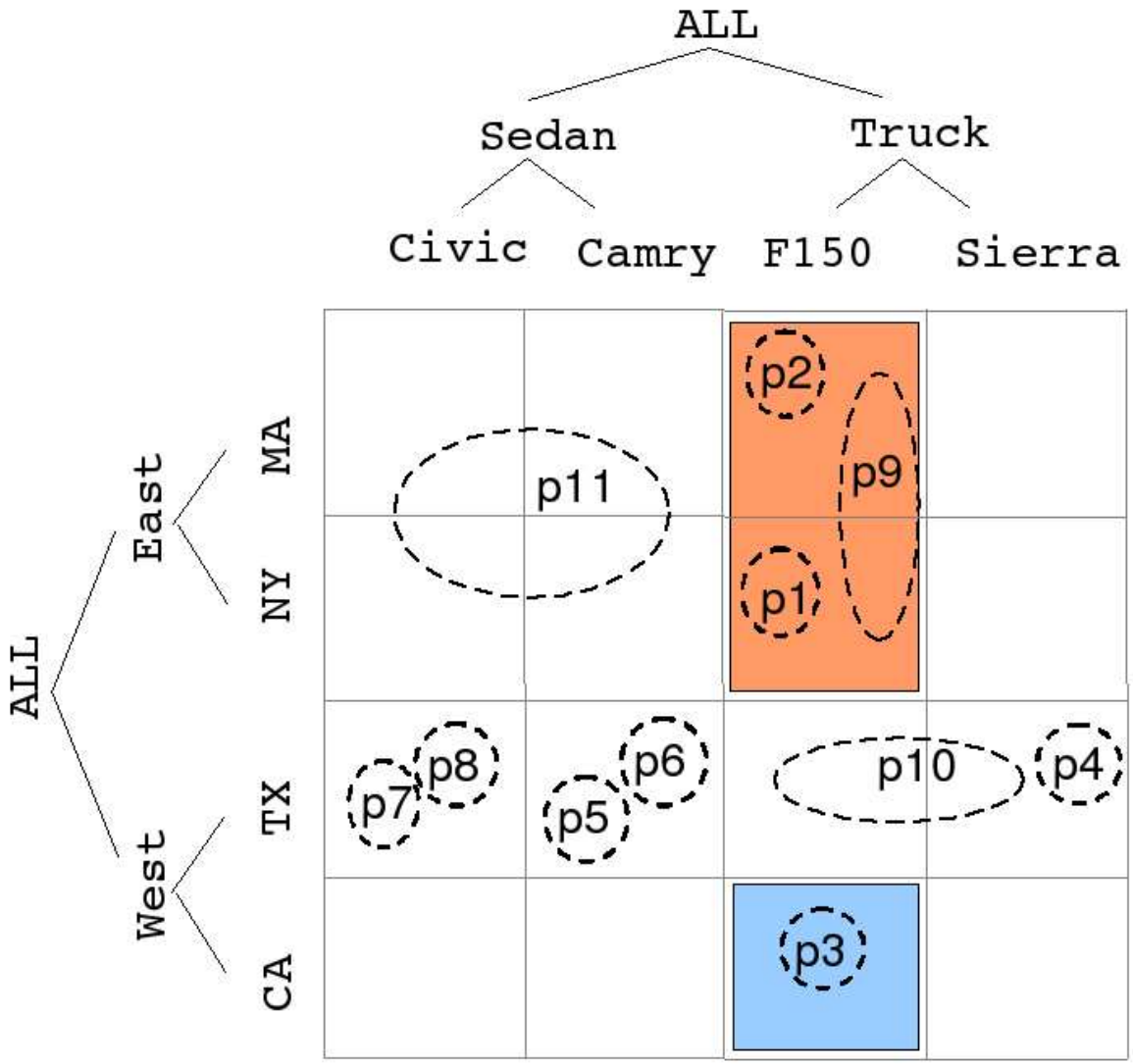
Data Represntation(contd..)

- Definition(**Fact Table Schemas and Instances**).
  - A *fact table schema* is $\langle A_1, A_2, ...A_k; M_1, ...M_n \rangle$ where
    (1) each dimension attribute $A_i, i \in 1...k$, has an associated domain $\text{dom}(A_i)$, that is imprecise and
    (2) each measure attribute $M_j, j \in 1...n$, has an associated domain $\text{dom}(M_j)$ that is either numeric or uncertain.

	Auto	*Loc*	*Repair*	*Text*	*Brake*
p1	F-150	NY	\$200	...	$\langle 0.8, 0.2 \rangle$
p2	F-150	MA	\$250	...	$\langle 0.9, 0.1 \rangle$
p3	F-150	CA	\$150	...	$\langle 0.7, 0.3 \rangle$
p4	Sierra	TX	\$300	...	$\langle 0.3, 0.7 \rangle$
p5	Camry	TX	\$325	...	$\langle 0.7, 0.3 \rangle$
p6	Camry	TX	\$175	...	$\langle 0.5, 0.5 \rangle$
p7	Civic	TX	\$225	...	$\langle 0.3, 0.7 \rangle$
p8	Civic	TX	\$120	...	$\langle 0.2, 0.8 \rangle$
p9	F150	East	\$140	...	$\langle 0.5, 0.5 \rangle$
p10	Truck	TX	\$500	...	$\langle 0.9, 0.1 \rangle$

  - A *database instance* of this fact table schema is collections of facts of the form $\langle a_1, a_2, ..a_k; m_1, m_2..., m_n \rangle$ where $a_1 \in dom(A_i), i \in 1...k$ and $m_j \in dom(M_j), j \in 1...n$.

Diagrammatic representation of Regions and Cells



Regions and Cells(contd..)

- Definition(**Regions and Cells**).
  - Consider a fact table schema with dimension attributes $A_1, A_2, ..., A_k,$. A vector $\langle c_1, c_2, ..., c_k \rangle$ is called a cell if every c_i is an element of base $A_i, i \in 1...k$.
  - A region of a dimension vector $\langle a_1, a_2, ..., a_k, \rangle$ is defined to be the set of cells $\langle c_1, c_2, ..., c_k | c_i \in a_i, i \in 1...k \rangle$ Let reg(r) denote a region associated with a fact r.

Data Represntation(contd..)

- Proposition
  - Consider a fact table schema with dimension attributes $A_1, A_2, ..., A_k$ that all have hierarchical domains. Consider a $k-$dimensional space in which each axis i is labelled with the leaf node dom(A_i). For every region, the set of all cells in the region is a contiguous $k-$dimensional hyper-reactangle that is orthogonal to the axes.

