Aggregation of Imprecise and Uncertain Information in Databases

Sally McClean, *Member*, *IEEE*, Bryan Scotney, and Mary Shapcott

Abstract—Information stored in a database is often subject to uncertainty and imprecision. Probability theory provides a well-known and well understood way of representing uncertainty and may thus be used to provide a mechanism for storing uncertain information in a database. We consider the problem of aggregation using an imprecise probability data model that allows us to represent imprecision by partial probabilities and uncertainty using probability distributions. Most work to date has concentrated on providing functionality for extending the relational algebra with a view to executing traditional queries on uncertain or imprecise data. However, for imprecise and uncertain data, we often require aggregation operators that provide information on patterns in the data. Thus, while traditional query processing is tuple-driven, processing of uncertain data is often attribute-driven where we use aggregation operators to discover attribute properties. The aggregation operator that we define uses the Kullback-Leibler information divergence between the aggregated probability distribution and the individual tuple values to provide a probability distribution for the domain values of an attribute or group of attributes. The provision of such aggregation operators is a central requirement in furnishing a database with the capability to perform the operations necessary for knowledge discovery in databases.

Index Terms—Relational databases, imprecise and uncertain data, partial probabilities, aggregation operators, knowledge discovery.

1 Introduction

There is an increasing demand for generalized databases to provide intelligent ways of storing and retrieving data. Frequently, real life data are *uncertain*, i.e., we are not certain about the truth of an attribute value, or *imprecise*, i.e., we are not certain about the specific value of an attribute but only that it takes one of a group of possible values where imprecision might occur naturally as a result of data being provided at different levels of the concept hierarchy. It is therefore important that appropriate functionality is provided for database systems to handle such imperfect information. A recent survey of various approaches to handling imperfect information in data and knowledge bases has been provided by Parsons [17].

A database model that is based on *partial values* and *partial probabilities* [8], [22], [7] has previously been proposed to handle both imprecise and uncertain data. Partial probabilities may be thought of as a generalization of null values where rather than not knowing anything about a particular attribute value, as is the case for null values, we may be more specific and identify the attribute value as belonging to a set of possible values. A partial value is therefore a set such that exactly one of the values in the set is the true value. Partial probabilities [6] take this generalization a stage further by assigning probabilities to the partial values. We may thus combine the strong imprecision handling capacity of partial values with the powerful uncertainty handling capabilities of probability theory. The underlying data that we consider is discrete, or, if continuous, must be discretized, by grouping, before inclusion in our model. Thus, by discretizing, we can ensure that the domain is finite which is an assumption of our approach.

Previous work, such as the possibilistic model based on fuzzy set theory [19] or the probabilistic model [4], has tended to focus on one or another aspect. A similar approach to our own is that in [5] which uses mass functions from the Dempster-Shafer theory of evidence to represent uncertain and imprecise data. However, while such a representation provides a useful framework for traditional database functionality, aggregation of data to provide descriptions of attributes is better supported by the probability model, which can draw on a rich repertoire of manipulation and analysis tools. The capacity to handle both stochastic and imprecise data is a major strength of our current approach. Such a model for the management of uncertain information includes both stochastic data, which is implicitly subject to randomness, and imprecise data, which is not crisp but instead falls in a set.

In this paper, we consider the problem of *aggregation* for the partial probability data model. Most previous work has concentrated on providing functionality that extends relational algebra with a view to executing traditional queries on uncertain or imprecise data. However, for such imperfect data, we often require aggregation operators that provide information on patterns in the data. Thus, while traditional query processing is tuple-specific where we need to extract individual tuples of interest, processing of uncertain data is often attribute-driven where we need to use aggregation operators to discover properties of attributes of interest. Thus, we might want to aggregate over individual tuples to provide summaries which describe relationships between attributes. Such a facility is a central

• *The authors are with the School of Information and Software Engineering, Faculty of Informatics, University of Ulster, Cromore Rd., Coleraine, BT52 1SA, Northern Ireland.*
E-mail: {si.mcclean, bw.scotney, cm.shapcott}@ulst.ac.uk.

Manuscript received 9 Jan. 1998; revised 9 July 1999; accepted 2 May 2000; posted to Digital Library 12 June 2001.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 106135.

requirement in providing a database with the capability to perform the operations necessary for knowledge discovery in databases, where we are frequently concerned with identifying interesting attributes or interesting relationships between attributes. A shortened version of this paper, which stresses the knowledge discovery aspects of our approach, has previously been presented [15].

2 Basic Concepts and Definitions

2.1 Partial Values and Partial Probabilities

In a generalized relational database, we consider an attribute A and tuple t of a relation $\boldsymbol{R}$ which has values which are imprecise in that, in any particular tuple, an attribute value may be a partial value. A partial value is defined as follows:

Definition 2.1. *A* partial value *is determined by a set of possible attribute values of tuple t of attribute A of which one and only one is the true value. We denote a partial value* η *by* $\eta = [a_r, \ldots a_s]$ *corresponding to a set of* h *possible values* $\{a_r, \ldots a_s\}$ *of the same domain, in which exactly one of these values is the true value of* η*. Here,* h *is the cardinality of* η*;* $\{a_r, \ldots a_s\}$ *is a subset of the domain set* $\{a_1, \ldots a_k\}$ *of attribute A of relation* $\boldsymbol{R}$ *and* $\mathrm{h} \leq \mathrm{k}$.

Example 2.1. We consider the attribute SMOKING_STATUS which has possible values "heavy," "light," "ex-heavy," "ex-light," "never." Then, ["heavy," "light"] is an example of a partial value. In this case, we know only that the individual is a smoker but not whether he or she is a heavy or a light smoker.

Database queries may require operations to be performed on partial values; this can result in a query being answered by so-called "maybe tuples" where the tuples have partial attribute values [22], [7]. In [22] and [6], probabilistic partial values are used, which leads to imprecise answer tuples of a query being associated with degrees of uncertainty and, hence, permits comparison between the imprecise maybe tuples in the query result. Partial probabilities are defined as follows:

Definition 2.2. *A* partial probability *denoted by* $\xi(\eta) = [p_r, \ldots, p_s]$ *is a vector which associates a probability* p_i *with each possible value* a_i *of the partial value* η *such that* $p_i = \mathrm{Probability}(a_i)$*, i.e.,* p_i *is the probability that the attribute value is* a_i*, where* $\sum_{\mathrm{i=r}}^{\mathrm{s}} p_{\mathrm{i}} = 1$.

Example 2.2. An example of a partial probability for the {"heavy," "light"} partial value is then given by:

$$\xi([''\text{heavy},''\ ''\text{light}'']) = (0.3, 0.7)$$

by which we mean that the probability of "heavy" is 0.3 and the probability of "light" is 0.7. Of course, we may not know this partial probability ab initio but might instead be able to compute it. This problem of the computation of partial probabilities is, in fact, a special case of the problem we consider in this paper.

Definition 2.3. *A* partition *of the set of values* $\mathrm{S} = \{\mathrm{v}_1, \ldots \mathrm{v}_k\}$ *of the domain of attribute A is given by a set of subsets* $\mathrm{S}_1, \ldots \mathrm{S}_g$ *of* S*, where:*

$$\bigcup_{\mathrm{i}=1}^{\mathrm{g}} \mathrm{S_i} = \mathrm{S} \text{ and } \mathrm{S_i} \bigcap \mathrm{S_j} = \emptyset \text{ for all } \mathrm{i} \neq \mathrm{j}.$$

Example 2.3. An example of a partition of the SMOKING_STATUS attribute is: ({"heavy," "light"}, {"ex-heavy," "ex-light"}, {"never"}).

