Many-Valued Disjunctive Logic Programs with Probabilistic Semantics

Thomas Lukasiewicz

Institut für Informationssysteme, Technische Universität Wien
Treitlstraße 3, A-1040 Wien, Austria
lukasiewicz@kr.tuwien.ac.at

Abstract. We present many-valued disjunctive logic programs in which classical disjunctive logic program clauses are extended by a truth value that respects the material implication. Interestingly, these many-valued disjunctive logic programs have both a probabilistic semantics in probabilities over possible worlds and a truth-functional semantics. We then define minimal, perfect, and stable models and show that they have the same properties like their classical counterparts. In particular, perfect and stable models are always minimal models. Under local stratification, the perfect model semantics coincides with the stable model semantics. Finally, we show that some special cases of propositional many-valued disjunctive logic programming under minimal, perfect, and stable model semantics have the same complexity like their classical counterparts.

1 Introduction

In the logic programming framework, there exist at least two main streams in handling uncertain knowledge. Many-valued and probabilistic logic programming aims to handle numerical uncertainty, whereas disjunctive logic programming deals with disjunctive knowledge and nonmonotonic negation. In this paper, we now propose a combination of both of them in a uniform framework.

This paper relies on probability theory as a commonly accepted formalism for handling numerical uncertainty. Probabilistic propositional logics and related languages are thoroughly studied in the literature (see especially [26] and [7]). Their extensions to probabilistic first-order logics can be classified into first-order logics in which probabilities are defined over a set of possible worlds and those in which probabilities are given over the domain (see especially [2] and [9]). The first ones are suitable for representing degrees of belief, while the latter are appropriate for describing statistical knowledge. In the present paper, we assume that probabilities are defined over a set of possible worlds.

Probabilistic reasoning in its full generality is a quite tricky task and very different from classical reasoning (see especially [19], [15], and [14]). It should generally be performed by global linear programming methods, rather than by local inference techniques. For this reason, it is generally also computationally more complex than classical reasoning.

In particular, the model and fixpoint characterization and the proof theory of classical definite logic programming generally do not carry over to probabilistic definite logic programming (as presented in [14]). Moreover, the tractability of special cases of classical logic programming generally does not carry over to the corresponding special cases of probabilistic logic programming.

However, we would like an approach to many-valued disjunctive logic programming that does not ignore the years of work in classical disjunctive logic programming. Furthermore, it would be nice if query processing in many-valued disjunctive logic programs is not computationally more complex than query processing in classical disjunctive logic programs.

The key to achieve all this is to increase the axioms of probability by an axiom that brings probabilistic logics closer to truth-functional logics [17].

In detail, our many-valued disjunctive logic programs have a probabilistic semantics in probabilities over possible worlds. Furthermore, the truth values of all clauses are truth-functionally defined on the truth values of atoms.

We showed in [17] and [18] that many-valued definite logic programming with this probabilistic semantics has a model and fixpoint characterization and a proof theory similar to classical definite logic programming. Moreover, special cases of many-valued logic programming with this semantics were shown to have the same computational complexity like their classical counterparts.

Many-valued definite logic programming with this probabilistic semantics has an important companion in the literature. More precisely, van Emden's quantitative deduction [31] can be given a probabilistic semantics by probabilities over possible worlds under the additional axiom. However, van Emden's quantitative deduction is based on a conditional probability semantics of the implication connective, while [17], [18], and the present paper use the material implication semantics. Interestingly, it turns out that the material implication is much closer to classical logic programming. In particular, the material implication is more suitable for additionally handling disjunction and nonmonotonic negation.

It is also important to point out that both many-valued definite logic programming with probabilistic semantics and van Emden's quantitative deduction are *approximations* of probabilistic logic programming. More precisely, our approach is an approximation of probabilistic logic programming under the material implication [18], while van Emden's quantitative deduction can be understood as an approximation of probabilistic logic programming under the conditional probability implication (as defined in [14]).

The literature contains many other approaches to many-valued logic programming (see, for example, [11], [31], [3], [8], and [20]) and probabilistic logic programming (see, for example, [23], [27], [24], [25], [4], [14], and [22]).

To our knowledge, this paper is the first to integrate numerical uncertainty in the form of probabilities over possible worlds, disjunction, and nonmonotonic negation in a uniform framework close to classical disjunctive logic programming.

The work closest in spirit to this paper is perhaps the one by Mateis [20]. It also combines numerical uncertainty, disjunctive knowledge, and nonmono-

tonic negation. Its uncertainty formalism, however, is based on t-norms and not on probabilities over possible worlds.

Ngo [25] also combines numerical uncertainty and disjunction. However, he does not consider nonmonotonic negation. Moreover, he does not allow numerical uncertainty on the rule level. Furthermore, his approach is closer to Bayesian networks than to classical disjunctive logic programming.

Finally, Ng and Subrahmanian [23] also deal with the combination of numerical uncertainty, disjunctive knowledge, and nonmonotonic negation. However, they also do not allow numerical uncertainty on the rule level. Moreover, their work is perhaps best described as logic programming under nonmonotonic negation about probabilistic disjunctions and conjunctions of atoms.

