

Doug Burdick · Prasad M. Deshpande · T.S. Jayram · Raghu Ramakrishnan · Shivakumar Vaithyanathan

OLAP Over Uncertain and Imprecise Data

Received: date / Accepted: date

Abstract We extend the OLAP data model to represent data ambiguity, specifically imprecision and uncertainty, and introduce an allocation-based approach to the semantics of aggregation queries over such data. We identify three natural query properties and use them to shed light on alternative query semantics. While there is much work on representing and querying ambiguous data, to our knowledge this is the first paper to handle both imprecision and uncertainty in an OLAP setting.

Keywords Aggregation · Imprecision · Uncertainty · Ambiguous

1 Introduction

In this paper, we extend the multidimensional OLAP data model to represent *data ambiguity*, specifically *imprecision* and *uncertainty*, and study possible semantics for aggregation queries over such data. While there is much work on representing and querying ambiguous data, and even some work in the context of OLAP, to our knowledge this is the first paper to identify criteria that must be satisfied by any approach to handling data ambiguity in an OLAP setting, and to use these criteria in a principled manner to arrive at appropriate semantics for queries. Our first criterion, called

consistency, accounts for the relationship between similar queries issued at related nodes in a domain hierarchy in order to meet users' intuitive expectations as they navigate up and down the hierarchy. The second criterion, called *faithfulness*, captures the intuition that more precise data should lead to better results. The third criterion, called *correlation-preservation*, essentially requires that the statistical properties of the data should not be affected by the allocation of ambiguous data records. While the last two criteria are not specific to OLAP, to our knowledge they have not been proposed previously.

We extend the usual OLAP data model in two fundamental ways. First, we relax the restriction that dimension attributes in a fact must be assigned leaf-level values from the underlying domain hierarchy, in order to model *imprecision*. For example, we can denote that a particular repair took place in Texas, without specifying a city. Clearly, this has implications for how we answer queries—for a query that aggregates repair costs in Austin, should the example repair be included, and if so, how? Our second extension is to introduce a new kind of measure attribute that represents uncertainty. Intuitively, an uncertain value encodes a range of possible values together with our belief in the likelihood of each possible value. Specifically, we represent a value for an uncertain measure as a *probability distribution function* (*pdf*) over values from an associated “base” domain.

Our contributions can be summarized as follows:

1. Generalization of the OLAP model to represent data ambiguity. To our knowledge, this is the first such generalization that addresses both imprecise dimension values and uncertain measure values.
2. The introduction of criteria—consistency, faithfulness, and correlation-preservation—that guide the choice of semantics for aggregation queries over ambiguous data.
3. A possible-worlds interpretation of data ambiguity that leads to a novel allocation-based approach to defining semantics for aggregation queries, and a careful study of choices arising in the treatment of data ambiguity, using the consistency, faithfulness, and correlation-preservation criteria.

Doug Burdick
1210 West Dayton Street, Madison, WI, 53715
E-mail: drburdick@cs.wisc.edu

Prasad Deshpande
650 Harry Road, San Jose, CA, 95125
E-mail: prasadesh@in.ibm.com

T.S. Jayram
650 Harry Road, San Jose, CA, 95125
E-mail: jayram@almaden.ibm.com

Raghu Ramakrishnan
1210 West Dayton Street, Madison, WI, 53715
E-mail: raghu@cs.wisc.edu

Shivakumar Vaithyanathan
650 Harry Road, San Jose, CA, 95125
E-mail: shiv@almaden.ibm.com

4. Algorithms for evaluating aggregation queries (including AVERAGE, COUNT, and SUM for ordinary measures, and LinOp for uncertain measures), together with a complexity analysis.
5. An experimental evaluation that addresses scalability as well as result quality.

1.1 Related Work

While there is an extensive literature on queries over ambiguous data, only a few papers [21, 25, 26, 14] have considered an OLAP setting. [21] is perhaps the closest to our work in that it considers the semantics of aggregation queries, but it does not consider uncertainty or investigate criteria that shed light on appropriate query semantics. [25, 26] consider the problem of imputing missing measure values, using linear equations, entropy maximization, cross-entropy minimization, or other constraint programming techniques. [14] describes a method to estimate the error of cube aggregates for certain data; the generalization of their method to uncertain data is not clear.

[18] and [6] study aggregation queries over imprecise data but do not consider data-level hierarchies (which are central to OLAP). We note that [18] also discusses uncertainty, and the approach in [6] leads to an exponential-time algorithm for SUM. [23] supports aggregate functions for uncertain data, but doesn't support imprecision or hierarchies.

The earliest work on aggregate queries over imprecise data is [6], and it was followed by [18, 7, 22]; however, none of these papers consider data-level hierarchies (which are central to OLAP). The approach in [6] leads to an exponential-time algorithm for SUM. [7] models uncertainty using intervals and pdfs and provides algorithms for aggregating them. [22] develops a linear programming based semantics for computing aggregates over probabilistic databases. We note that [18] also discusses uncertainty, and [23] supports aggregate functions for uncertain data, but doesn't support imprecision or hierarchies.

We believe that a key contribution of this paper is our methodology—identifying intuitive criteria such as consistency, faithfulness, and correlation-preservation and using them to study alternative query semantics is an approach that can be applied outside the OLAP setting (and indeed, faithfulness and correlation-preservation are not specific to OLAP). In this respect, our approach is similar in spirit to the use of *summarizability*, which was introduced to study the interplay between *properties of data* and the aggregation operators (i.e., what properties should the data possess for results of certain aggregation functions to be meaningful) [16, 17].

A number of papers consider imprecision and uncertainty for non-aggregation queries. The use of possible world semantics to represent imprecise data is discussed in [1]. [5, 11, 4, 15, 9] associate a probability distribution with the data (either at the data element or tuple level), and generalize re-

lational algebra operators to reflect the associated probability. [2, 3] seek to identify inconsistent data and to “repair” these databases to a consistent state; in contrast, we focus on imprecise yet consistent data, and do not consider integrity constraints (other than domain constraints). Various sources of data ambiguity are classified in [19, 20], together with approaches for representing and processing the ambiguity. [24] discusses the many similarities between statistical databases and OLAP.

2 Data Model

In this section we present our generalization of the standard multidimensional data model, incorporating imprecision and uncertainty.

2.1 Data Representation

Attributes in the standard OLAP model are of two kinds—*dimensions* and *measures*. We extend the model to support uncertainty in measure values and imprecision in dimension values.

Definition 1 (Uncertain Domains) An *uncertain domain* U over base domain B is the set of all possible probability distribution functions, or pdfs, over B . \square

Thus, each value u in U is a pdf that, intuitively, indicates our degree of belief that the “true” value being represented is b , for each b in the base domain B . For instance, instead of a single sales number, we might have a pdf over a base domain of sales-range numbers, e.g., $\{\langle \$0 - \$30, 0.2 \rangle, \langle \$31 - \$60, 0.6 \rangle, \langle \$61 - \$100, 0.2 \rangle\}$.

Definition 2 (Imprecise Domains) An *imprecise domain* I over a base domain B is a subset of the powerset of B with $\emptyset \notin I$; elements of I are called *imprecise values*. \square

Intuitively, an imprecise value is a non-empty *set* of possible base domain values. Allowing dimension attributes to have imprecise domains enables us, for example, to use the imprecise value `Wisconsin` for the location attribute in a data record if we know the sale occurred in Wisconsin but are unsure about the city.

Definition 3 (Fact Table Schemas and Instances) A *fact table schema* is $\langle A_1, A_2, \dots, A_k; M_1, \dots, M_n \rangle$ where (i) each dimension attribute $A_i, i \in 1 \dots k$, has an associated domain $\text{dom}(A_i)$ that is *imprecise*, and (ii) each measure attribute $M_j, j \in 1 \dots n$, has an associated domain $\text{dom}(M_j)$ that is either *numeric* or *uncertain*.

A *database instance* of this fact table schema is a collection of *facts* of the form $\langle a_1, a_2, \dots, a_k; m_1, \dots, m_n \rangle$ where $a_i \in \text{dom}(A_i), i \in 1 \dots k$ and $m_j \in \text{dom}(M_j), j \in 1 \dots n$. \square

Definition 4 (Regions and Cells) Consider a fact table schema with dimension attributes A_1, A_2, \dots, A_k . A vector $\langle c_1, c_2, \dots, c_k \rangle$ is called a *cell* if every c_i is a single-valued element of A_i , $i \in 1 \dots k$. The *region* of a *dimension vector* $\langle a_1, a_2, \dots, a_k \rangle$ is defined to be the set of cells $\{ \langle c_1, c_2, \dots, c_k \rangle \mid c_i \in a_i, i \in 1 \dots k \}$. Let $\text{reg}(r)$ denote the region associated with a fact r . \square

If all a_i are singleton sets, the observation is *precise*, and describes a region consisting of a single cell. If one or more a_i are non-singleton sets, the observation is *imprecise* and describes a set of larger k -dimensional regions. Each cell inside this region represents a possible completion of an imprecise fact, formed by replacing the set a_i with one of the base elements it contains. The process of completing every imprecise fact in this manner results in a *possible world* for the database (Section 5).

Proposition 1 Consider a fact table schema with dimension attributes A_1, A_2, \dots, A_k . Together these describe a k -dimensional space in which each axis i is labeled with the single-valued elements of $\text{dom}(A_i)$. If all dimension attributes are imprecise, the set of all cells in the region described by each dimension vector is a collection of non-contiguous, non-overlapping k -dimensional hyper-rectangles, each orthogonal to the axes.

2.2 Dimensions in OLAP

Often in OLAP, each dimension has an associated hierarchy structure. For example, the location dimension might have attributes *City* and *State*, with *State* denoting generalizations of *City*. Allowing dimension attributes to have imprecision can result in values for these attributes to be unknown at lower levels of the hierarchy. This is illustrated by the example for imprecision above where we know *State* = Wisconsin in the sale record but not the value for *City*. (The alternative view that this entity represents the aggregate sale for the Wisconsin is not considered in this paper as it does not reflect imprecision in the dimension attribute.) This suggests a natural special case of imprecise domains called *hierarchical domains*, which we define next.

Definition 5 (Hierarchical Domains) A domain H over base domain B is defined to be an imprecise domain over B such that (1) H contains every singleton set (i.e., corresponds to some element of B) and (2) for any pair of elements $h_1, h_2 \in H$, $h_1 \supseteq h_2$ or $h_1 \cap h_2 = \emptyset$. \square

Intuitively, each singleton set is a leaf node in the domain hierarchy and each non-singleton set in H is a non-leaf node; thus, Madison, Milwaukee, etc. are leaf nodes with parent Wisconsin (which, in turn might have USA as its parent).

Hierarchical domains allow an intuitive geometric interpretation for precise and imprecise facts. For a fact, if

	<i>Loc</i>	<i>Auto</i>	<i>Repair</i>
p1	NY	F-150	\$200
p2	MA	F-150	\$250
p3	CA	F-150	\$150
p4	TX	Sierra	\$300
p5	TX	Camry	\$325
p6	TX	Camry	\$175
p7	TX	Civic	\$225
p8	TX	Civic	\$120
p9	East	F150	\$140
p10	TX	Truck	\$500

Table 1 Sample data for Example 1

each dimension value a_i correspond to leaf nodes, the observation is *precise*, and again describes a region consisting of a single cell. If one or more A_i are assigned non-leaf nodes, the observation is *imprecise* and describes a larger k -dimensional region. 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 sub-tree for hierarchical domains rooted at a_i . If every dimension attribute has a hierarchical domain, we have an intuitive interpretation of each fact in the database as a (closed) region in a k -dimensional space.

Proposition 2 If all dimension attributes have hierarchical domains, the set of all cells in the region described by each dimension vector is a contiguous k -dimensional hyper-rectangle orthogonal to the axes.

Example 1 (Imprecision in Hierarchical Domains) Consider the scenario of a car manufacturer using a CRM application to track and manage service requests across its worldwide dealer operations. A fact table illustrating such data is shown in Table 1. Each fact describes an “incident”. The first two columns are dimension attributes *Automobile (Auto)* and *Location (Loc)*, and take values from their associated hierarchical domains. The structure of these domains and the regions of the facts are shown in Figure 1. Precise facts, p1–p8 in Table 1, have leaf nodes assigned to both dimension attributes and are mapped to the appropriate cells in Figure 1. Facts p9 and p10, on the other hand, are imprecise. Fact p9 is imprecise because the *Location* dimension is assigned to the non-leaf node *East* and its region contains the cells (NY, F150) and (MA, F150). Similarly, the region for p10 contains the cells (TX, F150) and (TX, Sierra). Each fact contains a value for the numeric measure attribute *Repair* denoting the repair cost associated with the incident.