Here, we extend the definition of [22] and [6] to that of a partial probability distribution so that probabilities are assigned to each partial value of a partition.

Definition 2.4. *A* partial probability distribution *is a vector of probabilities* $\varphi(\eta) = (p_1, \ldots p_e)$ *which is associated with a partition formed by partial values* $\eta = (\eta_1, \ldots \eta_e)$ *of attribute A in tuple t. Here,* p_{i} *is the probability that the attribute value is a member of partial value* η_{i}*. Also,*

$$\sum_{\mathrm{i}=1}^{\mathrm{e}} p_{\mathrm{i}} = 1.$$

Example 2.4. An example of a partial probability distribution on the above partition is then:

$$\varphi(\{''\text{heavy},''\ ''\text{light}''\},\ \{''\text{ex-heavy},''\ ''\text{ex-light}''\},\ \{''\text{never}''\}) = (0.4, 0.35, 0.25).$$

This distribution means that the probability of being either a heavy or a light smoker is 0.4, the probability of being an ex-heavy or ex-light smoker is 0.35, and the probability of never having been a smoker is 0.25.

The general problem that we consider in this paper is the situation where each tuple's value in each attribute is a partial probability distribution and we want to compute the probability of each attribute value. Of course, some of the partial values may be singletons with crisp probability values. The related problem of finding the partial probabilities, as described in [22] and [6], is a special case where some of the attribute values are known in advance to have zero probabilities, and they are not included in the (single) partial value. Thus, in the partial probability model, as defined in Definition 2.2, probabilities are assigned to the values within a single partial value, whereas in the partial probability distribution model, as defined in Definition 2.4, we want to assign probabilities to values within a *set* of partial values.

An extended relational model in which the tuples consist of partial values was described in [8] and [7]. A relation consisting of tuples with probabilistic partial values was defined in [22] and termed a *probabilistic partial relation*. We extend this data model to an extended relational database model based on a partial probability distribution. A similar data model has previously been proposed by Barbará et al. [4], who introduced the term *probability data model* (PDM). They also provided some extended relational operators, such as *project* and *join*, for use in conjunction with this model. Their model differs from ours in that they do not include partial values, as we have defined them; this allows

TABLE 1
An Imprecise Probablilty Data Model

ID	SEX	SMOKING_STATUS	HYPERTENSION	HEART_DISEASE
001	<{M},0.0> <{F},1.0>	<{S},1.0> <{E},0.0> <{N},0.0>	<{S,M},1.0> <{N},0.0>	<{S},0.75> <{M,N},0.25>
002	<{M},1.0> <{F},0.0>	<{S},0.0> <{E},1.0> <{N},0.0>	<{S,M,N},1.0>	<{S},0.75> <{M},0.2> <{N},0.05>
003	<{M},0.0> <{F},1.0>	<{S},0.0> <{E},0.2> <{N},0.8>	<{S,M},0.0> <{N},1.0>	<{S,M},0.0> <{N},1.0>
004	<{M},1.0> <{F},0.0>	<{S},0.9> <{E},0.1> <{N},0.0>	<{S},1.0> <{M,N},0.0>	<{S},1.0> <{M,N},0.0>
005	<{M},0.0> <{F},1.0>	<{S},0.0> <{E},1.0> <{N},0.0>	<{S,M},0.0> <{N},1.0>	<{S},0.2> <{M,N},0.8>
006	<{M},1.0> <{F},0.0>	<{S},1.0> <{E},0.0> <{N},0.0>	<{S,M},1.0> <{N},0.0>	<{S},1.0> <{M,N},0.0>
007	<{M},1.0> <{F},0.0>	<{S},0.0> <{E},0.0> <{N},1.0>	<{S,M},0.0> <{N},1.0>	<{S},0.1> <{M},0.2> <{N},0.7>
008	<{M},1.0> <{F},0.0>	<{S},1.0> <{E},0.0> <{N},0.0>	<{S},1.0> <{M,N},0.0>	<{S},0.9> <{M},0.05> <{N},0.05>
009	<{M},0.0> <{F},1.0>	<{S},0.75> <{E},0.25> <{N},0.0>	<{S,M},1.0> <{N},0.0>	<{S},0.0 > <{M},1.0> <{N},0.0>
010	<{M},1.0> <{F},0.0>	<{S},1.0> <{E},0.0> <{N},0.0>	<{S,M},0.0> <{N},1.0>	<{S},0.2> <{M,N},0.8>

Legend: *SEX: M—male, F—female; SMOKING_STATUS: S—smoker, E—ex-smoker, N—never; HYPERTENSION: S—severe, M—mild, N—no hypertension; HEART_DISEASE: S—severe, M—mild, N—no heart disease.*

us to handle imprecise as well as uncertain data. However, they instead focus on the joint distribution of several values occurring simultaneously; we have not included such an option for the moment although it would be straightforward to extend our approach in this way. We will extend the functionality of [4] to provide aggregation operators that are more geared toward knowledge discovery than the conventional relational operators.

Definition 2.5. *An* imprecise probability data model *(IPDM) $\boldsymbol{R}$ extends the relational model to describe a probability distribution of partial values for domains $D_1, D_2, \ldots D_n$ of attributes $A_1, A_2, \ldots A_n$, where each tuple t of $\boldsymbol{R} \subseteq P_1 \times P_2 \times \ldots P_n$ and P_i is the set of all the partial probability distributions on domain D_i. Each cell of the table is a probability distribution specified on a partition of the appropriate domain into partial values.*

An example of an imprecise probability data model is presented in Table 1. We note in passing that this relation is similar to that described in [5] which uses Dempster-Shafer mass functions where we use probabilities. It is clear that partial probability distributions will provide useful mechanisms for representing imprecise and uncertain information in similar circumstances to those discussed in [5]. However, in addition, an imprecise probability data model furnishes us with powerful data manipulation and analysis capabilities that in turn provide a framework suitable for rule induction and knowledge discovery.

We can see from Table 1 that this general relational data model allows us to represent a number of other models as special cases. For example, the attribute SEX is crisp with no uncertain values, i.e., we know in each case if a value is M or F with probability 1 and these attribute values are neither imprecise nor uncertain. The attribute SMOKING_STATUS consists entirely of crisp probabilistic values, i.e., in each case, we know the individual's smoking status in terms of a probability distribution giving probabilities of each of the possible values—smoker, ex-smoker, or never smoked. Therefore, for this attribute, we have uncertainty without imprecision. The attribute HYPERTENSION, on the other hand, consists of true partial values since the partition is not the same in all cases. These attribute values are therefore imprecise but certain since all probabilities are either one or zero. The final attribute, HEART_DISEASE is a true partial probability distribution since probabilities in this case are not all 0 or 1. Therefore, in this case, values are both imprecise and uncertain.

We note that, while Table 1 represents the logical structure of the partial probability distribution relation, in practice, those data that are crisp would be stored in a database in a more efficient representation. Thus, for example, since we know that an individual 1 is certainly a smoker, we can represent this information as in the traditional relational model and transform to a partial probability distribution when an appropriate query is executed.