The main contributions of this paper can be summarized as follows:

- We present many-valued disjunctive logic programs in which classical disjunctive logic program clauses are extended by a truth value that respects the material implication. These programs have both a probabilistic semantics in probabilities over possible worlds and a truth-functional semantics.
- We define minimal, perfect, and stable models and show that they have the same properties like their classical counterparts. In particular, perfect and stable models are always minimal. Furthermore, under local stratification, the perfect model semantics coincides with the stable model semantics.
- We show that the problems of deciding whether a ground program has a minimal, perfect, or stable model have the same complexity like their classical counterparts. Moreover, we show that some special cases of propositional query processing under minimal, perfect, and stable model semantics have the same complexity like their classical counterparts.

The rest of this paper is organized as follows. In Section 2, we describe the technical background in probabilistic first-order logics over possible worlds. Sections 3 and 4 introduce many-valued disjunctive logic programs. In Section 5, we focus on their minimal, perfect, and stable models. Section 6 concentrates on the complexity of many-valued disjunctive logic programming. In Section 7, we summarize the main results and give an outlook on future research.

Note that all proofs are given in full detail in [16].

2 Technical Preliminaries

In this section, we focus on the technical background. We briefly describe first-order logics of probability and their semantics in Pr- and Pr*-interpretations.

2.1 Pr-Interpretations

We now briefly summarize how (a quantifier-free fragment of) classical first-order logics can be given a probabilistic semantics in which probabilities are defined

over a set of possible worlds. We basically follow the work by Halpern [9], which we adapt to our needs in the logic programming framework.

Let Φ be a first-order vocabulary that contains a set of function symbols and a set of predicate symbols (as usual, *constant symbols* are function symbols of arity zero). Let $\mathcal{X}$ be a set of variables.

We define *terms* by induction as follows. A term is a variable from $\mathcal{X}$ or an expression $f(t_1, \ldots, t_k)$, where f is a function symbol of arity $k \geq 0$ from Φ and $t_1, \ldots, t_k$ are terms. We define *classical formulas* by induction as follows. If p is a predicate symbol of arity $k \geq 0$ from Φ and $t_1, \ldots, t_k$ are terms, then $p(t_1, \ldots, t_k)$ is a classical formula (called *atom*). If F and G are classical formulas, then $\neg F$ and $(F \wedge G)$ are classical formulas. A *probabilistic formula* is an expression $\mathsf{prob}(F) \geq c$, where F is a classical formula and c is a real number from $[0, 1]$.

We abbreviate $(F \vee G)$ and $(F \leftarrow G)$ by $\neg(\neg F \wedge \neg G)$ and $\neg(\neg F \wedge G)$, respectively. We adopt the usual conventions to eliminate parentheses in combination with these abbreviations. Literals, positive literals, and negative literals are defined as usual. Terms, classical formulas, and probabilistic formulas are *ground* iff they do not contain any variables. The notions of substitutions, ground substitutions, and ground instances of classical formulas are defined as usual. The latter is assumed to be canonically extended to probabilistic formulas.

An *interpretation* I is a subset of the Herbrand base HB_Φ over Φ. A *variable assignment* σ is a mapping that assigns to each variable from $\mathcal{X}$ an element from the Herbrand universe HU_Φ over Φ. It is by induction extended to terms by $\sigma(f(t_1, \ldots, t_k)) = f(\sigma(t_1), \ldots, \sigma(t_k))$ for all terms $f(t_1, \ldots, t_k)$. The *truth* of classical formulas F in I under σ, denoted $I \models_\sigma F$, is inductively defined as follows (we write $I \models F$ if F is ground):

- $I \models_\sigma p(t_1, \ldots, t_k)$ iff $p(\sigma(t_1), \ldots, \sigma(t_k)) \in I$.
- $I \models_\sigma \neg F$ iff not $I \models_\sigma F$, and $I \models_\sigma (F \wedge G)$ iff $I \models_\sigma F$ and $I \models_\sigma G$.

A *probabilistic interpretation (Pr-interpretation)* $\boldsymbol{p} = (\mathcal{I}, \mu)$ consists of a set $\mathcal{I}$ of classical interpretations (called *possible worlds*) and a discrete probability function μ on $\mathcal{I}$ (that is, a mapping μ from $\mathcal{I}$ to the real interval $[0, 1]$ such that all $\mu(I)$ with $I \in \mathcal{I}$ sum up to 1 and that the number of all $I \in \mathcal{I}$ with $\mu(I) > 0$ is countable). The *truth value* $\boldsymbol{p}_\sigma(F)$ of a formula F in the Pr-interpretation $\boldsymbol{p}$ under a variable assignment σ is defined by (we write $\boldsymbol{p}(F)$ if F is ground):

$$\boldsymbol{p}_\sigma(F) \;=\; \sum_{I \in \mathcal{I},\, I \models_\sigma F} \mu(I)\,. \tag{1}$$

