Appears in: ILPS'94.

Probabilistic Deductive Databases*

Laks V.S. Lakshmanan[†] and Fereidoon Sadri[‡]

[†] Dept. of Computer Science, Concordia University
Montreal, Canada, *email: laks@cs.concordia.ca*

[‡] Dept. of Mathematics, University of North Carolina at Greensboro
Greensboro, NC, USA, *email: sadri@uncg.edu*

1 Introduction

The Problem: Knowledge-base (KB) systems must typically deal with imperfection in knowledge, *e.g.* in the form of incompleteness, inconsistency, uncertainty, to name a few. Currently KB system development is mainly based on the expert system technology. Expert systems, through their support for rule-based programming, uncertainty, etc. offer a convenient framework for KB system development. But they require the user to be well versed with the low level details of system implementation. The manner in which uncertainty is handled has little mathematical basis. There is no decent notion of query optimization, forcing the user to take the responsibility for an efficient implementation of the KB system. We contend KB system development can and should take advantage of the deductive database technology, which overcomes most of the above limitations. An important problem here is to extend deductive databases (DDBs) into providing a systematic basis for rule-based programming with imperfect knowledge. In this paper, we are interested in an extension capable of handling probabilistic knowledge.

Previous Work: Recent developments in DDBs (and logics in general) have led to frameworks capable of handling various forms of imperfection in knowledge. Abiteboul, et. al. [1], Liu [16], and Dong and Lakshmanan [5] dealt with DDBs with incomplete information in the form of null values. Kifer and Lozinskii [13] have developed a logic for reasoning with inconsistency. Most of the works dealing with uncertainty in knowledge-bases employ one of the following formalisms: (1) a form of fuzzy logic (programming) (*e.g.* van Emden [20], Steger et. al. [19], and Fitting [8]), (2) annotated logic programming (*e.g.* see Kifer and Li [12] and Kifer and Subrahmanian [14]), (3) evidence theoretic logic programming (*e.g.* see Baldwin [2]), and (4) probabilistic logic programming (see below).

Ng and Subrahmanian [17, 18] have recently proposed an interesting scheme for logic programming with uncertainty modeled using probabilities. Syntactically, the framework shares the notation with annotated logic program-

*This research was supported in part by grants from the NSERC (Canada) and FCAR (Quebec), and was performed while the second author was at Concordia University.

ming. In [17] only annotations involving constant probability ranges were allowed. This was extended in [18] by allowing annotation variables and functions. They developed fixpoint and model-theoretic semantics, and provided a sound and weakly complete proof procedure. Güntzer et. al. [11] have proposed a sound (propositional) probabilistic calculus based on conditional probabilities, for reasoning in the presence of incomplete information. Although they make use of a datalog-based interface to implement this calculus, they do not extend DDBs with the ability to handle uncertain facts and rules. Besides these works, a wealth of literature is available on probabilistic logic alone (*e.g.* see Carnap [4]and Fagin, Halpern, and Meggido [6]). As pointed out in [18] it is unclear how these can form a basis for probabilistic logic programming. An interested reader is referred to [18] for a detailed account of these and other works.

Contributions: We associate a *confidence level* with facts and rules (of a deductive database). A confidence level comes with both a *belief* and a *doubt*(in what is being asserted) [see Section 2 for a motivation]. Belief and doubt are subintervals of $[0, 1]$ representing probability ranges. Confidence levels have an interesting algebraic structure called *trilattices* as their basis (Section 3). In addition to providing an algebraic footing for our framework, trilattices also shed light on the relationship between our work and earlier works and offer useful insights. We develop a probabilistic calculus for combining confidence levels associated with basic events into those for compound events (Section 4). Instead of committing to any specific rules for combining confidences, we propose a framework which allows a user to choose an appropriate "mode" from a collection of available ones. We develop a generalized framework for rule-based programming with probabilistic knowledge, based on this calculus. We provide the declarative and fixpoint semantics for such programs and establish their equivalence (Section 5). We also provide a sound and complete proof procedure (Section 6). We study the termination and complexity issues of such programs and show: (1) the closure ordinal of T_P can be as high as ω in general (but no more), and (2) when only *positive correlation* is used for disjunction[1], the data complexity of such programs is polynomial time. We also compare our work with related work and bring out the advantages and generality of our approach (Section 7). Section 8 presents our conclusions and future research directions. Complete details of the framework presented here and the proofs can be found in [15].

2 Motivation

In this section, we discuss the motivation for our work as well as comment on our design decisions for this framework. The motivation for using probability theory as opposed to other formalisms for representing uncertainty has been discussed at length in the literature [4, 17]. Probability theory is perhaps the best understood and mathematically well-founded paradigm in which uncertainty can be modeled and reasoned about. Two possibilities for associating probabilities with facts and rules in a DDB are van Emden's style of associating confidences with rules as a whole [20], or the annotation style of Kifer and Subrahmanian [14].The first approach is more suited for truth-functional derivation of probabilities whereas the second is ideal for subjective derivations of probabilities. (It is shown in [14] that the second approach subsumes the first.) In this paper, our interest is in truth-functional

[1]Other modes can be used (for conjunction/disjunction) in the "non-recursive part" of the program.

derivation of probabilities, and we choose the first option.

A second issue is whether we should insist on precise probabilities or allow intervals (or ranges). Firstly, probabilities derived from any sources may have tolerances associated with them. Even experts may feel more comfortable with specifying a range rather than a precise probability. Secondly, Fenstad [7] has shown (also see [17]) that when enough information is not available about the interaction between events, the probability of compound events cannot be determined precisely: one can only give (tight) bounds. Thus, we associate ranges of probabilities with facts and rules.

A last issue is the following. Suppose (uncertain) knowledge contributed by an expert corresponds to the formula F. In general, we cannot assume the expert's knowledge is perfect. This means he does not necessarily know *all* situations in which F holds. Nor does he know *all* situations where F fails to hold (*i.e.* $\neg F$ holds). He models the proportion of the situations where he knows F holds as his *belief* in F and the proportion of situations where he knows $\neg F$ holds as his *doubt*. There could be situations, unknown to our expert, where F holds (or $\neg F$ holds). These unknown situations correspond to the gap in his knowledge. Thus, as far as he knows, F is *unknown* or *undefined* in these remaining situations.

The following example illustrates such a scenario. (The figures in all our examples are ficticious.) Consider the results of Gallup polls conducted before the recent Canadian federal elections.

