Extending Disjunctive Logic Programming by T-norms*

Cristinel Mateis

Information Systems Department, TU Vienna
A-1040 Vienna, Austria

email address: mateis@dbai.tuwien.ac.at

Abstract. This paper proposes a new knowledge representation language, called QDLP, which extends DLP to deal with uncertain values. A certainty degree interval (a subinterval of $[0, 1]$) is assigned to each (quantitative) rule. Triangular norms (T-norms) are employed to define calculi for propagating uncertainty information from the premises to the conclusion of a quantitative rule. Negation is considered and the concept of stable model is extended to QDLP. Different T-norms induce different semantics for one given quantitative program. In this sense, QDLP is parameterized and each choice of a T-norm induces a different QDLP language. Each T-norm is eligible for events with determinate relationships (e.g., independence, exclusiveness) between them. Since there are infinitely many T-norms, it turns out that there is a family of infinitely many QDLP languages. This family is carefully studied and the set of QDLP languages which generalize traditional DLP is precisely singled out. Finally, the complexity of the main decisional problems arising in the context of QDLP (i.e., Model Checking, Stable Model Checking, Consistency, and Brave Reasoning) is analyzed. It is shown that the complexity of the relevant fragments of QDLP coincides exactly with the complexity of DLP. That is, reasoning with uncertain values is more general and not harder than reasoning with boolean values.

1 Introduction

Disjunctive logic programs are logic programs where disjunction is allowed in the heads of the rules and negation may occur in the bodies of the rules. Such programs are nowadays widely recognized as a valuable tool for knowledge representation and commonsense reasoning [3, 16, 22]. An important merit of disjunctive logic programming (DLP) is its capability to model incomplete knowledge [3, 22]. DLP has a very high expressive power. In [14] it is proved that, under stable model semantics, disjunctive programs capture the complexity class Σ_2^P, that is, they allow us to express every property which is decidable in non-deterministic polynomial time with an oracle in NP. Thus, DLP can express real world situations that cannot be represented by disjunction-free programs.

However, real-life applications often need to deal with uncertain information and quantitative data which cannot be represented in DLP. The usual logical reasoning in terms of the truth values **true** and **false** are insufficient for the purposes of several real-life applications. Image databases, sensor data, temporal indeterminacy, information

* Work partially supported by the Austrian Science Fund (FWF) under grants N Z29-INF and P12344-INF.

retrieval are only a few of the domains where uncertainty occurs [20]. Consider for instance a robot which moves and changes direction according to a prefixed route and to the coordinates received from a sensor. Since sensor data may be subject to error and sensors may have different reliability, a formalism able to deal with uncertain information is needed to encode the control mechanism of the robot. (See section 4 for the example on this subject.)

Many frameworks for multivalued logic programming have been proposed to handle uncertain information. There is a split in the AI community between (i) those who attempt to deal with uncertainty using non-numerical techniques [8, 9, 12], (ii) those who use numerical representations of uncertainty but, believing that probability calculus is inadequate for the task, invent entirely new calculi, such as Dempster-Shafer calculus [10, 17, 32], fuzzy logic [6, 7, 15, 18, 19, 33, 34], and (iii) those who remain within the traditional framework of probability theory, while attempting to equip the theory with computational facilities needed to perform AI tasks [2, 24–27, 29, 30].

We propose an approach to define the representation, inference, and control of uncertain information in the framework of DLP which is closely related to the second of the above categories.

The main contributions of the paper are the following.