2.3 Information Extracted from Text

Consider another example from an automotive CRM application, with the fact table shown in Table 2. Similar to Example 1, every row in the fact table describes an “incident”, with a hierarchical dimension *Location (Loc)* and a standard

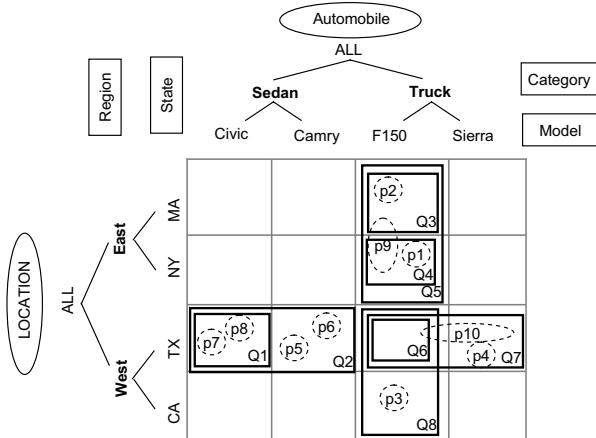


Fig. 1 Multidimensional view of Example 1

numerical measure *Repair*. In this new example, each fact now has a *Text* attribute containing unstructured (i.e. “free”) text recorded with the incident (e.g., typed into the system by a call-center operator). Information contained in this unstructured text is unavailable for conventional OLAP analysis. Standard approaches from text analytics can be used to extract structured information from the text, specifically *dimensions* and *measures*. As the following examples demonstrate, this extracted information often contains ambiguity which must be accounted for.

First, we consider an example for extracting dimension information from text. In the example, we assume a Dealer annotator which identifies auto dealership names was run over the *Text* column in the fact table. The output from this annotator will be stored in the *Dealer* dimension attribute. The structure of the Dealer dimension (which is explained in the next section) and the region for each fact are shown in Figure 2.

In the *Dealer* dimension, the base domain includes individual auto dealer names and NONE, which is assigned to facts mentioning no auto dealer. From each fact’s text attribute, the annotator may extract several dealer names or none at all. From the example, fact p_1 ’s text mentions two dealers, Capitol Chevy and Bernal BMW, abbreviated to CC and BB respectively in the *Dealer* column. Thus, p_1 is imprecise in the *Dealer* dimension, and maps to cells (NY, Capitol Chevy) and (NY, Bernal BMW). Fact p_6 mentions no dealer name, and is assigned the value NONE.

With imprecise dimensions, it is possible for facts to map to non-contiguous multi-dimensional regions. In Figure 2, we see the region for fact p_7 maps to disjoint cells (TX, Capitol Chevy) and (TX, Fremont Ford). This arises from p_7 taking two non-adjacent leaf values in the *Dealer* dimension. Notice there exists no single ordering of leaf values for *Dealer* which allows cells for both facts p_1 and p_7 to form a contiguous region. Despite this

	Loc	Repair	Text	Dealer	Brake
p1	NY	\$200	...	{CC, BB}	(0.8, 0.2)
p2	MA	\$250	...	{BB, LL}	(0.9, 0.1)
p3	CA	\$150	...	{FF}	(0.7, 0.3)
p4	TX	\$300	...	{FF}	(0.3, 0.7)
p5	TX	\$325	...	{LL}	(0.7, 0.3)
p6	TX	\$175	...	{NONE}	(0.5, 0.5)
p7	TX	\$225	...	{CC, FF}	(0.3, 0.7)
p8	TX	\$120	...	{CC}	(0.2, 0.8)
p9	East	\$140	...	{BB, LL}	(0.5, 0.5)
p10	TX	\$500	...	{LL}	(0.9, 0.1)

Table 2 Sample Data for Example 2

distinction between imprecise and hierarchical dimensions, both are treated identically in our data model.

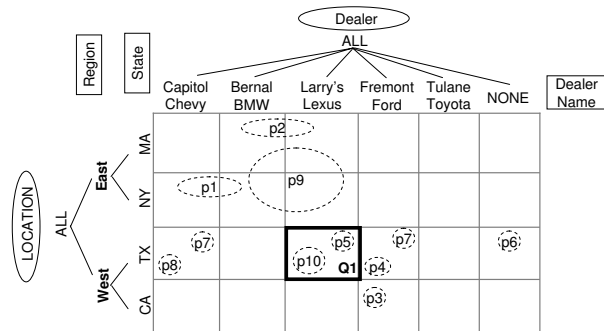


Fig. 2 Multidimensional View of Example 2

We now consider extracting *measures* from the *Text* attribute. For example, we may seek to classify incidents based on the type of problem (e.g., “brake”, “transmission”, “engine noise” etc.) described in the text. For each problem type, we assume there exists a classifier (e.g., [27]) that outputs a discrete probability distribution function (pdf) indicating the likelihood a piece of text describes such a problem. The pdf output reflects the uncertainty inherent in such classifiers. In the example, there is a single topic “Brake”, and the classifier outputs a pdf over two values, Yes and No, indicating whether the fact’s text describes a brake problem or not. The pdf is stored in the uncertain measure attribute *Brake* as a pair of probabilities.

3 Queries

While the OLAP paradigm offers a rich array of query operators, the basic query consists of selecting a meaningful value from $dom(A_i)$ for each dimension i , and applying an aggregation operator to a particular measure attribute.

Definition 6 (Queries) For each dimension i , define a *query domain*, denoted by $qdom(A_i)$, to be some non-empty subset of $dom(A_i)$. A query Q over a database

D with schema $\langle A_1, \dots, A_k; M_1, \dots, M_n \rangle$ has the form $Q(a_1, \dots, a_k; M_i, \mathcal{A})$, where: (i) each $a_i \in qdom(A_i)$ and together a_1, \dots, a_k describe the k -dimensional region being queried, (ii) M_i describes the measure of interest, and (iii) \mathcal{A} is an aggregation function. \square

While each element a_i in $dom(A_i)$ may appear in the dimension vector for a fact, it may not be possible to assign a meaningful semantic interpretation to queries including a_i . For example, consider the Dealer dimension in Figure 2. Since the Dealer dimension has an imprecise domain, each fact mentions an arbitrary set of dealer names. However, the user may decide only queries containing either a single dealer name or all dealer names return meaningful results. Thus, $qdom(Dealer)$ is limited to the elements appearing in Figure 2. Hierarchical domains are an example of a special case where $qdom(A_i) = dom(A_i)$. In the example, $qdom(Location) = dom(Location)$, since *Location* is a hierarchical dimension. In this paper, we choose to focus mainly on the case where the query domain is hierarchical since it allows for a richer interpretation of the query results, such as in standard OLAP systems.

The result of query Q is obtained by applying \mathcal{A} to a set of facts $FIND\text{-}RELEVANT(a_1, \dots, a_k, D)$. The function $FIND\text{-}RELEVANT$ identifies the set of facts in D deemed “relevant” to the query region, and the appropriate definition of this function is an important issue addressed in this paper. All precise facts within the query region are naturally included, but we have an important design decision with respect to imprecise facts. We have three options: ignore all imprecise facts (the None option), include only those contained in the query region (the Contains option), or include all imprecise facts whose region overlaps the query region (Overlaps option).

All other queries (such as *roll-up*, *slice*, *drill-down*, *pivot*, etc.) can be described in terms of repeated applications of basic queries. We therefore concentrate on studying the semantics of basic queries in light of our two data model extensions; the extension to the full array of OLAP query operators is straightforward, and is omitted.

3.1 Using Imprecise Facts To Answer Queries

How to handle imprecise facts when answering queries is a central issue, which we now illustrate through an example. In later sections, we study the various options for determining the facts relevant to a query more rigorously. We consider aggregate queries of the type “*What are the repair costs for F150’s in the East*”?, i.e., a SUM aggregate value for the measure attribute *Repair* in the region denoted by $(F150, East)$. All queries are depicted in Figure 1 as boxes enclosing the query region. For instance, the above example query corresponds to Q5 in Figure 1.

For queries whose regions do not overlap any imprecise facts, e.g., Q1 and Q2, the set of relevant facts is clear. For other queries, e.g., Q5, this is trickier. If we use the None option, the result of Q5 is $\mathcal{A}(p1, p2)$; the imprecise fact

p9 is ignored. If we use the Contains option, the result is $\mathcal{A}(p1, p2, p9)$. Which answer is better? Using p9 to answer Q5 seems reasonable since the region for Q5 contains p9, and the result reflects all available data. However, there is a subtle issue with using the Contains option to determine relevant facts. In standard OLAP, the answer for Q5 is the aggregate of answers for Q3 and Q4, which is clearly is not the case now, since $Q3 = \mathcal{A}(p2)$ and $Q4 = \mathcal{A}(p1)$.

Observing that p9 “overlaps” the cells $c1 = (F150, NY)$ and $c2 = (F150, MA)$, we may choose to *partially* assign p9 to both cells, a process we refer to as *allocation*. The partial assignment is captured by the weights w_{c1} and w_{c2} , such that $w_{c1} + w_{c2} = 1$, which reflect the effect p9 should have on the aggregate value computed for cells $c1$ and $c2$, respectively. If the Overlaps option is used, then $Q3 = \mathcal{A}(p2, w_{c1} * p9)$ and $Q4 = \mathcal{A}(p1, w_{c2} * p9)$. Observe the user’s “expected” relationship between Q3, Q4, and Q5, which we refer to as *consistency*, is now maintained. In addition to consistency, there is a notion of result quality relative to the quality of the data input to the query, which we refer to as *faithfulness*. For example, the answer computed for Q3 should be of higher quality if p9 were precisely known.

To further illustrate the role of allocation, consider query Q6. If p10 is allocated to all cells in its region then Q6 can be answered. Otherwise, the answer to Q6 is undefined, as in regular OLAP. Although allocation can be accomplished in several ways it is reasonable to expect that allocation is query independent. For example, Q7 and Q8 must be answered using the same allocation for p10.

Consistency and faithfulness are discussed further in Sections 4 and 6. A discussion of the possible-world semantics underlying allocation is presented in Section 5, and allocation algorithms are discussed in Section 7.

3.2 Aggregating Uncertain Measures

Consider a query of the type “How likely are brake problems for the dealer Larry’s Lexus in TX?” This corresponds to query Q1 in Figure 2 where the aggregation is over the uncertain measure ‘Brake’. The answer to this query is an aggregation over the pdfs in p5 and p10 for the Brake measure. This notion of aggregating pdfs is closely related to the problem studied in the statistics literature under the name of *opinion pooling* [13]. Informally, the opinion pooling problem is to provide a *consensus* opinion from a set of opinions Θ . The opinions in Θ as well as the consensus opinion are represented as pdfs over a discrete domain O .

Many pooling operators have been studied, and the *linear operator* $LinOp$ is among the most widely used. $LinOp(\Theta)$ produces a consensus pdf \bar{P} that is a weighted *linear* combination of the pdfs in Θ , i.e., $\bar{P}(x) = \sum_{P \in \Theta} w_P \cdot P(x)$, for $x \in O$. Here, the weights are non-negative quantities summing to one. Unless there is some form of prior knowledge, we assume that the weights are uniform, i.e., $w_P = 1/|\Theta|$, in which case $\bar{P}(x)$ is just the average of the probabilities $P(x)$ for $P \in \Theta$. It is straightforward to com-

pute LinOp using aggregation functions in current OLAP systems.

While uncertainty is an important aspect of handling ambiguity in OLAP, for the rest of this paper we focus more on the issue of imprecision in dimension attributes while answering queries, and suggest the necessary extensions to the case where uncertain measures are involved.

4 OLAP Requirements

In providing support for OLAP-style queries in the presence of imprecision and uncertainty, we argue that the answers to these queries should meet a reasonable set of requirements that can be considered generalizations of requirements met by queries in standard OLAP systems. We propose two requirements for handling imprecision, namely *consistency* and *faithfulness*, which apply to both numeric and uncertain measures. (Some requirements for handling uncertainty have been proposed in [12].) We use these requirements to argue that only the Overlaps option for handling imprecision results in well-behaved queries in the context of OLAP.

4.1 Consistency

The intuition behind the consistency requirement is that a user expects to see some natural relationships hold between the answers to aggregation queries associated with different (connected) regions in a hierarchy.