1. Of the people surveyed, between 50% and 53% of the people in the age group 19 to 30 favor the liberals.
2. Between 30% and 33% of the people in the above age group favor the reformists.
3. Between 5% and 8% of the above age group favor the tories.

The reason we have ranges for each category is that usually some tolerance is associated with the results coming from such polls. Also, we do not make the proportion of undecided people explicit as our interest is in determining the support for the different parties. Suppose we assimilate the information above in a probabilistic framework. For each party, we compute the probability that a *randomly* chosen person from the sample population of the given age group will (not) vote for that party. We transfer this probability as the *subjective* probability that *any* person from that age group (in the actual population) will (not) vote for the party. The conclusions are given below, where $vote(X, P)$ says X will vote for party P, $age\text{-}group1(X)$ says X belongs to the age group specified above. $liberals$, $reform$, and $tories$ are constants, with the obvious meaning.

1. $vote(X, liberals) \xleftarrow{\langle[\mathbf{0.5},\mathbf{0.53}],[\mathbf{0.35},\mathbf{0.41}]\rangle} age\text{-}group1(X)$.
2. $vote(X, reform)$: $\xleftarrow{\langle[\mathbf{0.3},\mathbf{0.33}],[\mathbf{0.55},\mathbf{0.61}]\rangle} age\text{-}group1(X)$.
3. $vote(X, tories)$: $\xleftarrow{\langle[\mathbf{0.05},\mathbf{0.08}],[\mathbf{0.8},\mathbf{0.86}]\rangle} age\text{-}group1(X)$.

As usual, each rule is implicitly universally quantified outside the entire rule. Each rule is expressed in the form $A \xleftarrow{\langle[\boldsymbol{\alpha},\boldsymbol{\beta}],\ [\boldsymbol{\gamma},\boldsymbol{\delta}]\rangle} Body$, where $\alpha, \beta, \gamma, \delta \in [0, 1]$. We usually require that $\alpha \leq \beta$ and $\gamma \leq \delta$. With each rule, we have associated two intervals. $[\alpha, \beta]$ ($[\gamma, \delta]$) is the *belief* (*doubt*) the expert has in the rule. Notice that from his knowledge, the expert can only conclude that the proportion of people he *knows* favor *reform* or *tories* will not vote for *liberals*. Thus the probability that a person in the age group 19-30 will not vote for liberals, according to the expert's knowledge, is in the range $[0.35, 0.41]$, obtained by summing the endpoints of the belief ranges for reform and tories. Notice that in this case $\alpha + \delta$ (or $\beta + \gamma$) is not necessarily 1. This shows we cannot regard the expert's doubt as the complement (w.r.t. 1) of his belief. Thus, if we have to model what *necessarily* follows according to the expert's knowledge, then we must carry both the belief and the doubt explicitly. Kifer and Li [12] and Baldwin [2] have argued that incorporating both belief and doubt (called disbelief there) is useful in dealing with incomplete knowledge, where different evidences may contradict each other. However, in their frameworks, doubt need not be maintained explicitly. For suppose we have a belief b and a disbelief d associated with a phenomenon. Then they can both be absorbed into one range $[b, 1-d]$ indicating that the effective certainty ranges over this set. The difference with our framework, however, is that we model what is *known* definitely, as opposed to what is *possible*. This makes (in our case) an explicit treatment of belief and doubt mandatory.

3 The Algebra of Confidence Levels

Fitting [9] has shown that *bilattices* (introduced by Ginsburg [10]) lead to an elegant framework for quantified logic programming involving both belief and doubt. In this section, we shall see that a notion of *trilattices* naturally arises with confidence levels. We shall establish the structure and properties of trilattices here, which will be used in later sections.

Definition 3.1 *Denote by $\mathcal{C}[0,1]$ the set of all closed subintervals over $[0,1]$. Consider the set $\mathcal{L}_c =_{def} \mathcal{C}[0,1] \times \mathcal{C}[0,1]$. We denote the elements of $\mathcal{L}_c$ as $\langle [\alpha,\beta],\ [\gamma,\delta] \rangle$. Define the following orders on this set. Let $\langle [\alpha_1,\beta_1],\ [\gamma_1,\delta_1] \rangle$, $\langle [\alpha_2,\beta_2],\ [\gamma_2,\delta_2] \rangle$ be any two elements of $\mathcal{L}_c$.*

$\langle [\alpha_1,\beta_1],\ [\gamma_1,\delta_1] \rangle \leq_t \langle [\alpha_2,\beta_2],\ [\gamma_2,\delta_2] \rangle$ *iff* $\alpha_1 \leq \alpha_2,\ \beta_1 \leq \beta_2$; *and* $\gamma_2 \leq \gamma_1,\ \delta_2 \leq \delta_1$.

$\langle [\alpha_1,\beta_1],\ [\gamma_1,\delta_1] \rangle \leq_k \langle [\alpha_2,\beta_2],\ [\gamma_2,\delta_2] \rangle$ *iff* $\alpha_1 \leq \alpha_2,\ \beta_1 \leq \beta_2$; *and* $\gamma_1 \leq \gamma_2,\ \delta_1 \leq \delta_2$.

$\langle [\alpha_1,\beta_1],\ [\gamma_1,\delta_1] \rangle \leq_p \langle [\alpha_2,\beta_2],\ [\gamma_2,\delta_2] \rangle$ *iff* $\alpha_1 \leq \alpha_2,\ \beta_2 \leq \beta_1$; *and* $\gamma_1 \leq \gamma_2,\ \delta_2 \leq \delta_1$.

Some explanation is in order. The order $\leq_t$ can be considered the *truth* ordering: "truth" relative to the expert's knowledge increases as belief goes up and doubt comes down. The order $\leq_k$ is the *knowledge* (or information) ordering: "knowledge" (*i.e.* the extent to which the expert commits his opinion on an assertion) increases as both belief and doubt increase. The order $\leq_p$ is the *precision* ordering: "precision" of information supplied increases as the probability intervals become narrower. The first two orders are analogues of similar orders in bilattices. The third one, however, has no counterpart there. It is straightforward to see that each of the orders $\leq_t, \leq_k, \leq_p$ is a partial order. $\mathcal{L}_c$ has a least and a greatest element w.r.t. each of these orders. In the following, we give the definition of meet and join w.r.t. the $\leq_t$ order. Operators w.r.t. the other orders have a similar definition.