A probabilistic formula $\mathsf{prob}(F) \geq c$ *is true* in $\boldsymbol{p}$ under σ iff $\boldsymbol{p}_\sigma(F) \geq c$. The formula $\mathsf{prob}(F) \geq c$ *is true* in $\boldsymbol{p}$, or $\boldsymbol{p}$ is a *model* of $\mathsf{prob}(F) \geq c$, denoted $\boldsymbol{p} \models \mathsf{prob}(F) \geq c$, iff $\mathsf{prob}(F) \geq c$ is true in $\boldsymbol{p}$ under all variable assignments σ. The Pr-interpretation $\boldsymbol{p}$ is a *model* of a set of probabilistic formulas $\mathcal{P}$, denoted $\boldsymbol{p} \models \mathcal{P}$, iff $\boldsymbol{p}$ is a model of all probabilistic formulas in $\mathcal{P}$. The set of probabilistic formulas $\mathcal{P}$ is *satisfiable* iff a model of $\mathcal{P}$ exists. The formula $\mathsf{prob}(F) \geq c$ is a *tight logical consequence* of $\mathcal{P}$, denoted $\mathcal{P} \models_{tight} \mathsf{prob}(F) \geq c$, iff c is the infimum of $\boldsymbol{p}_\sigma(F)$ subject to all models $\boldsymbol{p}$ of $\mathcal{P}$ and all variable assignments σ.

For Pr-interpretations $\boldsymbol{p} = (\mathcal{I}, \mu)$ with $\mu(I) = 1$ for some possible world $I \in \mathcal{I}$, we use $\mathit{classical}(\boldsymbol{p})$ to denote this I. For a set of probabilistic formulas $\mathcal{P}$, we use $\mathit{classical}(\mathcal{P})$ to denote the set of all F with $\mathsf{prob}(F) \geq 1 \in \mathcal{P}$.

2.2 Pr*-Interpretations

We now define Pr*-interpretations by restricting Pr-interpretations (that is, by assuming another axiom besides the axioms of probability):

A *Pr*-interpretation* is a Pr-interpretation $\boldsymbol{p}$ with:

$$\boldsymbol{p}(A \wedge B) = \min(\boldsymbol{p}(A), \boldsymbol{p}(B)) \text{ for all } A, B \in HB_{\Phi}\,. \tag{2}$$

Note that the condition $\boldsymbol{p}(A \wedge B) = \min(\boldsymbol{p}(A), \boldsymbol{p}(B))$ is just assumed for ground atoms A and B. This condition brings probabilistic logics over possible worlds closer to truth-functional logics.

It is important to point out that we do not assume that (2) always holds in the part of the real world that we want to model. The axiom (2) is simply a *technical assumption* that carries us to a form of many-valued logic programming that *approximates* probabilistic logic programming (see Section 5.1). It makes a global probabilistic semantics over possible worlds match with the truth-functionality that stands behind logic programming techniques.

Interestingly, the axiom (2) is equivalent to the assumption of a subset relationship between possible worlds as follows.

Theorem 1. *Let $\boldsymbol{p} = (\mathcal{I}, \mu)$ be a Pr-interpretation. Let $\mathcal{I}_+ = \{I \in \mathcal{I} \mid \mu(I) > 0\}$ and for all ground atoms A let $\mathcal{I}_+(A) = \{I \in \mathcal{I}_+ \mid I \models A\}$. Then the condition (2) is equivalent to each of the following conditions (3) and (4):*

$$\mathcal{I}_+(A) \subseteq \mathcal{I}_+(B) \;\; \textit{or} \;\; \mathcal{I}_+(A) \supseteq \mathcal{I}_+(B) \;\; \textit{for all } A, B \in HB_{\Phi} \tag{3}$$

$$I_1 \subseteq I_2 \;\; \textit{or} \;\; I_1 \supseteq I_2 \;\; \textit{for all } I_1, I_2 \in \mathcal{I}_+\,. \tag{4}$$

The next theorem shows that the truth value of certain ground formulas under Pr*-interpretations is truth-functionally defined on the truth values of their components. In particular, the truth value of all ground classical clauses is truth-functionally defined on the truth values of their ground atoms. Note that the truth functions are the same as in the nondenumerable infinite-valued Łukasiewicz logic $\mathrm{Ł}_{\aleph_1}$ (see [30] for a survey).

Theorem 2. *For all Pr*-interpretations $\boldsymbol{p} = (\mathcal{I}, \mu)$, all ground classical formulas F, and all ground classical formulas G and H that are built without the logical connectives $\neg$ and $\leftarrow$:*

$$\boldsymbol{p}(\neg F) = 1 - \boldsymbol{p}(F) \tag{5}$$

$$\boldsymbol{p}(G \wedge H) = \min(\boldsymbol{p}(G), \boldsymbol{p}(H)) \tag{6}$$

$$\boldsymbol{p}(G \vee H) = \max(\boldsymbol{p}(G), \boldsymbol{p}(H)) \tag{7}$$

$$\boldsymbol{p}(G \leftarrow H) = \min(1, \boldsymbol{p}(G) - \boldsymbol{p}(H) + 1)\,. \tag{8}$$

The following theorem shows that Pr*-interpretations give a natural probabilistic semantics to van Emden's quantitative deduction [31] in which the implication connective is interpreted as conditional probability (note that this result implies that van Emden's quantitative deduction is an approximation of probabilistic logic programming under the conditional probability implication).