Definition 7 (α -consistency) Let $\alpha(x, x_1, x_2, \dots, x_p)$ be a predicate such that each argument of α takes on values from the range of a fixed aggregation operator \mathcal{A} . Consider a collection of queries Q, Q_1, \dots, Q_p such that (1) the query region of Q is partitioned by the query regions of Q_1, \dots, Q_p , i.e., $\text{reg}(Q) = \cup_i \text{reg}(Q_i)$ and $\text{reg}(Q_i) \cap \text{reg}(Q_j) = \emptyset$ for every $i \neq j$, and (2) each query specifies that \mathcal{A} be applied to the same measure attribute. Let $\hat{q}, \hat{q}_1, \dots, \hat{q}_m$ denote the associated set of answers on D . We say that an algorithm satisfies α -consistency with respect to \mathcal{A} if $\alpha(\hat{q}, \hat{q}_1, \dots, \hat{q}_p)$ holds for every database D and for every such collection of queries Q, Q_1, \dots, Q_p . \square

This notion of consistency is in the spirit of *summarizability*, introduced in [16, 17], although the specific goals are different. Given the nature of the underlying data, only some aggregation functions are appropriate, or have the behavior the user expects.

4.2 Specific Forms of Consistency

We now define appropriate consistency predicates for the aggregation operators considered in this paper, using the notations given in the definition of α -consistency.

Definition 8 (Sum-consistency) Sum-consistency is defined as $\hat{q} = \sum_i \hat{q}_i$. \square

The above is an intuitive notion of consistency for SUM and COUNT. Since SUM is a distributive function, SUM for a query region should equal the value obtained by adding the results of SUM for the query sub-regions that partition the region. All statements for SUM in this paper are applicable to COUNT as well, and will not be explicitly mentioned in the interest of space.

Definition 9 (Boundedness-consistency) For a numeric measure, this consistency predicate is defined as $\min_i \{\hat{q}_i\} \leq \hat{q} \leq \max_i \{\hat{q}_i\}$. For an uncertain measure, the above inequalities should hold for the probabilities associated with all elements in the base domain. Formally, if $\hat{q}(o)$ is the probability for each element o , then $\min_i \{\hat{q}_i(o)\} \leq \hat{q}(o) \leq \max_i \{\hat{q}_i(o)\}$. \square

Boundedness-consistency is intuitively appropriate for any kind of averaging operator for numeric measures and aggregation operator for uncertain measures. In particular, AVERAGE for a query region should be within the bounds of AVERAGE for the query sub-regions that partition the region. In the case of LinOp, the same property should hold element-wise for the associated pdfs.

An important consequence of the various α -consistency properties defined above is that the Contains option is unsuitable for handling imprecision, as shown below:

Theorem 1 *There exists a SUM aggregate query which violates Sum-consistency when the Contains option is used to find relevant imprecise facts in FIND-RELEVANT.*

Similar theorems can be shown for other aggregation operators as well, but we omit them in the interest of space.

4.3 Faithfulness

Starting with a database D , suppose we increase imprecision in D by mapping facts in the database to larger regions. We expect that the answer to any query Q on this new database D' will be different from the original answer. Faithfulness is intended to capture the intuitive property that this difference should be as small as possible. Since an aggregation algorithm only gets to see D' as its input and is not aware of the “original” database D one cannot hope in general to state precise lower and upper bounds for this difference. Our aim instead will be to state weaker properties that characterize this difference, e.g., whether it is monotonic with respect to the amount of imprecision. The following definition is helpful in formalizing faithfulness.

Definition 10 (Measure-similar Databases) We say that two databases D and D' are *measure-similar* if D' is obtained from D by (arbitrarily) modifying (only) 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' . \square

Consider a query Q such that every fact region is either completely contained within the query region of Q or completely

disjoint from it. In such a situation, it is reasonable to treat the facts as if it were precise with respect to Q since the imprecision in the facts does not cause ambiguity with respect to the query region of Q . The first form of faithfulness formalizes this property, and is illustrated in Figure 3a.

Definition 11 (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 $\text{reg}(r)$ and $\text{reg}(r')$ are completely contained in $\text{reg}(Q)$ or both are completely disjoint from $\text{reg}(Q)$. We say that an algorithm satisfies *basic faithfulness* with respect to an aggregation function \mathcal{A} if for every query Q that uses \mathcal{A} , the algorithm gives identical answers for every pair of measure-similar databases D and D' that are identically precise with respect to Q . \square

Basic faithfulness enables us to argue that the None option of handling imprecision by ignoring all imprecise records is inappropriate, as we intuitively expect:

Theorem 2 *SUM, COUNT, AVERAGE and LinOp violate basic faithfulness when the None option is used to handle imprecision.*

Theorems 1 and 2 demonstrate the unsuitability of the Contains and None options for handling imprecision, and we do not consider them further. The remaining option, namely Overlaps, is the focus of our efforts for the rest of the paper, and it raises the challenge of how to handle relevant facts that partially overlap the query region. We tackle this problem in later sections using an allocation-based approach to summarizing the “possible worlds” that may have led to the imprecise dataset.

The next form of faithfulness is intended to capture the same intuition as basic faithfulness in the more complex setting of imprecise facts that partially overlap a query. We first define an ordering that compares the amount of imprecision in two databases with respect to a query Q , in order to reason about the answers to Q as the amount of imprecision grows.

Definition 12 (Partial order \preceq_Q) 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 pair of facts $r \in D$ and $r' \in D'$ such that $\text{reg}(r')$ is obtained from $\text{reg}(r)$ by adding a cell $c \notin \text{reg}(Q) \cup \text{reg}(r)$. Define the partial order \preceq_Q to be the reflexive, transitive closure of I_Q . \square

Figure 3b illustrates the definition of \preceq_Q for a query Q —the amount of imprecision for every fact $r' \in D'$ is larger than that of the corresponding fact $r \in D$ but only in the cells outside the query region. The reason for this restriction is that allowing r' to have a larger projection inside the query region does not necessarily mean that it is less relevant to Q than r (cf. basic faithfulness).

Definition 13 (β -faithfulness) Let $\beta(x_1, x_2, \dots, x_p)$ be a predicate such that the value taken by each argument of β belongs to the range of a fixed aggregation operator \mathcal{A} .

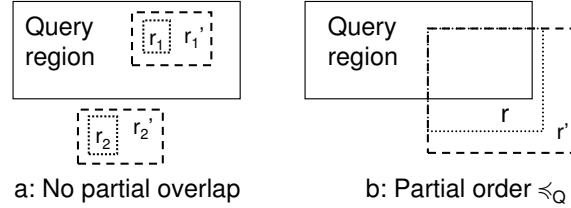


Fig. 3 Two forms of Faithfulness

We say that an algorithm satisfies β -faithfulness with respect to \mathcal{A} if for any query Q compatible with \mathcal{A} , and for any set of databases $D_1 \preceq_Q D_2 \preceq_Q \dots \preceq_Q D_p$, the predicate $\beta(\hat{q}_1, \dots, \hat{q}_p)$ holds true where \hat{q}_i denotes the answer computed by the algorithm on D_i , i in $1 \dots p$. \square

4.4 Specific Forms of Faithfulness

We now discuss how β -faithfulness applies to the aggregation operations considered in this paper.

SUM: If we consider SUM over non-negative measure values, the intuitive notion of faithfulness is that as the data in a query region becomes imprecise and grows outside the query region, SUM should be non-increasing.

Definition 14 (Sum-faithfulness) Sum-faithfulness is defined as follows: if $D_1 \preceq_Q D_2$, then $\hat{q}_{D_1} \geq \hat{q}_{D_2}$. \square

AVERAGE and LinOp: It is difficult, unfortunately, to define an appropriate instance of β -faithfulness for AVERAGE and LinOp. Consider how the AVERAGE behaves as facts in a query region become more imprecise and grow outside the query region: SUM for the query region diminishes, but the count also decreases. Since both the numerator and denominator are decreasing, the value of AVERAGE could either increase or decrease. The same observation applies to LinOp as well.

5 Allocation

In this section, we describe the basic framework for handling imprecision when the Overlaps option is used to find relevant facts. For the remainder of the paper, we assume the Overlaps option is used. For each imprecise fact, we consider all possible completions of the fact to cells in its region and associate a weight with that completion. The weights intuitively represent our belief regarding the possible completions. This is formally defined as follows:

Definition 15 (Allocation Policy and Extended Data Model) Let r be a fact in the fact table. For each cell $c \in \text{reg}(r)$, the *allocation* of fact r to cell c , denoted by $p_{c,r}$, is a non-negative quantity denoting the weight of completing r to cell c . We require that $\sum_{c \in \text{reg}(r)} p_{c,r} = 1$. An *allocation policy* is a procedure that takes as its input a fact

	ID	Loc	Auto	Repair	Alloc
1	p1	NY	F-150	\$200	1
2	p2	MA	F-150	\$250	1
3	p3	CA	F-150	\$150	1
4	p4	TX	Sierra	\$300	1
5	p5	TX	Camry	\$325	1
6	p6	TX	Camry	\$175	1
7	p7	TX	Civic	\$225	1
8	p8	TX	Civic	\$120	1
9	p9	MA	F150	\$140	0.6
10	p9	NY	F150	\$140	0.4
11	p10	TX	F150	\$500	0.3
12	p10	TX	Sierra	\$500	0.7

Table 3 Extended database for Table 1

table consisting of imprecise facts and produces as output the allocations of all the imprecise facts in the table. The result of applying such a policy to a database D is an *extended database* D^* ($EDBD^*$). The schema of D^* , referred to as the *extended data model* (EDM), contains all the columns of D plus additional columns to keep track of the cells that have strictly positive allocations. Suppose that fact r in D has a unique identifier denoted by $ID(r)$. Corresponding to each fact $r \in D$, we create a set of fact(s) $\langle ID(r), r, c, p_{c,r} \rangle$ in D^* for every $c \in reg(r)$ such that $p_{c,r} > 0$ and $\sum p_{c,r} = 1$. By default, each precise fact has a single allocation of 1 for the cell to which it maps. \square

Example 2 Consider the fact table in Table 1, whose multidimensional view is shown in Figure 1. An extended database for this fact table is shown in Table 3. Here, the extended database contains the allocations of imprecise facts $p9$ and $p10$. For example, allocations for $p9$ are 0.6 and 0.4 for cells $(MA, F150)$ and $(NY, F150)$ respectively. Similarly, the allocations for $p10$ be 0.3 and 0.7 corresponding to $(TX, F150)$ or $(TX, Sierra)$.

Consider the second example which contains dimensions extracted from text (cf. Table 2 and its corresponding multidimensional view, Figure 2). Here the allocations will correspond to cells containing cities and individual dealer names (including NONE). Suppose the allocations for $p1$ equal 0.6 and 0.4 for cells $(NY, Capitol Chevy)$ and $(NY, Bernal BMW)$. The resulting extended database D^* will consist of two entries corresponding to fact $p1$ —one belonging to $(NY, Capitol Chevy)$ with weight 0.6 and another to $(NY, Bernal BMW)$ with weight 0.4, both tagged with the same identifier. The other imprecise facts are handled similarly. Facts $p2, p4, p7$ each have at most 2 corresponding entries in D^* , while fact $p9$ has at most 4 entries in D^* .

From the two preceding examples, we see that both imprecise and hierarchical dimensions are handled identically in our data model. Allocation policies used to assign the allocation weights are described in detail in Section 7. The size of the extended database increases only linearly in the number of imprecise facts. However, since the region of an imprecise fact is exponentially large in the number of dimen-

sion attributes which are assigned non-singleton sets, care must be taken in determining the cells that get positive allocations.

6 Query Semantics

Consider the Extended Database D^* produced using an allocation policy \mathcal{A} . We now give a probabilistic interpretation of D^* motivated by the condition that allocations of each imprecise fact sum to 1. Let the input fact table $D = \{r_1, r_2, \dots, r_N\}$. Associated with each fact $r \in D$ is a random variable $C(r)$ which indicates the possible completions of r ; $C(r)$ equals c with probability $p_{c,r}$. We view the allocation weight $p_{c,r}$ as the *probability* that fact r completes to c . D^* is obtained by applying \mathcal{A} to D , and contains for each fact $r \in D$ a set of facts representing possible completions of r along with the probability $p_{c,r}$ of this completion.

We can interpret D^* in terms of the following probabilistic process. For each $r \in D$, we pick a cell c and modify the dimension attributes in r so that the resulting fact belongs to cell c . The end result of this process is a database containing only precise facts. Such a database is referred to as a *possible world* for the input fact table D , a notion that has been considered previously [1, 10]. The *weight* of a possible world is defined to be the probability of generating that world; hence, the weights of the possible worlds sum to 1. To formalize this, we associate the joint random variable $\mathcal{C} = (C(r_1), C(r_2), \dots, C(r_N))$ with the above process. Each value for \mathcal{C} is a set of precise facts $\{C(r_1), C(r_2), \dots, C(r_N)\}$ representing one of the possible worlds encoded by EDB D^* . Thus, we use \mathcal{C} to refer to the set of possible worlds.