Definition 3.2 *Let $\langle \mathcal{L}_c, \leq_t, \leq_k, \leq_p \rangle$ be as defined in Definition 3.1. Then the meet and join corresponding to the truth order, denoted $\otimes_t, \oplus_t$, are defined as follows.*

1. $\langle[\alpha_1,\beta_1],\ [\gamma_1,\delta_1]\rangle \otimes_t \langle[\alpha_2,\beta_2],\ [\gamma_2,\delta_2]\rangle = \langle[min\{\alpha_1,\alpha_2\}, min\{\beta_1,\beta_2\}], [max\{\gamma_1,\gamma_2\}, max\{\delta_1,\delta_2\}]\rangle$.
2. $\langle[\alpha_1,\beta_1],\ [\gamma_1,\delta_1]\rangle \oplus_t \langle[\alpha_2,\beta_2],\ [\gamma_2,\delta_2]\rangle = \langle[max\{\alpha_1,\alpha_2\}, max\{\beta_1,\beta_2\}], [min\{\gamma_1,\gamma_2\}, min\{\delta_1,\delta_2\}]\rangle$.

The top and bottom elements w.r.t. the various orders are as follows. The subscripts indicate the associated orders, as usual.
$\top_t = \langle[1,1],[0,0]\rangle$, $\perp_t = \langle[0,0],[1,1]\rangle$,
$\top_k = \langle[1,1],[1,1]\rangle$, $\perp_k = \langle[0,0],[0,0]\rangle$,
$\top_p = \langle[1,0],[1,0]\rangle$, $\perp_p = \langle[0,1],[0,1]\rangle$.

$\top_t$ corresponds to total belief and no doubt; $\perp_t$ is the opposite. $\top_k$ represents maximal information (total belief and doubt), to the point of being probabilistically inconsistent: belief and doubt probabilities sum to more than 1; $\perp_k$ gives the least information: no basis for belief or doubt; $\top_p$ is maximally precise, to the point of making the intervals empty (and hence inconsistent, in a non-probabilistic sense); $\perp_p$ is the least precise, as it imposes only trivial bounds on belief and doubt probabilities.

Fitting [9] defines a bilattice to be *interlaced* whenever the meet and join w.r.t. any order of the bilattice are monotone w.r.t. the other order. He shows that it is the interlaced property of bilattices that makes them most useful and attractive. We say that a trilattice is *interlaced* provided the meet and join w.r.t. any order are monotone w.r.t. any other order. We have

Lemma 3.1 *The trilattice $\langle\mathcal{L}_c, \leq_t, \leq_k, \leq_p\rangle$ defined above is interlaced.*

We say a confidence level $\langle[\alpha,\beta],\ [\gamma,\delta]\rangle$ in a trilattice is *consistent* provided, (i) $\alpha \leq \beta$ and $\gamma \leq \delta$, and (ii) $\alpha + \delta \leq 1$ and $\beta + \gamma \leq 1$. In this paper, we assume we only use consistent confidence levels in a (probabilistic) deductive database. Trilattices are of independent interest in their own right, from an algebraic point of view. We also stress that they can be used as a basis for developing quantified/annotated logic programming schemes (which need not be probabilistic). This will be pursued in a future paper.

4 A Probabilistic Calculus

Given the confidence levels for (basic) events, how are we to derive the confidence levels for compound events which are based on them? Since we are working with probabilities, our combination rules must respect probability theory. We need a model of our knowledge about the interaction between events. A simplistic model studied in the literature (*e.g.* , see Barbara et. al. [3]) assumes *independence* between all pairs of events. This is highly restrictive and is of limited applicability. A general model, studied by Ng and Subrahmanian [17, 18] is that of *ignorance*: assume no knowledge about event interaction. Although this is the most general possible situation, it can be overly conservative when *some* knowledge is available, concerning some of the events. We argue that for "real-life" applications, no single model of event interaction would suffice. Indeed, we need the ability to "parameterize" the model used for event interaction, depending on what *is* known about the events themselves. In this section, we develop a probabilistic calculus which allows the user to select an appropriate "mode" of event interaction, out of several choices, to suit his needs.

Let **L** be an arbitrary, but fixed, first-order language with finitely many constants, predicate symbols, infinitely many variables, and no function symbols[2]. We use (ground) atoms of **L** to represent basic events. We blur the distinction between an event and the formula representing it. Our objective is to characterize confidence levels of boolean combinations of events involving the connectives $\neg, \wedge, \vee$, under various modes (see below).

Let F and G represent two arbitrary ground (*i.e.* variable-free) formulas. For a formula F, $conf(F)$ will denote its confidence level. In the following, we describe the various modes informally and state our results on the confidence levels of conjunction and disjunction under these modes. The complete details are discussed in [15].

1. Ignorance: This is the most general situation possible: nothing is assumed/known about event interaction between F and G. The extent of the interaction between F and G could range from maximum overlap to minimum overlap.

2. Independence: This is a well-known mode. It simply says (non-)occurrence of one event does not influence that of the other.

3. Positive Correlation: This mode corresponds to the knowledge that the occurrences of two events overlap as much as possible. This means the conditional probability of one of the events (the one with the larger probability) given the other is 1. Because of the independent nature of belief and disbelief several possibilities arise.

4. Negative Correlation: This is the exact opposite of positive correlation: the occurrences of the events overlap minimally.

5. Mutual Exclusion: This is a special case of negative correlation, where we know that the sum of probabilities of the events does not exceed 1.

We have the following results.

Proposition 4.1 *Let F be any event, and let $conf(F) = \langle[\alpha,\beta],\ [\gamma,\delta]\rangle$. Then $conf(\neg F) = \langle[\gamma,\delta],\ [\alpha,\beta]\rangle$. Thus, negation simply swaps belief and doubt.*

Theorem 4.1 *Suppose F and G are any formulas and assume their confidence levels are consistent. Then the confidence levels of the formulas $F \wedge G$ and $F \vee G$, obtained under the various modes above are all consistent.*

The following theorem establishes the confidence levels of compound formulas as a function of those of the constituent formulas, under various modes. For brevity, we give the result for *ignorance* and *positive correlation* only. The reader is refereed to [15] for the complete details.