Theorem 3. *For all Pr*-interpretations $\boldsymbol{p}$, all real numbers $c \in [0,1]$, and all ground atoms $H, B_1, \ldots, B_k$ with $k \geq 0$:*

$$\boldsymbol{p}(H) \;\geq\; c \cdot \min(\boldsymbol{p}(B_1), \ldots, \boldsymbol{p}(B_k)) \;\; \textit{iff}$$
$$\boldsymbol{p}(B_1 \wedge \cdots \wedge B_k) = 0 \;\; \textit{or} \;\; \boldsymbol{p}(H \mid B_1 \wedge \cdots \wedge B_k) \;\geq\; c\,.$$

Note that for $\boldsymbol{p}(B_1 \wedge \cdots \wedge B_k) > 0$, the expression $\boldsymbol{p}(H \mid B_1 \wedge \cdots \wedge B_k)$ is defined as $\boldsymbol{p}(H \wedge B_1 \wedge \cdots \wedge B_k) \,/\, \boldsymbol{p}(B_1 \wedge \cdots \wedge B_k)$. Note also that for $k = 0$, we naturally define both $\min(\boldsymbol{p}(B_1), \ldots, \boldsymbol{p}(B_k))$ and $\boldsymbol{p}(B_1 \wedge \cdots \wedge B_k)$ as 1.

Finally, we show that Pr*-interpretations are already uniquely determined by the truth values they give to all ground atoms:

Theorem 4. *Let $\boldsymbol{p} = (\mathcal{I}, \mu)$ be a Pr*-interpretation with $\mu(I) > 0$ for all $I \in \mathcal{I}$. Then $\boldsymbol{p}$ is uniquely determined by all pairs $(A, \boldsymbol{p}(A))$ with $A \in HB_\Phi$.*

Hence, Pr*-interpretations can be identified with mappings from HB_Φ to the real interval $[0,1]$. Since such mappings can also be viewed as fuzzy sets, we get the following natural subset relation on Pr*-interpretations. For Pr*-interpretations $\boldsymbol{p}$ and $\boldsymbol{q}$, we say $\boldsymbol{p}$ is a *subset* of $\boldsymbol{q}$, denoted $\boldsymbol{p} \subseteq \boldsymbol{q}$, iff $\boldsymbol{p}(A) \leq \boldsymbol{q}(A)$ for all $A \in HB_\Phi$. We use $\boldsymbol{p} \subset \boldsymbol{q}$ as an abbreviation for $\boldsymbol{p} \subseteq \boldsymbol{q}$ and $\boldsymbol{p} \neq \boldsymbol{q}$.

For sets of probabilistic formulas $\mathcal{P}$ and probabilistic formulas $\mathsf{prob}(F) \geq c$, we write $\mathcal{P} \models^\star_{tight} \mathsf{prob}(F) \geq c$, iff c is the infimum of $\boldsymbol{p}_\sigma(F)$ subject to all Pr*-interpretations $\boldsymbol{p}$ that are models of $\mathcal{P}$ and all variable assignments σ.

3 Many-Valued Disjunctive Logic Programs

We are now ready to define many-valued disjunctive logic programs. We start by defining many-valued disjunctive logic program clauses, which are special probabilistic formulas that are interpreted under Pr*-interpretations:

A *many-valued disjunctive logic program clause* (or simply *clause*) is a probabilistic formula of the following kind:

$$\mathsf{prob}(A_1 \vee \cdots \vee A_l \leftarrow B_1 \wedge \cdots \wedge B_m \wedge \neg C_1 \wedge \cdots \wedge \neg C_n) \geq c\,,$$

where $A_1, \ldots, A_l, B_1, \ldots, B_m, C_1, \ldots, C_n$ are atoms, $l, m, n \geq 0$, and $c \in [0,1]$ is rational. It is abbreviated by $(A_1 \vee \cdots \vee A_l \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n)[c,1]$. We call $A_1 \vee \cdots \vee A_l$ its *head*, $B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n$ its *body*, and c its *truth value*. Such a clause is called an *integrity clause* iff $l = 0$, a *fact* iff $l > 0$ and $m + n = 0$, and a *rule* iff $l > 0$ and $m + n > 0$. A *many-valued disjunctive logic program* (or simply *program*) is a finite set of clauses.

Given a program P, we identify Φ with the vocabulary $\Phi(P)$ that consists of all the function and predicate symbols in P. We use HB_P to denote the Herbrand base over $\Phi(P)$. We use $ground(P)$ to denote the set of all ground instances of clauses from P with respect to $\Phi(P)$.

Given a program P, we do not need all the real numbers in $[0,1]$ to characterize the semantics of P. More precisely, the least set of equidistant rational numbers from $[0,1]$ that contains 0, 1, and all the rational numbers occurring in P is sufficient (see Theorem 7). Hence, we define the *set of truth values* of P, denoted $TV(P)$, as the least set of rational numbers $\{\frac{0}{n-1}, \frac{1}{n-1}, \ldots, \frac{n-1}{n-1}\}$, where $n \geq 2$ is a natural number, that contains all the rational numbers occurring in P. The program P is *n-valued* iff $|TV(P)| = n$.