Example 3 Consider the fact table in Table 1. The Extended Database produced as a result of some allocation policy is shown in Table 3, The multidimensional views of Table 1, and the possible worlds encoded by Table 3, are shown in Figure 4. The two choices of possible completions for both $p9$ and $p10$ result in $(2 * 2)$ possible worlds $\{D_1, D_2, D_3, D_4\}$ as shown in the figure. Each world D_i has an associated weight w_i . For example, one possible assignment of values to the w_i is $\{w_1, w_2, w_3, w_4\} = \{0.25, 0.25, 0.25, 0.25\}$.

Observe the query semantics for any query Q are well-defined for each possible world, which is a database consisting only of precise facts. Thus, we consider the multiset of results $\{v_1, \dots, v_m\}$ obtained by executing Q on each of the possible worlds $\{D_1, \dots, D_m\}$ ¹. The main problem tackled in the rest of this section is to define the appropriate semantics for summarizing this multiset of answers to produce a single answer. Two issues make this challenging. First, we need to ensure the desiderata of consistency and faithfulness are satisfied. Second, we require efficient algorithms for computing the summarized answer.

¹ The answers form a multiset because multiple possible worlds may give the same answer.

We now address the first issue. Since the weights of the possible worlds sum to 1, we can interpret $\{v_1, \dots, v_m\}$ as a distribution of answers to query Q over the possible worlds. Let $Q(\mathcal{C})$ be a random variable associated with this distribution of values. For any value $v \in \{v_1, \dots, v_m\}$, we have $\Pr[Q(\mathcal{C}) = v] = \sum_{i:v_i=v} w_i$, where w_i is the weight of possible world D_i . For this discussion, we assume v_i are numerical; in Section 6.1.3 we consider aggregating uncertain measures. One useful summarization of this distribution is the expected value of $Q(\mathcal{C})$, $E[Q(\mathcal{C})]$, and the following theorem offers some justification for its use.

Theorem 3 *Basic faithfulness is satisfied if answers to queries are computed using the expected value of the answers in each of the possible worlds.* \square

We now present algorithms for efficiently computing the expected value of the answers over the possible worlds, $E[Q(\mathcal{C})]$. This is challenging because the number of possible worlds grows exponentially in the number of imprecise facts. A *naive algorithm* for finding $E[Q(\mathcal{C})]$ would: 1) enumerate all possible worlds, 2) compute $Q(D_i) = v_i$ for each possible world D_i , and 3) compute the expected value $E[Q(\mathcal{C})] = \sum_{i=1}^m v_i * w_i$. Clearly, the exponential number of possible worlds makes this approach infeasible. Therefore, the key is to compute $E[Q(\mathcal{C})]$ without explicitly materializing all possible worlds. To this end, we propose algorithms for answering Q using the Extended Database D^* , which only grows *linearly* in the number of imprecise facts.

We will show the naive algorithm over the possible worlds is *equivalent* to an algorithm over the entries in the Extended Database D^* , i.e., $Q(D^*) = E[Q(\mathcal{C})]$, provided certain conditions on the assignments of weights to the possible worlds are met. In other words, if the (discrete) pdf p assigned to \mathcal{C} meets certain conditions, then we can achieve the desired possible worlds query semantics without enumerating all possible worlds. We will give the conditions that such pdfs must meet at the end of this section.

These conditions should become clear after we derive the “framework” for algorithms which use the Extended Database D^* to answer Q . Fix a query Q , which has associated region q and aggregation operator \mathcal{A} . Let $\text{Comp}(r, c)$ denote the *precise fact* obtained by the completion of r to cell c . In the Extended Database D^* , $\text{Comp}(r, c)$ is stored along with its allocation $p_{c,r}$. The set of facts that potentially contribute to the answer are those that have positive allocation to q , and are given by $\mathcal{R}(Q) = \{r \mid \text{Comp}(r) \cap q \neq \emptyset\}$. We say that $\mathcal{R}(Q)$ is the set of *candidate facts* for query Q . For each candidate fact r , let Y_r be the 0-1 random variable for the event “ $C(r)$ maps to a cell in q ”. We have,

$$\Pr[Y_r = 1] = \sum_{c: c \in \text{reg}(Q)} p_{c,r} \quad (1)$$

By definition $\Pr[Y_r = 1]$ is the sum of the w_i of the possible worlds D_i in which r completes to a cell in q . Since Y_r is a 0-1 random variable, $\Pr[Y_r = 1] = E[Y_r]$. *Notice the right-hand side of Equation 1 can be obtained from the Extended Database D^* . With a slight abuse of notation, we*

say that $E[Y_r]$ is the *allocation* of r to the query Q ; it is *full* if $E[Y_r] = 1$ and *partial* otherwise.

Consider computing the answer to Q for a possible world \mathcal{C} . Let $r.M$ be the measure attribute on which Q is executed. Then, the (multiset) of values relevant to query Q in possible world D_i equals $\{r.M \mid Y_r = 1\}$, and

$$Q(\mathcal{C}) = v_i = \mathcal{A}(\{r.M \mid Y_r = 1\}) \quad (2)$$

Thus, we only require the values for $\Pr[Y_r = 1]$ to describe their joint distribution, which are obtainable from D^* (i.e., they are the $p_{c,r}$ values associated with each entry in D^*). Observe, this *marginalization condition* is the only assumption regarding the joint distribution of the Y_r we have made so far (i.e., for each Y_r , the individual marginal distribution of the joint distribution w.r.t. Y_r equals $P[Y_r = 1]$). This suggests the following “framework” for *EDB-based Algorithms* for obtaining $E[Q(\mathcal{C})]$:

Step 1: Identify the set of candidate facts $r \in \mathcal{R}(Q)$ 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 in D^* that share the same identifier in the ID column and then summing the allocations (i.e., the $p_{c,r}$) within each group. At the end of this step, we have a set of facts containing for each fact $r \in \mathcal{R}(Q)$, the allocation of r to Q and the measure value associated with r . Note this step depends only on the query region q .

Step 2: Apply the aggregation operator \mathcal{A} specified by Q to the result from Step 1. This step is specialized to each \mathcal{A} , and leads to the different algorithms below. We observe for some aggregation operators, merging this second step with the first is possible in order to gain further savings, e.g., the expected value of SUM can be computed thus. This extra optimization step will not be discussed further.

For what assignment of weights to the possible worlds does $Q(D^*) = E[Q(\mathcal{C})]$? We will answer this question in two steps. First, we will make the independent completion assumption, i.e., that each fact r is completed independently of the other facts in D . This corresponds to assuming the $C(r)$ are jointly independent, and specifies a unique weight assignment to the possible worlds (i.e., unique pdf p_{IND} over \mathcal{C}). Second, we show that if any pdf p_M satisfying the marginal distributions given by the $C(r)$ is used to assign weights w_i to \mathcal{C} , then the expected value over the possible worlds \mathcal{C} remains the same.

6.1 Independent Completion Assumption

For the results presented in this section, we make the independent completion assumption. The assumption that all $C(r)$ are jointly independent can be captured by the following random process: Independently, for each fact r , we pick a cell c with probability $p_{c,r}$ and modify the dimension attributes in r so the resulting fact belongs to c . The probability assigned to each possible world generated by this process equals the product of the allocations $p_{c,r}$ where fact r

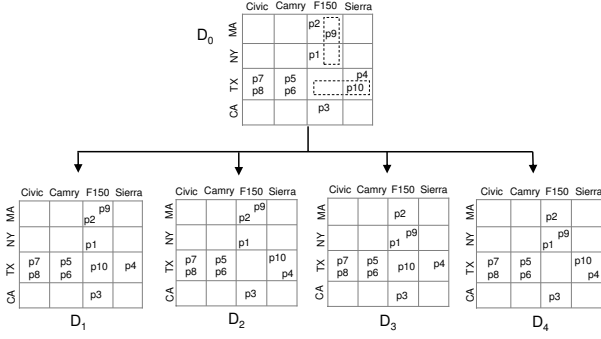


Fig. 4 Possible Worlds

is completed to cell c in that world. Weights assigned in this manner are referred to as weights assigned by the *independent completion assumption*.

Example 4 Consider the possible worlds shown in Figure 4 and described in Example 3. From the independent completion assumption, the weights of the possible worlds can be derived using the allocations of p_9 and p_{10} . The allocations of the remaining facts equal 1 by default since they are precise. The weight associated with D_1 equals $0.6 \cdot 0.3$, corresponding to the allocation 0.6 of p_9 to $(MA, F150)$ and the allocation 0.3 of p_{10} to $(TX, F150)$. The weights for the other possible worlds can be derived analogously, and the weights for all worlds are $\{w_1, w_2, w_3, w_4\} = \{0.18, 0.42, 0.12, 0.28\}$. \square

We now show that our desired possible world query semantics hold for several aggregation operators given the independent completion assumption.

6.1.1 SUM

The random variable corresponding to the answer for a SUM query Q is given by $Q(C) = \sum_{r \in \mathcal{R}(Q)} v_r Y_r$, where v_r is the value of the numerical measure for record r . Using this expression, we can efficiently compute the expectation for SUM:

Theorem 4 *The value of SUM can be computed in a single pass through EDB D^* , and is equivalent to the expected value of computing sum over the possible worlds given the independent completion assumption. The expectation for SUM satisfies Sum-Consistency.* \square

For SUM, β -faithfulness can be violated if the extended data model was built using arbitrary allocation policies. We define a class of allocation policies for which we can guarantee faithfulness. Such allocation policies will be discussed in Section 7.

Definition 16 (Monotone Allocation Policy) Let D and D' be two similar data sets with the property that the associated regions are identical for every pair of corresponding facts, except for a single pair (r, r') , $r \in D, r' \in D'$ such that

$\text{reg}(r') = \text{reg}(r) \cup \{c^*\}$, for some cell c^* . Fix an allocation policy A , and let $p_{c,r}$ (resp. $p'_{c,r}$) denote the resulting allocations in D (resp. D') computed with respect to A . We say that A is a *monotonic* allocation policy if $p_{c,s} \geq p'_{c,s}$ for every fact s and for every cell $c \neq c^*$. \square

Monotonicity is a strong but reasonable and intuitive property of allocation policies. When the database has no imprecision, there is a unique possible world with weight 1. But as the amount of imprecision increases, the set of possible worlds will increase as well. Monotone allocation policies restrict the way in which the weights for the larger set of possible worlds are defined. In particular, as a region gets larger, allocations for the old cells are redistributed to the new cells.

Theorem 5 *The expectation of SUM satisfies Sum-faithfulness if the allocation policy used to build the extended database is monotone.* \square

6.1.2 AVERAGE

In this case, the random variable corresponding to the answer is given by $Q(C) = \frac{\sum_{r \in \mathcal{R}(Q)} v_r Y_r}{\sum_{r \in \mathcal{R}(Q)} Y_r}$. Unfortunately, computing even the expectation becomes difficult because of the appearance of Y_r in both the numerator and denominator. As shown in the following theorem, we devise a non-trivial algorithm for AVERAGE.

Theorem 6 *Let n and m be the number of partially and completely allocated facts in a query region, respectively. The exact expected value of AVERAGE can be computed in time $O(m+n^3)$, with n passes over the set of candidate facts. Also, the value of exact AVERAGE computed using EDB D^* is equal to the expected value over the possible worlds, given the independent completion assumption.* \square

The above algorithm is based on computing an appropriate recurrence that results from expanding the expression for the expectation in a suitable manner. While this algorithm is theoretically feasible, the cost of computing the exact AVERAGE is high if the number of partially allocated facts for Q is high. To address this, we consider an *alternative* estimate for AVERAGE which does not involve summarizing the answers to AVERAGE on each of the possible worlds. Instead, we will consider expected values of SUM and COUNT on the possible worlds and take their ratio. Formally, this yields

$$\frac{E[\text{SUM}]}{E[\text{COUNT}]} = \frac{E[\sum_{r \in \mathcal{R}(Q)} v_r Y_r]}{E[\sum_{r \in \mathcal{R}(Q)} Y_r]}$$

Theorem 7 *An alternative estimate for AVERAGE can be computed in time $O(m+n)$ using a single pass over the set of candidate facts in EDB D^* , and is equivalent to the expected value computed directly over the possible worlds, given the independent completion assumption. The relative error of the estimate is negligible when $n \ll m$.* \square

The assumption of $n \ll m$ in the theorem above is reasonable for most databases since we expect that the fraction of

facts with missing values that contribute to any query will be small.