Theorem 4.2 *Let F and G be any events and let $conf(F) = \langle[\alpha_1,\beta_1],[\gamma_1,\delta_1]\rangle$ and $conf(G) = \langle[\alpha_2,\beta_2],\ [\gamma_2,\delta_2]\rangle$. Then the confidence levels of the compound events $F \wedge G$ and $F \vee G$ are given as follows. (In each case the subscript denotes the mode.)*

$conf(A \wedge_{ig} B) = \langle[max\{0, \alpha_1+\alpha_2-1\}, min\{\beta_1,\beta_2\}], [max\{\gamma_1,\gamma_2\}, min\{1, \delta_1+\delta_2\}]\rangle$.

$conf(A \vee_{ig} B) = \langle[max\{\alpha_1,\alpha_2\}, min\{1,\beta_1+\beta_2\}], [max\{0, \gamma_1+\gamma_2-1\},$

[2]These restraints are imposed for technical reasons.

$min\{\delta_1, \delta_2\}]\rangle$.
$conf(A \wedge_{pc} B) = \langle [min\{\alpha_1, \alpha_2\}, min\{\beta_1, \beta_2\}], [max\{\gamma_1, \gamma_2\}, max\{\delta_1, \delta_2\}]\rangle$.
$conf(A \vee_{pc} B) = \langle [max\{\alpha_1, \alpha_2\}, max\{\beta_1, \beta_2\}], [min\{\gamma_1, \gamma_2\}, min\{\delta_1, \delta_2\}]\rangle$.

5 Probabilistic Deductive Databases

In this section, we develop a framework for probabilistic deductive databases using a language of probabilistic programs (p-programs). We make use of the probabilistic calculus developed in Section 4 and develop the syntax and declarative semantics for programming with confidence levels. We also provide the fixpoint semantics of programs in this framework and establish its equivalence to the declarative semantics. We will use the first-order language **L** of Section 4 as the underlying logical language in this section.

Syntax of p-Programs: A *rule* is an expression of the form $A \stackrel{\mathbf{c}}{\leftarrow} B_1, \ldots, B_m$, $m \geq 0$, where A, B_i are atoms and $c \equiv \langle [\alpha, \beta], [\gamma, \delta]\rangle$ is the confidence level associated with the rule[3]. When $m = 0$, we call this a *fact*. All variables in the rule are assumed to be universally quantified outside the whole rule, as usual. We restrict attention to range restricted rules, as is customary. A *probabilistic rule* (p-rule) is a triple $(r;\ \mu_r, \mu_p)$, where r is a rule, μ_r is a mode indicating how to conjoin the confidence levels of the subgoals in the body of r (with that of r itself), and μ_p is a mode indicating how the confidence levels of different derivations of an atom involving the head predicate of r are to be disjoined. We say μ_r (μ_p) is the mode associated with the body (head) of r. We refer to r as the underlying rule of this p-rule. When r is a fact, we omit μ_r for obvious reasons. A *probabilistic program* (p-program) is a finite collection of p-rules such that whenever there are p-rules whose underlying rules define the same predicate, the mode associated with their head is identical. This last condition ensures different rules defining the same predicate q agree on the manner in which confidences of identical q-atoms generated by these rules are to be combined. The notions of Herbrand universe H_P and Herbrand base B_P associated with a p-program P are defined as usual. A p-rule is ground exactly when every atom in it is ground. The Herbrand instantiation P^* of a p-program is defined in the obvious manner. The following example illustrates our framework.

Example 5.1 *People are assessed to be at high risk for various diseases, depending on factors such as age group, family history (w.r.t. the disease), etc. Accordingly, high risk patients are administered appropriate medications, which are prescribed by doctors among several alternative ones. Medications cause side effects, sometimes harmful ones, leading to other symptoms and diseases*[4]*. Here, the extent of risk, administration of medications*[5]*, side effects (caused by medications), and prognosis are all uncertain phenomena, and we associate confidence levels with them. The following program is a sample of the uncertain knowledge related to these phenomena.*

---

[3] We assume only consistent confidence levels henceforth (see Section 3).

[4] Recent studies on the effects of certain medications on high risk patients for breast cancer provide one example of this.

[5] Uncertainty in this is mainly caused by the choices available and the fact that even under identical conditions doctors need not prescribe the same drug. The probabilities here can be derived from statistical data on the relative frequency of prescriptions of drugs under given conditions.

1. $(high\text{-}risk(X,D) \xleftarrow{\langle[\mathbf{0.65},\mathbf{0.65}],[\mathbf{0.1},\mathbf{0.1}]\rangle} midaged(X), family\text{-}history(X,D);$ $ind,\ \_).$

2. $(takes(X,M) \xleftarrow{\langle[\mathbf{0.40},\mathbf{0.40}],[\mathbf{0},\mathbf{0}]\rangle} high\text{-}risk(X,D),\ medication(D,M);$ $ign,\ \_).$

3. $(prognosis(X,D) \xleftarrow{\langle[\mathbf{0.70},\mathbf{0.70}],[\mathbf{0.12},\mathbf{0.12}]\rangle} high\text{-}risk(X,D);\ \ ign,\ pc).$

4. $(prognosis(X,D) \xleftarrow{\langle[\mathbf{0.20},\mathbf{0.20}],[\mathbf{0.70},\mathbf{0.70}]\rangle} takes(X,M), side\text{-}effects(M,D);$ $ind,\ pc).$

We can assume an appropriate set of facts (the EDB) in conjunction with the above program. For rule 1, it is easy to see that each ground atom involving the predicate high-risk has at most one derivation. Thus, a disjunctive mode for this rule will be clearly redundant, and we have suppressed it for convenience. A similar remark holds for rule 2. Rule 1 says that if a person is midaged and the disease D has struck his ancestors, then the confidence level in the person being at high risk for D is given by propagating the confidence levels of the body subgoals and combining them with the rule confidence in the sense of $\wedge_{ind}$*. This could be based on an expert's belief that the factors* $midaged$ *and* $family\text{-}history$ *contributing to high risk for the disease are independent. Each of the other rules has a similar explanation. For the last rule, we note that the potential of a medication to cause side effects is an intrinsic property independent of whether one takes the medication. Thus the conjunctive mode used there is independence. Finally, for simplicity, we show each interval in the above rules as a point probability. Still, note that the confidences for atoms derived from the program will be genuine intervals.*