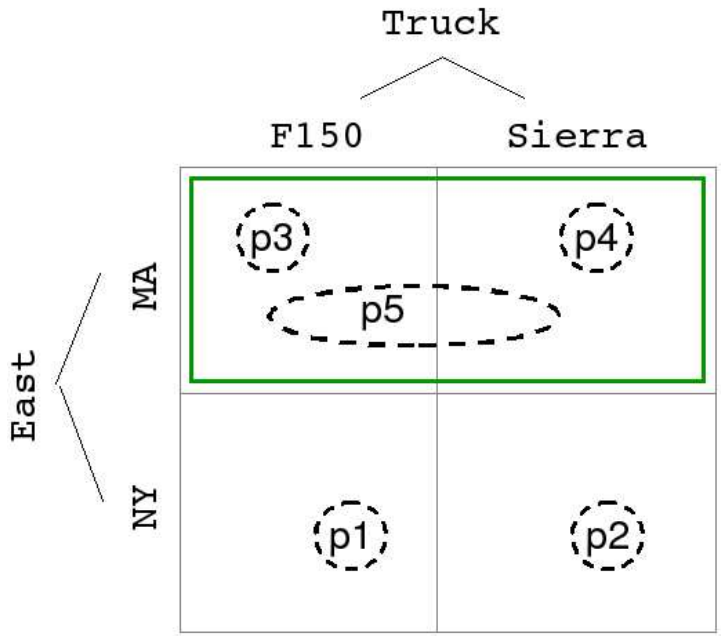
Uncertain Data Representation

- Classify the incident based on the type of problem.
- The subjective nature of text precludes the unambiguity.
- Define an uncertain measure whose values are represented as *pdfs* over the set of problem type.
- A text analyzer analyzes the text and outputs *pdf* over the classifiers for each problem type.

ID	Auto	Loc	Rep	Text	Brake
p1	F150	NY	200	.....	<0.8,0.2>

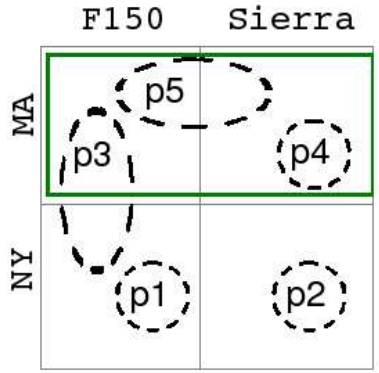
Queries

- Definition (**Queries and Query Results**)
  - Aquery Q over a database D with schema $\langle A_1, A_2, ..., A_k; M_1, M_2, ..., M_n$ has form $Q(a_1, a_2, ..., a_K; M_i; \mathring{A})$, where
    (1) $a_1, a_2, ..., a_k$ describes the $k-$dimensional region being queried
    (2) M_i describes the measure of interest and
    (3) $\mathring{A}$ is an aggregation function.
  - The result of Q is obtained by applying $\mathring{A}$ to a set of facts FIND-RELEVENT$(a_1, a_2, ..., a_k, D)$
- **Query:** *What is the total repair cost of Trucks in MA?*



FIND-RELEVENT

- Identifies the set of the facts in D deemed relevant to query region.
- All precise facts are naturally included.
- What about imprecise facts?
  - None option
  - Contains option
  - Overlaps option
- **Query:** *What is the total repair cost of Trucks in MA?*



Aggregating Uncertain Measures

- **Query:** *How likely are the brake problems for Sedans in Texas*
- **Answer:** Aggregation over the pdfs for p_5, p_6, p_7, p_8
- Aggregation is same as evaluating expected value of a random variable.
- $\overline{P(x)} = \sum w_p * P(x)$
- Unless there is some prior knowledge, we assume the weights are uniform.
- In case of uniform weights $\overline{P(x)}$ is average of probabilities.

p5	Camry	TX	\$325	...	$\langle 0.7, 0.3 \rangle$
p6	Camry	TX	\$175	...	$\langle 0.5, 0.5 \rangle$
p7	Civic	TX	\$225	...	$\langle 0.3, 0.7 \rangle$
p8	Civic	TX	\$120	...	$\langle 0.2, 0.8 \rangle$

OLAP Requirement

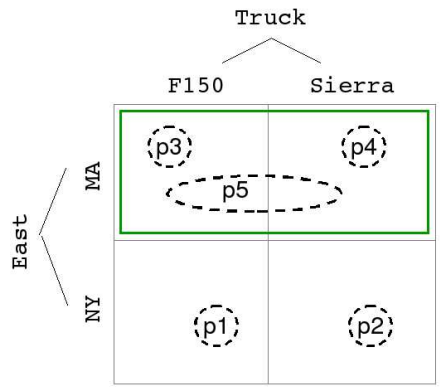
In providing support for OLAP-style queries in the presence of imprecision and uncertainty, the answers to these queries should meet reasonable set of requirements.

- Consistency
- Faithfulness

Consistency

- The user expects to see some natural relationship holds between the answers to aggregation queries associated with different (connected)regions in a hierarchy.

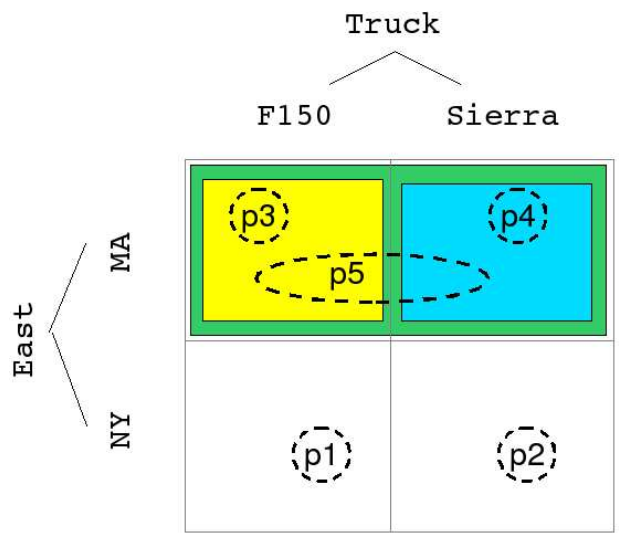
- Specific forms of Consistency
  - Sum-Consistency
  - Boundedness-Consistency



- **Theorem** *There exists a SUM aggregate query which violates Sum-Consistency when* **Contains** *option is used to find relevent imprecise facts in* FIND-RELEVENT.

