Extension of the Relational Algebra to Probabilistic Complex Values

Thomas Eiter[1], Thomas Lukasiewicz[1], and Michael Walter[2,3]

[1] Institut und Ludwig Wittgenstein Labor für Informationssysteme, TU Wien
Favoritenstraße 9–11, A-1040 Wien, Austria
{eiter, lukasiewicz}@kr.tuwien.ac.at

[2] Institut für Informatik, Universität Gießen
Arndtstraße 2, D-35392 Gießen, Germany

Abstract. We present a probabilistic data model for complex values. More precisely, we introduce probabilistic complex value relations, which combine the concept of probabilistic relations with the idea of complex values in a uniform framework. We then define an algebra for querying database instances, which comprises the operations of selection, projection, renaming, join, Cartesian product, union, intersection, and difference. We finally show that most of the query equivalences of classical relational algebra carry over to our algebra on probabilistic complex value relations. Hence, query optimization techniques for classical relational algebra can easily be applied to optimize queries on probabilistic complex value relations.

1 Introduction

Databases are a central component of many business and information systems. During the past two decades, relational database systems have replaced earlier systems and have become a standard for data management. Various needs in practice, however, cannot be appropriately managed with current commercial database systems, which are largely based on the plain relational data model.

An important such need is the integration of models of uncertainty into databases. Applications that involve uncertainty abound. Research on this issue can be divided into four categories [5]: (1) handling null values, (2) retrieval of incomplete data, (3) management of vague data (for example, "John is tall") in fuzzy set approaches, and (4) management of ambiguous or imprecise data (for example, "John's salary is between 50K and 70K") in probabilistic approaches. Another important issue is the management of structured objects. The relational model has been early generalized for storing structured objects [16], which was a step towards object-oriented database systems (see [1] for a background on complex values and a historic account).

[3] Current address: IBM Germany, Überseering 24, D-22297 Hamburg, Germany. E-mail: MiWalter@de.ibm.com.

In many applications, especially in the business domain, uncertainty stems from ambiguity rather than from vagueness. Several models for incorporating ambiguity into the relational model have thus been proposed so far (e.g., [4, 2, 5, 12, 8]; see Section 7 for a discussion of these and further approaches). Informally, they attach a probability to each tuple and/or to each value in a set of possible values of an imprecise attribute of a tuple. The operations of relational algebra are then generalized to combine these probabilities in a suitable way, adopting some underlying assumptions (like independence or disjointness of events).

In this paper, we extend this line of research to databases storing structured objects. In detail, we present a probabilistic data model for complex values. To our knowledge, there is not much work in this direction so far. A probabilistic version of NF2 relations has been presented in [7] (building on earlier work [8] and making some limiting assumptions; see Section 7 for a comparison to our work).

Our model generalizes a similar model of annotated tuples [12] to complex values [1]. Informally, every complex value v is associated with a probability interval $[l, u]$ and an event e, forming a quadruple (v, l, u, e). The interval $[l, u]$ represents the likelihood that v belongs to the database, and e records information about how this value was derived. Note that interval probabilities provide a greater flexibility than point probabilities, and seem better suited especially to represent imprecise and subjective probabilistic knowledge. Moreover, we use intervals also for technical reasons (under the assumed probabilistic knowledge, even when using only point probabilities, we generally specify *a set* of probability distributions, rather than *a unique single* distribution). The following is an example of a relation containing probabilistic complex values:

v		l	u	e
patient	diseases	0.7	0.9	$e_1 \vee e_2$
John	{lung cancer, tuberculosis}			
patient	diseases	0.5	0.7	e_3
Jack	{leprosy}			

Observe that in the above model, probabilities are assigned to a complex value as a whole, and no impreciseness in attributes, given by sets of values with probabilities attached, can be explicitly expressed in the language. It has been shown in [12] that each tuple with imprecise attribute values can be represented by an annotated tuple, and that such a representation can be efficiently computed. Applying similar techniques, complex values with imprecise attributes may be represented by probabilistic complex values as above. To keep our model simple, we do not consider imprecise attributes here.

On top of our model, we define a relational algebra that uses, following [12], generic functions $\otimes, \oplus, \ominus$ for computing the probability range of the conjunction, disjunction, and difference of two events e_1 and e_2 from the probability ranges $[l_1, u_1]$ and $[l_2, u_2]$ of e_1 and e_2, respectively. Instances of these functions are selected according to the relationship between e_1 and e_2. For example, if e_1 and e_2 are independent, then the probability range of $e_1 \wedge e_2$ is given by $[l_1 \cdot l_2, u_1 \cdot u_2]$. However, if nothing is known about how e_1 and e_2 relate, then it is given by

$[\max(0, l_1 + l_2 - 1), \min(u_1, u_2)]$. Such generic functions allow us to remove the explicit or implicit assumptions about joint occurrence of tuples and/or attribute values in relations that are present in other approaches (see [4, 2, 5, 8]).

We further refine [12] by giving a model-theoretic definition of probabilistic combination strategies, which assumes a probabilistic semantics in which probability distributions are defined over a set of possible worlds. Furthermore, we propose probabilistic difference strategies that are not necessarily derived from conjunction strategies and negation. We show that these refinements lead to more precise results in certain cases (see Example 2.8).

The main contributions of this work can be briefly summarized as follows:

- We give a model-theoretic definition of probabilistic conjunction, disjunction, and difference strategies, which is based on a possible worlds semantics.
- We present a data model that is based on probabilistic complex value relations, which generalizes previous data models in the literature [12, 1].
- We define an algebra for querying database instances.
- We present equivalence results for query expressions, which can be used for query optimization. Since our model generalizes the one in [12], these results also apply to [12] as a special case. Note that the results on query equivalences in [12] are limited to the relationship between compaction and the standard relational algebra operators.

The rest of this paper is organized as follows. Section 2 gives some preliminaries. Sections 3 and 4 define the data model and the algebra. In Sections 5 and 6, we focus on query equivalences and optimization. Section 7 discusses related work, and Section 8 gives a short summary and an outlook on future research.

Note that for space limitations, proofs of the results are omitted (detailed proofs of the results are essentially given in [18]).

2 Probabilistic Background

In this section, we describe the probabilistic background of our approach to probabilistic complex value databases. We assume a semantics in which probabilities are defined over a set of possible worlds (see especially [3, 9, 17, 10]). Note that we adopt some technical notions from [13, 14].

The main aim of this section is to give a model-theoretic definition of probabilistic conjunction, disjunction, and difference strategies, which have been introduced by an axiomatic characterization in [12]. Given the probability ranges of two events e_1 and e_2, these strategies compute the probability range of the events $e_1 \vee e_2$, $e_1 \wedge e_2$, and $e_1 \wedge \neg e_2$, respectively. We allow a variety of different probabilistic conjunction, disjunction, and difference strategies to take into account the different dependencies between the two events e_1 and e_2.

We assume a set of *basic events* $\mathcal{B} = \{b_1, b_2, \ldots, b_n\}$ with $n \geq 1$. The set of *events* is the closure of $\mathcal{B}$ under the Boolean operations $\neg$ and $\wedge$. As usual, we use $(e_1 \vee e_2)$ to abbreviate $\neg(\neg e_1 \wedge \neg e_2)$. We use $\bot$ and $\top$ to abbreviate the *false event* $(b_1 \wedge \neg b_1)$ and the *true event* $\neg(b_1 \wedge \neg b_1)$, respectively. A *probabilistic pair* $(e, [l, u])$ consists of an event e and an interval $[l, u] \subseteq [0, 1]$, where l and u are rational numbers. We use $\bot$ and $\top$ to abbreviate $(\bot, [0, 0])$ and $(\top, [1, 1])$, respectively.

Informally, each tuple t in our probabilistic complex value database will be associated with a probabilistic pair $(e, [l, u])$. Intuitively, this means that t is identified with e and that the probability of t lies in the interval range $[l, u]$. More precisely, each tuple in a base relation will be assigned a probabilistic pair $(e, [l, u])$ with a basic event e. Moreover, each derived tuple t will be assigned a general probabilistic pair $(e, [l, u])$, where e encodes some information on how t is computed from tuples in the base relations.

An *implication* (resp., *negative correlation*, *positive correlation*, *independence*) *formula* is an expression $a \to b$ (resp., $NC(a, b)$, $PC(a, b)$, $Ind(a, b)$) with events a and b. A *dependence information* on two events a and b is a subset of $KB^\star(a, b) = \{Ind(a, b),\ NC(a, b),\ PC(a, b),\ a \to b,\ b \to a,\ a \wedge b \to \bot,\ \top \to a \vee b\}$.

Informally, to express that the probability of two events e_1 and e_2 lies in the intervals $[l_1, u_1]$ and $[l_2, u_2]$, respectively, we will use the two probabilistic pairs $(e_1, [l_1, u_1])$ and $(e_2, [l_2, u_2])$, respectively. The relationship between e_1 and e_2 will then be encoded by some dependence information $KB \subseteq KB^\star(e_1, e_2)$.

A *classical interpretation* I is a truth assignment to the basic events in $\mathcal{B}$, which is extended to all events as usual (that is, $(e_1 \wedge e_2)$ *is true in* I iff e_1 and e_2 are true in I, and $\neg e$ *is true in* I iff e is not true in I). We write $I \models e$ iff e is true in I. We use $\mathcal{I}_\mathcal{B}$ to denote the set of all classical interpretations on $\mathcal{B}$.