Crucially, the truth value of all ground clauses under Pr*-interpretations is truth-functionally defined on the truth values of their ground atoms:

Theorem 5. *A ground clause* $(A_1 \vee \cdots \vee A_l \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n)[c,1]$ *is true in a Pr*-interpretation* $\boldsymbol{p}$ *iff the following condition holds:*

$$\max(\boldsymbol{p}(A_1), \ldots, \boldsymbol{p}(A_l), \boldsymbol{p}(C_1), \ldots, \boldsymbol{p}(C_n)) \;\geq\; c - 1 + \min(\boldsymbol{p}(B_1), \ldots, \boldsymbol{p}(B_m)) \,.$$

Note that the maximum and the minimum of an empty list of arguments are canonically defined as 0 *and* 1*, respectively.*

We finally define queries and their correct and tight answers:

A *many-valued query* (or simply *query*) is an expression $\exists(F)[t,1]$, where F is a ground classical formula and t is a variable or a rational number from $[0,1]$. We call the query $\exists(G)[t,1]$ a *positive* query and the query $\exists(\neg G)[t,1]$ a *negative* query if G is built without the logical connectives $\neg$ and $\leftarrow$.

Given the queries $\exists(F)[c,1]$ and $\exists(F)[x,1]$ to a program P, where $c \in [0,1]$ and $x \in \mathcal{X}$, we define their desired semantics in terms of correct and tight answers with respect to a set $\boldsymbol{M}(P)$ of models of P as follows.

The *correct answer* for $\exists(F)[c,1]$ to P *under* $\boldsymbol{M}(P)$ is Yes if $c \leq \inf\{\boldsymbol{p}(F) \mid \boldsymbol{p} \in \boldsymbol{M}(P)\}$ and No otherwise. The *tight answer* for $\exists(F)[x,1]$ to P *under* $\boldsymbol{M}(P)$ is the substitution $\theta = \{x/d\}$, where $d = \inf\{\boldsymbol{p}(F) \mid \boldsymbol{p} \in \boldsymbol{M}(P)\}$.

Many-valued query processing generalizes the classical cautious inference:

Theorem 6. *Let P be a 2-valued program and let $\boldsymbol{M}(P)$ be a set of models $(\mathcal{I}, \mu)$ of P with $\mu(\mathcal{I}) \subseteq \{0,1\}$. The correct answer for the query $\exists(F)[1,1]$ to P under $\boldsymbol{M}(P)$ is* Yes *iff F is true in all models from classical$(\boldsymbol{M}(P))$.*

4 Example

Assume that we have the following knowledge about roads and the reachability of places through roads: The probability that the road r is closed or that the road s is closed is greater than 0.5. The probability that r connects the place a with the place b is greater than 0.8. The probability that s connects b with c is greater than 0.7. The probability that we can reach Y through X if there is a

road from X to Y that is not closed is greater than 0.9. The probability that we can reach Z through X if we can reach Z through Y and Y through X is greater than 0.9. This knowledge can be expressed by the following program P (r, s, a, b, and c are constant symbols and R, X, Y, and Z are variables):

$$\begin{aligned} P = \{&(closed(r) \vee closed(s) \leftarrow)[0.5, 1], \\ &(road(r, a, b) \leftarrow)[0.8, 1],\ (road(s, b, c) \leftarrow)[0.7, 1], \\ &(reach(X, Y) \leftarrow road(R, X, Y), \neg closed(R))[0.9, 1], \\ &(reach(X, Z) \leftarrow reach(X, Y), reach(Y, Z))[0.9, 1]\}\,. \end{aligned}$$

We may ask for the tight lower bound of the probability that we can reach c through a. This can be expressed by the query $\exists(reach(a, c))[U, 1]$, where U is a variable. To give its tight answer, we must specify a set of models of P.

Some models $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, $\boldsymbol{p}_3$, and $\boldsymbol{p}_4$ of P are shown in Table 1 (we assume that $\boldsymbol{p}_i(A) = 0$ for all ground atoms A that are not mentioned). The tight answer for $\exists(reach(a, c))[U, 1]$ to P under $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \boldsymbol{p}_4\}$ is given by $\{U/0\}$, whereas the tight answer for $\exists(reach(a, c))[U, 1]$ to P under $\{\boldsymbol{p}_1, \boldsymbol{p}_2\}$ is given by $\{U/0.5\}$. Hence, as far as the query $\exists(reach(a, c))[U, 1]$ is concerned, $\{\boldsymbol{p}_1, \boldsymbol{p}_2\}$ seems to describe the intended meaning of P better than $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \boldsymbol{p}_4\}$.

Table 1. Some models of the program P

	$closed(r)$	$closed(s)$	$road(r, a, b)$	$road(s, b, c)$	$reach(a, b)$	$reach(b, c)$	$reach(a, c)$
$\boldsymbol{p}_1$	0.5	0	0.8	0.7	0.7	0.6	0.5
$\boldsymbol{p}_2$	0	0.5	0.8	0.7	0.7	0.6	0.5
$\boldsymbol{p}_3$	0	0.6	0.8	0.7	0.7	0	0
$\boldsymbol{p}_4$	0	0.7	0.8	0.7	0	0	0

The models $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, $\boldsymbol{p}_3$, and $\boldsymbol{p}_4$ are some *minimal models* of P (with respect to the subset relationship defined in Section 2.2), whereas the models $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are the only *perfect* and *stable models* of the *locally stratified* program P.

We will introduce these notions in the following section.

5 Model Semantics

In this section, we define minimal, perfect, and stable models of many-valued disjunctive logic programs, and we discuss some of their properties.

5.1 Minimal Models

We now define minimal models of many-valued disjunctive logic programs.