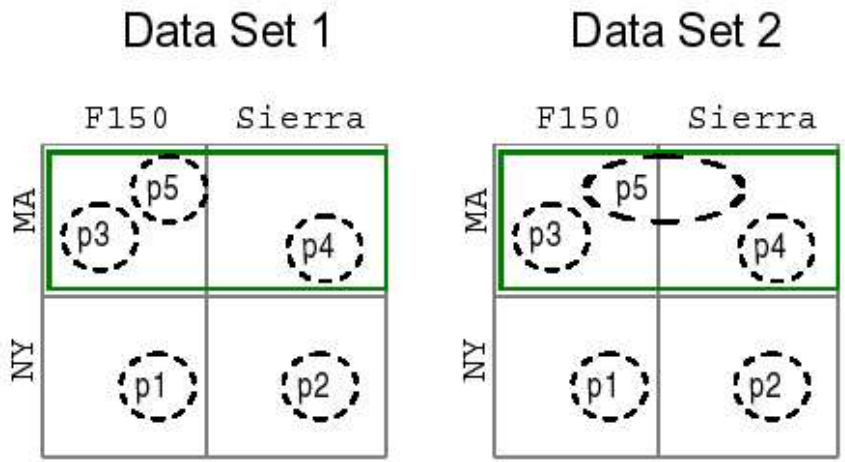
Motivating Example for Allocation of *imprecise* facts

- p_5 overlaps *yellow* and *blue* cells.
- Partially assign p_5 to both cells.
- $weight(yellow) = W_{yellow}$
- $weight(blue) = W_{blue}$
- $W_{yellow} + W_{blue} = 1$
- $RepairCost(MA, F150) = p_3 + W_{yellow} * p_5$
- $RepairCost(MA, Sierra) = p_4 + W_{blue} * p_5$
- $RepairCost(East, Sierra) = p_3 + p_4 + p_5$



Faithfulness

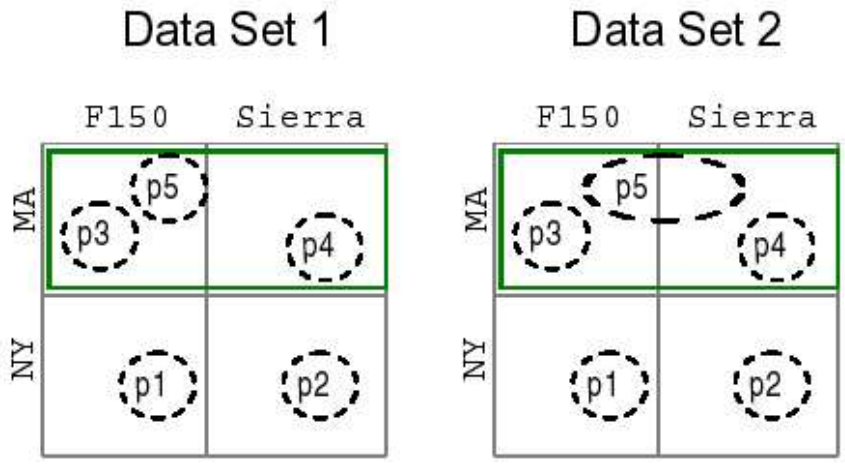
- Faithfulness captures the intuition that more precise data should give better results.
- Definition(**Measure-similar Databases**)
  - We say that two databases D and D' are measure-similar if D' is obtained from D by modifying the dimension attribute values in each fact r. Let $r' \in D'$ denote the fact obtained by modifying $r \in D$; we say that *r corresponds to r'*.



- Types of Faithfulness
  - Basic faithfulness
  - β-faithfulness

Basic faithfulness

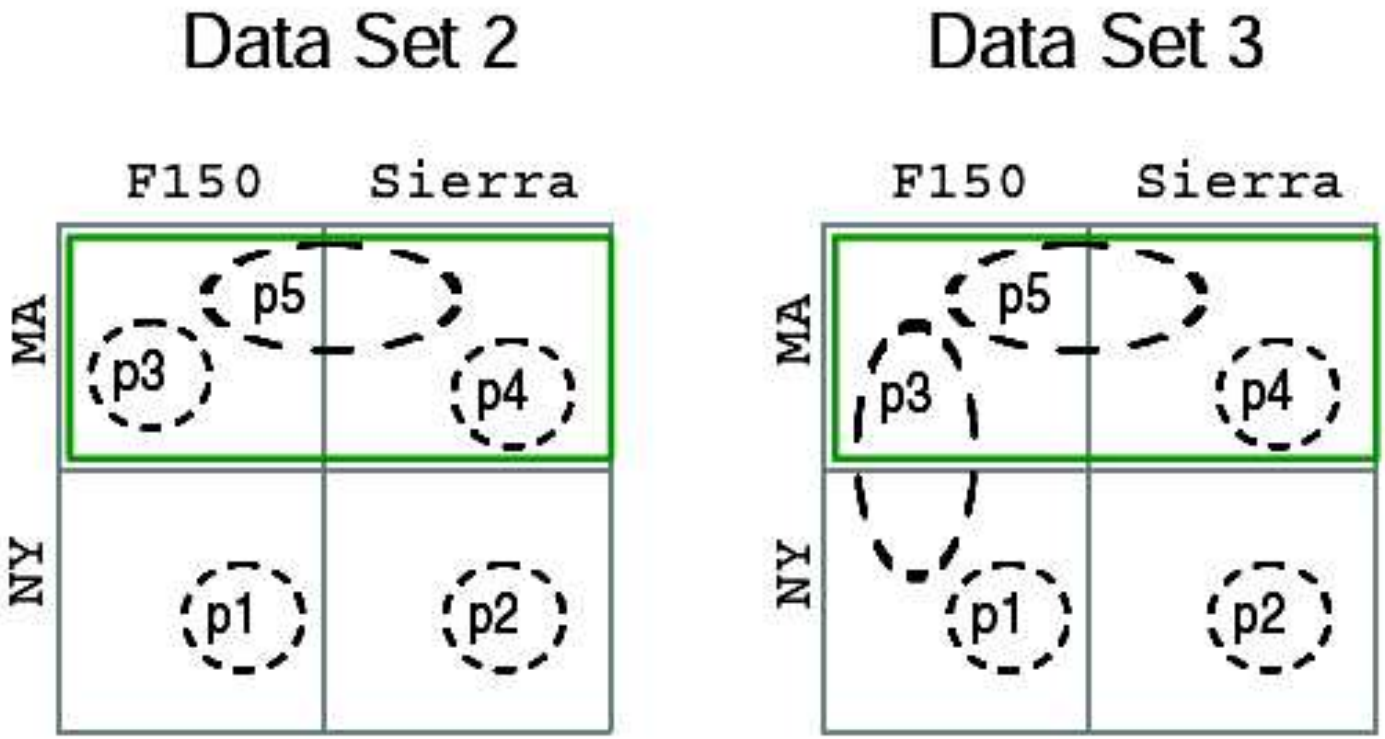
- We say that two measure-similar databases D and D' are identically precise with respect to query Q if for every pair of corresponding facts $r \in D$ and $r' \in D'$, either both reg(r) and reg(r') are completely contained in reg(Q) or both are completely disjoint from reg(Q). We say that an algorithm satisfies *Basic faithfulness* with respect to aggregation function $\mathring{A}$, if the algorithm gives identical answers for every pair of measure-similar databases D and D' that are identically precise with respect to Q.