A Valuation Based Semantics: We develop the declarative semantics of p-programs based on the notion of valuations. Let P be a p-program. A *probabilistic valuation* is a function $v : B_P \rightarrow \mathcal{L}_c$ which associates a confidence level with each ground atom in B_P. We define the satisfaction of p-programs under valuations, w.r.t. the truth order $\leq_t$ of the trilattice (see Section 4)[6]. We say a valuation v *satisfies* a ground p-rule $\rho \equiv (A \xleftarrow{\mathbf{c}} B_1, \ldots, B_m,\ \mu_r, \mu_p)$, denoted $\models_v \rho$, provided $c \wedge_{\mu_r} v(B_1) \wedge_{\mu_r} \cdots \wedge_{\mu_r} v(B_m) \leq_t v(A)$. The intended meaning is that in order to satisfy this p-rule, v must assign a confidence level to A that is no less true (in the sense of $\leq_t$) than the result of the conjunction of the confidences assigned to B_i's by v and the rule confidence c, in the sense of the mode μ_r. Even when a valuation satisfies (all ground instances of) each rule in a p-program, it may not satisfy the p-program as a whole. The reason is that confidences coming from different derivations of atoms are combined strengthening the overall confidence. Thus, we need to impose the following additional requirement.

Let $\rho \equiv (r \equiv A \xleftarrow{\mathbf{c}} B_1, \ldots, B_m;\ \ \mu_r, \mu_p)$ be a ground p-rule, and v a valuation. Then we denote by $rule\text{-}conf(A, \rho, v)$ the confidence level propagated to the head of this rule under the valuation v and the rule mode μ_r, given by the expression $c \wedge_{\mu_r} v(B_1) \wedge_{\mu_r} \cdots \wedge_{\mu_r} v(B_m)$. Let $P^* = P_1^* \cup \cdots \cup P_k^*$ be the partition of P^* such that (i) each P_i^* contains all (ground) p-rules which define the same atom, say A_i, and (ii) A_i and A_j are distinct, whenever $i \neq j$. Suppose μ_i is the mode associated with the head of the p-rules in P_i^*. We denote by $atom\text{-}conf(A_i, P, v)$ the confidence level determined for

[6] Satisfaction can be defined w.r.t. each of the 3 orders of the trilattice, giving rise to different interesting semantics. Their discussion is beyond the scope of this paper.

the atom A_i under the valuation v using the program P. This is given by the expression $\vee_{\mu_i}\{rule\text{-}conf(A_i, \rho, v) | \rho \in P_i*\}$. We now define satisfaction of p-programs.

Definition 5.1 *Let P be a p-program and v a valuation. Then v satisfies P, denoted $\models_v P$ exactly when v satisfies each (ground) p-rule in P^*, and for all atoms $A \in B_P$, $atom\text{-}conf(A, P, v) \leq_t v(A)$.*

The additional requirement ensures the valuation assigns a strong enough confidence to each atom so it will support the combination of confidences coming from a number of rules (pertaining to this atom). A p-program P logically implies a p-fact $A \xleftarrow{\mathbf{c}}$, denoted $P \models A \xleftarrow{\mathbf{c}}$, provided every valuation satisfying P also satisfies $A \xleftarrow{\mathbf{c}}$. We next have

Proposition 5.1 *Let v be a valuation and P a p-program. Suppose the mode associated with the head of each p-rule in P is positive correlation. Then $\models_v P$ iff v satisfies each rule in P^*.*

The above proposition shows that when positive correlation is the only disjunctive mode used, satisfaction is very similar to the classical case.

For the declarative semantics of p-programs, we need something like the "least" valuation satisfying the program. It is straightforward to show that the class of all valuations Υ from B_P to $\mathcal{L}_c$ itself forms a trilattice, complete with all the 3 orders and the associated meets and joins. They are obtained by a pointwise extension of the corresponding order/operation on the trilattice $\mathcal{L}_c$. We give one example. For valuations u, v, $u \leq_t v$ iff $\forall A \in B_P$, $u(A) \leq_t v(A)$; $\forall A \in B_P$, $(u \otimes_t v)(A) = u(A) \otimes_t v(A)$. One could investigate "least" w.r.t. each of the 3 orders of the trilattice. In this paper, we confine attention to the order $\leq_t$. The least (greatest) valuation is then the valuation **false** (**true**) which assigns the confidence level $\bot_t$ ($\top_t$) to every ground atom. We now have

Lemma 5.1 *Let P be any p-program and u, v be any valuations satisfying P. Then $u \otimes_t v$ is also a valuation satisfying P. In particular, $\otimes_t\{v | \models_v P\}$ is the least valuation satisfying P.*

We take the least valuation satisfying a p-program as characterizing its declarative semantics.

Example 5.2 *Consider the following p-program P.*

1. $(A \xleftarrow{\langle[\mathbf{0.5,0.7}],\ [\mathbf{0.3,0.45}]\rangle} B;\ ind,\ pc)$. *2.* $(A \xleftarrow{\langle[\mathbf{0.6,0.8}],\ [\mathbf{0.1,0.2}]\rangle} C;\ ign,\ pc)$.
3. $(B \xleftarrow{\langle[\mathbf{0.9,0.95}],\ [\mathbf{0,0.1}]\rangle};\ \_,\ ind)$. *4.* $(C \xleftarrow{\langle[\mathbf{0.7,0.8}],\ [\mathbf{0.1,0.2}]\rangle};\ \_,\ ind)$.

In the following we show three valuations v_1, v_2, v_3, of which v_1 and v_3 satisfy P, while v_2 does not. In fact, v_3 is the least valuation satisfying P.

val	B	C	A
v_1	$\langle[0.9, 1],\ [0, 0]\rangle$	$\langle[0.8, 0.9],\ [0.05, 0.1]\rangle$	$\langle[0.5, 0.9],\ [0, 0]\rangle$
v_2	$\langle[0.9, 1],\ [0, 0]\rangle$	$\langle[0.9, 1],\ [0, 0]\rangle$	$\langle[0.5, 0.7],\ [0.1, 0.4]\rangle$
v_3	$\langle[0.9, 0.95],\ [0, 0.1]\rangle$	$\langle[0.7, 0.8],\ [0.1, 0.2]\rangle$	$\langle[0.45, 0.8],\ [0.1, 0.4]\rangle$

Fixpoint Semantics: We associate an "immediate consequence" operator T_P with a p-program P, defined as follows.