A model $\boldsymbol{p}$ of a program P is a *minimal model* of P iff no model of P is a proper subset of $\boldsymbol{p}$. $\boldsymbol{MM}(P)$ denotes the set of all minimal models of P.

Crucially, as far as minimal models of a program P are concerned, we can restrict our attention to the finite number of truth values in $TV(P)$:

Theorem 7. *All minimal models of a program P map into $TV(P)$.*

Given a positive query to a program P, the tight answer under $\boldsymbol{MM}(P)$ describes a tight logical consequence under Pr*-interpretations. Moreover, it *approximates* a tight logical consequence under Pr-interpretations. That is, inference in Pr*-interpretations is an approximation of inference in Pr-interpretations.

Theorem 8. *Let P be a program.*

a) The tight answer for a positive query $\exists(G)[x,1]$ to P under $\boldsymbol{MM}(P)$ is given by $\{x/d\}$, where d such that $P \models^\star_{tight} \mathsf{prob}(G) \geq d$.

b) If the tight answer for a positive query $\exists(G)[x,1]$ to P under $\boldsymbol{MM}(P)$ is given by $\{x/d\}$, then $[0,d]$ contains the unique c with $P \models_{tight} \mathsf{prob}(G) \geq c$.

Finally, many-valued minimal models generalize classical minimal models:

Theorem 9. *Let P be a 2-valued program. The set classical$(\boldsymbol{MM}(P))$ coincides with the set of all minimal models of classical(P).*

5.2 Perfect Models

We now extend the notion of perfect models [28] to many-valued disjunctive logic programs. For this purpose, we must first define a priority relation on ground atoms and a preference relation on Pr*-interpretations.

The priority relation on ground atoms is simply defined like in [28]: For a program P, the priority relation $\prec$ and the auxiliary relation $\preceq$ are the least binary relations on HB_P with the following properties. If $ground(P)$ contains a clause with the atom A in the head and the negative literal $\neg C$ in the body, then $A \prec C$. If $ground(P)$ contains a clause with the atom A in the head and the positive literal B in the body, then $A \preceq B$. If $ground(P)$ contains a clause with the atoms A and A' in the head, then $A \preceq A'$. If $A \prec B$, then $A \preceq B$. If $A \preceq B$ and $B \preceq C$, then $A \preceq C$. If $A \preceq B$ and $B \prec C$, then $A \prec C$. If $A \prec B$ and $B \preceq C$, then $A \prec C$. We say that the ground atom B *has higher priority than* the ground atom A iff $A \prec B$. The preference relation on Pr*-interpretations is defined as follows. For Pr*-interpretations $\boldsymbol{p}$ and $\boldsymbol{q}$, we say $\boldsymbol{p}$ is *preferable* to $\boldsymbol{q}$, denoted $\boldsymbol{p} \ll \boldsymbol{q}$, iff $\boldsymbol{p} \neq \boldsymbol{q}$ and for each $A \in HB_P$ with $\boldsymbol{p}(A) > \boldsymbol{q}(A)$ there is some $B \in HB_P$ with $\boldsymbol{q}(B) > \boldsymbol{p}(B)$ and $A \prec B$. We write $\boldsymbol{p} \lll \boldsymbol{q}$ iff $\boldsymbol{p} \ll \boldsymbol{q}$ or $\boldsymbol{p} = \boldsymbol{q}$.

We are now ready to define perfect models.

A model $\boldsymbol{q}$ of a program P is a *perfect model* of P iff no model of P is preferable to $\boldsymbol{q}$. We use $\boldsymbol{PM}(P)$ to denote the set of all perfect models of P.

Every many-valued perfect model is a minimal model:

Theorem 10. *Every perfect model of a program P is a minimal model of P.*

Many-valued perfect models generalize classical perfect models:

Theorem 11. *Let P be a 2-valued program. The set classical($\boldsymbol{PM}(P)$) coincides with the set of all perfect models of classical(P).*

5.3 Perfect Models under Local Stratification

We now concentrate on perfect models of locally stratified programs.

Locally stratified classical disjunctive logic programs without integrity clauses always have a perfect model [28]. We now show that the same holds for locally stratified many-valued disjunctive logic programs without integrity clauses.

A program P without integrity clauses is *locally stratified* iff HB_P can be partitioned into sets $H_1, H_2, \ldots$ (called *strata*) such that for each clause

$$(A_1 \vee \cdots \vee A_l \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n)[c, 1] \in \mathit{ground}(P)\,,$$

there exists an $i \geq 1$ such that all $A_1, \ldots, A_l$ belong to H_i, all $B_1, \ldots, B_m$ belong to $H_1 \cup \cdots \cup H_i$, and all $C_1, \ldots, C_n$ belong to $H_1 \cup \cdots \cup H_{i-1}$.

Given such a partition $H_1, H_2, \ldots$ of HB_P (which is called a *local stratification* of P) with $i \geq 1$, we use P_i to denote the set of all clauses from $\mathit{ground}(P)$ whose heads belong to H_i. Moreover, we define $H_i^\star = H_1 \cup \cdots \cup H_i$, $P_i^\star = P_1 \cup \cdots \cup P_i$, and $\boldsymbol{h}_i^\star = \boldsymbol{HB}_P|H_i^\star$, where $\boldsymbol{HB}_P = \{(A, 1) \mid A \in HB_P\}$.