Theorem *SUM, COUNT, AVERAGE violate basic faithfulness when* **None** *option is used*

$\beta-$faithfulness

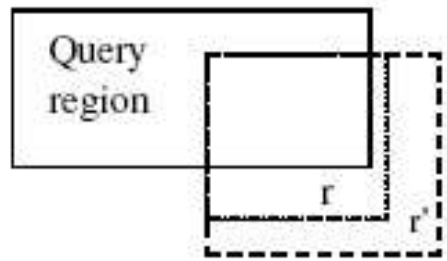
Query: *What is the total repair cost of Trucks in MA?*



Answer(Data-set-2) $\leqq$ Answer(Data-set-3)

$\beta-$faithfulness

- Definition(partial order$\preceq$)
  - Fix a query Q. We say that the relation $I_Q(D, D')$ holds on two measure-similar databases D and D' if all pairs of corresponding facts in D and D' are identical, except for a single fact $r \in D$ and $r' \in D'$ such that reg(r') is obtained from reg(r) by adding a cell $c \notin reg(Q) \cup reg(r)$



- Definition($\beta-$faithfulness)
  - Let $\beta(x_1, x_2, ..., x_p)$ be a predicate such that the value taken by each argument of β belongs to range of a fixed aggregation operator $\mathring{A}$.
  - We say that an algorithm satisfies $\beta-$faithfulness with respect to $\mathring{A}$ if for any query Q compatible with $\mathring{A}$, and for any set of databases $D_1 \preceq D_2 \preceq ... \preceq D_p$ the predicate $\beta(\overline{q_1}, ..., \overline{q_p}))$ holds true where $\overline{q_i}$denotes the answer computed by algorithm on D_i, i in $1...p$

Queries with the Overlaps option

- Notions of possible worlds and Extended Data Model
- Summarizing Possible Worlds
- Allocations and allocation policies
- Results

Possible Worlds

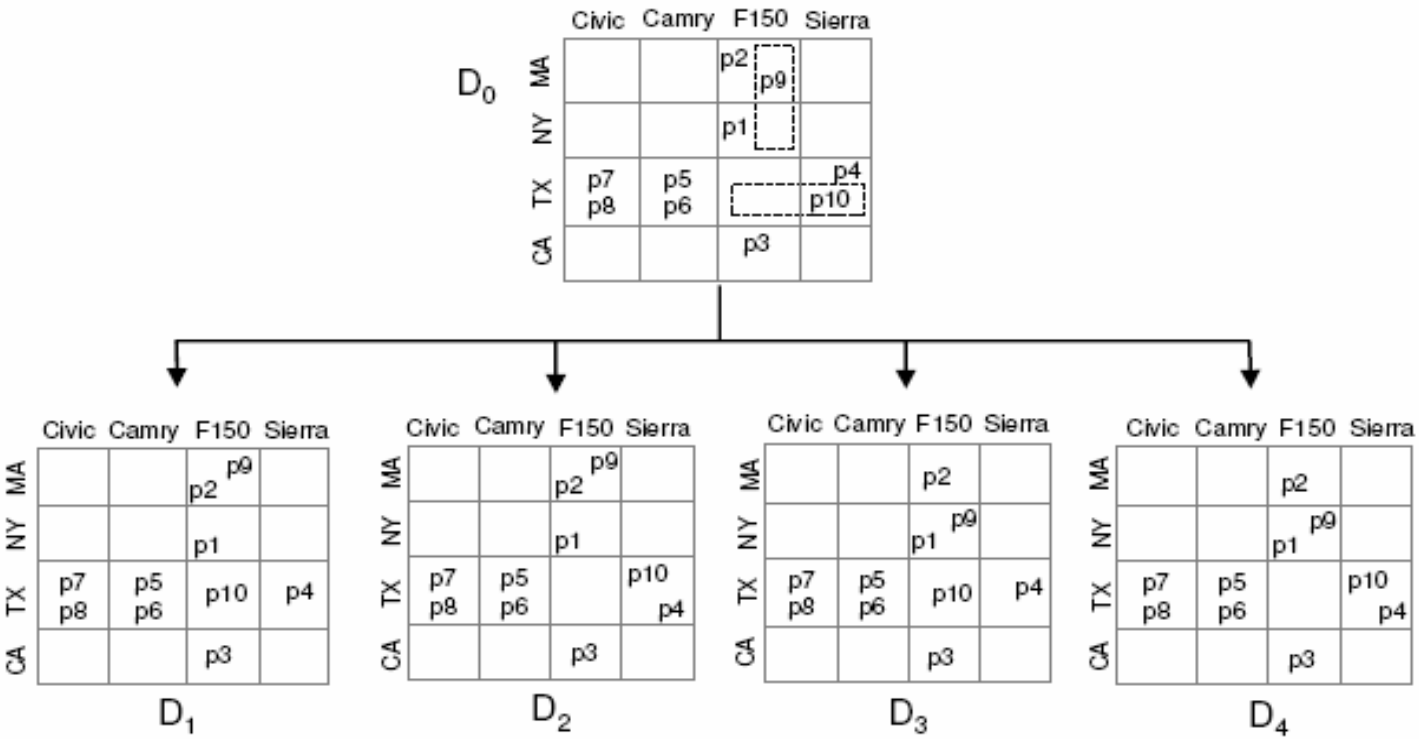


Figure 3: Possible Worlds

Possible Worlds

- Consider an imprecise fact r which maps to a region R of cells.
- Each cell inside this region represents a possible completion of an imprecise fact, formed by replacing non-leaf node a_i with a leaf node from the subtree rooted at a_i.
- Repeating this process for every imprecise fact in D leads to a database D' that contains only precise facts. We call D' a *possible world* for D, and the multiple choices for eliminating imprecision lead to a set of possible worlds for D.
- Possible world query semantics

  Given all possible worlds together with their probabilities, queries are easily answered (using expected values)

Possible world query semantics

- The allocation weights encode a set of possible worlds,{D1, . . . ,Dm} with associated weights w1, . . . ,wm. The answer to a query Q is a multiset1 {v1, . . . , vm}.
- Consider the multiset {v1, . . . vm} of possible answers to a query Q. We define the *answer variable* Z associated with Q to be a random variable with pdf

$$\Pr[Z = v_i] = \sum_{j \ s.t. \ v_i = v_j} w_j, i, j$$

- *Basic faithfulness is satisfied if answers to queries are computed using the expected value of the answer variable.*
- The above approach complicates matters because the number of possible
- worlds grows exponentially in the number of imprecise facts. Allocations can compactly encode this exponentially large set but the challenge now is to summarize without having to explicitly use the allocations to iterate over all possible worlds.