- We define a new knowledge representation language, called *Quantitative Disjunctive Logic Programming* (QDLP), extending DLP to deal with uncertain values.
- We define a mechanism of reasoning with uncertainty through rule chaining by using the well-studied and mathematically clean notion of T-norm. In particular, we consider a p-parameterized family of T-norms. Each T-norm is eligible for events with determinate relationships (e.g., independence, exclusiveness) between them. Different T-norms induce different semantics for one given quantitative program. Thus, QDLP is parameterized and each choice of a T-norm induces a different QDLP language. There are infinitely many T-norms, hence there are infinitely many QDLP languages. Importantly, the T-norm may be chosen according to the level of knowledge of the relationships between the atoms (events) of the program.
- We single out precisely the fragments from the QDLP family which are generalizations of DLP. Basically, a fragment QF of QDLP induced by a T-norm $T^{(p)}$, $p \in [-\infty, +\infty]$, is a generalization of DLP iff to each program P from DLP corresponds a program QP in QF such that the set of all stable models of P is exactly the set of all stable models of QP under the semantics induced by $T^{(p)}$.
- We show that the Quantitative Logic Programming Language proposed by van Emden in [34] coincides with the disjunction-free fragment of QDLP induced by the T-norm T_3.
- We analyze the complexity of the main decisional problems arising in QDLP. We classify precisely (i.e., by completeness results) the complexity of all relevant fragments of QDLP (i.e., of the QDLP languages which truly generalize DLP) for the T-norm T_3. Importantly, the addition of uncertainty does not cause any computational overhead, as the complexity of QDLP is exactly the same as the complexity of DLP. In other words, uncertainty comes for free!

For space limitation, we omit the proofs of the results reported in section 6.2 and 7. The proofs of all results along with further material and details are reported in the long version of the paper [23] which can be retrieved from the mentioned web address.

2 Preliminaries: Triangular Norms and Conorms

The triangular norms (T-norms) and conorms (T-conorms) form the basis for the various uncertainty calculi discussed in this paper. We will denote a T-norm by T and a T-conorm by S. One of the advantages of these operators is their low computational complexity.

The T-norms and T-conorms are functions $T, S : [0,1] \times [0,1] \rightarrow [0,1]$ which satisfy the following properties:

$T(a,0) = T(0,a) = 0$	$S(1,a) = S(a,1) = 1$	$[boundary]$
$T(a,1) = T(1,a) = a$	$S(0,a) = S(a,0) = a$	$[boundary]$
$T(a,b) \leq T(c,d)$	$S(a,b) \leq S(c,d)$ if $a \leq c,\ b \leq d$	$[monotonicity]$
$T(a,b) = T(b,a)$	$S(a,b) = S(b,a)$	$[commutativity]$
$T(a,T(b,c)) = T(T(a,b),c)$	$S(a,S(b,c)) = S(S(a,b),c)$	$[associativity]$

Intuitively, $T(a,b)$ (resp., $S(a,b)$) assigns a certainty value to the composition of two events e_1 and e_2 whose certainty values are a and b. Usually, the composition of e_1 and e_2 is the conjunction (resp., disjunction) under certain conditions (e.g., independence, mutual exclusiveness).

Although defined as two-place functions, the T-norms and T-conorms can be used to represent the composition of a larger number of events. Because of the associativity property, it is possible to define recursively $T(x_1, \ldots, x_n, x_{n+1})$ and $S(x_1, \ldots, x_n, x_{n+1})$ for $x_1, \ldots, x_{n+1} \in [0,1]$ as:

$$T(x_1, \ldots, x_n, x_{n+1}) = T(T(x_1, \ldots, x_n), x_{n+1})$$
$$S(x_1, \ldots, x_n, x_{n+1}) = S(S(x_1, \ldots, x_n), x_{n+1})$$

Some typical T-norms and T-conorms are the following:

$T_0(a,b) = \begin{cases} min(a,b) & \text{if } max(a,b) = 1 \\ 0 & \text{otherwise} \end{cases}$	$S_0(a,b) = \begin{cases} max(a,b) & \text{if } min(a,b) = 0 \\ 1 & \text{otherwise} \end{cases}$
$T_1(a,b) = max(0, a+b-1)$	$S_1(a,b) = min(1, a+b)$
$T_{1.5}(a,b) = max(0, \sqrt{a} + \sqrt{b} - 1)^2$	$S_{1.5}(a,b) = 1 - max(0, \sqrt{1-a} + \sqrt{1-b} - 1)^2$
$T_2(a,b) = ab$	$S_2(a,b) = a + b - ab$
$T_{2.5}(a,b) = \frac{ab}{a+b-ab}$	$S_{2.5}(a,b) = \frac{a+b-2ab}{1-ab}$
$T_3(a,b) = min(a,b)$	$S_3(a,b) = max(a,b)$