Every model of a locally stratified program is subsumed by a perfect model:

Theorem 12. *For every model $\boldsymbol{q}$ of a locally stratified program P, there exists a perfect model $\boldsymbol{p}$ of P such that $\boldsymbol{p} \ll\!= \boldsymbol{q}$.*

The next theorem shows that each perfect model of a locally stratified program has a natural characterization by iterative minimal models.

Theorem 13. *Let P be a program and let $H_1, H_2, \ldots$ be a local stratification of P. The $Pr^\star$-interpretation $\boldsymbol{q}$ is a perfect model of P iff*

1. *the $Pr^\star$-interpretation $\boldsymbol{q}|H_1$ is a minimal model of P_1 and*
2. *for all $i \geq 2$, the $Pr^\star$-interpretation $\boldsymbol{q}|H_i^\star$ is a minimal element in the set of all models $\boldsymbol{o} \subseteq \boldsymbol{h}_i^\star$ of P_i with $\boldsymbol{o}|H_{i-1}^\star = \boldsymbol{q}|H_{i-1}^\star$.*

Finally, the following theorem shows that locally stratified programs without disjunction always have a unique perfect model.

Theorem 14. *Every disjunction-free locally stratified program P has a unique perfect model $\boldsymbol{p}$ such that $\boldsymbol{p} \ll\!= \boldsymbol{q}$ for all models $\boldsymbol{q}$ of P.*

5.4 Stable Models

We now extend the notion of stable models [29] to many-valued disjunctive logic programs. For this purpose, we must slightly generalize clauses as follows.

An *extended many-valued disjunctive logic program clause* (or simply *extended clause*) is an expression of the following kind:

$$(A_1 \vee \cdots \vee A_l\,;\, d \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n)[c, 1]\,,$$

where $A_1, \ldots, A_l, B_1, \ldots, B_m, C_1, \ldots, C_n$ are atoms, $l, m, n \geq 0$, c is a rational number from $[0, 1]$, and d is a real number from $[0, 1]$. It *is true* in a Pr*-interpretation $\boldsymbol{p}$ under a variable assignment σ iff $\max(\boldsymbol{p}_\sigma(A_1), \ldots, \boldsymbol{p}_\sigma(A_l), \boldsymbol{p}_\sigma(C_1), \ldots, \boldsymbol{p}_\sigma(C_n), d) \geq c - 1 + \min(\boldsymbol{p}_\sigma(B_1), \ldots, \boldsymbol{p}_\sigma(B_m))$.

We next generalize the classical Gelfond-Lifschitz transformation:

For a program P and a Pr*-interpretation $\boldsymbol{q}$, the expression $P/\boldsymbol{q}$ denotes the set of extended clauses that is obtained from $ground(P)$ by replacing every clause $(A_1 \vee \cdots \vee A_l \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_n)[c, 1]$ by the extended clause

$$(A_1 \vee \cdots \vee A_l\,;\, \max(\boldsymbol{q}(C_1), \ldots, \boldsymbol{q}(C_n)) \leftarrow B_1, \ldots, B_m)[c, 1]\,.$$

We are now ready to define stable models as follows.

A Pr*-interpretation $\boldsymbol{q}$ is a *stable model* of a program P iff $\boldsymbol{q}$ is a minimal model of $P/\boldsymbol{q}$. We use $\boldsymbol{SM}(P)$ to denote the set of all stable models of P.

Every stable model is also a minimal model:

Theorem 15. *Every stable model of a program P is a minimal model of P.*

The next theorem shows that for locally stratified programs, the notion of stable models coincides with the notion of perfect models.

Theorem 16. *The set of stable models of a locally stratified program P coincides with the set of perfect models of P.*

Many-valued stable models generalize classical stable models:

Theorem 17. *Let P be a 2-valued program. The set $classical(\boldsymbol{SM}(P))$ coincides with the set of all stable models of $classical(P)$.*

6 Computational Complexity

We now show that some decision problems related to many-valued disjunctive logic programs have the same complexity like their classical counterparts [6].

We first concentrate on the problems of deciding whether a ground program has a minimal, perfect, or stable model.

Theorem 18. *a) The problem of deciding whether a ground program P has a minimal model is NP-complete. b) The problem of deciding whether a ground program P has a perfect model is Σ_2^P-complete. c) The problem of deciding whether a ground program P has a stable model is Σ_2^P-complete.*

We next focus on some decision problems related to propositional query processing under minimal, perfect, and stable model semantics.

Theorem 19. *The problem of deciding whether* Yes *is the correct answer for a ground positive or negative query* $\exists(F)[c,1]$ *to a ground program* P *under* $\boldsymbol{M}(P)$ *is* Π_2^P*-complete for every* $\boldsymbol{M}(P)$ *among* $\boldsymbol{MM}(P)$, $\boldsymbol{PM}(P)$, *and* $\boldsymbol{SM}(P)$.

7 Summary and Outlook

We presented many-valued disjunctive logic programs with probabilistic semantics in which classical disjunctive logic program clauses are extended by a truth value that respects the material implication. We showed that they have a natural minimal, perfect, and stable model semantics, which generalize the minimal, perfect, and stable model semantics of classical disjunctive logic programs. We also showed that some decision problems related to ground many-valued disjunctive logic programs under minimal, perfect, and stable model semantics have the same computational complexity like their classical counterparts.