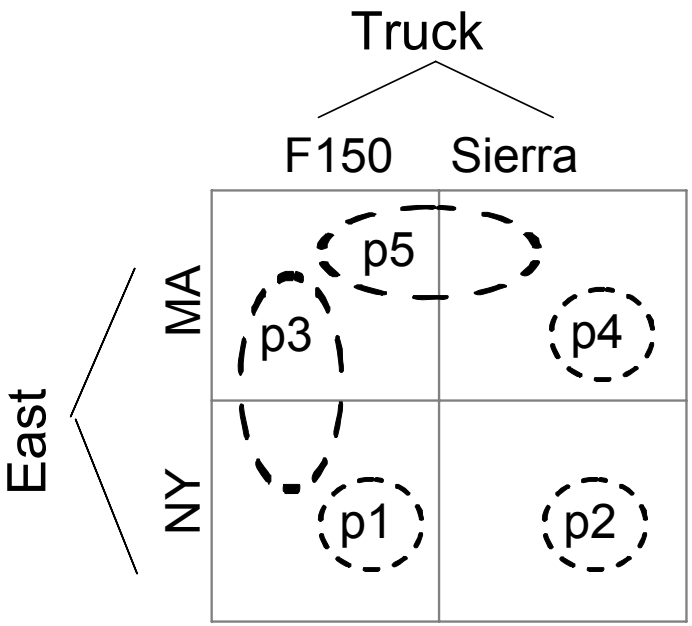
Extended Data Model

- If there are k imprecise facts in a dataset D, and the region for the i^{th} imprecise fact contains c_i cells, the number of possible worlds is

$$\prod_{i=1}^{k} c_i$$

- i.e. number of possible worlds is exponential!
- To tackle the complexity due to this exponential number of possible worlds, we consider each imprecise fact r and assign a probability for its “true” value being c, for each cell c in its region. The assignments for all imprecise facts collectively (and implicitly) associate probabilities (weights) with each possible world.

Storing Allocations using Extended Data Model



ID	FactID	Auto	Loc	Repair	Weight
1	1	F150	NY	100	1.0
2	2	Sierra	NY	500	1.0
3	3	F150	MA	150	0.6
4	3	F150	NY	150	0.4
5	4	Sierra	MA	200	1.0
6	5	F150	MA	100	0.5
7	5	Sierra	MA	100	0.5

Allocation

- Allocation gives facts weighted assignments to possible completions, leading to an extended version of the data (Allocated Database)
  - The schema of Allocated Database contains all the columns of D plus additional columns to keep track of the cells that have strictly positive allocations.
  - Size increase is linear in number of (completions of) imprecise facts
  - Queries operate over this extended version
- Key contributions:
  - Appropriate characterization of the large space of allocation policies
  - Designing efficient allocation policies that take into account the correlations in the data

Summarizing PossibleWorlds

- We answer query Q in the extended data model in two steps:
- **Step 1:** We identify the set of candidate facts r Є R(Q) and compute the corresponding allocations to Q. The former is accomplished by using a filter for the query region whereas the latter is accomplished by identifying groups of facts that share the same identifier in the ID column and then summing up the allocations within each group. At the end of this step, we have a set of facts that contains for each fact r Є R(Q), the allocation of r to Q and the measure value associated with r. Note that this step depends only on the query region q.
- **Step 2:** This step is specialized to the aggregation operator, and two comments are in order. First, we seek to identify the information necessary to compute the summarization while circumventing the enumeration of possible worlds. Second, it is possible in some cases to merge this second step with the first in order to gain further savings, e.g., the expected value of SUM can be computed thus.

Results on Query Semantics

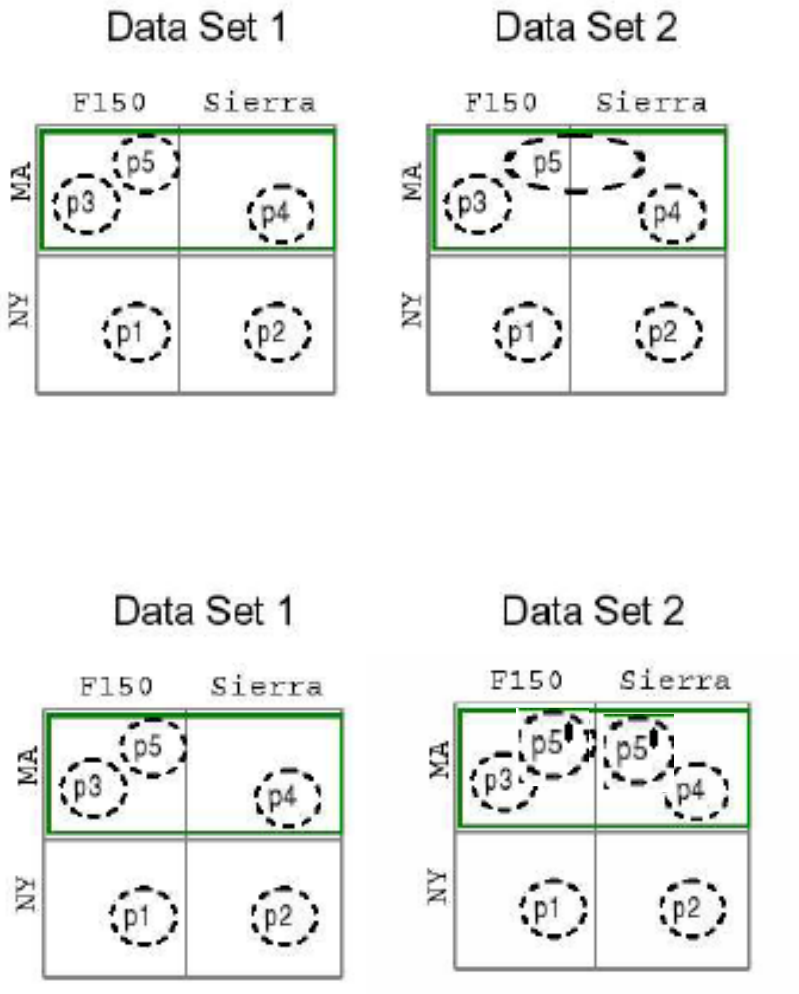
- Evaluating queries over extended version of data yields expected value of the aggregation operator over all possible worlds
  - intuitively, the correct value to compute
- Efficient query evaluation algorithms for SUM, COUNT
  - consistency and faithfulness for SUM, COUNT are satisfied under appropriate conditions
- Dynamic programming algorithm for AVERAGE
  - Unfortunately, consistency does not hold for AVERAGE