Definition 5.2 *Let P be a p-program and P^* its Herbrand instantiation. Then T_P is a function $T_P : \Upsilon \rightarrow \Upsilon$, defined as follows. For any probabilistic valuation v, and any ground atom $A \in B_P$, $T_P(v)(A) = \vee_{\mu_p}\{c_A | \exists$ a p-rule $(A \stackrel{c}{\leftarrow} B_1, \ldots, B_m,\ \mu_r,\ \mu_p) \in P^*$, s.t. $c_A = c \wedge_{\mu_r} v(B_1) \wedge_{\mu_r} \cdots \wedge_{\mu_r} v(B_m)\}$.*

Call a valuation v *consistent* provided for every atom A, $v(A)$ is consistent, as defined in Section 3.

Theorem 5.1 *(1) T_P always maps consistent valuations to consistent valuations. (2) T_P is monotone and continuous.*

We define bottom-up iterations based on T_P in the usual manner.
$T_P \uparrow 0 =$ **false** (which assigns the truth-value $\perp_t$ to every ground atom).
$T_P \uparrow \alpha = T_P(T_P \uparrow \alpha - 1)$, for a successor ordinal α.
$T_P \uparrow \alpha = \oplus_t\{T_P \uparrow \beta | \beta < \alpha\}$, for a limit ordinal α.
We have the following results.

Proposition 5.2 *Let v be any valuation and P be a p-program. Then v satisfies P iff $T_P(v) \leq_t v$.*

Theorem 5.2 *Let P be a p-program. Then the following claims hold.*
(i) $lfp(T_P) = \otimes_t\{v | \models_v P\} =$ the $\leq_t$-least valuation satisfying P.
(ii) For a ground atom A, $lfp(T_P)(A) = c$ iff $P \models A \stackrel{c}{\leftarrow}$.

The *closure ordinal* of T_P is the smallest ordinal α such that $T_P \uparrow \alpha = lfp(T_P)$.

Theorem 5.3 *Let P be any p-program. Then the closure ordinal of T_P is at most ω. Further, suppose for all recursive predicates, P only associates positive correlation as the disjunctive mode. Then its closure ordinal is finite.*

We show in Section 7 that there are p-programs for which the closure ordinal does reach ω.

6 Proof Theory

Since confidences coming from different derivations of a fact are combined, we need a notion of disjunctive proof-trees. We note that the notions of substitution, unification, etc. are analogous to the classical ones. A variable appearing in a rule is *local* if it only appears in its body.

Definition 6.1 *Let G be a(n atomic) goal and P a p-program. Then a disjunctive proof-tree (DPT) for G w.r.t. P is a tree T defined as follows.*
1. T has two kinds of nodes: rule *nodes and* goal *nodes. Each rule node is labeled by an instance of a rule in P and a substitution. Each goal node is labeled by an atomic goal. The root is a goal node labeled G.*
2. Let u be a goal node labeled by an atom A. Then every child (if any) of u is a rule node labeled (r', θ), where r' is an instance of a rule r in P after its head is unified with A using the mgu θ. In this case, the mode associated with the body of r is the mode of node u.
3. Whenever an atom B occurs in the body of a rule labeling a node u, u has a goal child v labeled B. The mode associated with the head of any rule defining B is the mode of v.
4. For any two substitutions π, θ occurring in T, $\pi(V) = \theta(V)$, for every variable V. In other words, all substitutions occurring in T are compatible.

A node without children is called a leaf. A DPT T is *proper* provided whenever T has a goal leaf labeled A, there is no rule in P whose head is unifiable with A. We only consider proper DPTs unless otherwise specified. A rule leaf is a *success* node while a goal leaf is a *failure* node.

Confidences are associated with (finite) DPTs as follows.

Definition 6.2 *Let P be a p-program, G a goal, and T any finite DPT for G w.r.t. P. We associate confidences with the nodes of T as follows.*
1. Each failure nodes gets the confidence $\langle [0,0],\ [1,1] \rangle$ (see Section 3). Each success node labeled (r', θ), where r' is an instance of rule $r \in P$, and c is the confidence of rule r, gets the confidence c.
2. Suppose u is a rule node labeled (r', θ), with a mode μ_r, where r' is an instance of a rule r such that the confidence of r is c, and the confidences of the children of u are $c_1, \ldots, c_m$. Then u gets the confidence $c \wedge_{\mu_r} c_1 \wedge_{\mu_r} \cdots \wedge_{\mu_r} c_m$.
3. Suppose u is a goal node labeled A, with a mode μ_p such that the confidences of its children are $c_1, \ldots, c_k$. Then u gets the confidence $c_1 \vee_{\mu_p} \cdots \vee_{\mu_p} c_k$.

We have the following theorems. For technical details, the reader is referred to [15].

Theorem 6.1 (Soundness) *Let P be a p-program and G a goal. If there is a DPT for G w.r.t. P with an associated confidence c at its root, then $c \leq_t lfp(T_P)(G)$.*

Theorem 6.2 (Completeness) *Let P be a p-program and G a goal such that for some number $k < \omega$, $lfp(T_P)(G) = T_P \uparrow k(G)$. Then there is a finite DPT T for G w.r.t. P with an associated confidence c at its root, such that $lfp(T_P)(G) \leq_t c$.*

In particular, when modes other than positive correlation are not used as a disjunctive mode for recursive predicates, the theorems guarantee that the exact confidence associated with the goal can be determined by constructing a complete DPT for it. Even when these modes *are* used indiscriminately, we can still obtain the confidence associated with the goal with an arbitrarily high degree of accuracy, by constructing DPTs of appropriate height.

7 Termination and Complexity

In this section, we first compare our work with that of Ng and Subrahmanian [17, 18] (see Section 1 for a general comparison with non-probabilistic frameworks). First, let us examine the (only) "mode" for disjunction used by them[7]. They combine the confidences of an atom A coming from different derivations by taking their intersection. Indeed, the bottom of their lattice is a valuation (called "formula function" there) that assigns the interval $[0, 1]$ to every atom. From the trilattice structure, it is clear that (i) their bottom corresponds to $\perp_p$, and (ii) their disjunctive mode corresponds to $\oplus_p$.

Example 7.1 *r_1: $p(X, Y) : [V_1 \times V_3, V_2 \times V_4] \leftarrow e(X, Z) : [V_1, V_2],\ \ p(Z, Y) : [V_3, V_4]$.*

[7]Their framework allows an infinite class of "conjunctive modes". Also, recall they represent only beliefs.

r_2: $p(X,Y):[V_1,V_2] \leftarrow e(X,Y):[V_1,V_2]$.
r_3: $e(1,2):[1,1]$.
r_4: $e(1,1):[0.9,0.9]$.