A *probabilistic interpretation* Pr is a mapping $Pr\colon \mathcal{I}_\mathcal{B} \to [0, 1]$ such that all $Pr(I)$ with $I \in \mathcal{I}_\mathcal{B}$ sum up to 1. It is extended to all events e as follows:

$$Pr(e) = \textstyle\sum_{I \in \mathcal{I}_\mathcal{B},\, I \models e} Pr(I)\,.$$

It is important to point out that probabilistic interpretations are defined on the set of all classical interpretations and not on the set of all basic events. That is, we do not assume that the basic events are pairwise mutually exclusive. Moreover, we do not assume that the basic events are pairwise independent.

The *truth* of probabilistic pairs, implication formulas, negative correlation formulas, positive correlation formulas, and independence formulas F in a probabilistic interpretation Pr, denoted $Pr \models F$, is defined as follows:

$$\begin{aligned}
Pr \models (e, [l, u]) &\ \text{ iff }\ Pr(e) \in [l, u]\,,\\
Pr \models a \to b &\ \text{ iff }\ Pr(a \wedge b) = Pr(a)\,,\\
Pr \models NC(a, b) &\ \text{ iff }\ Pr(a \wedge \neg b) = \min(Pr(a), Pr(\neg b))\,,\\
Pr \models PC(a, b) &\ \text{ iff }\ Pr(a \wedge b) = \min(Pr(a), Pr(b))\,,\\
Pr \models Ind(a, b) &\ \text{ iff }\ Pr(a \wedge b) = Pr(a) \cdot Pr(b)\,.
\end{aligned}$$

A probabilistic interpretation Pr is a *model* of a formula F iff $Pr \models F$. Pr is a *model* of a set of formulas $\mathcal{F}$, denoted $Pr \models \mathcal{F}$, iff Pr is a model of all $F \in \mathcal{F}$. The set $\mathcal{F}$ is *satisfiable* iff a model of $\mathcal{F}$ exists. A formula F is a *logical consequence* of $\mathcal{F}$, denoted $\mathcal{F} \models F$, iff each model of $\mathcal{F}$ is also a model of F.

The next result shows that there are combinations of probabilistic pairs $(e_1, [l_1, u_1])$ and $(e_2, [l_2, u_2])\}$ with dependence information $KB \subseteq KB^\star(e_1, e_2)$ that are unsatisfiable (note that this fact remains unmentioned in [12]).

Lemma 2.1. *Let $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_2, u_2])$ be two probabilistic pairs and let KB be a dependence information on e_1 and e_2 such that $KB \cup \{p_1, p_2\}$ is satisfiable. Then, all the following conditions hold:*

1. If $KB \models e_1 \wedge e_2 \rightarrow \bot$*, then* $l_1 + l_2 \leq 1$.
2. If $KB \models e_1 \rightarrow e_2$*, then* $l_1 \leq u_2$.
3. If $KB \models e_2 \rightarrow e_1$*, then* $l_2 \leq u_1$.
4. If $KB \models \top \rightarrow e_1 \vee e_2$*, then* $u_1 + u_2 \geq 1$.

Example 2.2. Let p_1 and p_2 be given by $(e, [.5, 1])$ and $(\neg e, [.6, 1])$, respectively, and let $KB = \emptyset$. Then, $KB \cup \{p_1, p_2\}$ is not satisfiable, since $KB \models e \wedge \neg e \rightarrow \bot$ and $0.5 + 0.6 > 1$ (intuitively, the lower bound 0.6 for the probability of $\neg e$ implies the upper bound $0.4 < 0.5$ for the probability of e).

We next define the notion of tight logical consequence for probabilistic pairs. A probabilistic pair $(e, [l, u])$ is a *tight logical consequence* of a satisfiable set of formulas $\mathcal{F}$, denoted $\mathcal{F} \models_{tight} (e, [l, u])$, iff l and u are the infimum and supremum, respectively, of $Pr(e)$ subject to all models Pr of $\mathcal{F}$.

We are now ready to define probabilistic conjunctions, disjunctions, and differences of probabilistic pairs. Let $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_2, u_2])$ be two probabilistic pairs and let $KB \subseteq KB^\star(e_1, e_2)$ such that $KB \cup \{p_1, p_2\}$ is satisfiable. The *probabilistic conjunction*, *disjunction*, and *difference* of p_1 and p_2 under KB, denoted $p_1 \otimes_{KB} p_2$, $p_1 \oplus_{KB} p_2$, and $p_1 \ominus_{KB} p_2$, respectively, are defined as the probabilistic pairs $(e_1 \wedge e_2, [l, u])$, $(e_1 \vee e_2, [l, u])$, and $(e_1 \wedge \neg e_2, [l, u])$, respectively, where $[l, u]$ such that $(e_1 \wedge e_2, [l, u])$, $(e_1 \vee e_2, [l, u])$, and $(e_1 \wedge \neg e_2, [l, u])$, respectively, are tight logical consequences of $KB \cup \{p_1, p_2\}$ (note that the structure of KB ensures that both l and u are rational).

Informally, to compute the probability range $[l, u]$ of $e_1 \wedge e_2$, $e_1 \vee e_2$, or $e_1 \wedge \neg e_2$ from the probability ranges $[l_1, u_1]$ and $[l_2, u_2]$ of e_1 and e_2, respectively, we first collect all available dependence information $KB \subseteq KB^\star(e_1, e_2)$ on e_1 and e_2. We then check whether $KB \cup \{(e_1, [l_1, u_1]), (e_2, [l_2, u_2])\}$ is satisfiable. If this is the case, it just remains to compute the unique tight logical consequences $(e_1 \wedge e_2, [l, u])$, $(e_1 \vee e_2, [l, u])$, or $(e_1 \wedge \neg e_2, [l, u])$ of $KB \cup \{(e_1, [l_1, u_1]), (e_2, [l_2, u_2])\}$. Both the satisfiability check and the computation of the tight logical consequence can *effectively* be done by using linear and nonlinear programming techniques.

We next introduce the notions of probabilistic conjunction, disjunction, and difference *strategies*. Let us first give some motivating background.

In query expressions of our algebra, each occurrence of a probabilistic combination operator stands for the probabilistic combination of several pairs of probabilistic pairs. Moreover, each such occurrence will be parameterized with a dependence information $KB_{st} \subseteq KB^\star(a, b)$, where a and b are two new distinct basic events. For example, $r_1 \cup^{\oplus_{in}} r_1$ will denote the union of two relations r_1 and r_2 under the dependence information $KB_{in} = \{Ind(a, b)\}$. This dependence information KB_{st} can now be used in the following ways:

DYNAMIC DEPENDENCE: Given two probabilistic pairs $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_1, u_2])$, we compute $p_1 \otimes_{KB} p_2$, $p_1 \oplus_{KB} p_2$, and $p_1 \ominus_{KB} p_2$, where KB is given by $KB_{st} \cup \{e_1 \rightarrow a, a \rightarrow e_1, e_2 \rightarrow b, b \rightarrow e_2\}$.

STATIC DEPENDENCE: For any two probabilistic pairs $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_1, u_2])$, we compute $p'_1 \otimes_{KB_{st}} p'_2$, $p'_1 \oplus_{KB_{st}} p'_2$, and $p'_1 \ominus_{KB_{st}} p'_2$, where $p'_1 = (a, [l_1, u_1])$ and $p'_2 = (b, [l_1, u_2])$, instead of $p_1 \otimes_{KB} p_2$, $p_1 \oplus_{KB} p_2$, and $p_1 \ominus_{KB} p_2$, respectively (that is, we ignore the concrete events e_1 and e_2).

DYNAMIC DEPENDENCE has nice properties from the semantic point of view, since it exploits all the (locally) available dependence information. STATIC DEPENDENCE, in contrast, implies nice computational properties. Firstly, the dependence information KB_{st} is in itself complete (that is, we do not have to compute any other relationships that are implicitly encoded in the structure of some concrete events e_1 and e_2). Secondly, some specific dependence information KB_{st} often implies nice algebraic properties of $\otimes_{KB_{st}}$, $\oplus_{KB_{st}}$, and $\ominus_{KB_{st}}$, which are known in advance, and can thus be exploited for query optimization.

In the sequel, we implicitly assume STATIC DEPENDENCE (which can be seen as one of the main motivations behind probabilistic combination *strategies*).

Let $KB \subseteq KB^\star(a, b)$ be a dependence information on two distinct basic events a and b. The ***probabilistic conjunction*** (resp., *disjunction*, *difference*) ***strategy*** for KB is the unique function $\otimes$ (resp., $\oplus$, $\ominus$) that associates any two probabilistic pairs $(e_1, [l_1, u_1])$ and $(e_2, [l_2, u_2])$, where $KB \cup \{(a, [l_1, u_1]), (b, [l_2, u_2])\}$ is satisfiable, with the probabilistic pair $(a, [l_1, u_1]) \otimes_{KB} (b, [l_2, u_2])$ (resp., $(a, [l_1, u_1]) \oplus_{KB} (b, [l_2, u_2])$, $(a, [l_1, u_1]) \ominus_{KB} (b, [l_2, u_2])$).

Example 2.3. Let $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_2, u_2])$ be two probabilistic pairs. The probabilistic conjunction, disjunction, and difference strategies for KB among $\emptyset$, $\{Ind(a, b)\}$, $\{PC(a, b)\}$, $\{a \rightarrow b\}$, and $\{a \wedge b \rightarrow \bot\}$ (we refer to these cases by *ignorance*, *independence*, *positive correlation*, *left implication*, and *mutual exclusion*, respectively) are given in Table 1. Note that the probabilistic conjunction and disjunction strategies for ignorance, independence, positive correlation, and mutual exclusion are already known from [12].