Alternative Semantics for AVERAGE

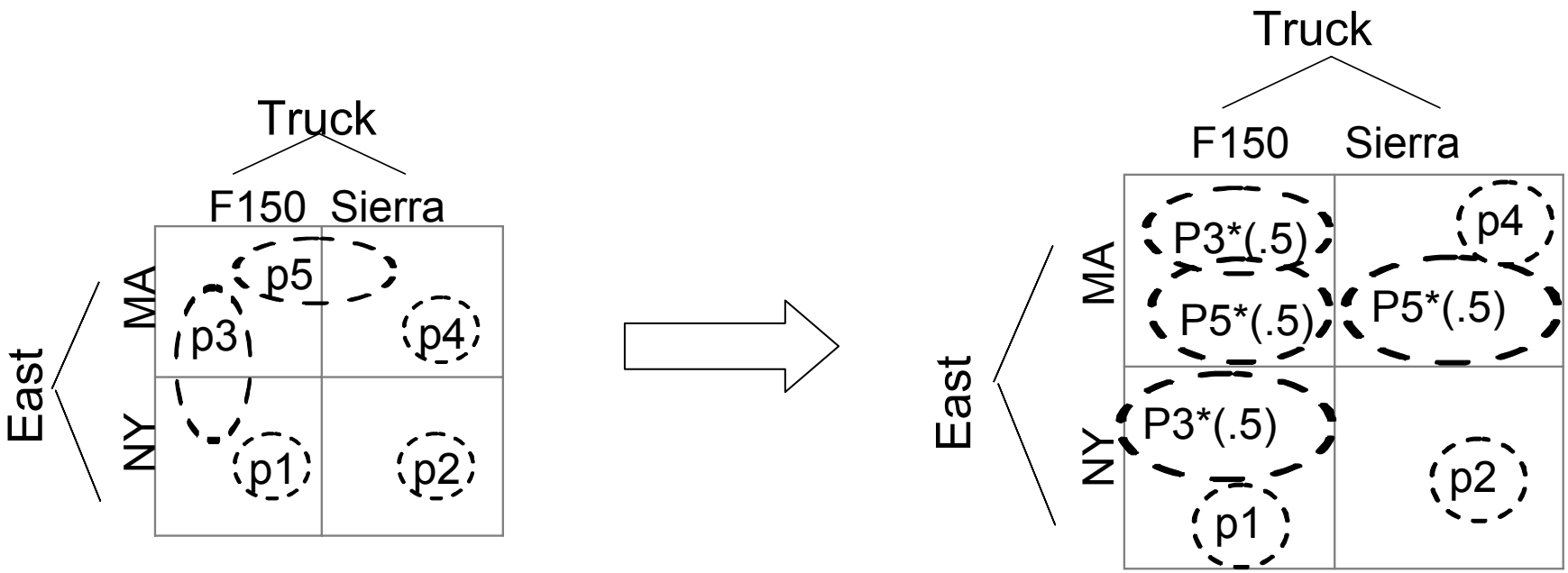
- APPROXIMATE AVERAGE
  - E[SUM] / E[COUNT] instead of E[SUM/COUNT]
  - simpler and more efficient
  - satisfies consistency
  - extends to aggregation operators for uncertain measures

Allocation Policies

- Faithfulness can be violated if the extended data model is built using arbitrary allocation policies.
- Monotone Allocation Policy
  - Restricts the way in which the weights for the larger set of possible worlds are defined.
  - As a region gets larger allocations for the old cells are redistributed to new cells
  - E.g. Uniform Allocation Policy



Uniform Allocation policy

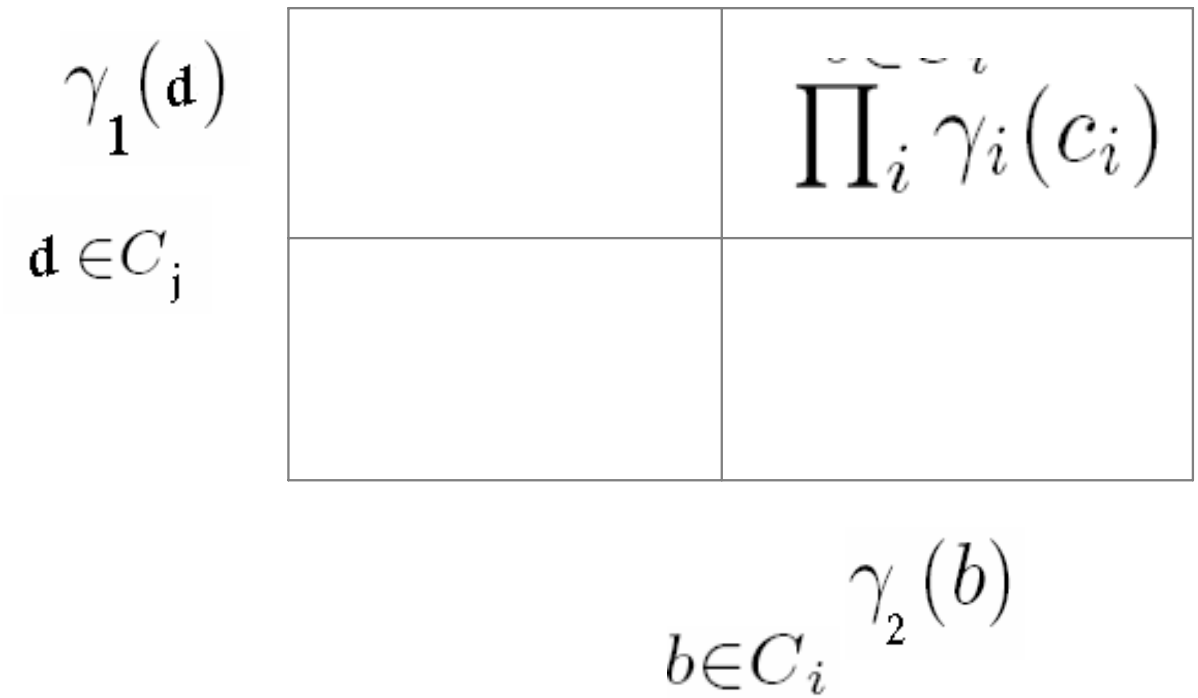


Allocation Policies

- Dimension Independent Policies

  allocation $p_{c,r}$ equals the probability cell c is chosen

$$\text{if } c = (c_1, c_2, \ldots, c_k), \text{ then } p_{c,r} = \prod_i \gamma_i(c_i).$$



Allocation Policies

- Measure-oblivious Allocation

  An allocation policy is said to be *measure-oblivious* if the following holds.

- Let D be any database and let D' be obtained from D by possibly modifying the measure attribute values in each fact r arbitrarily but keeping the dimension attribute values in r intact. Then, the allocations produced by the policy are identical for corresponding facts in D and D'.
- Eg. Uniform Allocation policy
- Correlation-Preserving Allocation

  Allocation policy A is *correlation-preserving* if for every database D, the correlation distance of A with respect D is the minimum over all policies.