It is important to note that

$$T_0 \leq T_1 \leq T_{1.5} \leq T_2 \leq T_{2.5} \leq T_3$$
$$S_3 \leq S_{2.5} \leq S_2 \leq S_{1.5} \leq S_1 \leq S_0$$

T_1 is appropriate to perform the intersection of lower probability bounds (uncertainty values) and captures the notion of the worst case, where the arguments are considered

as mutually exclusive as possible. T_3 is appropriate to represent the intersection of upper probability bounds and captures the notion of the best case, where one argument attempts to subsume the others. T_2 is the classical probabilistic operator that assumes independence of arguments and its dual T-conorm S_2 is the usual additive measure for the union.

Schweizer and Sklar [31] proposed a parameterized family, denoted by $T(a, b, p)$, where a and b are the T-norm's arguments and p is the parameter that spans the space of T-norms from T_0 to T_3:

$$T(a,b,p) = \begin{cases} (a^{-p} + b^{-p} - 1)^{-\frac{1}{p}} & \text{if } a^{-p} + b^{-p} \geq 1 \text{ when } p < 0 \\ 0 & \text{if } a^{-p} + b^{-p} \leq 1 \text{ when } p < 0 \\ lim_{p \to 0} T(a,b,p) = ab & \text{when } p \to 0 \\ (a^{-p} + b^{-p} - 1)^{-\frac{1}{p}} & \text{when } p > 0 \end{cases}$$

Let $\overline{\mathbf{R}} = [-\infty, +\infty]$ and $\overline{\mathbf{R}^+} = [0, +\infty]$. Given a real number $p \in \overline{\mathbf{R}}$, we denote by $T^{(p)}$ the member of the family of T-norms induced by p. Note that we allow p to be assigned the infinite values $-\infty$ and $+\infty$. Figure 1 illustrates how $T^{(p)}$ spans over the real numbers, so for example $T^{(-\infty)} = T_0$, $T^{(-1)} = T_1$, $T^{(0)} = T_2$, and $T^{(+\infty)} = T_3$.

p	$-\infty$	-1	-0.5	0	1	$+\infty$
$T(a,b,p)$	T_0	T_1	$T_{1.5}$	T_2	$T_{2.5}$	T_3

Fig. 1. Spanning of the T-norms over the real numbers

For suitable negation operators $N(a)$, such as $N(a) = 1 - a$, T-norms and T-conorms are duals in the sense of the following generalization of DeMorgan's law:

$$S(a,b) = N(T(N(a), N(b)))$$
$$T(a,b) = N(S(N(a), N(b)))$$

This duality implies that given the negation operator $N(a) = 1 - a$, the selection of a T-norm uniquely constrains the selection of the T-conorm.

The dual parameterized family of T-conorm, denoted by $S(a, b, p)$ is defined as $S(a, b, p) = 1 - T(1 - a, 1 - b, p)$. Given a real number $p \in \overline{\mathbf{R}}$ we denote by $S^{(p)}$ the member of the family of T-conorms induced by p. So for example $S^{(-\infty)} = S_0$, $S^{(-1)} = S_1$, $S^{(0)} = S_2$, and $S^{(+\infty)} = S_3$.

Theorem 1. *The evaluation of the T-norms and T-conorms at the extremes of the unity interval $[0, 1]$ satisfies the truth tables of the logical operators AND and OR, respectively.* □

3 Syntax of QDLP

A $term$ is either a constant or a variable[1]. An $atom$ is $a(t_1, ..., t_n)$, where a is a $predicate$ of arity n and $t_1, ..., t_n$ are terms. A $literal$ is either a $positive\ literal$ p or a $negative\ literal$ $\neg p$, where p is an atom.