Table 1. Examples of probabilistic combination strategies

ignorance	$p_1 \otimes_{ig} p_2 = (e_1 \wedge e_2, [\max(0, l_1 + l_2 - 1), \min(u_1, u_2)])$ $p_1 \oplus_{ig} p_2 = (e_1 \vee e_2, [\max(l_1, l_2), \min(1, u_1 + u_2)])$ $p_1 \ominus_{ig} p_2 = (e_1 \wedge \neg e_2, [\max(0, l_1 - u_2), \min(u_1, 1 - l_2)])$
independence	$p_1 \otimes_{in} p_2 = (e_1 \wedge e_2, [l_1 \cdot l_2, u_1 \cdot u_2])$ $p_1 \oplus_{in} p_2 = (e_1 \vee e_2, [l_1 + l_2 - l_1 \cdot l_2, u_1 + u_2 - u_1 \cdot u_2])$ $p_1 \ominus_{in} p_2 = (e_1 \wedge \neg e_2, [l_1 \cdot (1 - u_2), u_1 \cdot (1 - l_2)])$
positive correlation	$p_1 \otimes_{pc} p_2 = (e_1 \wedge e_2, [\min(l_1, l_2), \min(u_1, u_2)])$ $p_1 \oplus_{pc} p_2 = (e_1 \vee e_2, [\max(l_1, l_2), \max(u_1, u_2)])$ $p_1 \ominus_{pc} p_2 = (e_1 \wedge \neg e_2, [\max(0, l_1 - u_2), \max(0, u_1 - l_2)])$
left implication	$p_1 \otimes_{li} p_2 = (e_1 \wedge e_2, [l_2, \min(u_1, u_2)])$ $p_1 \oplus_{li} p_2 = (e_1 \vee e_2, [\max(l_1, l_2), u_1])$ $p_1 \ominus_{li} p_2 = (e_1 \wedge \neg e_2, [\max(0, l_1 - u_2), u_1 - l_2])$
mutual exclusion	$p_1 \otimes_{me} p_2 = (e_1 \wedge e_2, [0, 0])$ $p_1 \oplus_{me} p_2 = (e_1 \vee e_2, [\min(1, l_1 + l_2), \min(1, u_1 + u_2)])$ $p_1 \ominus_{me} p_2 = (e_1 \wedge \neg e_2, [l_1, \min(u_1, 1 - l_2)])$

We next focus on the properties of probabilistic combination strategies. Let us first introduce some necessary notations.

For probabilistic pairs $p = (e, [l, u])$, we write $\iota(p)$ to denote $[l, u]$. For intervals $[r_1, s_1], [r_2, s_2] \subseteq [0, 1]$, we write $[r_1, s_1] \leq [r_2, s_2]$ to denote $r_1 \leq r_2$ and $s_1 \leq s_2$. For probabilistic pairs p_1 and p_2, we write $p_1 \subseteq p_2$, $p_1 \equiv p_2$, $p_1 \leq p_2$, and $p_1 \geq p_2$ to denote $\iota(p_1) \subseteq \iota(p_2)$, $\iota(p_1) = \iota(p_2)$, $\iota(p_1) \leq \iota(p_2)$, and $\iota(p_1) \geq \iota(p_2)$, respectively.

Let $KB \subseteq KB^\star(a, b)$ with two distinct basic events a and b. The probabilistic conjunction (resp., disjunction) strategy $\otimes$ (resp., $\oplus$) for KB is *commutative* iff $p_1 \otimes p_2 \equiv p_2 \otimes p_1$ (resp., $p_1 \oplus p_2 \equiv p_2 \oplus p_1$) for all probabilistic pairs p_1 and p_2. It is *associative* iff $(p_1 \otimes p_2) \otimes p_3 \equiv p_1 \otimes (p_2 \otimes p_3)$ (resp., $(p_1 \oplus p_2) \oplus p_3 \equiv p_1 \oplus (p_2 \oplus p_3)$) for all probabilistic pairs p_1, p_2, and p_3. We say that $\otimes$ is *distributive* over $\oplus$ iff $p_1 \otimes (p_2 \oplus p_3) = (p_1 \otimes p_2) \oplus (p_1 \otimes p_3)$ for all probabilistic pairs p_1, p_2, and p_3.

For associative probabilistic disjunction strategies $\oplus$ and probabilistic pairs $p_1, p_2, \ldots, p_k$ with $k \geq 1$, we write $\bigoplus_{i \in [1:k]} p_i$ to denote $p_1 \oplus p_2 \oplus \cdots \oplus p_k$.

Example 2.4. It can easily be verified that the probabilistic conjunction and disjunction strategies for ignorance, independence, positive correlation, and mutual exclusion are all commutative and associative. Moreover, for positive correlation, the conjunction strategy is distributive over the disjunction strategy.

Rather than defining probabilistic conjunction and disjunction strategies by a rigorous foundation on probability theory, the work in [12] gives a characterization by the postulates Bottomline, Ignorance, Identity, Annihilator, Commutativity, Associativity, and Monotonicity (see [12] for details).

Indeed, all probabilistic conjunction and disjunction strategies satisfy Ignorance, (a slight variant of) Identity, and Annihilator.

Lemma 2.5. *Let $KB \subseteq KB^\star(a, b)$ with two distinct basic events a and b. Let $\otimes$, $\oplus$, and $\ominus$ be the probabilistic combination strategies for KB. Then, the conditions shown in Table 2 hold for all probabilistic pairs $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_2, u_2])$ such that $KB \cup \{(a, [l_1, u_1]), (b, [l_2, u_2])\}$ is satisfiable.*

Table 2. Properties of probabilistic combination strategies

Ignorance	$p_1 \otimes p_2 \subseteq p_1 \otimes_{ig} p_2$
	$p_1 \oplus p_2 \subseteq p_1 \oplus_{ig} p_2$
	$p_1 \ominus p_2 \subseteq p_1 \ominus_{ig} p_2$
Identity	$p_1 \otimes p_2 \equiv p_2$, if $[l_1, u_1] = [1, 1]$, $a \wedge b \rightarrow \bot \notin KB$, and $a \rightarrow b \notin KB$
	$p_1 \otimes p_2 \equiv p_1$, if $[l_2, u_2] = [1, 1]$, $a \wedge b \rightarrow \bot \notin KB$, and $b \rightarrow a \notin KB$
	$p_1 \oplus p_2 \equiv p_2$, if $[l_1, u_1] = [0, 0]$, $\top \rightarrow a \vee b \notin KB$, and $b \rightarrow a \notin KB$
	$p_1 \oplus p_2 \equiv p_1$, if $[l_2, u_2] = [0, 0]$, $\top \rightarrow a \vee b \notin KB$, and $a \rightarrow b \notin KB$
	$p_1 \ominus p_2 \equiv p_1$, if $[l_2, u_2] = [0, 0]$, $\top \rightarrow a \vee b \notin KB$, and $a \rightarrow b \notin KB$
Annihilator	$p_1 \otimes p_2 \equiv \bot$, if $[l_1, u_1] = [0, 0]$ or $[l_2, u_2] = [0, 0]$
	$p_1 \oplus p_2 \equiv \top$, if $[l_1, u_1] = [1, 1]$ or $[l_2, u_2] = [1, 1]$
	$p_1 \ominus p_2 \equiv \bot$, if $[l_1, u_1] = [0, 0]$ or $[l_2, u_2] = [1, 1]$

However, there are probabilistic conjunction and disjunction strategies $\otimes$ and $\oplus$ that do *not* satisfy the postulate BOTTOMLINE, which says that $p_1 \otimes p_2 \leq (e_1 \wedge e_2, [\min(l_1, l_2), \min(u_1, u_2)])$ and $p_1 \oplus p_2 \geq (e_1 \vee e_2, [\max(l_1, l_2), \max(u_1, u_2)])$, respectively, for all probabilistic pairs $p_1 = (e_1, [l_1, u_1])$ and $p_2 = (e_2, [l_2, u_2])$ such that $p_1 \otimes p_2$ and $p_1 \oplus p_2$, respectively, are defined.

Example 2.6. We show that the probabilistic conjunction and disjunction strategies for left implication do not satisfy BOTTOMLINE. Let the two probabilistic pairs p_1 and p_2 be given by $(e_1, [.2, .7])$ and $(e_2, [.5, .8])$, respectively. It can easily be verified that $\{b \rightarrow a, (a, [.2, .7]), (b, [.5, .8])\}$ is satisfiable. Moreover, we get:

$$\begin{aligned} p_1 \otimes_{li} p_2 &= (e_1 \wedge e_2, [.5, .7]), \text{ but } [\min(l_1, l_2), \min(u_1, u_2)] = [.2, .7]\,, \\ p_1 \oplus_{li} p_2 &= (e_1 \vee e_2, [.5, .7]), \text{ but } [\max(l_1, l_2), \max(u_1, u_2)] = [.5, .8]\,. \end{aligned}$$

Furthermore, there are probabilistic conjunction and disjunction strategies that do *not* satisfy COMMUTATIVITY (that is, that are *not* commutative).