We now compare our two solutions for AVERAGE, namely the exact and the alternative estimate in terms of the requirements. First, we can show that:

Theorem 8 *The expectation of the AVERAGE computed from the extended data model satisfies basic faithfulness but violates Boundedness-consistency.* \square

On the other hand:

Theorem 9 *The alternative estimate for AVERAGE defined above satisfies Boundedness-consistency and basic faithfulness.* \square

The above theorems show the tradeoff in answering queries between accuracy (the expected value is a good estimator of the distribution of answers) and consistency. Given the efficiency and the desiderata aspects, and the small relative error (under reasonable conditions) for the alternative estimate, we propose using this estimate for answering queries.

6.1.3 Uncertain Measures

In Section 6.1.3 we proposed LinOP as a reasonable aggregation operator for uncertain measures. We now address the issue of summarizing LinOP over the possible worlds. One approach is to compute LinOP over all the facts in all the worlds simultaneously, where the facts in a world D_i are weighted by the probability of that world w_i . This is somewhat analogous to the alternative estimate for AVERAGE described above.²

Definition 17 (AggLinOp) Let D_1, D_2, \dots, D_m be the possible worlds with weights w_1, w_2, \dots, w_m respectively. Fix a query Q , and let $W(r)$ denote the set of i 's such that the cell to which r is mapped in D_i belongs to $\text{reg}(Q)$. AggLinOp is defined as

$$\frac{\sum_{r \in \mathcal{R}(Q)} \sum_{i \in W(r)} v_r w_i}{\sum_{r \in \mathcal{R}(Q)} \sum_{i \in W(r)} w_i},$$

where the vector v_r represents the measure pdf of r . \square

Similar to the approximate estimate for AVERAGE, AggLinOp can be computed efficiently, and satisfies similar kinds of requirements.

Theorem 10 *AggLinOp can be computed in a single pass over the set of candidate facts, and satisfies Boundedness-Consistency and Basic Faithfulness. The value of AggLinOp computed using the EDB D^* is equivalent to expected answer to Q evaluated over all possible worlds, given the independent completion assumption.* \square

² A summarizing estimate for uncertain measures that is analogous to the exact estimate for AVERAGE can also be defined but is not considered here because it has the same drawbacks.

6.2 Relaxing the Independent Completion Assumption

We have shown our desired possible worlds query semantics can be maintained by algorithms using the EDB D^* to answer queries if we make the independent completion assumption. Ideally, we would like to “relax” this assumption by showing that for a set of pdfs \mathcal{P} over the possible worlds \mathcal{C} , the desired possible world query semantics still hold, or that $Q(D^*) = E[Q(\mathcal{C})]$. Observe that the unique pdf defined by the independent completion assumption p_{IND} is contained in \mathcal{P} . This would represent a significant degree of flexibility, since D^* becomes a semantically valid representation instead for a collection of weight assignments to the possible worlds. Observe, that D^* remains the same, but the weights W_i assigned to the possible worlds D_i now vary.

Recall the allocation weights $p_{c,r}$ are the probability that fact r completes to c . A pleasing alternative view of the $C(r)$ is as *marginal distributions* over the possible worlds, with each $p_{c,r} = \sum_{i: \text{Comp}(r,c) \text{ in } D_i} w_i$. Then, each $p \in \mathcal{P}$ can be viewed as a pdf over \mathcal{C} satisfying the constraints given by these marginal distributions. Let \mathcal{C}_I designate using p_{IND} to assign w_i to the possible worlds \mathcal{C} , and likewise \mathcal{C}_M designate using any $p \in \mathcal{P}$ to assign the w_i . Thus, we are interested in showing that $Q(D^*) = E[Q(\mathcal{C}_I)] = E[Q(\mathcal{C}_M)]$. The first equality was shown to hold for several aggregation operators in Section 6.1. The second equality is proven as part of the proof of the following claim.

Theorem 11 *Assume we are given an imprecise database D , and the corresponding EDB D^* . Let \mathcal{P} be the set of (discrete) pdfs assignable to the space of possible worlds \mathcal{C} such that each $p_M \in \mathcal{P}$ satisfying the marginal distributions given by the $C(r)$.*

If any $p_M \in \mathcal{P}$ is selected as the pdf over \mathcal{C} , then for any query Q using one of the following aggregation operators, the value of $Q(D^) = E[Q(\mathcal{C})]$:*

1. SUM
2. COUNT
3. Alternative AVERAGE
4. AggLinOp

\square

We assume the distributions of each $C(r)$ are specified by the allocation policy \mathcal{A} used to create the EDB D^* . The set of pdfs which satisfy the constraints outlined in Theorem 11, and from which p may be selected, is formally defined by the following system of linear equations.

Definition 18 (Set of Valid pdfs) Assume we are given the distributions for each $C(r)$. Then, the set of (discrete) pdfs over the possible worlds which satisfies the marginal distributions given by $C(r)$ are solutions to a linear system created as follows: Introduce a variable for each w_i , and an equation of the following form for each $p_{c,r} = \sum_{i: \text{Comp}(r,c) \text{ in } D_i} w_i$, and the $\sum_{i=1}^m w_i = 1$, where m is the number of possible worlds.

We refer to solutions $\langle w_1, \dots, w_m \rangle$ of this linear system as the set \mathcal{P} of pdfs over the possible worlds which are *valid*,

since they all satisfy the marginalization condition on such distributions given in Theorem 11. \square

Example 5 Consider Example 3, which results in the 4 possible worlds shown in Figure 4. The two choices of possible completions for both $p9$ and $p10$ result in $(2 * 2)$ possible worlds $\{D_1, D_2, D_3, D_4\}$ as shown in the figure. The set of pdfs for the possible worlds are all solutions to the following linear system

$$\begin{aligned} w_1 + w_2 &= 0.6 \\ w_3 + w_4 &= 0.4 \\ w_1 + w_3 &= 0.3 \\ w_2 + w_4 &= 0.7 \\ w_1 + w_2 + w_3 + w_4 &= 1 \end{aligned}$$

The system can be simplified (using basic algebra) to the following:

$$\begin{aligned} w_1 + w_2 &= 0.6 \\ w_3 + w_4 &= 0.4 \\ w_1 + w_3 &= 0.3 \end{aligned}$$

which has an infinite number of solutions, since the system is under-constrained (3 equations, 4 unknowns). Any assignment to $\{w_1, w_2, w_3, w_4\}$ which is a solution to this linear system defines a valid pdf for the possible worlds. \square

7 Allocation Policies

In the previous section, we designed efficient algorithms for various aggregation operators using the extended database, and proved several consistency and faithfulness properties. We now turn to the task of creating the extended database via appropriate allocation policies.

We first explain the assumptions behind our approach, and from this propose an objective function that allocation policies should attempt to maximize. Using this objective function, we derive a general allocation template in Section 7.1, and demonstrate how various allocation policies are instantiations of this template. In Section 7.2, we provide a mechanism allowing users to specify additional information that can be used to assign the allocation weights.

Recall that every imprecise fact r maps to a collection of cells denoted by $\text{reg}(r)$. Each cell c in $\text{reg}(r)$ is a candidate for completing r , and our goal is to design algorithms to obtain the allocation $p_{c,r}$ associated with this completion.

We assume the fact table D was created by randomly adding imprecision to facts in an unobservable precise fact table \widehat{D} . We would like to however capture the intuition that certain choices for \widehat{D} are preferable over others. To that end, we will assume that there is a statistical model for generating facts in \widehat{D} . This statistical model will be assumed to

belong to a family of statistical models parameterized by a set of arguments θ . Several choices exist for choosing a parametric family, and by choosing one, the user implicitly assumes the corresponding correlations in the data captured by the parametric family. These correlations can be classified as follows:

None: There is no correlation present.

Dimension-based: The correlation is only between the dimension attributes, and the values for measure attributes are assumed to be arbitrary. Here the parametric family is given by $P(A_1, \dots, A_k; \theta)$

General: The correlation is between the dimension as well as measure attributes. Here the parametric family is given by $P(A_1, \dots, A_k, M_1, \dots, M_n; \theta)$

When there is no correlation present, the most principled approach is the *uniform allocation policy*, where each fact r is uniformly allocated to every cell in $\text{reg}(r)$. This allocation policy can be shown to be monotone (cf. Definition 16).

For the dimension-based and general correlation, our goal is to compute the values for θ via an appropriate algorithm. The “best” estimate of θ is considered to be one that maximizes the likelihood of data D , under the assumption that the imprecision in D was created uniformly at random. Once θ has been determined, let $P(c; \theta)$ denote the probability of a cell c as determined by θ . The allocation of a fact r and cell c is given by:

$$p_{c,r} = \frac{P(c; \theta)}{\sum_{c' \in \text{reg}(r)} P(c'; \theta)}$$

Example 6 Consider the following dimension-based correlation. Let $P(c; \theta)$ be a parametric model denoting the probability of generating a precise fact that belongs to cell c under θ . In other words, $\theta = \{\theta(c) | c \text{ is a cell}\}$, and $P(c; \theta) = \theta(c)$. For the given fact table D , the likelihood function is given as follows:

$$\mathcal{L}(\theta) = \prod_{r \in D} \sum_{c \in \text{reg}(r)} \theta(c).$$

Note that the likelihood function includes both precise and imprecise facts. \square

The objective function for correlations involving measures can be obtained in a similar manner. For uncertain measures, this requires additional thought since the measure values are probability distributions. Fix a measure attribute, and let v_r denote the associated measure value; this is a vector of probabilities where $v_r(o)$ is the probability associated with the base domain element o . If v_r is viewed as an empirical distribution induced by a given sample (i.e., defined by relative frequencies of the base elements o in the sample), then uncertain measures are simply summaries of several individual observations for each fact. Consequently, the likelihood function for this case can be written as well.

7.1 Allocation Policy Template

The likelihood functions that we consider have a similar structure which is instantiated using different quantities. Maximizing the likelihood function is a non-linear optimization problem, and the standard approach for such problems is the well-known Expectation Maximization (EM) algorithm [8]. The details of the fairly standard derivation are omitted for clarity. The result of this derivation is an iterative algorithm in which certain quantities are updated iteratively. The equations for the iterative update follow a template involving a quantity $\delta(c)$. The exact definition of the quantity $\delta(c)$ depends on the particular likelihood function but the basic property is that $\delta(c)$ is determined by only the precise facts in D belonging to cell c , and remains fixed throughout the algorithm. We now introduce 2 quantities that vary during the algorithm: let $\Delta^{(t)}(c)$ be the updated quantity assigned to c during step t to account for all facts r overlapping c , and let $\Gamma^{(t)}(r)$ denote the quantity associated with fact r . Then, the iterative EM updates will have the following template:

$$\begin{aligned}\Gamma^{(t)}(r) &= \sum_{c': c' \in \text{reg}(r)} \Delta^{(t-1)}(c') \\ \Delta^{(t)}(c) &= \delta(c) + \sum_{r: c \in \text{reg}(r)} \frac{\Delta^{(t-1)}(c)}{\Gamma^{(t)}(r)}\end{aligned}$$

Instantiating this template using different quantities leads to different allocation policies. If this template is instantiated using counts of facts mapped to cells, we obtain *EM-Count* allocation. Instantiating this template with measure values results in the *EM-Measure* allocation policy. When the algorithm converges, there will be post-processing step in which both the parameters θ and the allocations can be recovered from Δ and Γ .

7.2 Including Additional Information

The allocation template presented above assigns allocation weights by only considering information in the given fact table D . While this approach is reasonable, it suffers from two potential weaknesses. First, the user may have additional information they would like used for assigning allocation weights. Second, the allocations are constrained to use only those cells for which there is at least one precise fact in D . We propose supporting prior information for the values of θ as a means to circumvent these problems. For the given quantity of interest, assume the user assigns an initial value to each cell. Together, the values assigned to the cells c are the user’s belief regarding the corresponding correlations present in the data. For example, if the quantity is count, the assigned values can be interpreted as encoding the user’s belief regarding the correlations for the cells, with the size of these initial values indicates the relative strength of the user’s belief.

A pleasing interpretation of this prior information is to regard it as a “pseudo” precise fact table D_p that is implicitly observed along with the given fact table D . This interpretation of priors is consistent with the “method of equivalent data”.

With this interpretation, the following straightforward procedure can be used to include prior information:

1. Initialize all cell quantities $\delta(c)$ using D_p .
2. Update cell quantities using the precise portion of the given D .
3. Update $\Delta(c)$ using the EM-equations. The final allocation weights are obtained as before.