[1] Note that function symbols are not considered in this work.

A *positive (disjunctive) quantitative rule* r is a clause of the form:

$$h_1 \vee \cdots \vee h_n \overset{[x,y]}{\longleftarrow} b_1, \cdots, b_k, \qquad n \geq 1,\ \ k \geq 0$$

where $h_1, \cdots, h_n, b_1, \cdots, b_k$ are atoms and $0 < x \leq y \leq 1$. The interval $[x, y]$ is the certainty degree interval of the rule (i.e., the strength of the rule implication) and it is a measure of the reliability of the rule. $h_1 \vee \cdots \vee h_n$ is the head of the quantitative rule and it is a non-empty disjunction of atoms. $b_1, \cdots, b_k$ is the body of the quantitative rule and it is a (possibly empty) conjunction of atoms. If the body is empty (i.e., $k = 0$) and the head contains exactly one atom (i.e., $n = 1$), the rule is a fact whose certainty degree interval coincides with the strength of the implication.

A *positive (disjunctive) quantitative program* is a finite set of positive quantitative disjunctive rules.

4 Semantics of QDLP

Let $\mathcal{P}$ be a positive disjunctive quantitative program. The *Herbrand universe* $U_\mathcal{P}$, the *Herbrand base* $B_\mathcal{P}$, and $ground(\mathcal{P})$ of $\mathcal{P}$ are defined like in DLP.

Once we defined the syntax of quantitative rules, we need to evaluate the satisfiability of premises, to propagate uncertainty through rule chaining and to consolidate the same conclusion derived from different rules.

A *quantitative interpretation* I of $\mathcal{P}$ is a mapping which assigns to each atom $A \in B_\mathcal{P}$ a certainty degree interval $[x_A, y_A] \subseteq [0, 1]$. We write $I(A) = [x_A, y_A],\ \forall\ A \in B_\mathcal{P},\ [x_A, y_A] \subseteq [0, 1]$.

It is worth noting that a quantitative program $\mathcal{P}$ has infinitely many quantitative interpretations because each atom $A \in B_\mathcal{P}$ can be assigned infinitely many intervals $[x_A, y_A] \subseteq [0, 1]$. This is an important difference w.r.t. (function-free) DLP, where each program has always a finite number of Herbrand interpretations.

Let p be any real number inducing $T^{(p)}$ from the family of T-norms. We denote by $\mathcal{T}^{(p)}$ (resp., $\mathcal{S}^{(p)}$) the generalization of the T-norm $T^{(p)}$ (resp., T-conorm $S^{(p)}$) whose arguments are intervals instead of single values, e.g., $\mathcal{T}^{(p)}([a, b], [c, d]) = [T^{(p)}(a, c), T^{(p)}(b, d)]$.

Now that we know what a quantitative interpretation I is, the first thing to straighten out is when a rule r is true w.r.t. I and what is the role of p. To this end, we first define the way the certainty degree intervals of the atoms of a conjunction or disjunction are combined. In particular, we define

1. The certainty degree interval of a (possibly empty) conjunction C of atoms from $B_\mathcal{P}$, $C = b_1 \wedge \ldots \wedge b_m$, w.r.t. I and p:

$$I^{(p)}(C) = \begin{cases} [1, 1] & \text{if } m = 0 \quad (i.e., C = \emptyset) \\ \mathcal{T}^{(p)}(I(b_1), \ldots, I(b_m)) & \text{if } m > 0 \end{cases}$$

2. The certainty degree interval of a non-empty disjunction D of atoms from $B_\mathcal{P}$, $D = h_1 \vee \ldots \vee h_n$, w.r.t. I and p:

$$I^{(p)}(D) = \mathcal{S}^{(p)}(I(h_1), \ldots, I(h_n)).$$

Given two certainty degree intervals $[a, b]$ and $[p, q]$, then $[a, b] \leq [p, q]$ iff $a \leq p$ and $b \leq q$. Moreover, $[a, b] < [p, q]$ iff (i) $[a, b] \leq [p, q]$, and (ii) $a < p$ or $b < q$.