Data such as we have envisaged may arise in a number of ways [4]. First, such uncertainty and imprecision may be specified by the domain expert, e.g., a doctor may not want to be precise about whether a patient has mild or severe hypertension or may want to specify probabilities of heart disease rather than giving an exact diagnosis. Another possibility is that such data are extracted from another database that allows us to calculate the probabilities. Imprecision may occur naturally as part of the missing value structure, may refine null values or other more imprecise values using database integrity constraints or other domain knowledge, or may arise as a result of the integration of heterogeneous distributed databases.

3 Aggregation of Crisp Probabilities

In this paper, we consider the problem of aggregation for the imprecise probability data model. The objective is to provide an aggregation operator that allows us to aggregate individual tuples to form summary tables which describe the proportion (or number) of tuples with each possible value in the cross-product of the domains. For data that are imprecise and uncertain, we require an aggregation operator which provides information on relationships between attributes. The derivation of such aggregates from imprecise and uncertain data, in general, is a difficult task for which we will draw on existing results from statistics. As we will discuss further in Section 6, such functionality is useful for knowledge discovery in databases where we are frequently concerned with identifying interesting attributes or interesting relationships between attributes.

We are required to define an aggregate operator which handles all data described in Table 1, i.e., crisp and certain (SEX), crisp and uncertain (SMOKING_STATUS), imprecise and certain (HYPERTENSION), and imprecise and uncertain (HEART_DISEASE), in a consistent and uniform manner. Therefore, in this section, we begin by considering cases 1 and 3 only, i.e., the cases of crisp and certain, and crisp and uncertain data. In the first of these cases, the aggregation operator should collapse to the well-known count operator which simply counts the numbers of tuples in which each value of an attribute occurs.

We begin by defining an aggregation (count) operator for the certain, crisp case and then generalize until we have covered all the types of data illustrated in Table 1.

Definition 3.1. *The* simple count *operator operates on a number of distinct values of attribute A_j of relation $\boldsymbol{R}$. Denoted* count $(\boldsymbol{R}.\ A_j)$*, it is defined as a vector-valued function:*

$$count(\boldsymbol{R}.\ A_j) = (n_1, \dots n_k),$$

where A_j has domain $D_j = \{v_1, \dots v_k\}$ and

$$n_i = \sum_{r=1}^{m} I(t_r.A_j = v_i) \text{ for } i = 1, \dots k.$$

Here, I *is an indicator function which is 1 if $(t_r.A_j = v_i)$ and 0 otherwise, where $t_r.A_j$ is the value of attribute A_j in tuple t_r;* m *is the cardinality of A_j.*

Definition 3.2. *The* simple aggregate *operator operates on a number of distinct values of attribute A_j of relation $\boldsymbol{R}$. Denoted $sagg(\boldsymbol{R}.A_j)$, it is defined as a vector-valued function:*

$$sagg(\boldsymbol{R}.A_j), = (\pi_1, \dots \pi_n),$$

where $\pi_i = n_i / \sum_{r=1}^{n} n_r$ and the n_is are as defined for the simple count.

Example 3.1. These definitions of course relate to our simplest scenario, when the domain of A_j is both crisp and certain. Thus, for the attribute SEX in Table 1 ($\boldsymbol{R}$), we obtain:

$$count(\boldsymbol{R}.\text{SEX}) = (6, 4) : sagg(\boldsymbol{R}.\text{SEX}) = (0.6, 0.4),$$

where the first element of each tuple refers to the value "male" and the second value refers to the value "female."

We now define an aggregation operator that is appropriate to data that are both crisp and uncertain. In this case, attribute A_j of tuple t_r takes value v_i with probability p_{ri} for $r = 1, \dots m$ and $i = 1, \dots k$; here, r indexes the tuple while i indexes the attribute value.

Definition 3.3. *The* probability aggregate *operator operates on a number of distinct values of attribute A_j of relation $\boldsymbol{R}$. Denoted* pagg$(\boldsymbol{R}.A_j)$*, it is defined as a vector-valued function:*

$$pagg(\boldsymbol{R}.A_j), = (\pi_1, \dots \pi_k),$$

where

$$\pi_i = \left(\sum_{r=1}^{m} p_{ri} \right) / \text{m for } i = 1, \dots k.$$

Here, $\varphi(v_1, \dots v_k) = (p_{r1}, \dots p_{rk})$ is the partial probability distribution for tuple t_r of attribute A_j.

Example 3.2. These definitions relate to our next scenario, when the domain of A_j is crisp but uncertain. Thus, for the attribute SMOKING_STATUS in Table 1 ($\boldsymbol{R}$), we obtain:

$$pagg(\mathbf{R}.SMOKING\_STATUS) = (0.565, 0.255, 0.18),$$

where the first element of the vector refers to the value "S," the second element refers to the value "E," and the third element refers to the value "N."

As we can see from this example, the pagg operator produces a probability distribution which is an average of the individual (tuple) probability distributions. We may justify this approach via the law of total probability, as follows:

Theorem 3.1. *The probability distribution described by the probability aggregate operator is equivalent to the probability distribution of attribute values.*

Proof. We assume that $\mathrm{Prob}(t_\mathrm{r}) = 1/\mathrm{m}$ for $\mathrm{r} = 1, \ldots \mathrm{m}$, i.e., all tuples are equally likely to occur. Then,

$$\begin{aligned}\mathrm{Prob}(\mathrm{v_j}) &= \sum_{\mathrm{r}=1}^{\mathrm{m}} \mathrm{Prob}(t_\mathrm{r}.A_\mathrm{j} = \mathrm{v_j} \mid t_\mathrm{r}).\mathrm{Prob}(t_\mathrm{r}) \\ &= \frac{1}{\mathrm{m}} \sum_{\mathrm{r}=1}^{\mathrm{m}} p_{\mathrm{ri}} \text{ for } \mathrm{j} = 1, \ldots \mathrm{k}.\end{aligned}$$

□

Theorem 3.2. *The probability distribution described by the probability aggregate operator is a proper probability distribution.*

Proof. A proper probability distribution is one in which all the components add to 1. In this case:

$$\begin{aligned}\sum_{\mathrm{i}=1}^{k} \pi_\mathrm{i} &= \sum_{\mathrm{i}=1}^{k} \left(\sum_{\mathrm{r}=1}^{\mathrm{m}} p_{\mathrm{ri}} \right) / \mathrm{m} = \sum_{\mathrm{r}=1}^{\mathrm{m}} \left(\sum_{\mathrm{i}=1}^{\mathrm{k}} p_{\mathrm{ri}} \right) / \mathrm{m} \\ &= \sum_{\mathrm{r}=1}^{\mathrm{m}} (1)/\mathrm{m} = 1.\end{aligned}$$

□

Theorem 3.3. *The simple aggregate of a number of distinct values on attribute A_j of relation $\boldsymbol{R}$ is a special case of the probability distribution described by the probability aggregate operator.*

Proof. In the case when all attribute values are crisp and precise, we describe their values by probabilities that are either 1 (if the value is achieved), or 0 (if the value is not achieved). Summation of the 1s and division by the cardinality m gives the required result. □

We note in passing that the probability aggregate operator is a special case of the weighted sum operator which was defined in [14] in the context of integrating distributed databases.

4 Aggregation of Partial Probabilities

4.1 The Algorithm

In this section, we develop an approach which allows us to aggregate attribute values for any attribute with values which are expressed as probability distributions, i.e., any attribute of a partial probability distribution relation such as that presented in Table 1. Thus, this general aggregation operator ($gagg$) must include as special cases each of the operators ($sagg$ and $pagg$) developed in Section 3 for crisp data.