- correlation distance

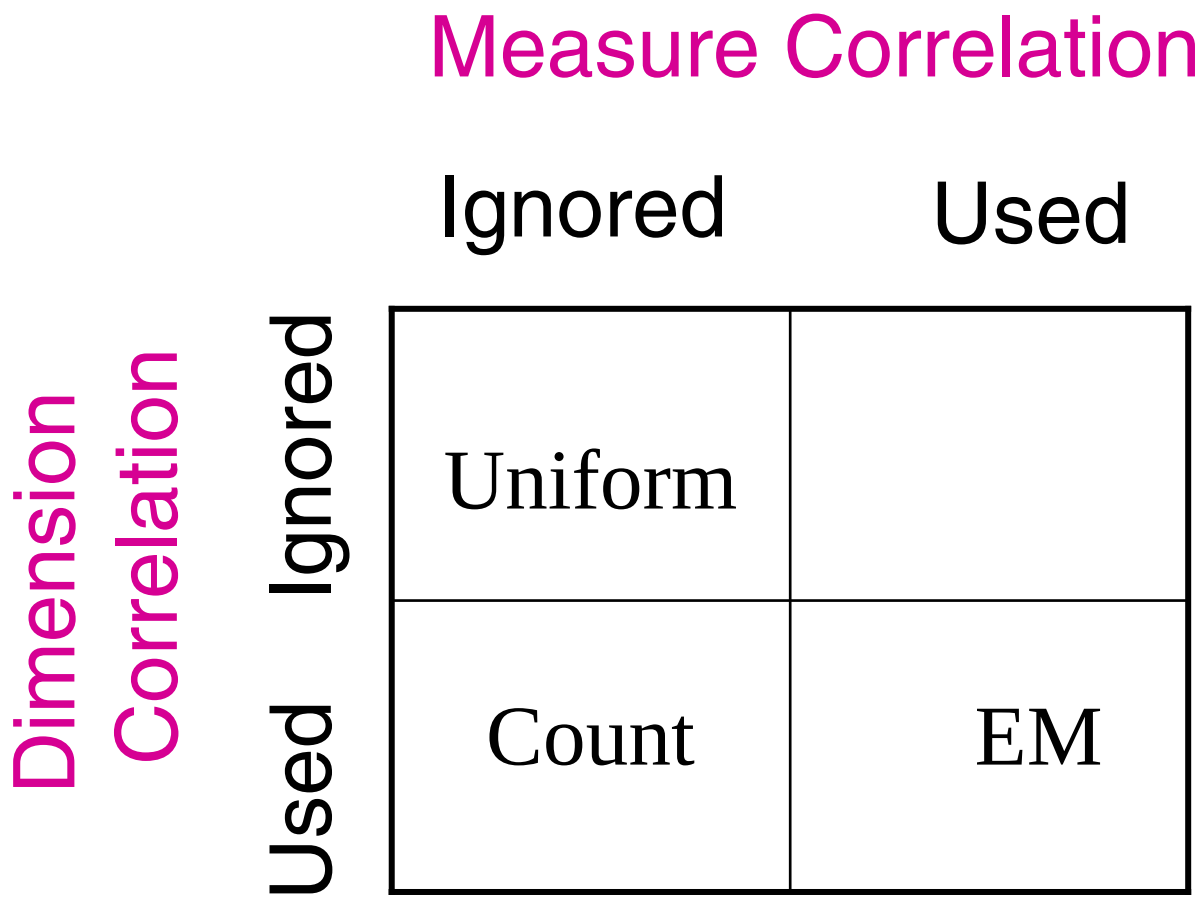
$$\Delta(\mathrm{corr}(D_0), \sum_i w_i \cdot \mathrm{corr}(D_i))$$

Correlation-based Allocation

- Involves defining an objective function to capture some underlying correlation structure
  - a more stringent requirement on the allocations
  - solving the resulting optimization problem yields the allocations

- EM (Expectation Minimization)-based iterative allocation policy
  - interesting highlight: allocations are re-scaled iteratively by computing appropriate aggregations

Classifying Allocation Policies

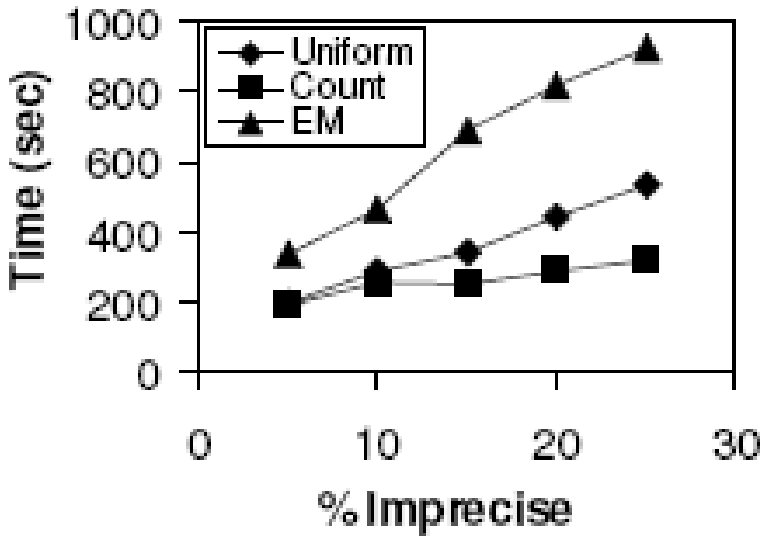


Experiments

- No materialized views or indices were built on the data
- Experiments using both a numeric measure and an uncertain measure (over a base domain of size 2) were conducted. All dimensions had hierarchical domains with three levels. For three of these hierarchical domains, the root of the corresponding tree had 5 children; every root child had 10 children each (resulting in 50 leaf nodes); the corresponding branching factors for the remaining dimension was 10 and 10, respectively (100 leaf nodes). Thus, there are 12.5 million cells in the multidimensional space.
- The initial data consisted of 1 million facts (density=1/12.5 = 8%), each generated by choosing (with uniform probability) a leaf node from the appropriate hierarchy for each dimension. Imprecision was introduced by replacing the leaf node for a dimension with an appropriate parent in the hierarchy.
- For 50% of the imprecise facts, a second dimension was made imprecise as well (e.g., if 10% of the facts were imprecise, 5% were imprecise in 1 dimension and 5% imprecise in 2 dimensions).

Scalability of the Extended Data Model

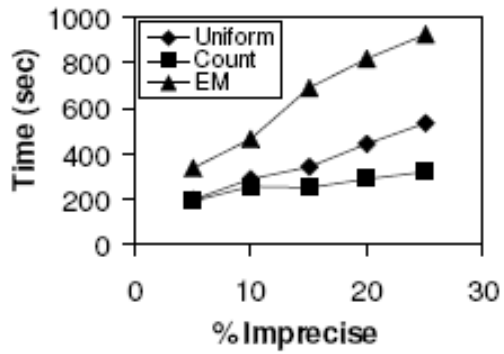
- Running time for Different allocation policies increase (almost) linearly with respect to the number of imprecise records.
- The running time has 2 components,
  - one for processing the input data
  - and the other for writing out the facts to the extended data model



(a) Allocation Algorithm Running Time

Scalability of the Extended Data Model

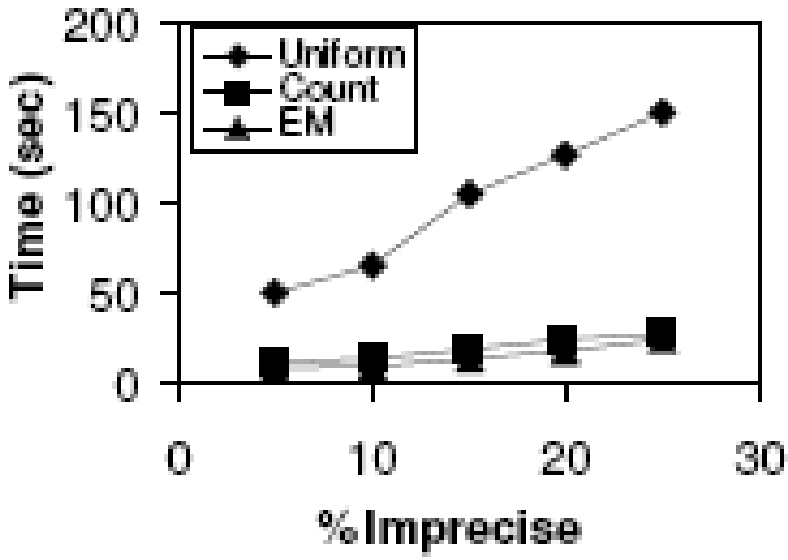
- For EM the first component is high, since it is an iterative algorithm requiring multiple scans. This explains the reason for longer running time than Uniform and Count which require only a single scan.
- The larger running time for Uniform with respect to Count is due to the second component. Since the input data density is low, Uniform allocates to many empty cells, so the number of allocated facts created by Uniform is significantly larger than Count and EM. For example, with 25% imprecision, Uniform had 14.5 million facts whereas Count and EM each had 2.3 million facts. This relative difference between Uniform and Count should increase as the input data density decreases.