Example 2.7. We show that the probabilistic conjunction and disjunction strategies for left implication do not satisfy COMMUTATIVITY. Let the two probabilistic pairs p_1 and p_2 be given by $(e_1, [.2, .7])$ and $(e_2, [.5, .6])$, respectively. It can easily be verified that $\{b \rightarrow a, (a, [.2, .7]), (b, [.5, .6])\}$ is satisfiable. We get:

$$\begin{aligned} p_1 \otimes_{li} p_2 = (e_1 \wedge e_2, [.5, .6]) \not\equiv (e_2 \wedge e_1, [.2, .6]) = p_2 \otimes_{li} p_1\,, \\ p_1 \oplus_{li} p_2 = (e_1 \wedge e_2, [.5, .7]) \not\equiv (e_2 \wedge e_1, [.5, .6]) = p_2 \oplus_{li} p_1\,. \end{aligned}$$

Finally, we give an example in which our rigorous model-theoretic approach yields more precise intervals than the axiomatic approach in [12].

Example 2.8. Let us consider the case of left implication. Let the two probabilistic pairs p_1 and p_2 be given by $(e_1, [.2, .7])$ and $(e_2, [.5, .6])$, respectively. It can easily be verified that $\{b \rightarrow a, (a, [.2, .7]), (b, [.5, .6])\}$ is satisfiable. We get:

$$\begin{aligned} p_1 \otimes_{li} p_2 &= (e_1 \wedge e_2, [.5, .6])\,, \\ p_1 \oplus_{li} p_2 &= (e_1 \vee e_2, [.5, .7])\,, \\ p_1 \ominus_{li} p_2 &= (e_1 \wedge \neg e_2, [0, .2])\,. \end{aligned}$$

The approach in [12], in contrast, just produces the probability ranges $[.2, .6]$, $[.5, .7]$, and $[0, .5]$, respectively, which are obtained from p_1 and p_2 by the computations $p_1 \otimes_{pc} p_2 = (e_1 \wedge e_2, [.2, .6])$, $p_1 \oplus_{pc} p_2 = (e_1 \vee e_2, [.5, .7])$, and $p_1 \ominus p_2 = (e_1, [.2, .7]) \otimes_{ig} (\neg e_2, [.4, .5]) = (e_1 \wedge \neg e_2, [0, .5])$, respectively.

3 Data Model

In this section, we define probabilistic complex value relations.

Let $\mathcal{A}$ be a nonempty set of *attribute names* and let $\mathcal{T}$ be a nonempty set of *atomic types*. We define *complex value types* (or simply *types*) by induction as follows. Every atomic type from $\mathcal{T}$ is a type. If T is a type, then the set $\{T\}$

is a type. If $A_1, \ldots, A_k$ with $k \geq 1$ are distinct attribute names from $\mathcal{A}$ and $T_1, \ldots, T_k$ are types, then the mapping $T = \{(A_1, T_1), \ldots, (A_k, T_k)\}$ is a type (called *tuple type*). It is abbreviated by $[A_1 \colon T_1, \ldots, A_k \colon T_k]$. We call $A_1, \ldots, A_k$ the *top-level attribute names* of T. We use $T.A_i$ to denote T_i.

Every atomic type $T \in \mathcal{T}$ is assigned a *domain* $\mathrm{dom}(T)$. We define *complex values* by induction as follows. For all atomic types $T \in \mathcal{T}$, every $v \in \mathrm{dom}(T)$ is a complex value of type T. If $v_1, \ldots, v_k$ with $k \geq 0$ are complex values of type T, then the set $\{v_1, \ldots, v_k\}$ is a complex value of type $\{T\}$. If $A_1, \ldots, A_k$ with $k \geq 1$ are distinct attribute names from $\mathcal{A}$ and $v_1, \ldots, v_k$ are complex values of types $T_1, \ldots, T_k$, then the mapping $\{(A_1, v_1), \ldots, (A_k, v_k)\}$ is a complex value of type $[A_1 \colon T_1, \ldots, A_k \colon T_k]$. It is abbreviated by $[A_1 \colon v_1, \ldots, A_k \colon v_k]$.

Given a tuple type T, a *complex value tuple* (*cv-tuple*) of type T is simply a complex value of type T. For cv-tuples $t = [A_1 \colon v_1, \ldots, A_k \colon v_k]$, we use $v.A_i$ to denote v_i. A *complex value relation* (*cv-relation*) of type T is a finite set of cv-tuples of type T.

A *probabilistic complex value tuple* (*pcv-tuple*) of type T is a mapping t of the kind $\{(\mathsf{data}, v), (\mathsf{lb}, r), (\mathsf{ub}, s), (\mathsf{path}, e)\}$, where v is a cv-tuple of type T, r and s are rational numbers with $0 \leq r \leq s \leq 1$, and e is an event. It is abbreviated by $[\mathsf{data} \colon v,\ \mathsf{lb} \colon r,\ \mathsf{ub} \colon s,\ \mathsf{path} \colon e]$. We use $t.\mathsf{data}$, $t.\mathsf{lb}$, $t.\mathsf{ub}$, $t.\mathsf{path}$, and $t.\mathsf{prob}$ to denote v, r, s, e, and $([r, s], e)$, respectively. A *probabilistic complex value relation* (*pcv-relation*) of type T is a finite set of pcv-tuples of type T.

A *base pcv-relation* is a pcv-relation r such that $t.\mathsf{path} \in \mathcal{B}$ for all $t \in r$ and that $t_1.\mathsf{path} \neq t_2.\mathsf{path}$ for all $t_1, t_2 \in r$ with $t_1.\mathsf{data} \neq t_2.\mathsf{data}$. A *probabilistic complex value database* is a finite set of base pcv-relations that are defined over pairwise disjoint sets of basic events.

A pcv-relation r is called *compact* iff $t_1.\mathsf{prob} = t_2.\mathsf{prob}$ for all $t_1, t_2 \in r$ with $t_1.\mathsf{data} = t_2.\mathsf{data}$. In the sequel, we identify compact pcv-relations r of type R with pairs (δ, μ), where δ is a cv-relation of type R and μ is a mapping from δ to the set of all probabilistic pairs.

Each non-compact pcv-relation can be transformed into a compact pcv-relation by applying a compaction operation: Given a pcv-relation r and a commutative and associative probabilistic disjunction strategy $\oplus$, the *compaction of r under $\oplus$*, denoted $\kappa^{\oplus}(r)$, is defined as the pcv-relation (δ, μ), where:

- $\delta = \{v \mid \exists\, w \in r \colon w.\mathsf{data} = v\}$,
- $\mu(v) = \bigoplus_{w \in r,\, w.\mathsf{data} = v} w.\mathsf{prob}$ for all $v \in \delta$.

In the sequel, we thus assume that all pcv-relations are compact. Moreover, we assume that $t.\mathsf{prob} \not\equiv \bot$ for all tuples t in a pcv-relation r (that is, we make the closed world assumption that $t.\mathsf{prob} \equiv \bot$ for all $t \notin r$).

Example 3.1. A compact pcv-relation is shown in Table 3. It contains information about chemical experiments in which several substances are checked for a certain ingredient. Each experiment yields information about the substance analyzed ($\mathsf{substance\text{-}id}$), the laboratory ($\mathsf{lab}$), the date ($\mathsf{date}$), the assistants who did the experiment ($\mathsf{assistants}$), and the result (α-result).

The probabilistic information in this pcv-relation can be interpreted as subjective belief of an agent. Each pcv-tuple t then means that the probability that t.data holds in the real world ranges from t.lb to t.ub. For example, the first pcv-tuple t could say that the probability that substance S89 was analyzed in laboratory L17 on 02/17/99 by Jack and Jill with positive α-result lies between 0.7 and 0.8 (this could mean that we are unsure about the whole information in t.data or that we are just unsure about the α-result). Another possible interpretation is that the probability that the positive α-result describes correctly the properties of substance S89 lies between 0.7 and 0.8.

Table 3. A compact pcv-relation

data					lb	ub	path
substance-id	α-arrangement			α-result	0.7	0.8	e_1
	lab	date	assistants				
S89	L17	02/17/99	{Jack, Jill}	+			
substance-id	α-arrangement			α-result	0.6	0.9	e_2
	lab	date	assistants				
S89	L10	02/13/99	{Joe, Jim}	+			
substance-id	α-arrangement			α-result	0.5	0.7	e_3
	lab	date	assistants				
S64	L12	02/17/99	{Janet, Jill}	−			

4 Algebra

In this section, we define an algebra on probabilistic complex value relations.

4.1 Selection, Projection, and Renaming

We first define the selection operation on pcv-relations.

Let x be a *tuple variable*. We define *terms* by induction as follows. A term is a complex value v, the tuple variable x, or an expression $t.A$, where t is a term and A belongs to $\mathcal{A} \cup \{\mathsf{data}, \mathsf{lb}, \mathsf{ub}\}$. We define *selection conditions* by induction as follows. If t_1 and t_2 are terms and θ belongs to $\{=, \leq, \in, \subseteq\}$, then $t_1 \, \theta \, t_2$ is a selection condition (called *atomic selection condition*). If ϕ_1 and ϕ_2 are selection conditions, then $\neg\phi_1$ and $(\phi_1 \wedge \phi_2)$ are selection conditions. We use $(\phi_1 \vee \phi_2)$ to abbreviate $\neg(\neg\phi_1 \wedge \neg\phi_2)$.