Notation 4.1. *As before, we consider an attribute A_j of a partial probability relation $\mathbf{R}$ with corresponding domain $D_\mathrm{j} = \{\mathrm{v}_1, \ldots \mathrm{v_k}\}$ which has tuples $t_1, \ldots t_m$. Then, the value of the rth tuple of A_j is a probability distribution described by the vectors:*

$$t_\mathrm{r}.A_\mathrm{j} = \left(\mathrm{f}_{\mathrm{r1}}^{(\mathrm{j})}, \ldots, \mathrm{f}_{\mathrm{rg_r}}^{(\mathrm{j})}\right)$$

and

$$\mathbf{P}_\mathrm{r} = \left(\mathrm{S}_{\mathrm{r1}}^{(\mathrm{j})}, \ldots, \mathrm{S}_{\mathrm{rg_r}}^{(\mathrm{j})}\right),$$

where $\mathbf{P}_\mathrm{r}$ is a partition of the domain D_j and

$$\mathrm{f}_{\mathrm{r}\ell}^{(\mathrm{j})} = \mathrm{Prob}\left\{\text{tuple value is a member of } \mathrm{S}_{\mathrm{r}\ell}^{(\mathrm{j})}\right\}.$$

Here, $\mathrm{g_r}$ is the number of sets in the partition for tuple t_r. We further define:

$$q_{\mathrm{ir}\ell} = \begin{cases} 1 & \text{if } \mathrm{v_i} \in \mathrm{S}_{\mathrm{r}\ell}^{(\mathrm{j})} \\ 0 & \text{otherwise.} \end{cases}$$

This notation is of a general form that can be used to describe any of the data types we have in mind, as exemplified in Table 1.

Definition 4.1. *The* general aggregate *of a number of probability distribution values on attribute A_j of relation $\boldsymbol{R}$, denoted $gagg(\boldsymbol{R}.A_\mathrm{j})$, is defined as a vector-valued function: $gagg(\boldsymbol{R}.A_\mathrm{j}), = (\pi_1, \ldots \pi_k)$, where the π_is are computed from the iterative scheme:*

$$\pi_\mathrm{i}^{(n)} = \pi_\mathrm{i}^{(n-1)} \left(\sum_{\mathrm{r}=1}^{\mathrm{m}} \sum_{\ell=1}^{\mathrm{g_r}} \left(\mathrm{f}_{\mathrm{r}\ell}^{(\mathrm{j})} q_{\mathrm{ir}\ell} / \sum_{\mathrm{u}=1}^{\mathrm{k}} \pi_\mathrm{u}^{(n-1)} q_{\mathrm{ur}\ell} \right) \right) / \mathrm{m} \text{ for } \mathrm{i} = 1, \ldots \mathrm{k}.$$

Here, $\pi_\mathrm{i}^{(n)}$ is the value of π_i at the nth iteration. This formula may be regarded as an iterative scheme which at each step apportions the data to the partial values according to the current values of the probabilities. Thus, for example, if individual 1 has mild or no heart disease with probability 0.25 and the probabilities of mild and no heart disease are 0.2 and 0.3, respectively, then we calculate a contribution $0.25^* \frac{0.2}{0.2+0.3} = 0.1$ to mild and $0.25^* \frac{0.3}{0.2+0.3} = 0.15$ to no heart disease.

Where the data are crisp, i.e., if we are trying to estimate π_i from a singleton partition set $\mathrm{S}_{\mathrm{ri}}^{(\mathrm{j})}$ of tuple r, then the contribution to π_i is simply $\mathrm{f}_{\mathrm{ri}}^{(\mathrm{j})}$. Otherwise, we apportion the probability $\mathrm{f}_{\mathrm{ri}}^{(\mathrm{j})}$ to π_b, where $\mathrm{v_b} \in \mathrm{S}_{\mathrm{ri}}^{(\mathrm{j})}$ in the proportion

$$\pi_\mathrm{j} / \left(\sum_{\mathrm{j} \in \mathrm{S}_{\mathrm{bi}}^{(\mathrm{j})}} \pi_\mathrm{j} \right).$$

In the next section, we show that this formula produces solutions for the π_is which minimize the Kullback-Leibler information divergence. We illustrate the algorithm in the following example:

Example 4.1. We illustrate how the algorithm works by considering the first two tuples of the attribute HEART_DISEASE in Table 1.

Here,

$$g_1 = 2; S_{11} = \{S\}; S_{12} = \{M, N\}; f_{11} = 0.75; f_{12} = 0.25;$$
$$q_{111} = 1; q_{212} = 1; q_{312} = 1;$$

and

$$g_2 = 3; S_{21} = \{S\}; S_{22} = \{M\}, S_{23} = \{N\}; f_{21} = 0.75;$$
$$f_{22} = 0.2; f_{23} = 0.05; q_{121} = 1; q_{222} = 1; q_{323} = 1.$$

The iteration scheme in Definition 4.1 is then given by:

$$\pi_1^{(n)} = \pi_1^{(n-1)}\left(0.75/\pi_1^{(n-1)} + 0.75/\pi_1^{(n-1)}\right)/2$$
$$\pi_2^{(n)} = \pi_2^{(n-1)}\left(0.2/\pi_2^{(n-1)} + 0.25/\left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)\right)/2$$
$$\pi_3^{(n)} = \pi_3^{(n-1)}\left(0.05/\pi_3^{(n-1)} + 0.25/\left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)\right)/2$$

from which we obtain the result that:

$$\pi_1 = 0.75 \text{ and}$$
$$\pi_2^{(n)} = 0.1 + 0.125\pi_2^{(n-1)}/\left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)$$
$$\pi_3^{(n)} = 0.025 + 0.125\pi_3^{(n-1)}/\left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right).$$

By iteration, or, in this case, simply substitution, we obtain the result that $\pi_1 = 0.75$, $\pi_2 = 0.2$, and $\pi_3 = 0.05$.

Theorem 4.1. *The aggregation operators defined in Section 3 for crisp data are special cases of the general aggregate operator as defined in Definition 4.1.*

Proof. We have already shown in Theorem 3.3 that the simple aggregate is a special case of the probability aggregate, so it suffices to show that the probability aggregate is a special case of the general aggregate. In this case, all the sets $S_{r\ell}^{(j)}$ of each partition $\mathbf{P}_r = \left(S_{r1}^{(j)}, \ldots, S_{rg_r}^{(j)}\right)$ are singleton sets and for each case the corresponding partition is $\mathbf{P}_r = \left(S_{r1}^{(j)}, \ldots, S_{rk}^{(j)}\right)$, where k is the number of values in the domain D_j of attribute A_j. The iterative scheme in Definition 4.1 then becomes:

$$\pi_i^{(n)} = \pi_i^{(n-1)}\left(\sum_{r=1}^{m} f_{ri}/\pi_i^{(n-1)}\right)/m = \frac{1}{m}\sum_{r=1}^{m} f_{ri} \text{ for } i = 1, \ldots k,$$

as before. □

Our definition of the general aggregate operator therefore provides a general approach that can aggregate both imprecise and uncertain data in a consistent manner. The way in which the algorithm treats imprecise data is intuitively appealing since imprecise data values are apportioned to the appropriate aggregated probabilities according to the probabilities of their taking each of the possible values. However, in addition, we now go on to show that the general aggregation algorithm also provides an approach that has a strong theoretical underpinning.