An interesting topic of future research is to explore other semantics of nonmonotonic negation in many-valued disjunctive logic programs. Moreover, it would be very interesting to elaborate a fixpoint semantics and a proof theory for many-valued disjunctive logic programs.

Acknowledgments

I am very grateful to Thomas Eiter, Georg Gottlob, Nicola Leone, and Cristinel Mateis for useful discussions.

Some of this work was done while I was supported by a DFG grant.

References

1. K. R. Apt. Logic programming. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 10, pages 493–574. MIT Press, 1990.
2. F. Bacchus, A. Grove, J. Y. Halpern, and D. Koller. From statistical knowledge bases to degrees of beliefs. *Artif. Intell.*, 87:75–143, 1996.
3. J. F. Baldwin. Evidential support logic programming. *Fuzzy Sets Syst.*, 24:1–26, 1987.
4. A. Dekhtyar and V. S. Subrahmanian. Hybrid probabilistic programs. In *Proc. of the 14th International Conference on Logic Programming*, pages 391–405, 1997.
5. T. Eiter and G. Gottlob. Complexity aspects of various semantics for disjunctive databases. In *Proc. of the 12th ACM Symposium on Principles of Database Systems*, pages 158–167. ACM Press, 1993.
6. T. Eiter and G. Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Ann. Math. Artif. Intell.*, 15:289–323, 1995.
7. R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Inf. Comput.*, 87:78–128, 1990.

8. M. Fitting. Bilattices and the semantics of logic programming. *J. Logic Program.*, 11(1–2):91–116, 1991.
9. J. Y. Halpern. An analysis of first-order logics of probability. *Artif. Intell.*, 46:311–350, 1990.
10. M. Kifer and V. S. Subrahmanian. Theory of generalized annotated logic programming and its applications. *J. Logic Program.*, 12(3–4):335–367, 1992.
11. J.-L. Lassez and M. J. Maher. Optimal fixedpoints of logic programs. *Theor. Comput. Sci.*, 39:15–25, 1985.
12. J. W. Lloyd. *Foundations of Logic Programming.* Springer, Berlin, 2nd ed., 1987.
13. J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming.* MIT Press, Cambridge, MA, 1992.
14. T. Lukasiewicz. Probabilistic logic programming. In *Proc. of the 13th Biennial European Conf. on Artificial Intelligence*, pages 388–392. J. Wiley & Sons, 1998.
15. T. Lukasiewicz. Local probabilistic deduction from taxonomic and probabilistic knowledge-bases over conjunctive events. *Int. J. Approx. Reas.*, 21(1):23–61, 1999.
16. T. Lukasiewicz. Many-valued disjunctive logic programs with probabilistic semantics. Technical Report 1843-99-09, Institut für Informationssysteme, Technische Universität Wien, 1999. ftp://ftp.kr.tuwien.ac.at/pub/tr/rr9909.ps.gz.
17. T. Lukasiewicz. Many-valued first-order logics with probabilistic semantics. In *Proc. of the Annual Conference of the European Association for Computer Science Logic*, 1998, volume 1584 of *LNCS*, pages 415–429. Springer, 1999.
18. T. Lukasiewicz. Probabilistic and truth-functional many-valued logic programming. In *Proc. of the 29th IEEE International Symposium on Multiple-Valued Logic*, pages 236–241, 1999.
19. T. Lukasiewicz. Probabilistic deduction with conditional constraints over basic events. *J. Artif. Intell. Res.*, 10:199–241, 1999.
20. C. Mateis. *A Quantitative Extension of Disjunctive Logic Programming.* Doctoral Dissertation, Technische Universität Wien, 1998.
21. J. Minker. Overview of disjunctive logic programming. *Ann. Math. Artif. Intell.*, 12:1–24, 1994.
22. R. T. Ng. Semantics, consistency, and query processing of empirical deductive databases. *IEEE Trans. Knowl. Data Eng.*, 9(1):32–49, 1997.
23. R. T. Ng and V. S. Subrahmanian. A semantical framework for supporting subjective and conditional probabilities in deductive databases. *J. Autom. Reasoning*, 10(2):191–235, 1993.
24. R. T. Ng and V. S. Subrahmanian. Stable semantics for probabilistic deductive databases. *Inf. Comput.*, 110:42–83, 1994.
25. L. Ngo. Probabilistic disjunctive logic programming. In *Proc. of the 12th Conf. on Uncertainty in Artificial Intelligence*, pages 397–404. Morgan Kaufmann, 1996.
26. N. J. Nilsson. Probabilistic logic. *Artif. Intell.*, 28:71–88, 1986.
27. D. Poole. Probabilistic Horn abduction and Bayesian networks. *Artif. Intell.*, 64:81–129, 1993.
28. T. C. Przymusinski. On the declarative semantics of stratified deductive databases and logic programs. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 193–216. Morgan Kaufmann, 1988.
29. T. C. Przymusinski. Stable semantics for disjunctive programs. *New Generation Comput.*, 9:401–424, 1991.
30. N. Rescher. *Many-valued Logic.* McGraw-Hill, New York, 1969.
31. M. H. van Emden. Quantitative deduction and its fixpoint theory. *J. Logic Program.*, 3(1):37–53, 1986.