The *interpretation* of a term t in a pcv-tuple w, denoted $[t]_w$, is inductively defined by $[v]_w = v$, $[x]_w = w$, and $[t.A]_w = [t]_w.A$ if $[t]_w.A$ is defined.

A selection condition ϕ is *applicable* to a tuple type T iff for all atomic selection conditions $t_1 \, \theta \, t_2$ that occur in ϕ and all pcv-tuples w of type T: $[t_1]_w$, $[t_2]_w$, and $[t_1]_w \, \theta \, [t_2]_w$ are defined. A selection condition ϕ is *probability-free* iff it does not contain the expressions lb and ub.

For selection conditions ϕ that are applicable to T, the *satisfaction* of ϕ in a pcv-tuple w of type T, denoted $w \models \phi$, is inductively defined as follows:

- $w \models t_1 \, \theta \, t_2$ iff $[t_1]_w \, \theta \, [t_2]_w$,
- $w \models \neg\phi$ iff not $w \models \phi$,
- $w \models (\phi \wedge \psi)$ iff $w \models \phi$ and $w \models \psi$.

Let $r = (\delta, \mu)$ be a pcv-relation of type R and let ϕ be a selection condition that is applicable to R. The *selection on r with respect to ϕ*, denoted $\sigma_\phi(r)$, is the pcv-relation (δ', μ') of type R, where:

- $\delta' \; = \; \{v \,|\, \exists\, w \in r \colon w.\mathsf{data} = v, \; w \models \phi\}$,
- $\mu'(v) \; = \; \mu(v)$ for all $v \in \delta'$.

Example 4.1. Let us take the pcv-relation given in Table 3. Let the selection condition ϕ be defined by

$$\phi \; = \; ((\, x.\mathsf{lb} \geq 0.7 \; \vee \; x.\mathsf{data}.\alpha\text{-}\mathsf{result} = -) \; \wedge \\ \mathsf{Jack} \in x.\mathsf{data}.\alpha\text{-}\mathsf{arrangement.assistants}\,)\,.$$

The result of the selection operation with respect to ϕ is shown in Table 4.

Table 4. Result of selection

data					lb	ub	path
substance-id	α-arrangement			α-result			
	lab	date	assistants				
S89	L17	02/17/99	{Jack, Jill}	+	0.7	0.8	e_1

We next concentrate on the projection operation on pcv-relations. We define the *subtype* relationship on all types by induction as follows. If T is an atomic type from $\mathcal{T}$, then T be a subtype of T. If T_1 is a subtype of T_2, then $\{T_1\}$ is a subtype of $\{T_2\}$. If $A_1, \ldots, A_k$ with $k \geq 1$ are distinct attribute names from $\mathcal{A}$ and $T_1, \ldots, T_l, S_1, \ldots, S_k$ with $l \leq k$ are types such that $T_1, \ldots, T_l$ are subtypes of $S_1, \ldots, S_l$, then $[A_1 \colon T_1, \ldots, A_l \colon T_l]$ is a subtype of $[A_1 \colon S_1, \ldots, A_k \colon S_k]$.

Let v be a complex value of type T and let S be a subtype of T. The *projection of v to S*, denoted $\pi_S(v)$, is by induction defined as follows. If S is an atomic type, then $\pi_S(v) = v$. If $v = \{v_1, \ldots, v_k\}$ and $S = \{S'\}$, then $\pi_S(v) = \{\pi_{S'}(v_1), \ldots, \pi_{S'}(v_k)\}$. If $v = [A_1 \colon v_1, \ldots, A_k \colon v_k]$ and $S = [A_1 \colon S_1, \ldots, A_l \colon S_l]$, then $\pi_S(v) = [A_1 \colon \pi_{S_1}(v_1), \ldots, A_l \colon \pi_{S_l}(v_l)]$.

Let r be a cv-relation of type R and let S be a subtype of R. The *projection of r to S*, denoted $\pi_S(r)$, is the cv-relation $\{\pi_S(t) \,|\, t \in r\}$ of type S.

Let $r = (\delta, \mu)$ be a pcv-relation of type R, let S be a subtype of R, and let $\oplus$ be a commutative and associative disjunction strategy. The *projection of r to S under $\oplus$*, denoted $\pi^{\oplus}_S(r)$, is defined as the pcv-relation (δ', μ') of type S, where:

- $\delta' \; = \; \{v \in \pi_S(\delta) \,|\, \bigoplus_{w \in \delta,\, \pi_S(w) = v} \, \mu(w) \not\equiv \bot\}$,
- $\mu'(v) \; = \; \bigoplus_{w \in \delta,\, \pi_S(w) = v} \, \mu(w)$ for all $v \in \delta'$.

Example 4.2. The projection of the pcv-relation from Table 3 to a tuple type over the attribute names substance-id and α-result under $\oplus_{pc}$ is given in Table 5.

Table 5. Result of projection

data		lb	ub	path
substance-id	α-result	0.7	0.9	$e_1 \vee e_2$
S89	+			
substance-id	α-result	0.5	0.7	e_3
S64	−			

We finally define the renaming operation on pcv-relations. Let T be a type and let $\mathbf{A}$ denote the set of all attribute names that occur in T. A *renaming condition* for T is an expression $B_1, \ldots, B_l \leftarrow C_1, \ldots, C_l$, where $B_1, \ldots, B_l$ is a sequence of distinct attribute names from $\mathcal{A}$ and $C_1, \ldots, C_l$ is a sequence of distinct attribute names from $\mathcal{A} - (\mathbf{A} - \{B_1, \ldots, B_l\})$.

Let T be a type and let $N = B_1, \ldots, B_l \leftarrow C_1, \ldots, C_l$ be a renaming condition for T. The *renaming of T with respect to N*, denoted $\rho_N(T)$, is obtained from T by replacing each attribute name B_i with $i \in [1\!:\!l]$ by the new attribute name C_i.

Let v be a complex value of type T and let $N = B_1, \ldots, B_l \leftarrow C_1, \ldots, C_l$ be a renaming condition for T. The *renaming of v with respect to N*, denoted $\rho_N(v)$, is obtained from v by replacing each attribute name B_i with $i \in [1\!:\!l]$ by the new attribute name C_i.

Let r be a cv-relation of type R and let N be a renaming condition for R. The *renaming of r with respect to N*, denoted $\rho_N(r)$, is the cv-relation $\{\rho_N(t) \mid t \in r\}$ of type $\rho_N(R)$. Let $r = (\delta, \mu)$ be a pcv-relation of type R and let N be a renaming condition for R. The *renaming of r with respect to N*, denoted $\rho_N(r)$, is the pcv-relation (δ', μ') of type $\rho_N(R)$, where:

- $\delta' = \rho_N(\delta)$,
- $\mu'(v) = \mu(\rho_N^{-1}(v))$ for all $v \in \delta'$.

4.2 Join and Cartesian Product

We now define the join operation on pcv-relations.

Two tuple types T_1 and T_2 over the sets of top-level attribute names $\mathbf{A}_2$ and $\mathbf{A}_2$, respectively, are *join-compatible* iff $T_1.A = T_2.A$ for all $A \in \mathbf{A}_1 \cap \mathbf{A}_2$. The *join* of two such types T_1 and T_2, denoted $T_1 \bowtie T_2$, is the tuple type T over $\mathbf{A}_1 \cup \mathbf{A}_2$, where $T.A = T_1.A$ if $A \in \mathbf{A}_1$ and $T.A = T_2.A$ if $A \in \mathbf{A}_2$.

Let v_1 and v_2 be two cv-tuples of join-compatible types T_1 and T_2 over $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively, with $v_1.A = v_2.A$ for all $A \in \mathbf{A}_1 \cap \mathbf{A}_2$. The *join* of v_1 and v_2, denoted $v_1 \bowtie v_2$, is defined as the cv-tuple v of type $T_1 \bowtie T_2$, where $v.A = v_1.A$ if $A \in \mathbf{A}_1$ and $v.A = v_2.A$ if $A \in \mathbf{A}_2$.

Let r_1 and r_2 be two cv-relations of join-compatible types R_1 and R_2 over $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively. The *join of r_1 and r_2*, denoted $r_1 \bowtie r_2$, is the cv-relation r of type $R_1 \bowtie R_2$ with $r = \{t_1 \bowtie t_2 \mid t_1 \in r_1, t_2 \in r_2, t_1.A = t_2.A \text{ for all } A \in \mathbf{A}_1 \cap \mathbf{A}_2\}$.

Let $r_1 = (\delta_1, \mu_1)$ and $r_2 = (\delta_2, \mu_2)$ be two pcv-relations of join-compatible types R_1 and R_2, respectively, and let $\otimes$ be a probabilistic conjunction strategy. The *join of* r_1 *and* r_2 *under* $\otimes$, denoted $r_1 \bowtie^{\otimes} r_2$, is defined as the pcv-relation (δ, μ) of type $R_1 \bowtie R_2$, where:

- $\delta = \{v \in \delta_1 \bowtie \delta_2 \mid \mu_1(\pi_{R_1}(v)) \otimes \mu_2(\pi_{R_2}(v)) \not\equiv \bot\}$,
- $\mu(v) = \mu_1(\pi_{R_1}(v)) \otimes \mu_2(\pi_{R_2}(v))$ for all $v \in \delta$.

Example 4.3. The join of the pcv-relations from Tables 5 and 6 under the probabilistic conjunction strategy $\otimes_{in}$ is given in Table 7.