4.2 The Theoretical Framework

Thus far, we have motivated our definition of the general aggregation operator by showing that it generalizes similar operators for crisp data and does so in a manner which apportions uncertain belief in an intuitive manner. We now go on to show that the iteration equations which define the aggregation in fact minimize the Kullback-Leibler information divergence between the aggregated probability distribution $\{\pi_i\}$ and the data $\{f_{rs}\}$. In statistical terminology, this is equivalent to maximizing the likelihood of the model given the data.

The *Kullback-Leibler information divergence* between two distributions $p = (p_1, \ldots p_n)$ and $q = (q_1, \ldots q_n)$ is defined as:

$$D(p \parallel q) = \sum_j p_j \log(p_j/q_j).$$

In our case, minimizing the Kullback-Leibler information divergence or equivalently maximizing the log-likelihood becomes:

$$\text{Maximize } W = \sum_{r=1}^{m}\sum_{\ell=1}^{g_r} f_{r\ell}^{(j)} \log\left(\sum_{i=1}^{k} q_{ir\ell}\pi_i\right) \text{ subject to } \sum_{i=1}^{k}\pi_i = 1.$$

Vardi and Lee [23] have shown that this problem belongs to a general class that they term Linear Inverse Problems (LININPOS). For such problems, they develop a general algorithm, which, in our case, is equivalent to the iterative scheme in Definition 4.1. Their general algorithm is then shown to converge monotonically to the solution of the minimum information divergence equation. This iterative scheme is in fact an example of the EM (expectation-maximization) algorithm [9], which has been widely used in statistics for the solution of incomplete data problems. For example, the EM algorithm has been used for the integration of distributed data [21] in a similar fashion to our present approach.

4.3 Reduction of the Attribute Values

Vardi and Lee [23] have shown that, in general, for such problems, the iterative scheme always converges, but where there is an identifiability problem the solution may not be unique. We must therefore determine whether a unique solution always exists to the problem we have posed and, if not, how we can find a solution. This problem relates to that described by Malvestuto [13] for database schema decomposability. In our context, nonidentifiability of attribute values arises in situations where there is not enough information in the different partitions to completely separate the values. This may be because the values in question always occur together in the partial values or because there is insufficient data to determine the exact probabilities at the required granularity. A reduction algorithm [10] has previously been developed which tells us the finest granularity at which we can assign probabilities as a result of combining data from different partial values. We now adapt this algorithm to our problem.

Definition 4.2. *We define a* concentration graph *for an attribute to be a multipartite graph* $G = (X_1 \cup X_2 \ldots \cup X_n \ldots \cup Y, E)$; X_i *is the set of nodes corresponding to partial values of the ith*

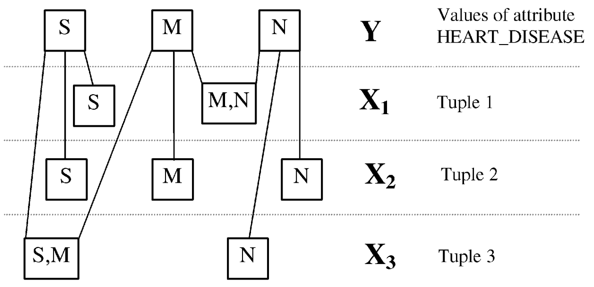

Fig. 1. An example of a concentration graph.

tuple value of an imprecise probability data model—these are now the elements of the partition; Y *is the set of nodes corresponding to the singleton values of the domain of a given attribute;* E *is the set of edges which join the* $\mathrm{X_i}$ *nodes to the* Y *nodes. We may here regard each subgraph* $\mathrm{G} = (\mathrm{X_i} \cup \mathrm{Y}, \mathrm{E})$ *to be bipartite for* $\mathrm{i} = 1, \ldots \mathrm{n}$*. Thus, each node in* Y *which represents a value contained in the partial value in* $\mathrm{X_i}$ *provides an edge. Essentially, the graph describes the concentration of extensional data on the imperfect domain values.*

A simple example of a concentration graph is obtained by considering the first three tuples of the attribute HEART_DISEASE in Table 1. The corresponding concentration graph is presented in Fig. 1. A concentration graph is so named because it describes which subsets of the attribute domain the probability is concentrated on.

Definition 4.3. *We define a* concentration matrix $\mathbf{C} = \{\mathrm{c_{ij}}\}$ *to be a matrix such that* $\mathrm{c_{ij}}$ *is equal to 1 if there is an edge joining vertex* $\mathrm{V_i}$ *of* Y *to vertex* $\mathrm{V_j}$ *of* X *and 0 otherwise.*

The concentration matrix corresponding to the graph in Fig. 1 is therefore:

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix},$$

where each column of $\mathbf{C}$ corresponds to a partial value in the data and the columns are in order of appearance within the tuple and then by tuple. Here, columns 1 and 2 refer to the partial values in tuple 1, columns 3, 4, and 5 refer to the partial values in tuple 2, and columns 6 and 7 refer to the partial values in tuple 3.

The refined concentration matrix $\mathbf{Q}$ is then obtained by collapsing $\mathbf{C}$ by eliminating replicated columns. Here, $\mathbf{Q} = \{\mathrm{q_{ij}}\}$, where the $\mathrm{q_{ij}}$s are defined as in Section 4.1 and each column of $\mathbf{Q}$ corresponds to a partial value present in the data. Corresponding to Fig. 1, we therefore obtain:

$$\mathbf{Q} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

We now consider $\mathbf{Q}$ in order to determine if there is a unique solution to the iteration scheme in Definition 4.1. Vardi and Lee [23] have shown that a sufficient condition for uniqueness is that the rows of the matrix $\mathbf{Q} = \{q_{\mathrm{ij}}\}$ are linearly independent. They also show that a necessary and sufficient condition for uniqueness is that the matrix equation $\mathbf{F} = \pi'.\mathbf{Q}$ does not have multiple solutions, where $\mathbf{Q} = \{q_{\mathrm{ij}}\}, \pi = \{\pi_{\mathrm{i}}\}, \pi'$ denotes the transpose of π, and . denotes matrix multiplication. The vector $\mathbf{F}$ consists of elements that are the sums of the probabilities $\mathrm{f_{rs}}$ aggregated over the set $\mathrm{S_{rs}}$. This vector is then normalized so that the elements sum to 1. Here, the π_{i}s and $\mathrm{f_{rs}}$s are as defined in Notation 4.1.

In our case, it is likely that we will have a large proportion of the data that is crisp and a much smaller proportion that is uncertain or imprecise. Typically, null values are relatively rare compared with values which are known exactly. We will therefore assume that there is always at least one crisp value for each possible domain value. In this case, we can permute the rows and columns of $\mathbf{Q}$ so that the first k columns represents the k singleton sub-partitions of $\mathbf{P}$. Where we have identical rows we collapse two rows into one and combine the values since, in this case, the values are not separable. Then, the first k rows and columns comprise an identity matrix and the rows of $\mathbf{Q}$ are therefore linearly independent. Here, the solution of the iterative scheme is clearly unique. More generally, if there is a large degree of imprecision, we may have possible values of D_{j} that are not crisp for any tuple in R. In this case, we need to test for uniqueness using the reduction algorithm in [10] since although the algorithm will still converge to a solution, it may not be very informative.