This is a pf-program in the framework of Ng and Subrahmanian [18]. Let us denote the operator T_P *defined by them as* T_P^{NS} *for distinguishing it from ours. It is not hard to see that* $lfp(T_P^{NS})$ *would assign an empty probability range for* $p(1,2)$. *This is quite unintuitive. Indeed, there is a definite path (with probability 1) corresponding to the edge* $e(1,2)$.
Suppose now rule r_3 *is replaced by* r_3': $e(1,2):[0,1]$. *In this case, the least fixpoint of* T_P^{NS} *is only attained at* ω *and it assigns the range* $[0,0]$ *to* $p(1,2)$. *Again, the result is unintuitive for this example. Since* T_P^{NS} *is not continuous, one can easily write programs such that no reasonable approximation to* $lfp(T_P^{NS})$ *can be obtained by iterating* T_P^{NS} *an arbitrary (finite) number of times. (*E.g. , *consider the program obtained by adding the rule* r_5: $q(X,Y):[1,1] \leftarrow p(X,Y):[0,0]$ *to* $\{r_1, r_2, r_3', r_4\}$.*) Notice that as long as one uses any arithmetic annotation function such that the probability of the head is less than the probability of the subgoals of* r_1 *(which is a reasonable annotation function), this problem will arise. The problem (for the unintuitive behavior) lies with the mode for disjunction. Our point, however, is* not *that intersection is a "wrong" mode. Rather, we stress that different combination rules (modes) are appropriate for different situations.*

Now, consider the p-program corresponding to the annotated program $\{r_1, \ldots, r_4\}$, *obtained by stripping off atom annotations in* r_1, r_2 *and shifting the annotations in* r_3, r_4 *to the associated rules. Also, associate the confidence level* $\langle[1,1],\ [0,0]\rangle$ *with* r_1, r_2. *For uniformity and ease of comparison, assume the doubt ranges are all* $[0,0]$. *As an example, let the conjunctive mode used in* r_1, r_2 *be independence and let the disjunctive mode used be positive correlation (or, in this case, even ignorance!). Then* $lfp(T_P)$ *would assign the confidence* $\langle[1,1],\ [0,0]\rangle$ *to* $p(1,2)$, *which agrees with our intuition. Consider the p-program corresponding to the rules* $\{r_1, r_2, r_3', r_4\}$, *obtained as suggested above. Suppose positive correlation (ignorance) is used for disjunction and independence for conjunction, as before. Then* $lfp(T_P)$ *would assign the confidence level* $\langle[0,1]\ [0,0]\rangle$ *to* $p(1,2)$. *This again agrees with our intuition. As a last example, suppose we start with the confidence* $\langle[0,0.1],\ [0,0]\rangle$ *for* $e(1,2)$ *instead. Then under positive correlation (for disjunction)* $lfp(T_P)(p(1,2)) = \langle[0,0.1],\ [0,0]\rangle$ *while ignorance leads to* $lfp(T_P)(p(1,2)) = \langle[0,1],\ [0,0]\rangle$. *The former makes more intuitive sense, although the latter (more conservative under* $\leq_p$*) is obviously* not *wrong. Also, in the latter case, the* lfp *is reached only at* ω.

We define the *data complexity* [21] of a p-program P as the time complexity of computing the least fixpoint of T_P as a function of the size of the database, *i.e.* the number of constants occurring in P[8]. It is well known that the data complexity for datalog programs is polynomial. An important question concerning any extension of DDBs to handle uncertainty is whether the data complexity is increased compared to datalog. We can show that under suitable restrictions (see below) the data complexity of p-programs is polynomial time. However, the proof cannot be obtained by (straightforward extensions of) the classical argument for the data complexity for datalog. In the classical case, once a ground atom is derived during bottom-up evaluation, future

[8]With many rule-based systems with uncertainty, we cannot always separate EDB and IDB predicates, which explains this slightly modified definition of data complexity.

derivations of it can be ignored. In programming with uncertainty, complications arise because we *cannot* ignore alternate derivations of the same atom: the confidences obtained from them need to be combined, reinforcing the overall confidence of the atom. This calls for a new proof technique. Our technique makes use of the following additional notions.

Define a *disjunctive derivation tree* (DDT) to be a finite DPT (see Section 6 for a definition) such that every rule and every goal labeling any node in the tree is ground (and hence all substitutions in this case will be ground substitutions and not necessarily mgu's). It is straightforward to show that the height of a DDT is always an odd number. We have the following results.

Proposition 7.1 *Let P be a p-program and A any ground atom in B_P. Suppose the confidence determined for A in iteration $k \geq 1$ of bottom-up evaluation is c. Then there exists a DDT T of height $2k-1$ for A such that the confidence associated with A by T is exactly c.*

Define a DDT to be *simple* provided there is no pair of goal nodes u and v in it such that (i) u and v are labeled by the same ground atom and (ii) u is an ancestor of v. The following is the key result for proving the polynomial time complexity bound.

Theorem 7.1 *Let P be a p-program such that only positive correlation is used as the disjunctive mode for recursive predicates. Let $max\{height(T_A) \mid A \in B_P$, and T_A is any simple DDT for $A\} = 2k-1$, $k \geq 1$. Then at most $k+1$ iterations of naive bottom-up evaluation are needed to compute the least fixpoint of T_P.*

It can be shown that the height of simple DDTs is polynomially bounded by the database size. This makes the above result significant. Finally, we have

Theorem 7.2 *Let P be a p-program such that only positive correlation is used as the disjunctive mode for recursive predicates. Then its least fixpoint can be computed in time polynomial in the database size. In particular, bottom-up naive evaluation terminates in time polynomial in the size of the database, yielding the least fixpoint.*

We remark that our proof of Theorem 7.2 implies a similar result for van Emden's framework. To our knowledge, this is the first polynomial time result for rule-based programming with (probabilistic) uncertainty[9]. We should point out that the polynomial time complexity is preserved whenever modes other than positive correlation are associated with non-recursive predicates (for disjunction). More generally, suppose R is the set of all recursive predicates and N is the set of non-recursive predicates in the KB, which are possibly defined in terms of those in R. Then any modes can be freely used with the predicates in N while keeping the data complexity polynomial. Finally, if we know that the data does not contain cycles, we can use any mode even with a recursive predicate and still have a polynomial time data complexity. In the full paper [15], we make a detailed comparison with non-probabilistic frameworks such as Kifer and Li [12] and Kifer and Subrahmanian [14]. In particular, the framework of annotation functions used