Table 6. Second input pcv-relation of join

data		lb	ub	path
substance-id	β-result	0.8	0.8	e_4
S89	$-$			
substance-id	β-result	0.8	1.0	e_5
S37	$+$			

Table 7. Result of join

data			lb	ub	path
substance-id	α-result	β-result	0.56	0.72	$(e_1 \vee e_2) \wedge e_4$
S89	$+$	$-$			

We next define the Cartesian product as a special case of join. Two tuple types T_1 and T_2 over the sets of top-level attribute names $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively, are *Cartesian-product-compatible* iff $\mathbf{A}_1$ and $\mathbf{A}_2$ are disjoint. The *Cartesian product* of such types T_1 and T_2, denoted $T_1 \times T_2$, is the tuple type $T_1 \bowtie T_2$.

Let r_1 and r_2 be two pcv-relations of Cartesian-product-compatible types R_1 and R_2, respectively, and let $\otimes$ be a probabilistic conjunction strategy. The *Cartesian product of* r_1 *and* r_2 *under* $\otimes$, denoted $r_1 \times^{\otimes} r_2$, is the pcv-relation $r_1 \bowtie^{\otimes} r_2$ of type $R_1 \times R_2$.

4.3 Union, Intersection, and Difference

We now define the union, intersection, and difference operation on pcv-relations.

Let $r_1 = (\delta_1, \mu_1)$ and $r_2 = (\delta_2, \mu_2)$ be two pcv-relations of the same type R. Let $\otimes$ / $\oplus$ / $\ominus$ be a probabilistic conjunction/disjunction/difference strategy.

The *union of* r_1 *and* r_2 *under* $\oplus$, denoted $r_1 \cup^{\oplus} r_2$, is the pcv-relation (δ, μ) of type R, where:

- $\delta = \{v \in \delta_1 - \delta_2 \mid \mu_1(v) \oplus \bot \not\equiv \bot\} \cup \{v \in \delta_2 - \delta_1 \mid \bot \oplus \mu_2(v) \not\equiv \bot\} \cup$
  $\{v \in \delta_1 \cap \delta_2 \mid \mu_1(v) \oplus \mu_2(v) \not\equiv \bot\}$,

- $\mu(v) = \begin{cases} \mu_1(v) \oplus \bot & \text{if } v \in \delta_1 - \delta_2 \\ \bot \oplus \mu_2(v) & \text{if } v \in \delta_2 - \delta_1 \\ \mu_1(v) \oplus \mu_2(v) & \text{if } v \in \delta_1 \cap \delta_2\,. \end{cases}$

Example 4.4. The union of the pcv-relations from Tables 5 and 8 under the probabilistic disjunction strategy $\oplus_{pc}$ is given in Table 9.

Table 8. Second input pcv-relation of union

data		lb	ub	path
substance-id	α-result	0.2	0.3	e_6
S89	$+$			
substance-id	α-result	0.3	0.4	e_7
S37	$-$			

Table 9. Result of union

data		lb	ub	path
substance-id	α-result	0.7	0.9	$e_1 \vee e_2 \vee e_6$
S89	$+$			
substance-id	α-result	0.3	0.4	e_7
S37	$-$			
substance-id	α-result	0.5	0.7	e_3
S64	$-$			

The *intersection of* r_1 *and* r_2 *under* $\otimes$, denoted $r_1 \cap^{\otimes} r_2$, is the pcv-relation (δ, μ) of type R, where:

- $\delta = \{v \in \delta_1 \cap \delta_2 \mid \mu_1(v) \otimes \mu_2(v) \not\equiv \bot\}$,
- $\mu(v) = \mu_1(v) \otimes \mu_2(v)$.

The *difference of* r_1 *and* r_2 *under* $\ominus$, denoted $r_1 -^{\ominus} r_2$, is the pcv-relation (δ, μ) of type R, where:

- $\delta = \{v \in \delta_1 - \delta_2 \mid \mu_1(v) \ominus \bot \not\equiv \bot\} \cup \{v \in \delta_1 \cap \delta_2 \mid \mu_1(v) \ominus \mu_2(v) \not\equiv \bot\}$,
- $\mu(v) = \begin{cases} \mu_1(v) \ominus \bot & \text{if } v \in \delta_1 - \delta_2 \\ \mu_1(v) \ominus \mu_2(v) & \text{if } v \in \delta_1 \cap \delta_2\,. \end{cases}$

4.4 Relationship to Classical Relational Algebra

It turns out that pcv-relations and the algebra on pcv-relations properly extend classical relations and the classical relational algebra.

Obviously, each tuple t in a classical relation r can be expressed by the pcv-tuple $\varepsilon(t) = [\mathsf{data}\colon t,\ \mathsf{lb}\colon 1,\ \mathsf{ub}\colon 1,\ \mathsf{path}\colon e_t]$ with $e_t \in \mathcal{B}$ in an appropriate pcv-relation $\varepsilon(r)$. Hence, it now remains to show that the results of our algebraic operations on such pcv-relations corresponds to the results of the classical algebraic operations on the original classical relations.

Note first that each event $t.\mathsf{path}$ in a pcv-tuple t with $[t.\mathsf{lb}, t.\mathsf{ub}] = [1, 1]$ is logically equivalent to $\top$. Moreover, we have the following result.

Lemma 4.5. *Let $\otimes$, $\oplus$, and $\ominus$ be the probabilistic combination strategies for any case among ignorance, independence, and positive correlation. Then, the result of applying $\otimes$, $\oplus$, and $\ominus$ to the probabilistic pairs $\bot$ and $\top$ is given in Fig. 1.*

$\otimes$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$

$\oplus$	$\bot$	$\top$
$\bot$	$\bot$	$\top$
$\top$	$\top$	$\top$

$\ominus$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$
$\top$	$\top$	$\bot$

Fig. 1. Probabilistic combinations of $\bot$ and $\top$

This result easily shows that under the probabilistic combination strategies for the cases ignorance, independence, and positive correlation, our selection, projection, renaming, join, Cartesian product, union, intersection, and difference on the pcv-relations $\varepsilon(r)$ correspond to the classical selection, projection, renaming, join, Cartesian product, union, intersection, and difference, respectively, on the original classical relations r.

4.5 Computational Complexity

We now show that under certain assumptions, the introduced operations on pcv-relations can be done in polynomial time in the size of the input relations. The results of this section are implicit in [18].

Let $KB \subseteq KB^\star(a, b)$ be a dependence information on two distinct basic events a and b. The satisfiability check for KB is *polynomial-time-computable* iff the satisfiability of $KB \cup \{(a, [l_1, u_1]), (b, [l_2, u_2])\}$ can be decided in polynomial time in the input size of l_1, u_1, l_2, u_2. The probabilistic conjunction (resp., disjunction, difference) strategy for KB is *polynomial-time-computable* iff $(a, [l_1, u_1]) \otimes_{KB} (b, [l_2, u_2])$ (resp., $(a, [l_1, u_1]) \oplus_{KB} (b, [l_2, u_2])$, $(a, [l_1, u_1]) \ominus_{KB} (b, [l_2, u_2])$) can be computed in polynomial time in the input size of l_1, u_1, l_2, u_2.

Example 4.6. The satisfiability checks and the probabilistic combination strategies for the cases ignorance, independence, positive correlation, left implication, and mutual exclusion are all polynomial-time-computable.

In the sequel, we implicitly assume that all involved satisfiability checks and probabilistic combination strategies are polynomial-time-computable. Our first result shows that then all unary operations can be done in polynomial time.

Theorem 4.7. *a) Given a not necessarily compact pcv-relation r of type R, and a commutative and associative probabilistic disjunction strategy $\oplus$, the compaction $\kappa^{\oplus}(r)$ can be computed in polynomial time in the input size of r.*

b) Given a pcv-relation r of type R, a selection condition ϕ applicable to R, and a renaming condition N for R, both the selection $\sigma_\phi(r)$ and the renaming $\rho_N(r)$ can be computed in linear time in the input size of r.

c) Given a pcv-relation r of type R, a subtype S of R, and a commutative and associative probabilistic disjunction strategy $\oplus$, the projection $\pi_S^{\oplus}(r)$ can be computed in polynomial time in the input size of r.

The next result deals with the binary operations on pcv-relations.

Theorem 4.8. *a) Given two pcv-relations r_1 and r_2 of join-compatible (resp., Cartesian-product-compatible) types R_1 and R_2, and a probabilistic conjunction strategy $\otimes$, the join $r_1 \bowtie^{\otimes} r_2$ (resp., Cartesian product $r_1 \times^{\otimes} r_2$) can be computed in polynomial time in the input size of r_1 and r_2.*

b) Given two pcv-relations r_1 and r_2 of the same type R, and a probabilistic conjunction (resp., disjunction, difference) strategy $\otimes$ (resp., $\oplus$, $\ominus$), the intersection $r_1 \cap^{\otimes} r_2$ (resp., union $r_1 \cup^{\oplus} r_2$, difference $r_1 -^{\ominus} r_2$) can be computed in polynomial time in the input size of r_1 and r_2.

Note that the same results also hold for the generalization of our algebraic operations to Dynamic Dependence, if we impose further restrictions on the events in the input relations. For example, if we assume that all events are conjunctions of basic events, or that all events are disjunctions of basic events, or that all events are defined on pairwise disjoint sets of basic events (in these cases, for any $KB \subseteq KB^{\star}(a, b)$, the concrete dependence information $\{F \in KB^{\star}(a,b) \mid KB \cup \{e_1 \rightarrow a, a \rightarrow e_1, e_2 \rightarrow b, b \rightarrow e_2\} \models F\}$ can be computed in polynomial time in the input size of a and b).