4.4 Application

In our simulations, we have used the scaled frequencies (the $\mathrm{f_{rs}}$s) of all the crisp data as an initial value for the general aggregation operator. We obtain this initial value from the vector $\mathbf{F}$, as defined in the last section; we extract those elements of $\mathbf{F}$ which correspond to singleton values, in our notation, the first k elements of $\mathbf{F}$ and scale these so they add to 1. This is the answer we would get if we applied the simple aggregation operator and ignored all imprecise data. Such a starting value should therefore prove reasonably close to the final answer in situations where the number of imprecise tuples is low and, therefore, leads to fast convergence. An alternative starting value that is easy to compute is to assume all probabilities are initially equal.

We now illustrate the approach by applying the algorithm to the attribute HEART_DISEASE in Table 1.

Example 4.2. In this case,

$$\mathbf{Q} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

corresponding to $\{\mathrm{S}\}$, $\{\mathrm{M}\}$, $\{\mathrm{N}\}$, and $\{\mathrm{M,N}\}$, respectively, where we have permuted the rows and columns to reflect a logical order of subsets. The vector $\mathbf{F}$ which was defined in Section 4.2 is then given by:

$$\mathbf{F} = (0.49, 0.145, 0.18, 0.185).$$

The iterative scheme for this data thus is:

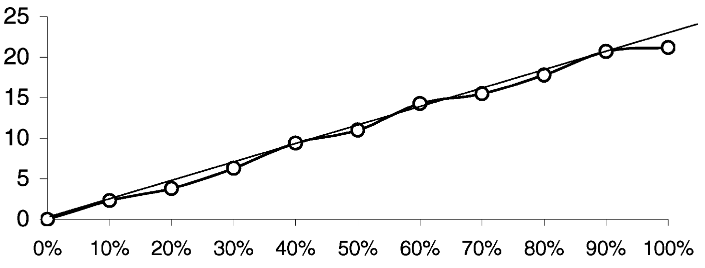

Fig. 2. Number of iterations by percentage of imprecision.

$$\pi_1^{(n)} = 0.49$$
$$\pi_2^{(n)} = 0.145 + 0.185^{*}\pi_2^{(n-1)} / \left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)$$
$$\pi_3^{(n)} = 0.18 + 0.185^{*}\pi_3^{(n-1)} / \left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)$$

from which we obtained the result that:

$$\pi_1 = 0.49, \pi_2 = 0.228, \text{ and } \pi_3 = 0.282$$

in 1 iteration (to three decimal places), using a start value of $(0.601, 0.178, 0.221)$.

In order to test the performance of the *gagg* operator, we have carried out simulations based on the example in Table 1. This was accomplished by generating 100 cases from a multinomial distribution with probabilities 0.6, 0.175, and 0.225 of having severe, mild, and no heart disease, respectively. In each case, this was repeated 10 times for imprecision at 10 percent, 20 percent, ...90 percent, and 100 percent. Here, imprecision is defined as the proportion of the data which is not precise. It is simulated by randomly assigning the appropriate fraction of heart disease cases to either of the imprecise sets {severe, mild} or {mild, no}. The results of our simulation are presented in Fig. 2, where convergence is assumed to have occurred when all probabilities agree with their values at the previous iteration to three decimal places. We see that the number of iterations increases linearly with the percentage of imprecision.

In general, it is recognized that the convergence of the EM algorithm may be slow [24]. However, for our current application, where the majority of the attribute values are crisp, convergence is expected to be reasonably fast, and our simulation results support this claim. In the special case when all of the values are crisp, convergence is achieved immediately in a single iteration. However, if there is a high proportion of imprecise values, then convergence of the EM algorithm may be slow, though there are a number of recent papers which suggest ways in which convergence can be accelerated, e.g., [11], [16].

The computation of the *gagg* operator divides into two main stages. The purpose of the first stage is to determine the frequencies of subsets (partial values) and the purpose of the second stage is to compute the underlying proportions of each attribute value. Here, the time complexity of the first stage depends on the way in which the imprecise and uncertain tuples are stored. Assume that we map the domain values to a finite set of integers and store the integers in the cells for the case of crisp values. A negative integer could be used as a pointer to a partial probability value and this would minimize the storage requirement. The partial probability value would be stored in a complex data structure, but this would only need to be used infrequently. A complete scan through the relation would be required to compute the frequencies (fs): It is unlikely that indexing on the storage structures would be used because of the small number of categories. Hence, the complexity would be $\mathrm{O(m)}$, where m is the number of tuples, and we are assuming that most of the cost is in reading data from disk.

The second stage of computation is the EM algorithm (Definition 4.1). Here, the complexity depends on the number of attribute values, i.e., k—the domain size, the number of distinct subsets in the whole dataset, and the imprecision—proportion of data values which are imprecise. The time complexity is then linear with respect to the number of attribute values, linear with respect to the degree of imprecision in the data (as our simulations have shown), and linear with respect to the number of distinct subsets in the data, provided that the number of noncrisp subsets is not too large compared with the number of crisp subsets. In practice, our confidence in the solution is likely to be low if there is a large number of distinct subsets so we advise that the algorithm be used only when the number of distinct subsets is relatively small. Nonetheless, our simulations have suggested that the time complexity may be linear even with quite a high number of distinct subsets in the data.

4.5 Other Approaches

Other authors ([4], [7]) have suggested that the singleton values within a partial value should divide the probability assigned to this partial value equally between them; this means that the conditional distribution of the singleton values which comprise the partial value are assumed uniform. Such an approach may be useful a priori, particularly when we are seeking to answer tuple-based queries, since it does not make any assumptions about preexisting knowledge. However, in our case, we seek new knowledge about the attributes and, therefore, use the a posteriori knowledge that we get from the rest of the data; this means that all the data are utilized. Thus, for example, if our database tells us that 90 percent of technical staff are programmers and 10 percent are systems analysts, then, for Fred who is either a programmer or a systems analyst, we assign a probability of 0.9 to his being a programmer and 0.1 to his being a systems analyst. As we have pointed out, our approach also has the advantage of maximizing the likelihood and minimizing the Kullback-Leibler information divergence.

5 Aggregation Involving Several Attributes

In the previous section, we provided a general aggregation operator for aggregating attribute values which are represented as imprecise and uncertain data. This operator is a vector valued function which operates on a single attribute by aggregating the individual probability distributions from each tuple to form a vector of probabilities of the occurrence of each possible value in the attribute domain. While it is useful to compute such aggregates, it is often of more interest to derive the joint probabilities of several attributes' values occurring simultaneously. This is particularly the case for knowledge discovery where, for example, we might want to investigate whether the probability of heart disease is higher for smokers than for nonsmokers.

In fact, our methodology of the previous section may be extended to cover the situation in which we wish to compute aggregates of several variables. In this case, we consider the cross product of domains $D_1, \dots D_n$. We define the cross product $\mathtt{D} = D_1 \mathrm{x} \dots \mathrm{x} D_n$. Let the values of domain D_j be given by $\{v_1^{(j)}, \dots, v_{k_j}^{(j)}\}$. Then, the values of $\mathtt{D}$ are of the form $\{v_{a_1}^{(1)} \mathrm{x} \dots \mathrm{x}\ v_{a_n}^{(n)}\}$, where $a_i \in \{1, \dots k_i\}; i = 1, \dots n$. The partial probability distribution then assigns probabilities $\{f_{rs}\}$ to partitions $\{S_{rs}\}$ of $\mathtt{D}$ for each tuple. The algorithm proceeds as in Definition 4.1. We illustrate this approach by deriving the joint distribution of the attributes SMOKING_STATUS and HEART_DISEASE in Table 1.