An alternative equivalent formulation of prior information is also possible. Instead of specifying an initial value for each cell c , the user instead suggests allocation weights for each imprecise fact r (i.e., assigns $p_{c,r}$ for each $c \in \text{reg}(r)$). If the priors are satisfiable, then it is possible to create a precise D_p which exactly encodes them. Priors specified in such a manner can be thought of as specifying linear constraints between cell values used for allocation. Notice this is equivalent to specifying a correlation structure between cell values.

8 Experiments

In this section we evaluate the main contributions of this paper, namely handling imprecision and uncertainty in an OLAP setting. To this end we designed and conducted experiments to evaluate both scalability and quality. The scalability experiments targeted the construction and the querying of the extended database; the quality experiments targeted the performance of different allocation policies under varying characteristics of the data.

8.1 Scalability of the Extended Data Model

The experiments were conducted on a 2.4GHz Pentium 4 machine with 1 GB physical memory and a single IDE disk. The back-end was a commercial relational database system with buffer pool size set to 100 MB. No materialized views or indices were built on the data.

To provide a controlled environment for evaluation, we used synthetically generated data consisting of 4 dimensions. 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).

Figure 5 plots the running time for the different allocation policies. Note that they all increase (almost) linearly with respect to the number of imprecise records. For all algorithms, the running time has 2 components, one for processing the input data and the other for writing out the facts to the extended database.

For both EM-Count and EM-Measure the first component is variable, and depends on the number of iterations being performed. To examine the effect of the iterations on performance, we ran both algorithms for 1 and 5 iterations. Since the running times are very similar for the two EM algorithms for a fixed number of iterations, the curves overlap in the figure. Running the iterative EM algorithms for a single iteration is the fairest comparison to Uniform. From the figure, we see both perform better than Uniform, with the larger running time explained due to the second component. Since the input data density is low, Uniform allocates to many empty cells, so the number of entries in the extended database created by Uniform is significantly larger than EM-Count and EM-Measure. For example, with 25% imprecision, Uniform generated an extended database 14.5 million facts whereas EM-Count and EM-Measure each generated databases with 2.3 million facts. This relative difference between Uniform and EM methods run with a single iteration should increase as the input data density decreases.

For a larger number of iterations, the first component dominates the running time in both EM algorithms, and Uniform outperforms both of them. This is not surprising, and shows that the number of iterations for the EM algorithms impacts performance.

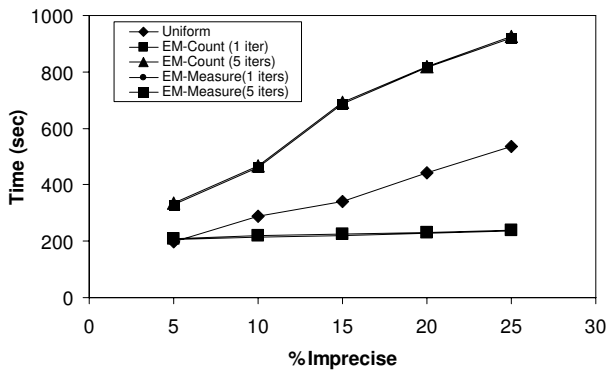


Fig. 5 Allocation Algorithm Performance Results

The second experiment evaluated the performance of standard OLAP point queries using the extended databases created above. For example, SUM can be calculated in SQL

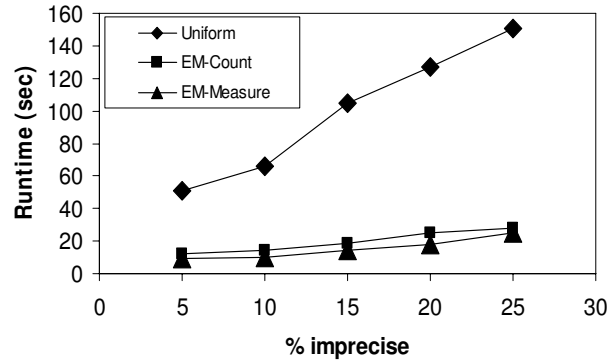


Fig. 6 Query Performance Results

using the following template:

```
SELECT dim-values, SUM (measure * weight),
FROM fact-table, dim-tables
WHERE qualification-list GROUP BY dim-values
```

To understand runtime behavior we randomly generated a total of 25 queries by choosing a random level and node for each dimension. Figure 6 shows the average query running time for SUM. Since the runtime behavior of LinOp and AVERAGE (approximate estimate) were similar, they are omitted in the interests of space. In general the running time was dominated by the I/O cost for scanning the extended database. As seen above, this is much higher for Uniform than for EM-Count or EM-Measure.

8.2 Quality of the Allocation Policies

These experiments evaluate how data characteristics affect the behavior of our proposed allocation policies. We define a new notion of density called *pseudo-density* as the expected number of facts in each non-empty cell for a precise database. Intuitively, high pseudo-density ensures that no empty cells are created as facts are made imprecise (i.e., cells with precise records thus better preserving dimension-value correlations reflected in the record count in each cell. The other characteristic that we chose to examine is measure correlation, which captures the effect of dimension values on the measure value.

For these experiments, we used synthetically generated data consisting of 2 dimensions with 128 leafs in each dimension (128x128 grid) and a single numeric measure. For all experiments, we start by generating a precise data set with the desired pseudo-density and measure correlation. Of the total grid, only 1 out of 8 cells have data records (i.e., regular density is fixed at 12.5%). We then select uniformly at random (without replacement) a percentage of records to make imprecise (between 10 - 30%). A record is made imprecise by extending it horizontally over 64 cells. From the resulting imprecise dataset, extended databases are created using different allocation policies (Uniform, EM-

Count, EM-Measure). For each extended database, we compute the SUM aggregate for each cell in the grid. The quality metric is the average percent difference for the results as compared to the original precise data set.

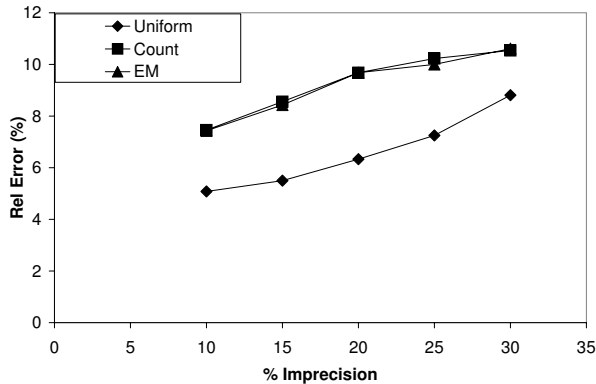


Fig. 7 Low Pseudo Density Dataset

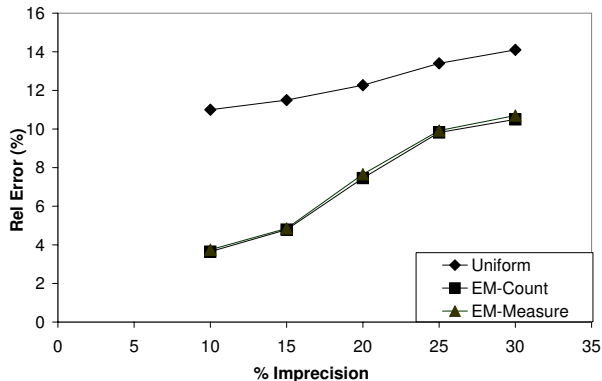


Fig. 8 High Pseudo Density, No Measure Correlation

Figures 7 through 9 show the results for experiments demonstrating the effects of pseudo-density. For the first experiment, the data was generated so no correlation exists between the measure and dimensions. The pseudo-density was set to 1 (i.e., each non-empty cell contains a single record), and the numerical measure was assigned by sampling from a Gaussian distribution with mean 5 and standard deviation 2. The results show that Uniform allocation policy has a lower relative error compared to EM-Count and EM-Measure. 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, neither EM-Count or EM-Measure will 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).

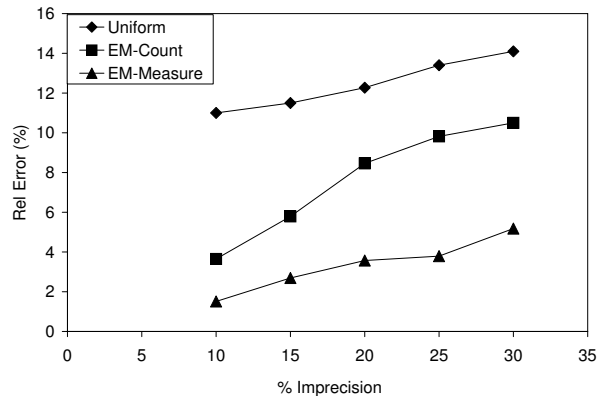


Fig. 9 High Pseudo Density, Measure Correlation

Figure 8 shows the results for a similar experiment, but with a dataset having a pseudo-density of 4. Again, there is no correlation between the measure and dimensions. Since the pseudo-density is higher, less dimension-value correlation information is lost as more records become imprecise. Thus EM-Count and EM-Measure result in better allocations, whereas Uniform suffers since it ignores the available correlation information and allocates to empty cells as well. EM-Measure performs similarly to EM-Count, but slightly worse. The reason is that EM-Measure assumes the correlation structure includes the measure, and this assumption was invalid for this dataset.

Figure 9 shows the results for a data set that has a high correlation between the measure and dimension values. The data was generated so that facts in the left half of the grid were assigned numerical measures drawn from a Gaussian distribution with a mean of 5 and variance of 0.5, while facts in the right-hand size were assigned values from a Gaussian with mean 1 and variance 0.5. The pseudo-density was still set to 4. The results show that EM-Measure now significantly outperforms both EM-Count and Uniform. This is because EM-Measure uses the correlation between the measure and dimensions while performing allocation, whereas EM-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.

9 Future Directions

An important aspect of this paper is handling uncertain measures as probability distribution functions (pdfs). The example data in Table 1 provides a conceptual view of this model with a “pdf” type column for Brake. Under the assumptions of the model discussed in this paper, adding a new uncertain measure (e.g., Transmission) would result in another column with the same type “pdf”. An obvious generalization is to capture the relationships between these two un-

certain measures. Consider a query of the type "How likely are Brake and Transmission problems in Camrys driven in Texas?". This more complicated aggregation query requires additional dependency information between the two uncertain measures and this can be captured as a set of constraints either provided by the user or learned from the data. More generally we believe that the approach initiated here generalizes to handle more general aspects of imprecision and uncertainty-handling in a DBMS, and we are actively investigating this generalization.