5 Query Equivalences

We now concentrate on query equivalences. We implicitly assume that all involved conjunction and disjunction strategies are commutative and associative.

Our first result shows that the join operation can be reduced to renaming, Cartesian product, selection, and projection like in classical relational algebra.

Theorem 5.1. *Let r_1 and r_2 be pcv-relations of join-compatible types R_1 and R_2, respectively. Let $\otimes$ and $\oplus$ be a probabilistic conjunction and disjunction strategy. Let $\mathbf{A}_1$ and $\mathbf{A}_2$ denote the sets of top-level attribute names of R_1 and R_2, respectively. Let ρ_N replace each $A \in \mathbf{A}_1 \cap \mathbf{A}_2$ by the new attribute name A'. Let ϕ be the conjunction of all $x.A' = x.A$ with $A \in \mathbf{A}_1 \cap \mathbf{A}_2$. Let $R = R_1 \bowtie R_2$.*

$$r_1 \bowtie^{\otimes} r_2 = \pi_R^{\oplus}(\sigma_{\phi}(\rho_N(r_1) \times^{\otimes} r_2)) \,. \tag{1}$$

We next show that most other query equivalences of the classical relational algebra also carry over to our algebra on pcv-relations. The following theorem shows that selections with respect to conjunctive selection conditions can be decomposed, and that sequences of selections can be reordered.

Theorem 5.2. *Let r be a pcv-relation of type R. Let ϕ_1 and ϕ_2 be two selection conditions that are applicable to R.*

$$\sigma_{\phi_1 \wedge \phi_2}(r) = \sigma_{\phi_1}(\sigma_{\phi_2}(r)) \tag{2}$$

$$\sigma_{\phi_1}(\sigma_{\phi_2}(r)) = \sigma_{\phi_2}(\sigma_{\phi_1}(r)) \,. \tag{3}$$

The next result shows that sequences of projections can be reordered and that certain selections can be pushed through projections.

Theorem 5.3. *Let r be a pcv-relation of type R. Let T be a subtype of R and let S be a subtype of T. Let ϕ be a probability-free selection condition applicable to T. Let $\oplus$ be a probabilistic disjunction strategy.*

$$\pi_S^{\oplus}(\pi_T^{\oplus}(r)) = \pi_S^{\oplus}(r) \tag{4}$$

$$\pi_T^{\oplus}(\sigma_\phi(r)) = \sigma_\phi(\pi_T^{\oplus}(r))\,. \tag{5}$$

The following theorem shows that selections and projections can be pushed through renaming operations.

Theorem 5.4. *Let r be a pcv-relation of type R. Let N be a renaming condition for R. Let ϕ be a selection condition applicable to $\rho_N(R)$ and let S be a subtype of $\rho_N(R)$. Let ϕ' and S' be obtained from ϕ and S, respectively, by performing the renaming ρ_N^{-1}. Let $\oplus$ be a probabilistic disjunction strategy.*

$$\sigma_\phi(\rho_N(r)) = \rho_N(\sigma_{\phi'}(r)) \tag{6}$$

$$\pi_S^{\oplus}(\rho_N(r)) = \rho_N(\pi_{S'}^{\oplus}(r))\,. \tag{7}$$

The next theorem shows that joins are commutative and associative, and that certain selections and projections can be pushed through join operations.

Theorem 5.5. *Let r_1, r_2, and r_3 be pcv-relations of the pairwise join-compatible types R_1, R_2, and R_3, respectively. Let $\otimes$ and $\oplus$ be conjunction and disjunction strategies such that $\otimes$ is distributive over $\oplus$. Let ϕ_1 and ϕ_2 be probability-free selection conditions that are applicable to R_1 and R_2, respectively. Let S be a subtype of $R_1 \bowtie R_2$. Let $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}$ denote the sets of top-level attribute names of R_1, R_2, and S, respectively. Let the tuple type S_1 over $(\mathbf{A} \cup \mathbf{A}_2) \cap \mathbf{A}_1$ be defined by $S_1.A = S.A$ for all $A \in (\mathbf{A} - \mathbf{A}_2) \cap \mathbf{A}_1$ and $S_1.A = R_1.A$ for all $A \in \mathbf{A}_1 \cap \mathbf{A}_2$, and let the tuple type S_2 over $(\mathbf{A} \cup \mathbf{A}_1) \cap \mathbf{A}_2$ be defined by $S_2.A = S.A$ for all $A \in (\mathbf{A} - \mathbf{A}_1) \cap \mathbf{A}_2$ and $S_2.A = R_2.A$ for all $A \in \mathbf{A}_1 \cap \mathbf{A}_2$.*

$$r_1 \bowtie^{\otimes} r_2 = r_2 \bowtie^{\otimes} r_1 \tag{8}$$

$$(r_1 \bowtie^{\otimes} r_2) \bowtie^{\otimes} r_3 = r_1 \bowtie^{\otimes} (r_2 \bowtie^{\otimes} r_3) \tag{9}$$

$$\sigma_{\phi_1 \wedge \phi_2}(r_1 \bowtie^{\otimes} r_2) = \sigma_{\phi_1}(r_1) \bowtie^{\otimes} \sigma_{\phi_2}(r_2) \tag{10}$$

$$\pi_S^{\oplus}(r_1 \bowtie^{\otimes} r_2) = \pi_S^{\oplus}(\pi_{S_1}^{\oplus}(r_1) \bowtie^{\otimes} \pi_{S_2}^{\oplus}(r_2))\,. \tag{11}$$

As Cartesian product is a special case of join, we get the following corollary.

Corollary 5.6. *Let r_1, r_2, and r_3 be pcv-relations of the pairwise Cartesian-product-compatible types R_1, R_2, and R_3, respectively. Let $\otimes$ and $\oplus$ be conjunction and disjunction strategies such that $\otimes$ is distributive over $\oplus$. Let ϕ_1 and ϕ_2 be probability-free selection conditions that are applicable to R_1 and R_2, respectively. Let S be a subtype of $R_1 \times R_2$. Let $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}$ denote the sets of top-level attribute names of R_1, R_2, and S, respectively. Let the tuple type S_1*

over $\mathbf{A} \cap \mathbf{A}_1$ *be defined by* $S_1.A = S.A$ *for all* $A \in \mathbf{A} \cap \mathbf{A}_1$*, and let the tuple type* S_2 *over* $\mathbf{A} \cap \mathbf{A}_2$ *be defined by* $S_2.A = S.A$ *for all* $A \in \mathbf{A} \cap \mathbf{A}_2$*.*

$$r_1 \times^{\otimes} r_2 = r_2 \times^{\otimes} r_1 \tag{12}$$

$$(r_1 \times^{\otimes} r_2) \times^{\otimes} r_3 = r_1 \times^{\otimes} (r_2 \times^{\otimes} r_3) \tag{13}$$

$$\sigma_{\phi_1 \wedge \phi_2}(r_1 \times^{\otimes} r_2) = \sigma_{\phi_1}(r_1) \times^{\otimes} \sigma_{\phi_2}(r_2) \tag{14}$$

$$\pi^{\oplus}_{S}(r_1 \times^{\otimes} r_2) = \pi^{\oplus}_{S_1}(r_1) \times^{\otimes} \pi^{\oplus}_{S_2}(r_2)\,. \tag{15}$$

The next result finally shows that union and intersection operations are commutative and associative, and that certain selections can be pushed through union, intersection, and difference. Moreover, we show that projections can be pushed through union, and that intersection is a special case of join.

Theorem 5.7. *Let* r_1*,* r_2*, and* r_3 *be pcv-relations of type* R*. Let* $\otimes\,/\oplus\,/\ominus$ *be a probabilistic conjunction/disjunction/difference strategy. Let* ϕ *be a probability-free selection condition that is applicable to* R *and let* S *be a subtype of* R*.*

$$r_1 \cup^{\oplus} r_2 = r_2 \cup^{\oplus} r_1 \tag{16}$$

$$(r_1 \cup^{\oplus} r_2) \cup^{\oplus} r_3 = r_1 \cup^{\oplus} (r_2 \cup^{\oplus} r_3) \tag{17}$$

$$r_1 \cap^{\otimes} r_2 = r_2 \cap^{\otimes} r_1 \tag{18}$$

$$(r_1 \cap^{\otimes} r_2) \cap^{\otimes} r_3 = r_1 \cap^{\otimes} (r_2 \cap^{\otimes} r_3) \tag{19}$$

$$\sigma_{\phi}(r_1 \cup^{\oplus} r_2) = \sigma_{\phi}(r_1) \cup^{\oplus} \sigma_{\phi}(r_2) \tag{20}$$

$$\sigma_{\phi}(r_1 \cap^{\otimes} r_2) = \sigma_{\phi}(r_1) \cap^{\otimes} \sigma_{\phi}(r_2) \tag{21}$$

$$\sigma_{\phi}(r_1 -^{\ominus} r_2) = \sigma_{\phi}(r_1) -^{\ominus} \sigma_{\phi}(r_2) \tag{22}$$

$$\pi^{\oplus}_{S}(r_1 \cup^{\oplus} r_2) = \pi^{\oplus}_{S}(r_1) \cup^{\oplus} \pi^{\oplus}_{S}(r_2) \tag{23}$$

$$r_1 \cap^{\otimes} r_2 = r_1 \bowtie^{\otimes} r_2\,. \tag{24}$$

6 Query Optimization

In this section, we briefly concentrate on query optimization.