Example 5.1. The possible singleton values of the domain

$$\mathtt{D} = \mathrm{SMOKING\_STATUS} \times \mathrm{HEART\_DISEASE}$$

are: $\{S,S\}$, $\{S,M\}$, $\{S,N\}$, $\{E,S\}$, $\{E,M\}$, $\{E,N\}$, $\{N,S\}$, $\{N,M\}$, $\{N,N\}$ with corresponding probabilities $\pi_1, \dots \pi_9$. The subset of $\mathtt{D}$ which are represented in the data are: $\{S,S\}$, $\{S,M\}$, $\{S,N\}$, $\{E,S\}$, $\{E,M\}$, $\{E,N\}$, $\{N,S\}$, $\{N,M\}$, $\{N,N\}$, $\{S,\{M,N\}\}$, $\{\{E,\{M,N\}\}$. $\mathbf{Q}$ is then given by:

$$\mathbf{Q} = \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&1&0 \\ 0&0&1&0&0&0&0&0&0&1&0 \\ 0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0 \end{pmatrix}$$

and

$$\mathbf{F} = \{0.375, 0.08, 0.005, 0.105, 0.045, 0.025, 0.01, 0.02, 0.15, 0.105, 0.08\}.$$

The iterative scheme for this data is given by:

$$\pi_1^{(n)} = 0.375$$
$$\pi_2^{(n)} = 0.08 + 0.105^* \pi_2^{(n-1)} / \left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)$$
$$\pi_3^{(n)} = 0.005 + 0.105^* \pi_3^{(n-1)} / \left(\pi_2^{(n-1)} + \pi_3^{(n-1)}\right)$$
$$\pi_4^{(n)} = 0.105$$
$$\pi_5^{(n)} = 0.045 + 0.08^* \pi_5^{(n-1)} / \left(\pi_5^{(n-1)} + \pi_6^{(n-1)}\right)$$
$$\pi_6^{(n)} = 0.025 + 0.08^* \pi_6^{(n-1)} / \left(\pi_5^{(n-1)} + \pi_6^{(n-1)}\right)$$
$$\pi_7^{(n)} = 0.01$$
$$\pi_8^{(n)} = 0.02$$
$$\pi_9^{(n)} = 0.15$$

from which we obtained the result that:

$$\pi_1 = 0.375, \pi_2 = 0.179, \pi_3 = 0.011, \pi_4 = 0.105, \pi_5 = 0.096, \pi_6 = 0.054, \pi_7 = 0.01, \pi_8 = 0.02, \pi_9 = 0.15,$$

where convergence occurred in 1 iteration.

Here, we note that although for computational convenience we have used an example that has only one attribute which is fully imprecise, we may readily deal with cases when several attributes are both imprecise and uncertain by apportioning the imprecise data to corresponding sets. If necessary, we may use the reduction algorithm discussed in Section 4.3 to decide whether the probabilities may be computed uniquely. In the case where two rows of the refined concentration matrix are identical, we collapse these two rows into one and the corresponding probability is calculated for the sum of these probabilities (which cannot be determined separately). The reduction then determines whether, subsequent to collapsing, the iterative scheme can be solved uniquely.

6 Knowledge Discovery

In order to discover knowledge from databases, we are often concerned with inducing beliefs or rules from database tuples [2]. Rules tend to be based on sets of attribute values, partitioned into an antecedent and a consequent. A typical "if then" rule, of the form "if antecedent = true, then consequent = true," is given by "if an individual smokes and is hypertensive, then he/she is very likely to suffer from heart disease." Support for such a rule is based on the proportion of tuples in the database which have the specified attribute values in both the antecedent and the consequent. A major advantage of finding and evaluating such association rules is that often they require only standard database functionality, and they may be implemented using embedded SQL [2]. Such an approach has been explored in a number of papers, e.g., [1] for transaction data, and [3] using Evidence Theory for uncertain and imprecise data.

Using the approach that we have described in previous sections, we may use the general aggregation operator to derive joint probabilities and then compute conditional probabilities to assess the potential for new knowledge. Thus, for example, using the data in Table 1 we obtain:

probability (smokes and severe heart disease) $= 0.375$ and probability (smokes) $= 0.565$.

Therefore,

$$\text{probability(severe heart disease} \mid \text{smokes)} = 0.375/0.565 = 0.664.$$

However, the unconditional

$$\text{probability(severe heart disease)} = 0.49$$

is substantially smaller. It is therefore quite likely that we would induce the rule "smoking increases the chances of heart disease" from this data, subject to thresholding parameters.

The large itemset approach [1], commonly used in Data Mining, is appropriate to data that are taken from a shopping basket where each item, e.g., apples, oranges, etc. may or may not be included. Such data may be represented in our data model by a number of attributes which in statistical terminology represent dichotomous variables. Thus, we have an attribute "apple" which may take the value "yes" or "no," another attribute "orange" with the same possible values, etc. This data model is then a special case of our own where an attribute may have many values, including partial values and may also include uncertainty, represented by a probability distribution. The *Gagg* operator then generalizes the computation of cardinalities of itemsets by apportioning the partial memberships of generalized itemsets. In each case, we are calculating the numbers of members of appropriate sets but our case is more complex in that we do not always know if an item belongs to a set or not. Our *Gagg* operator may therefore be regarded as a generalization of the itemset approach which allows us to extend the computation of association rules to the case of imprecise and uncertain data.

Since the general aggregation operator allows us to compute joint probabilities for any combination of attributes, we may therefore use it to assess the validity of any rules of this type. Thus, if we wish to examine the rule:

$$A_1, \ldots A_r \rightarrow B_1, \ldots B_s,$$

where $A_1 \ldots A_r, B_1, \ldots B_s$ are all attributes of a relation $\mathbf{R}$, then we compute the joint probabilities:

$$\text{Prob}(A_1 \ldots A_r, B_1, \ldots B_s)$$

and $\text{Prob}(A_1 \ldots A_r)$ from which we may assess the validity of the rule by calculating the conditional probability:

$$\text{Prob}(B_1, \ldots, B_s | A_1 \ldots A_r) = \text{Prob}(A_1 \ldots A_r, B_1, \ldots B_s) / \text{Prob}(A_1 \ldots A_r).$$

By using such an approach, we may therefore determine the strength and support for various rules under consideration. Thresholding then takes place to decide if the candidate rules are sufficiently likely to merit the user being informed.

7 Summary and Further Work

It is often required to make decisions based on stochastic data that are subject to imperfections such as imprecision and uncertainty. Traditional database applications fail to handle such information satisfactorily, particularly with respect to the provision of appropriate functionality for knowledge discovery where there is a need for general attribute aggregation operators.

In this paper, we have considered an extended relational data model that can handle both imprecise and uncertain data. In this context, we have developed a general aggregation operator that allows us to determine a probability distribution for attribute values either for a single attribute or for the cross-product of a number of attributes. Such functionality has potential for knowledge discovery in databases where we may use it to assess the validity of potential association rules.

In previous work, we have developed a data model and suitable operators for storing and manipulating summary data held as aggregates [19], [20]. Such data may be combined at a high global level without having to revert to the original local data—for example, we may compute conditional probabilities, as discussed above. The extension of this approach to imprecise and uncertain data should facilitate the speed and efficiency of knowledge discovery using such imperfect data.