(a) Allocation Algorithm Running Time

Query Running Time Performance

- Figure shows the average query running time for SUM.
- In general the running time was dominated by the I/O cost for scanning the extended data model.
- As seen above, this is much higher for Uniform than for Count or EM.



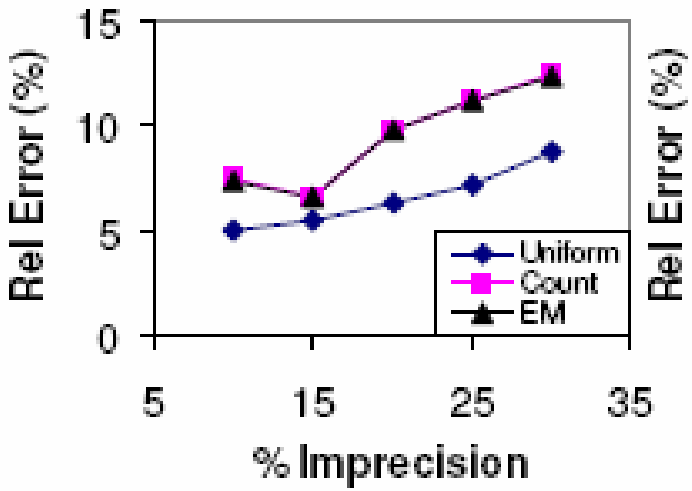
(b) Query Running Time Performance

Quality of the Allocation Policies

- These experiments evaluate how data characteristics affect the behavior of our proposed allocation policies.
- If all facts are precise, dependencies between dimensions are perfectly encoded in the cell counts. As facts become imprecise, a portion of this correlation information is lost. The strength of this encoding against such loss can be measured as the expected number of records in each non-empty cell.
- The other characteristic that we chose to examine is measure correlation, which captures the effect of dimension values on the measure value.

Quality of the Allocation Policies

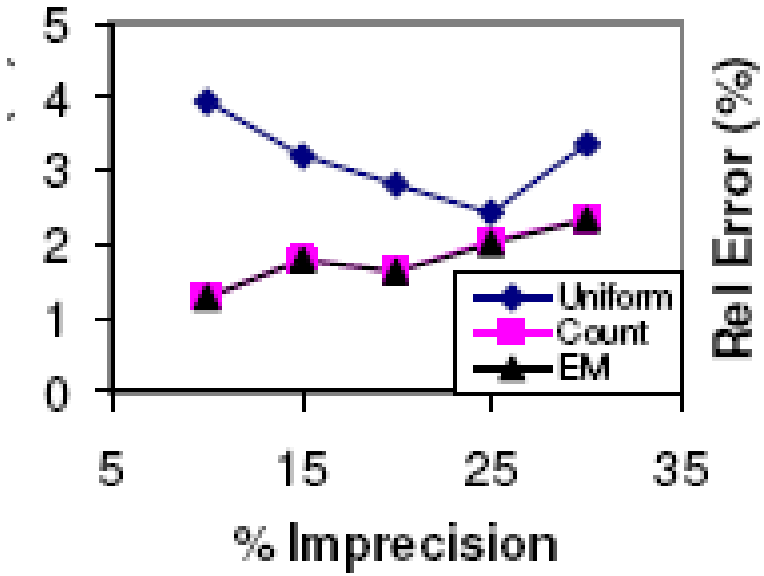
- The results show that Uniform allocation policy has a lower relative error compared to Count and EM. The reason for this is the loss of dimension-value correlation information when a record is made imprecise.
- For example, if a record *r* in cell *c* is made imprecise, *c* becomes empty, since *r* was the only record in that cell. During allocation, Count and EM will not allocate any portion of *r* to *c*. On the other hand, Uniform will allocate some of *r* to *c*, resulting in a better allocation (i.e., one that better reflects the correct answer).



(a) Low Pseudo Density Dataset

Quality of the Allocation Policies

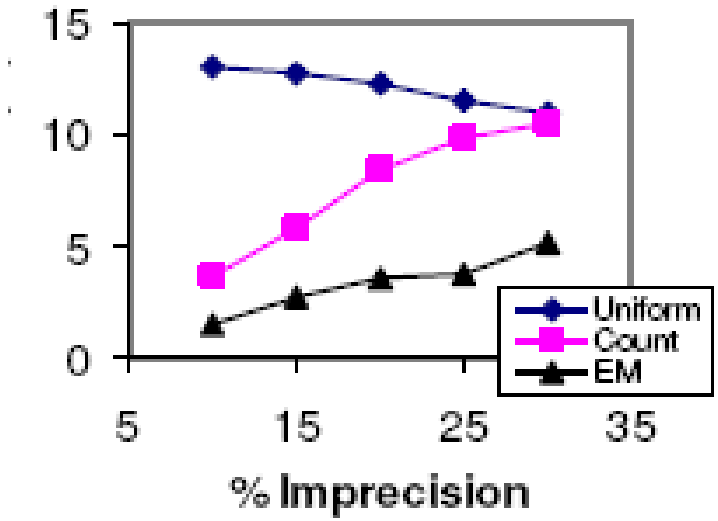
- Since the pseudo-density is higher, less dimension-value correlation information is lost as more records become imprecise. Thus Count and EM result in better allocations, whereas Uniform suffers since it ignores the available correlation information and allocates to empty cells as well.



(b) High Pseudo Density Dataset

Quality of the Allocation Policies- high correlation

- The results show that EM now significantly outperforms both Count and Uniform. This is because EM uses the correlation between the measure and dimensions while performing allocation, whereas Count does not.
- For example, consider a record *r* in the left half of the grid that is made imprecise to overlap some cells in the right half. Count will allocate *r* to the cells in the right half, whereas EM will allocate *r* only to the cells in the left half since it notices the correlation between the measure value of *r* and cells in the left half.



(c) Measure Correlated Dataset

Summary

- Allocation is the key to the framework
- Efficient algorithms for aggregation operators with appropriate guarantees of consistency and faithfulness
- Iterative algorithms for allocation policies