References

1. Abiteboul, S., Kanellakis, P.C., Grahne, G.: On the Representation and Querying of Sets of Possible Worlds. In: SIGMOD 1987
2. Arenas, M., Bertossi, L.E., Chomicki, J.: Consistent Query Answers in Inconsistent Databases. In: PODS 1999
3. Arenas, M., Bertossi, L.E., Chomicki, J., He, X., Raghavan, V., Spinrad, J.: Scalar Aggregation in Inconsistent Databases. *Theor. Comput. Sci.* **3**(296), 405–434 (2003)
4. Bell, D.A., Guan, J.W., Lee, S.K.: Generalized Union and Project Operations for Pooling Uncertain and Imprecise Information. *Data Knowl. Eng.* **18**(2), 89–117 (1996)
5. Cavallo, R., Pittarelli, M.: The Theory of Probabilistic Databases. In: VLDB 1987
6. Chen, A.L.P., Chiu, J.S., Tseng, F.S.C.: Evaluating Aggregate Operations over Imprecise Data. *IEEE TKDE* **8**(2), 273–284 (1996)
7. Cheng, R., Kalashnikov, D.V., Prabhakar, S.: Evaluating Probabilistic Queries over Imprecise Data. In: SIGMOD 2003
8. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B* (1977)
9. Dey, D., Sarkar, S.: PSQL: A Query Language for Probabilistic Relational Data. *Data Knowl. Eng.* **28**(1), 107–120 (1998). DOI [http://dx.doi.org/10.1016/S0169-023X\(98\)00015-9](http://dx.doi.org/10.1016/S0169-023X(98)00015-9)
10. Fuhr, N., Rasmussen, T.: A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Trans. Inf. Syst.* **15**(1), 32–66 (1997). DOI <http://doi.acm.org/10.1145/239041.239045>
11. Garcia-Molina, H., Porter, D.: The Management of Probabilistic Data. *IEEE TKDE* **4**, 487–501 (1992)
12. Garg, A., Jayram, T.S., Vaithyanathan, S., Zhu, H.: Model based opinion pooling. In: 8th International Symposium on Artificial Intelligence and Mathematics (2004)
13. Genest, C., Zidek, J.V.: Combining Probability Distributions: A Critique and An Annotated Bibliography (avec discussion). *Statistical Science* **1**, 114–148 (1986)
14. Kiviniemi, J., Wolski, A., Pesonen, A., Arminen, J.: Lazy Aggregates for Real-Time OLAP. In: DaWaK 1999
15. Lakshmanan, L.V.S., Leone, N., Ross, R., Subrahmanian, V.S.: ProbView: A Flexible Probabilistic Database System. *ACM TODS* **22**(3), 419–469 (1997)
16. Lenz, H.J., Shoshani, A.: Summarizability in OLAP and Statistical Data Bases. In: SSDBM 1997
17. Lenz, H.J., Thalheim, B.: OLAP Databases and Aggregation Functions. In: SSDBM 2001
18. McClean, S.I., Scotney, B.W., Shapcott, M.: Aggregation of Imprecise and Uncertain Information in Databases. *IEEE TKDE* **13**(6), 902–912 (2001)
19. Motro, A.: Accommodating Imprecision in Database Systems: Issues and Solutions. *SIGMOD Record* **19**(4), 69–74 (1990)
20. Motro, A.: Sources of Uncertainty, Imprecision and Inconsistency in Information Systems. In: Uncertainty Management in Information Systems, pp. 9–34 (1996)
21. Pedersen, T.B., Jensen, C.S., Dyreson, C.E.: Supporting Imprecision in Multidimensional Databases Using Granularities. In: SSDBM (1999)
22. Ross, R., Subrahmanian, V.S., Grant, J.: Aggregate Operators in Probabilistic Databases. *J. ACM* **52**(1), 54–101 (2005). DOI <http://doi.acm.org/10.1145/1044731.1044734>
23. Rundensteiner, E.A., Bic, L.: Evaluating Aggregates in Possibilistic Relational Databases. *Data Knowl. Eng.* **7**(3), 239–267 (1992). DOI [http://dx.doi.org/10.1016/0169-023X\(92\)90040-I](http://dx.doi.org/10.1016/0169-023X(92)90040-I)
24. Shoshani, A.: OLAP and Statistical Databases: Similarities and Differences. In: PODS 1997
25. Wu, X., Barbará, D.: Learning Missing Values from Summary Constraints. *SIGKDD Explorations* **4**(1), 21–30 (2002)
26. Wu, X., Barbará, D.: Modeling and Imputation of Large Incomplete Multidimensional Datasets. In: DaWaK (2002)
27. Zhu, H., Vaithyanathan, S., Joshi, M.V.: Topic Learning from Few Examples. In: PKDD 2003

A Proofs of Theorems

In this section, we describe the proofs of the various theorems stated in the main body of the paper. The numbering of the theorems here is consistent with that of the main body (i.e. Theorem A.1 corresponds to Theorem 1, Theorem A.2 to Theorem 2 etc.). We restate the theorems for the reader's convenience.

Theorem A1 *There exists a SUM aggregate query which violates Sum-Consistency when the Contains option is used to find relevant imprecise facts in FIND-RELEVANT.*

Proof Consider a fact table D with a single dimension and a single numerical measure called *Sales*. The dimension has two leaf values: *SJC* and *LAX*, and each leaf has the parent *CA*. Let $D = (SJC, 4), (LAX, 5), (CA, 6)$. Notice two of the facts are precise and the third is imprecise. Consider three queries:

$$\begin{aligned} Q_1 &= (SJC, Sales, SUM) \\ Q_2 &= (LAX, Sales, SUM); \quad \text{and} \\ Q_3 &= (CA, Sales, SUM) \end{aligned}$$

If Contains option is used, the imprecise fact on *CA* is not considered while answering queries on *SJC* or *LAX*. Thus the results for these queries are $\hat{q}_1 = 4$, $\hat{q}_2 = 5$ and $\hat{q}_3 = 15$. Observe these query results violate Sum-Consistency, which requires $\hat{q}_3 = \hat{q}_1 + \hat{q}_2$.

Theorem A2 *The aggregation operators SUM, COUNT, AVERAGE and LinOp violate Basic Faithfulness when the None option is used to handle imprecision.*

Proof Let D be a database containing only precise facts. Consider a query Q such that the query region has a non-empty set of facts S . Define the measure-similar database D' by making every fact in S imprecise but ensuring that the region of the resulting facts are still within the query region of Q . Under the definition of basic faithfulness, the answers for Q on both D and D' should be identical. But if the None option is used, the computation of Q on D' should ignore all the imprecise facts, resulting in the condition $\mathcal{A}(S) = \mathcal{A}(\emptyset)$, where \mathcal{A} is the aggregation function specified by Q . This is a contradiction for the aggregation functions mentioned in the theorem.

Theorem A3 *Basic Faithfulness is satisfied if answers to queries are computed using the expected value of the answer variable.*

Proof Consider two measure similar databases D and D' that are identically precise with respect to a query Q . Let $S = \{r \mid \text{reg}(r) \subseteq \text{reg}(Q)\}$ be the set of facts from D that are contained in the query region. In any possible world for D , the facts in S get mapped to some cell inside the query region, whereas facts not in S are mapped to a cell outside the query region, by the condition for basic faithfulness to be applicable. For readability, we refer to the random variable

$E[Q(\mathcal{C})]$ as Z . Thus, Z has the same value $\mathcal{A}(S)$ in all the possible worlds, and $E(Z) = \mathcal{A}(S)$.

Now consider database D' that is measure similar to D . Notice D' satisfies the condition for basic faithfulness also, by the definition of identically precise. Thus, all facts in both D and D' are either completely contained or completely outside the query region. Let $S' = \{r \mid \text{reg}(r) \subseteq \text{reg}(Q)\}$ be the set of facts from D' that are contained in the query region. Each fact in S' has exactly one corresponding fact in S , from the definition for identically precise. From measure similarity, we have $\mathcal{A}(S) = \mathcal{A}(S')$. It follows that $E(Z) = E(Z')$. Thus $E(Z)$ satisfies basic faithfulness.

Theorem A4 *The value of SUM can be computed in a single pass through EDB D^* , and is equivalent to the expected value of computing sum over the possible worlds given the independent completion assumption. The expectation for SUM satisfies Sum-Consistency.* \square

Proof For each fact $r \in \mathcal{R}(Q)$, let v_r be the associated measure value. Now the random variable associated with computing SUM is given by $Q(\mathcal{C}) = \sum_{r \in \mathcal{R}(Q)} v_r Y_r$. By linearity of expectation $E[Q(\mathcal{C})] = \sum_{r \in \mathcal{R}(Q)} v_r E[Y_r] = \sum_{r \in \mathcal{R}(Q)} \sum_{c:c \in \text{reg}(Q)} v_r * p_{c,r}$. The first expression corresponds to the expectation over the possible worlds, while the latter expression is the value computed using the Extended Database. Note that a single pass through D^* suffices to compute the expectation of SUM.

Turning to consistency, let Q, Q_1, Q_2, \dots, Q_p be queries for SUM where $\text{reg}(Q)$ is partitioned by $\text{reg}(Q_1), \dots, \text{reg}(Q_p)$. The completion of any fact r belongs to $\text{reg}(Q)$ if and only if it belongs to exactly one of $\text{reg}(Q_1), \dots, \text{reg}(Q_p)$. Thus, we have, $Y_{r,Q} = \sum_{i=1}^p Y_{r,Q_i}$. If Z_Q (resp. Z_{Q_i}) denotes the random variable corresponding to SUM for query Q (resp. Q_i), we have

$$\begin{aligned} Z_Q &= \sum_{r \in \mathcal{R}(Q)} v_r Y_{r,Q} = \sum_{r \in \mathcal{R}(Q)} \sum_{i=1}^p v_r Y_{r,Q_i} \\ &= \sum_{i=1}^p \sum_{r \in \mathcal{R}(Q)} v_r Y_{r,Q_i} = \sum_{i=1}^p \sum_{r \in \mathcal{R}(Q_i)} v_r Y_{r,Q_i} = \sum_{i=1}^p Z_{Q_i} \end{aligned}$$

The theorem follows because $E[Z_Q] = \sum_{i=1}^p E[Z_{Q_i}]$.

Theorem A5 *The expectation of SUM satisfies Sum-Faithfulness if the allocation policy used to build the extended data model is monotone.*

Proof Let $D \preceq_Q D''$, and let Q be a SUM query. To simplify the presentation, let \hat{q}_S denote the result of computing Q using database S . Consider the quantity $n(D, D'') = \sum_r \text{reg}(r) \Delta \text{reg}(r'')$, where Δ denotes symmetric difference. The quantity $n(D, D'')$ reflects the difference in the amount of imprecision between D and D'' with respect to the query.

The proof is by induction on $n = n(D, D'')$. For $n = 0$, there is nothing to prove. For $n > 1$, let D' be any database

such that $n(D, D') = 1$ and $n(D', D'') = n - 1$. Such a database can be shown to always exist. By induction, we have that $\hat{q}_{D'} \geq \hat{q}_{D''}$. It suffices to show that $\hat{q}_D \geq \hat{q}_{D'}$. Since $n(D, D') = 1$, this means the associated regions are identical for every pair of corresponding facts, except for a single pair (r, r') , such that $\text{reg}(r') \Delta \text{reg}(r) = 1$.

Because $D \preceq_Q D'$, it must be the case that $\text{reg}(r') = \text{reg}(r) \cup \{c^*\}$, for some cell $c^* \notin \text{reg}(Q)$. It follows that $\text{reg}(r) \cap \text{reg}(Q) = \text{reg}(r') \cap \text{reg}(Q)$. For any other fact s , $\text{reg}(s) \cap \text{reg}(Q) = \text{reg}(s) \cap \text{reg}(Q)$ as well. Since the allocation policy is monotone, and $c^* \notin \text{reg}(Q)$, for every fact s (including r) we have $\sum_{c \in \text{reg}(s) \cap \text{reg}(Q)} p_{c,r} \geq \sum_{c \in \text{reg}(s) \cap \text{reg}(Q)} p'_{c,r}$.

Therefore, the allocation of s to $\text{reg}(Q)$ in D is no less than the allocation in D' . The theorem follows by the definition of SUM.

The following technical lemma will be helpful towards proving Theorem 6.

Lemma A1 *Let X_1, X_2, \dots, X_n denote a set of independent 0-1 Bernoulli random variables with $\Pr[X_i = 1] = p_i$. Let $X = \frac{1}{m + \sum_i X_i}$. Then $E[X]$ can be computed in $O(n^2)$ time.*

Proof For every $0 \leq j \leq k \leq n$, let Z_{jk} denote the random variable $\frac{1}{m + j + \sum_{i>k} X_i}$ and let $A_{jk} = E[Z_{jk}]$. Because the X_i 's are independent, we have the following recurrence

$$\begin{aligned} A_{jk} &= E[Z_{jk}] \\ &= \Pr[X_{k+1} = 1] \cdot E[Z_{jk} \mid X_{k+1} = 1] \\ &\quad + \Pr[X_{k+1} = 0] \cdot E[Z_{jk} \mid X_{k+1} = 0] \\ &= p_{k+1} \cdot E[Z_{j+1, k+1}] + (1 - p_{k+1}) \cdot E[Z_{j, k+1}] \\ &= p_{k+1} A_{j+1, k+1} + (1 - p_{k+1}) A_{j, k+1}, \end{aligned}$$

for $k < n$, and $A_{jn} = \frac{1}{m+j}$. Thus, computing the recurrence can be achieved in $O(n^2)$ time. To complete the proof note that $E[X] = A_{00}$.

Using this lemma, we can estimate the complexity of computing an exact value for the AVERAGE function.