In addition, the approach that we have adopted may be used to derive other aggregates, such as the mean and standard deviation of numerical data, held in a format such as we have defined. For example, if our data were in the form of salary bands with corresponding probabilities, then the mean and standard deviation of salaries could be found from our probability estimates, using elementary formulae.

Further work will explore the possibility of including domain knowledge, such as that held as integrity constraints, into the model. Another area that we intend to investigate in the future is the provision of techniques for accelerating the algorithm. The building of a database machine to support the functionality offered by our model is another possible future direction since such a development is likely to provide performance improvements which would facilitate the practical use of our approach for knowledge discovery.

Acknowledgments

This work was partially funded by IDARESA (ESPRIT project no. 20478) and partially funded by ADDSIA (ESPRIT project no. 22950) which are both part of EUROSTAT's DOSIS (Development of Statistical Information Systems) initiative.

References

[1] R. Agrawal, H. Manilla, R. Srikant, H. Toivonen, and A.I. Verkami, "Fast Discovery of Association Rules," *Advances in Knowledge Discovery and Data Mining,* U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., pp. 307-328, 1996.
[2] S.S. Anand, B.W. Scotney, M.G. Tan, S.I. McClean, D.A. Bell, J.G. Hughes, and I.C. Magill, "Designing a Kernel for Data Mining," *IEEE Expert,* pp. 65-74, 1997.

[3] S.S. Anand, D.A. Bell, and J.G. Hughes, "EDM: A General Framework for Database Mining Based on Evidential Theory," *Data and Knowledge Eng. J.*, vol. 18, pp. 189-223, 1996.
[4] D. Barbará, H. Garcia-Molina, and D. Porter, "The Management of Probabilistic Data," *IEEE Trans. Knowledge and Data Eng.*, vol. 4, pp. 487-501, 1992.
[5] D.A. Bell, J.W. Guan, and S.K. Lee, "Generalized Union and Project Operations for Pooling Uncertain and Imprecise Information," *Data and Knowledge Eng.*, vol. 18, pp. 89-117, 1996.
[6] C.-S. Chang and A.L.P. Chen, "Determining Probabilities for Probabilistic Partial Values," *Proc. Int'l Conf. Data and Knowledge Systems for Manufacturing and Eng.*, pp. 277-284, 1994.
[7] A.L.P. Chen and F.S.C. Tseng, "Evaluating Aggregate Operations over Imprecise Data," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, pp. 273-284, 1996.
[8] L.G. Demichiel, "Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains," *IEEE Trans. Knowledge and Data Eng.*, vol. 4, pp. 485-493, 1989.
[9] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm (with discussion)," *J. R. Statistics Soc.*, vol. B39, pp. 1-38, 1977.
[10] L.A. Goodman, "Partitioning of Chi-Square, Analysis of Marginal Contingency Tables, and Estimation of Expected Frequencies in Multidimensional Contingency Tables," *J. Am. Statistical Assoc.*, vol. 66, pp. 339-344, 1971.
[11] J.M. Jamshidian and R.I. Jennrich, "Acceleration of the EM Algorithm by using Quasi-Newton Methods," *J. R. Statistics Soc.*, vol. B59, pp. 569-587, 1997.
[12] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo, "Finding Interesting Rules from Large Sets of Association Rules," *Proc. Third Int'l Conf. Information and Knowledge Management,* N.R. Adam, K.B. Bhargava, and Y. Yesha, eds. pp. 401-407, 1994.
[13] F.M. Malvestuto, "A Universal-Scheme Approach to Statistical Databases Containing Homogeneous Summary Tables," *ACM Trans. Database Systems,* vol. 18, pp. 678-708, 1993.
[14] S.I. McClean and B.W. Scotney, "Using Evidence Theory for the Integration of Distributed Databases," *Int'l J. Intelligent Systems,* vol. 12, no. 10, pp. 763-776, 1997.
[15] S.I. McClean, B.W. Scotney, and C.M. Shapcott, "Aggregation of Imprecise and Uncertain Information for Knowledge Discovery in Databases," *Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining (KDD '98),* pp. 269-270, 1998.
[16] G. Molenberghs and E. Goetghebeur, "Simple Fitting Algorithms for Incomplete Categorical Data," *J. R. Statistics Soc.*, vol. B59, pp. 401-414, 1997.
[17] S. Parsons, "Current Approaches to Handling Imperfect Information in Data and Knowledge Bases," *IEEE Trans. Knowledge and Data Eng.*, vol. 8, pp. 353-372, 1996.
[18] E.A. Rundersteiner and L. Bic, "Evaluating Aggregates in Possibilistic Relational Databases," *Data and Knowledge Eng. J.*, vol. 7, pp. 239-267, 1992.
[19] M.H. Sadreddini, D.A. Bell, and S.I. McClean, "A Model for Integration of Raw Data and Aggregate Views in Heterogeneous Statistical Databases," *Database Technology,* vol. 4, no. 2, pp. 115-127, 1991.
[20] M.H. Sadreddini, D.A. Bell, and S.I. McClean, "A Framework for Query Optimization in Distributed Statistical Databases," *Information and Software Technology,* vol. 6, pp. 363-377, 1992.
[21] B.W. Scotney, S.I. McClean, and M.C. Rodgers, "Optimal and Efficient Integration of Heterogeneous Summary Tables in a Distributed Database," *Data and Knowledge Eng. J.*, vol. 29, pp. 337-350, 2000.
[22] F.S.C. Tseng, A.L.P. Chen, and W.-P. Yang, "Answering Heterogeneous Database Queries with Degrees of Uncertainty," *Distributed and Parallel Databases,* vol. 1, pp. 281-302, 1993.
[23] Y. Vardi and D. Lee, "From Image Deblurring to Optimal Investments: Maximum Likelihood Solutions for Positive Linear Inverse Problems (with discussion)," *J. R. Statistics Soc. B*, vol. B55, no. 3, pp. 569-612, 1993.
[24] C.F.J. Wu, "On the Convergence Properties of the EM Algorithm," *The Annals of Statistic,* vol. 11, no. 1, pp. 95-103, 1983.

Sally McClean received her first degree in mathematics at Oxford University, then a MSc degree at Cardiff in mathematical statistics and operational research followed by a DPhil degree at the University of Ulster. She is currently a professor of mathematics in the School of Information and Software Engineering at the University of Ulster. Her research interests are in mathematical modeling, applied probability, multivariate statistical analysis, and applications of mathematical and statistical methods to computer science, particularly database technology. She has published and edited five books and more than 100 research papers in these subject areas. Dr. McClean is a fellow of the Royal Statistical Society, a member of the Operational Research Society, an associate fellow of the IMA, and a member of the IEEE.

Bryan Scotney received the BSc degree in mathematics from the University of Durham and the PhD degree in mathematics from the University of Reading. His thesis was on optimal error analysis for Petrov-Galerkin finite element methods. He is currently a reader in mathematics in the School of Information and Software Engineering at the University of Ulster. His research interests are in numerical analysis, mathematical modeling, and the application of mathematics to computer science, including statistical database technology and computer vision. He has coauthored one book and published more than 50 research papers in these areas.

Mary Shapcott is currently a senior lecturer in the School of Information and Software Engineering at the University of Ulster. Her research interests are mainly in the areas of bayesian belief networks, concurrent and distributed computing, distance education, and various aspects of database technology, particularly, statistical and probabilistic databases.