We have seen that most of the query equivalences of classical relational algebra also hold for the algebra on pcv-relations. Hence, we can use the same query equivalences like in classical relational algebra to move especially selections but also projections as much inside a given query expression as possible:

1. We can use (2) to break up conjunctive selection conditions.
2. We can use (3), (5), (6), (10), (14), (21), (20), and (22) to move selection operations as much inside the query expression as possible.
3. We can use (9), (13), (19), and (17) to structure the query expression in a better way.
4. We can use (4), (5), (7), (11), (15), and (23) to move projection operations as much inside the query expression as possible.

Example 6.1. Let T be an atomic type. Let $R_1 = [A\colon T, B_1\colon T, C_1\colon T]$, $R_2 = [A\colon T, B_2\colon T, C_2\colon T]$, $R_3 = [A\colon T, B_3\colon T, C_3\colon T]$, $S_1 = [A\colon T, C_1\colon T]$, $S_2 = [A\colon T, C_2\colon T]$, $S_3 = [A\colon T, C_3\colon T]$, and $S_3 = [A\colon T, C_1\colon T, C_2\colon T, C_3\colon T]$ be tuple types. Let r_1, r_2, and r_3 be pcv-relations over R_1, R_2, and R_3, respectively. Let ϕ_1 and ϕ_2 denote the selection conditions $x.\mathsf{data}.C_1 = v$ and $x.\mathsf{data}.C_2 = x.\mathsf{data}.C_3$, respectively, where v is a complex value of type T. Then, the query expression

$$\sigma_{\phi_1 \wedge \phi_2}(\pi_S^{\oplus_{pc}}((r_1 \bowtie^{\otimes_{pc}} r_2) \bowtie^{\otimes_{pc}} r_3))$$

can be transformed into the following equivalent query expression (since the conjunction strategy $\otimes_{pc}$ is distributive over the disjunction strategy $\oplus_{pc}$):

$$\pi_{S_1}^{\oplus_{pc}}(\sigma_{\phi_1}(r_1)) \bowtie^{\otimes_{pc}} \sigma_{\phi_2}(\pi_{S_2}^{\oplus_{pc}}(r_2) \bowtie^{\otimes_{pc}} \pi_{S_3}^{\oplus_{pc}}(r_3))\,.$$

7 Related Work

In this section, we give a brief comparison to related work in the literature.

Our approach generalizes annotated tuples of ProbView [12]. As argued in [12], ProbView generalizes various approaches (like, for example, [2, 4]). Cavallo and Pittarelli [4] view relations in a (flat) relational database as probability distribution functions, where tuples in the same relation are viewed as pairwise disjoint events whose probabilities sum up to 1. Drawbacks of this approach have been pointed out in [5]. An extension of the model using probability intervals, which are viewed as constraints on the probabilities, is reviewed in [15]. Barbará et al. [2] have considered a probabilistic extension to the relational model, in which imprecise attributes are modeled as probability distributions over finite sets of values. Their approach assumes that key attributes are deterministic (have probability 1) and that non-key attributes in different relations, are independent. No probabilities can be assigned to outmost tuples.

Fuhr and Rölleke [7] consider a probabilistic version of NF2 relations, extending their approach for flat tuples [8], and define a relational algebra for this model. Probabilities are assigned to tuples and to values of nested tuples (that is, set-valued attributes), which are viewed as events that have an associated event expression. The algebraic operators manipulate tuples by combining value and event expressions appropriately. The probability of a derived tuple is computed from the probabilities of initial tuples by taking into consideration its event expression. The approach in [7] defines an intensional semantics in which probabilities are defined through possible worlds. The evaluation method assumes that in nondeterministic relations (that is, relations with uncertain tuples), joint occurrence of two different values is either always independent or impossible. Our approach has no such severe restriction (we do not make any independence or mutual exclusion assumptions). Furthermore, [7] does not use probability intervals but a single probability value. On the other hand, the algebra in [7] provides nest and unnest operators, which we have not done here.

We see as major drawbacks of the approach in [7] the above non-dependency assumption on relations, which is rarely met in practice, and that the evaluation of event expressions takes exponential time in general, owing to complicated probability sums which need to be computed—this seems the price to pay for a smooth intensional semantics, though.

The probabilistic database model of Dey and Sarkar [5] assigns each tuple in a (flat) relational database a probability value in a special attribute. The classical relational operations are defined adopting different assumptions on the relationship between tuples; in particular, join assumes independence; union and difference assume positive correlation; and compaction assumes disjointness or positive correlation. Our model is more general, since it has complex values, intervals, and allows different strategies (reflecting different relationships between tuples) to be used in the algebraic operations. Based on [5], a probabilistic extension to SQL is developed in [6].

Zimányi [19] presents an approach to querying probabilistic databases based on probabilistic first-order logic. A probabilistic database is axiomatized as a first-order probabilistic theory, which has a possible worlds semantics as in [10]. Tuples t in relations have a trace formula ϕ attached, which intuitively selects the worlds in which t is true. The classical relational algebra operations are defined using formulas and manipulate trace formulas. A query is evaluated in two steps: the first step yields a relation of tuples with trace formulas, which are then evaluated using the assertions of the probabilistic theory to obtain a probability for each tuple. Compared to our approach, [19] aims at flat databases and considers no probability intervals (but mentions a possible extension). The approach assumes complete independence of occurrences of distinct tuples in the same or different relations. Furthermore, as with most other approaches, no query equivalences are discussed.

8 Summary and Conclusion

In this paper, we generalized the annotated tuple approach in [12] to complex value databases [1]. We presented a probabilistic data model for complex values and defined an algebra on top of it. Moreover, we presented query equivalences and briefly discussed their use for query optimization. It turned out that most of the equivalences of classical relational algebra hold in this generalized model. Hence, many classical query optimization techniques carry over to our model.

Several issues remain for further investigation. One such issue are other operators besides those of classical relation algebra, such as NF2 nest and unnest or the operators in [7, 5, 2, 15]. Moreover, it would be very interesting to more deeply investigate the relationship to purely extensional and purely intensional approaches to probabilistic databases. Finally, an intriguing issue is to extend our approach to object-oriented databases. To our knowledge, few approaches to probabilistic object-oriented databases have been proposed so far (see [11]).

Acknowledgments

We are very grateful to V. S. Subrahmanian for fruitful discussions. We also want to thank the referees for their useful comments.

This work has been partially supported by an NSF-DAAD grant and the Austrian Science Fund Project N Z29-INF.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, Reading, 1995.
2. D. Barbara, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *IEEE Transactions on Knowledge and Data Engineering*, 4(5):387–502, 1992.
3. R. Carnap. *Logical Foundations of Probability.* University of Chicago Press, Chicago, 1950.
4. R. Cavallo and M. Pittarelli. The theory of probabilistic databases. In *Proceedings of the 13th International Conference on Very Large Databases*, pages 71–81. Morgan Kaufmann, 1987.
5. D. Dey and S. Sarkar. A probabilistic relational model and algebra. *ACM Transactions on Database Systems*, 21(3):339–369, 1996.
6. D. Dey and S. Sarkar. PSQL: A query language for probabilistic relational data. *Data & Knowledge Engineering*, 28:107–120, 1998.
7. N. Fuhr and T. Rölleke. A probabilistic NF2 relational algebra for integrated information retrieval and database systems. In *Proceedings of the 2nd World Conference on Integrated Design and Process Technology*, pages 17–30. Society for Design and Process Science, 1996.
8. N. Fuhr and T. Rölleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems*, 15(1):32–66, 1997.
9. H. Gaifman. Concerning measures in first order calculi. *Israel Journal of Mathematics*, 2:1–18, 1964.
10. J. Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46(3):311–350, 1990.
11. Y. Kornatzky and S. E. Shimony. A probabilistic object-oriented data model. *Data & Knowledge Engineering*, 12:143–166, 1994.
12. L. V. S. Lakshmanan, N. Leone, R. Ross, and V. S. Subrahmanian. ProbView: A flexible probabilistic database system. *ACM Transactions on Database Systems*, 22(3):419–469, 1997.
13. T. Lukasiewicz. Local probabilistic deduction from taxonomic and probabilistic knowledge-bases over conjunctive events. *International Journal of Approximate Reasoning*, 21(1):23–61, 1999.
14. T. Lukasiewicz. Probabilistic deduction with conditional constraints over basic events. *Journal of Artificial Intelligence Research*, 10:199–241, 1999.
15. M. Pittarelli. An algebra for probabilistic databases. *IEEE Transactions on Knowledge and Data Engineering*, 6(2):293–303, 1994.
16. H.-J. Schek and P. Pistor. Data structures for an integrated data base management and information retrieval system. In *Proceedings of the 8th International Conference on Very Large Data Bases*, pages 197–207. Morgan Kaufmann, 1982.
17. D. Scott and P. Krauss. Assigning probabilities to logical formulas. In J. Hintikka and P. Suppes, editors, *Aspects of Inductive Logic*, pages 219–264. North-Holland, Amsterdam, 1966.
18. M. Walter. An extension of relational algebra to probabilistic complex values. Master's thesis, Universität Gießen, 1999.
19. E. Zimányi. Query evaluation in probabilistic relational databases. *Theoretical Computer Science*, 171(1–2):179–219, 1997.