Theorem A6 *Let n and m be the number of partially and completely allocated facts in a query region, respectively. The exact expected value of AVERAGE can be computed in time $O(m+n^3)$, with n passes over the set of candidate facts. Also, the value of exact AVERAGE computed using EDB D^* is equal to the expected value over the possible worlds, given the independent completion assumption. \square*

Proof Let $\mathcal{R}(Q)$ be the set of facts in D either partially or completely allocated to $\text{reg}(Q)$. Let S be the set of facts which are partially allocated to the query Q , and set $n = |\mathcal{S}(Q)|$. Let m equal the number of facts that are fully allocated to Q , and let A denote the sum of the corresponding measure values. Computing m and A can be accomplished in time $O(m+n)$. We have,

$$Q(C) = \frac{A + \sum_{r \in S} v_r Y_r}{m + \sum_{r \in S} Y_r}. \quad (3)$$

By linearity of expectation,

$$\begin{aligned} E[Z] &= \sum_{r \in S} E\left[\frac{A + v_r Y_r}{m + \sum_{r' \in S} Y_{r'}}\right] \\ &= \sum_{r \in S} (\Pr[Y_r = 1] \cdot E\left[\frac{A + v_r}{m + 1 + \sum_{\substack{r' \in S \\ r' \neq r}} Y_{r'}}\right] \\ &\quad + \Pr[Y_r = 0] \cdot E\left[\frac{A}{m + \sum_{\substack{r' \in S \\ r' \neq r}} Y_{r'}}\right]) \\ &= \sum_{r \in S} (\Pr[Y_r = 1] \cdot (A + v_r) \cdot E\left[\frac{1}{m + 1 + \sum_{\substack{r' \in S \\ r' \neq r}} Y_{r'}}\right] \\ &\quad + \Pr[Y_r = 0] \cdot A \cdot E\left[\frac{1}{m + \sum_{\substack{r' \in S \\ r' \neq r}} Y_{r'}}\right]) \end{aligned}$$

Each term in the summation above can be computed using Lemma A1 in time $O(n^2)$ from which the overall running time follows.

It can be seen that by ordering the computation of A_{jk} 's so that we iterate over each j for each value of k , we can compute the A_{jk} 's in one iteration of k , i.e. one scan of $\mathcal{R}(Q)$. Since we have to do this for each fact in S , we would need n scans of $\mathcal{R}(Q)$.

Notice that $P[Y_r = 1]$ is equivalent to the sum of the $p_{c,r}$ values for all $c \in \text{reg}(Q)$. Thus, evaluating the above equations using the Extended Database is equivalent to evaluating them over the possible worlds.

Theorem A7 *An alternative estimate for AVERAGE can be computed in time $O(m+n)$ using a single pass over the set of candidate facts in EDB D^* , and is equivalent to the expected value computed directly over the possible worlds, given the independent completion assumption. The relative error of the estimate is negligible when $n \ll m$. \square*

Proof The alternative estimate for AVERAGE is a fraction of SUM over COUNT. The expected value for both SUM and COUNT computed using the Expected Data Model and the possible worlds are identical. Thus, the two values for alternative AVERAGE are the same for both.

Write $Z = U/V$ where U and V are the numerator and denominator in Equation 3. Because $m \leq V \leq m+n$ always, we have that $U/(m+n) \leq Z \leq U/m$. It follows that $E[U]/(m+n) \leq E[Z] \leq E[U]/m$. Consider the estimator $A = E[U]/E[V]$ which also satisfies $E[U]/(m+n) \leq A \leq E[U]/m$. The relative error of the estimator A is therefore at most

$$\begin{aligned} \frac{E[U](\frac{1}{m} - \frac{1}{m+n})}{E[Z]} &= \frac{E[U]}{E[Z]} \left(\frac{1}{m} - \frac{1}{m+n} \right) \\ &\leq (m+n) \left(\frac{1}{m} - \frac{1}{m+n} \right) = \frac{n}{m} \end{aligned}$$

which is negligible when $n \ll m$.

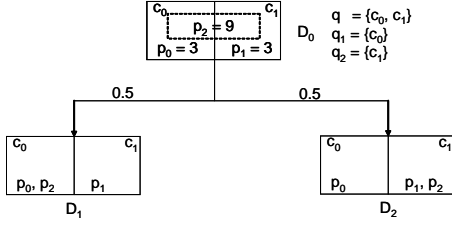


Fig. 10 AVERAGE computation

Theorem A8 *The expectation of the AVERAGE computed from the extended data model satisfies basic faithfulness but not Boundedness-Consistency.*

Proof The proof is through a counter example. Consider a database that has 3 facts as shown in Figure 10. p_0 and p_1 are exact data facts with measure values 3 each. p_2 is an imprecise data fact with measure value 9. The original database D_0 maps to two possible worlds D_1 and D_2 with equal weights 0.5 each. Now consider three queries: Q_1 that averages cell c_1 , Q_2 that averages cell c_2 and Q that averages over both c_1 and c_2 . Thus, Q_1 and Q_2 partition and cover the region of q . Table 4 summarizes the answers to the queries in each possible world and the expected values. Thus the expected values of Q , Q_1 and Q_2 do not satisfy the

	D_1	D_2	Expected Value
Q_1	6	3	4.5
Q_2	3	6	4.5
Q	5	5	5

Table 4 Expected value for AVERAGE

Boundedness-consistency requirements since 5 is out of the range of $[4.5, 4.5]$.

Theorem A9 *The alternative estimate for AVERAGE defined above satisfies Boundedness-Consistency and Basic Faithfulness.*

Proof The alternative estimate equals $E[Z] = E[U]/E[V]$ where $E[U]$ is the expected SUM and $E[V]$ is the expected COUNT. Consider a query Q and a set of queries Q_1, Q_2, \dots, Q_p whose associated regions partition that of Q . Let u_{Q^*} and v_{Q^*} denote $E[U]$ and $E[V]$ respectively for any query Q^* . Since SUM and COUNT satisfy Sum-Consistency, we have $u_Q = \sum_i u_{Q_i}$ and $v_Q = \sum_i v_{Q_i}$. Thus,

$$\hat{q} = \frac{u_Q}{v_Q} = \frac{\sum_i u_{Q_i}}{\sum_i v_{Q_i}} \geq \min_i \left(\frac{u_{Q_i}}{v_{Q_i}} \right) = \min_i (\hat{q}_i)$$

A similar argument shows that $\hat{q} \leq \max_i (\hat{q}_i)$.

Turning to basic faithfulness, let D_1 and D_2 be two similar databases that are identically precise with respect to a query Q . Since SUM and COUNT both satisfy basic faithfulness, we have $E(U_{D_1}) = E(U_{D_2})$ and $E(V_{D_1}) =$

$E(V_{D_2})$ if $D_1 = D_2$. It follows that $E(Z_{D_1}) = E(Z_{D_2})$ and $\hat{q}_{D_1} = \hat{q}_{D_2}$.

Theorem A10 *AggLinOp can be computed in a single pass over the set of candidate facts, and satisfies Boundedness-Consistency and Basic Faithfulness. The value of AggLinOp computed using the EDB D^* is equivalent to expected answer to Q evaluated over all possible worlds, given the independent completion assumption.* \square

Proof Recall the expression for AggLinOp given by Definition 17 for any query Q . Rearranging the terms in the numerator to get:

$$\sum_{r \in \mathcal{R}(Q)} \sum_{i \in \mathcal{S}(r)} v_r w_i = \sum_{r \in \mathcal{R}(Q)} v_r \left(\sum_{i \in \mathcal{S}(r)} w_i \right)$$

In our possible worlds interpretation, we assume that imprecise facts are assigned to the cells independent of each other. It follows that the expression $\sum_{i \in \mathcal{S}(r)} w_i$ equals the allocation of r to Q . We have,

$$\frac{\sum_{r \in \mathcal{R}(Q)} v_r \sum_{c \in C(r) \cap \text{reg}(Q)} p_{c,r}}{\sum_{r \in \mathcal{R}(Q)} \sum_{c \in (C(r) \cap Q)} p_{c,r}}$$

Recall that the allocations of candidate facts are computed in the first pass. It follows that AggLinOp can be computed in a single pass over $\mathcal{R}(Q)$ using the corresponding allocations. Thus, the answer computed using the Extended Database is equivalent to the expected value over the possible worlds.

Note that even though it is a single pass, the v_r s are vectors of size l each. So the computational complexity is $O(|\mathcal{R}(Q)|l)$.

Now considering consistency, let the queries q, q_1, \dots, q_p and associated query regions Q, Q_1, Q_2, \dots, Q_p be defined as before. We have,

$$\begin{aligned} \sum_{c \in (C(r) \cap Q)} p_{c,r} &= \sum_{c \in (C(r) \cap (Q_1 \cup Q_2 \cup \dots \cup Q_p))} p_{c,r} \\ &= \sum_{i=1}^p \sum_{c \in (C(r) \cap Q_i)} p_{c,r} \end{aligned}$$

Let U_Q and V_Q denote the numerator and denominator in the expression for AggLinOp. Using the above result we have,

$$\begin{aligned} U_q &= \sum_{r \in \mathcal{R}(Q)} v_r \sum_{c \in (C(r) \cap Q)} p_{c,r} \\ &= \sum_{i=1}^k \sum_{r \in \mathcal{R}(Q)} v_r \sum_{c \in (C(r) \cap \text{reg}(Q_i))} p_{c,r} \\ &= \sum_{i=1}^k \sum_{r \in \mathcal{R}(Q_i)} v_r \sum_{c \in (C(r) \cap Q_i)} p_{c,r} \\ &\text{since } (C(r) \cap Q_i) = \emptyset \text{ if } r \notin \mathcal{R}(Q_i) \\ &= \sum_{i=1}^k U_{Q_i} \end{aligned}$$

Similarly we have, $V_q = \sum_{i=1}^k V_{q_i}$. Using all these results and considering each component of the PDF separately we have,

$$\hat{q}(o) = \frac{U_q(o)}{V_q} = \frac{\sum_{i=1}^k U_{q_i}(o)}{\sum_{i=1}^k V_{q_i}} \geq \min_i \left(\frac{U_{q_i}(o)}{V_{q_i}} \right) = \min_i (\hat{q}_i(o))$$

A similar proof holds for max as well.

Turning to basic faithfulness, consider a database D that is precise w.r.t a query Q . Thus all facts in D are completely contained in the query region. Thus for any $r \in \mathcal{R}(Q)$, we have $\sum_{c \in (C(r) \cap Q)} p(c, r) = 1$ since r gets completely allocated within the query region. Thus the answer to Q can be written as,

$$\begin{aligned} \hat{q}_D &= \frac{\sum_{r \in \mathcal{R}(Q)} v_r}{\sum_{r \in \mathcal{R}(Q)} 1} \\ &= \frac{\sum_{r \in \mathcal{R}(Q)} v_r}{|\mathcal{R}(Q)|} \end{aligned}$$

Consider two databases such that $D_1 = D_2$ w.r.t query Q . Then by definition both are precise w.r.t the query Q and also $\mathcal{R}(Q)_{D_1} = \mathcal{R}(Q)_{D_2}$. It follows from the above discussion that $\hat{q}_{D_1} = \hat{q}_{D_2}$.

Theorem A11 *Assume we are given an imprecise database D , and the corresponding EDB D^* . Let \mathcal{P} be the set of (discrete) pdfs assignable to the space of possible worlds \mathcal{C} such that each $p_M \in \mathcal{P}$ satisfying the marginal distributions given by the $C(r)$.*

If any $p_M \in \mathcal{P}$ is selected as the pdf over \mathcal{C} , then for any query Q using one of the following aggregation operators, the value of $Q(D^) = E[Q(\mathcal{C})]$:*

1. SUM
2. COUNT
3. Alternative AVERAGE
4. AggLinOp

□

Proof First, we show that for all valid choices of p , $E[Q(\mathcal{C})]$ has the same value. Second, we show the $p_{c,r}$ values in the Extended Database D^* are unaffected by the choice of p . Thus, the EDB is identical for any valid p . Finally, we show the pdf defined when the independence assumption p_{IND} is valid. Since we have already shown that $E[Q(\mathcal{C})] = E[Q(EDB)]$ for all of the listed aggregation operators for this special pdf p , the proof is complete. We note these proofs did not make any assumptions regarding the joint distribution of $C(r)$.

First, we show $E[Q(\mathcal{C})]$ remains the same for all valid p . Consider the Y_r random variables for the event “ $C(r)$ maps to some cell in the region of Q ”. From Equation 1, the value for $P[Y_r = 1]$ (i.e., $E[Y_r]$) is unaffected by any joint distribution for $C(r)$ variables. As long as the $p_{c,r}$ values, the probabilities for individual completions of each fact r , remain the same, $E[Y_r]$ is unaffected. By the constraint that p must satisfy all $C(r)$, any choice of p will result in identical values

for all $p_{c,r}$. Thus, the $E[Y_r]$ remain the same. To determine the expected value of Q over the possible worlds, $E[Q(\mathcal{C})]$, only the individual $E[Y_r]$ are considered. Thus, $E[Q(\mathcal{C})]$ is the same for all valid p .

The $p_{c,r}$ values must remain the same for all valid p , since these define the distributions for $C(r)$ which are the definition of valid pdfs.

We now show p_{IND} satisfies the constraints defined by the distributions of the $C(r)$. Since each $C(r)$ is a vector which sums to 1, p_{IND} can be thought of as the cross-product of an arbitrary number of vectors which sum to 1. By the construction of p_{IND} , each of the $C(r)$ must be satisfied.

Thus, the proof is complete. □