[9] It is straightforward to show that the data complexity for the framework of [17] is polynomial, although the paper does not address this issue. However, that framework only allows constant annotations and is of limited expressive power.

in [14] enables an infinite family of modes to be used in propagating confidences from rule bodies to heads. The major differences with our work are (i) in [14] a fixed "mode" for disjunction is imposed unlike our framework, and (ii) they do not study the complexity of query answering, whereas we establish the conditions under which the important advantage of polynomial time data complexity of classical datalog can be retained. More importantly, our work has generated useful insights into how modes (for disjunction) affect the data complexity. Finally, a note about the use of positive correlation as the disjunctive mode for recursive predicates (when data might contain cycles). The rationale is that different derivations of such recursive atoms could involve some amount of overlap (the degree of overlap depends on the data). Now, positive correlation (for disjunction) tries to be conservative (and hence sound) by assuming the extent of overlap is maximal, so the combined confidence of the different derivations is the least possible (under $\leq_t$). Thus, it *does* make sense even from a practical point of view.

8 Conclusions

We motivated the problem of extending the DDB technology to cope with rule-based programming with imperfect knowledge. In this paper, we have taken a few steps in this direction, by developing a framework for programming with uncertainty, expressed in the form of belief and doubt associated with facts and rules in the knowledge-base. Belief and doubt are ranges of probabilities, and give rise to a natural notion of a trilattice. We developed a probabilistic calculus permitting different modes for combining confidence levels of events. We then developed the framework of p-programs for implementing probabilistic DDBs. p-Programs inherit the ability to "parameterize" the modes used for combining confidence levels, from our probabilistic calculus. We have developed a declarative semantics, a fixpoint semantics, and proved their equivalence. We have also provided a sound and complete proof-theory. We have shown that under disciplined use of modes, we can retain the (important) advantage of polynomial time data complexity of classical datalog, in this extended framework. We have also compared our framework with related work w.r.t. the aspects of termination and intuitive behavior (of the semantics). The parametric nature of modes in p-programs is shown to be a significant advantage w.r.t. these aspects. Also, the analysis of trilattices shows insightful relationships between previous work (*e.g.* , Ng and Subrahnmanian [17, 18]) and ours.

Interesting open issues which merit further research include (1) semantics of p-programs under various trilattice orders and various modes, including new ones, (2) query optimization, (3) handling inconsistency in a framework handling uncertainty, such as the one studied here.

Acknowledgments: The authors would like to thank Shiri Nematollaah for his careful reading of the manuscript, and the anonymous referees for their useful comments and suggestions.

References

[1] S. Abiteboul, P. Kanellakis, and G. Grahne. On the representation and querying of sets of possible worlds. *Theoretical Computer Science*, 78:159–187, 1991.

[2] J.F. Baldwin. Evidential support logic programming. *Journal of Fuzzy Sets and Systems*, 24:1–26, 1987.

[3] C. Barbara, H. Garcia-Molina, and D. Porter. A probabilistic relational data model. In *Proc. Advancing Database Technology EDBT 90*, pages 60–64, 1990.

[4] R. Carnap. *The Logical Foundations of Probability.* University of Chicago Press, 1962. 2nd. Edn.

[5] Fangqing Dong and Laks V. S. Lakshmanan. Deductive databases with incomplete information. In *Joint Int. Conf. and Symp. on Logic Programming*, Washington, D.C., Nov 1992.

[6] R. Fagin, J. Halpern, and N. Meggido. A logic for reasoning about probabilities. *Information and Computation*, 1992.

[7] J. E. Fenstad. The structure of probabilities defined on first-order languages. In R. C. Jeffrey, editor, *Studies in inductive logic and probabilities, volume 2*, pages 251–262. University of California Press, 1980.

[8] M. C. Fitting. Logic programming on a topological bilattice. *Fundamenta Informaticae*, 11:209–218, 1988.

[9] M. C. Fitting. Bilattices and the semantics of logic programming. *Journal of Logic Programming*, 11:91–116, 1991.

[10] M. Ginsburg. Multivalued logics: A uniform approach to reasoning in artificial intelligence. *Computational Intelligence*, 4:265–316, 1988.

[11] U. Güntzer, W. Kießling, and H. Thöne. New directions for uncertainty reasoning in deductive databases. In *Proc. ACM SIGMOD International Conf. on Management of Data*, pages 178–187, 1991.

[12] M. Kifer and A. Li. On the semantics of rule-based expert systems with uncertainty. In *2nd Intl. Conf. on Database Theory*, pages 102–117, August 1988.

[13] M. Kifer and E.L. Lozinskii. A logic for reasoning with inconsistency. In *Proc. 4th IEEE Symp. on Logic in Computer Science*, pages 253–262, Asilomar, CA, 1989.

[14] Michael Kifer and V. S. Subrahmanian. Theory of generalized annotated logic programming and its applications. *Journal of Logic Programming*, 12:335–367, 1992.

[15] Laks V.S. Lakshmanan and Fereidoon Sadri. A framework for modeling uncertainty in databases and knowledge-bases. Technical report, Concordia University, Montreal, Quebec, 1993.

[16] Y. Liu. Null values in definite programs. In *Proc. North American Conf. on Logic Programming*, pages 273–288, Austin, TX, Nov. 1990.

[17] R. Ng and V. S. Subrahmanian. Probabilistic logic programming. *Information and Computation*, 101(2):150–201, December 1992.

[18] R. T. Ng and V. S. Subrahmanian. A semantical framework for supporting subjective and conditional probabilities in deductive databases. Technical Report CS-TR-2563, University of Maryland, College Park, MD, November 1990.

[19] N. Steger, H. Schmidt, U. Güntzer, and W. Kießling. Semantics and efficient compilation for quantitative deductive databases. In *Proc. IEEE Int. Conf. on Data Engineering*, pages 660–669, 1989.

[20] M. H. van Emden. Quantitative deduction and its fixpoint theory. *Journal of Logic Programming*, 4(1):37–53, 1986.

[21] M. Y. Vardi. Querying logical databases. In *Proceedings of the 4th ACM Symp. on Principles of Database Systems*, pages 57–65, 1985.