We say that a rule $r \in ground(\mathcal{P})$, $H(r) \overset{[x,y]}{\longleftarrow} B(r)$, is p-satisfied w.r.t. I iff the following inequality is satisfied

$$I^{(p)}(H(r)) \geq \mathcal{T}_2(I^{(p)}(B(r)), [x, y]) \tag{1}$$

The member on the right-hand side of the inequality (1) represents the certainty degree interval propagated through the rule w.r.t. I and p.

The head event $H(r)$ depends on two events: (i) the rule reliability event, expressed through $[x, y]$, and (ii) the reliability event of the body of r w.r.t. I and p, given by $I^{(p)}(B(r))$. Intuitively, we can assume that the rule reliability is independent of the certainty degree intervals of the body literals, so that the two events are to be considered independent and for this reason we use $\mathcal{T}_2$ in (1).

A quantitative p-model of $\mathcal{P}$ is a quantitative interpretation M of $\mathcal{P}$ such that each rule $r \in ground(\mathcal{P})$ is p-satisfied w.r.t. M. Since the definition of quantitative p-model relies completely on the instantiation $ground(\mathcal{P})$ of $\mathcal{P}$, for simplicity, throughout the rest of this paper, we assume that $\mathcal{P}$ is a ground program (that can be either ground originally, or it is the instantiation $ground(\mathcal{P}')$ of a program $\mathcal{P}'$). The set of all p-models of $\mathcal{P}$ is denoted by $\mathcal{M}^{(p)}(\mathcal{P})$.

As previously noted, a quantitative program $\mathcal{P}$ has infinitely many quantitative interpretations. Thus, $\mathcal{P}$ may have (infinitely) many p-models. Therefore, it is useful to define an order relation between the p-models of $\mathcal{P}$ which makes possible to prefer some p-models to others. Since a p-model assigns certainty degree intervals to all atoms in $B_\mathcal{P}$, an order relation between p-models should be defined in terms of an order relation between intervals.

Given $M_1, M_2 \in \mathcal{M}^{(p)}(\mathcal{P})$, $M_1 \leq M_2$ iff $M_1(A) \leq M_2(A)$ for each $A \in B_\mathcal{P}$. Moreover, $M_1 < M_2$ iff (i) $M_1 \leq M_2$, and (ii) $\exists A \in B_\mathcal{P}$ s.t. $M_1(A) < M_2(A)$.

We are now in a position to define what a *minimal p-model* is. A p-model $M \in \mathcal{M}^{(p)}(\mathcal{P})$ is minimal iff there is no $N \in \mathcal{M}^{(p)}(\mathcal{P})$ such that $N < M$. The minimal p-model semantics of $\mathcal{P}$ is the set of all minimal p-models of $\mathcal{P}$ and is denoted by $\mathcal{MM}^{(p)}(\mathcal{P})$.

Once we fix p, we uniquely select a T-norm and its dual T-conorm which completely describe an uncertainty calculus. That is, according to the previous definitions, once we fix p, we define a semantics for $\mathcal{P}$, called the p-semantics. In this sense, we say that the semantics of the quantitative programs is parameterized and the choice of a T-norm induces the semantics of a quantitative program. Moreover, different T-norms induce different semantics in general. Since we can fix p in infinitely many ways, we can define infinitely many semantics for $\mathcal{P}$. The T-norm may be chosen according to the level of knowledge of the relationships between the atoms of $\mathcal{P}$.

Example 1. Consider the ground program $\mathcal{P}$ consisting of the following rules

$$a \vee c \overset{[0.9,1]}{\longleftarrow} . \qquad u \overset{[0.8,0.8]}{\longleftarrow} a,\ b . \qquad w \overset{[0.5,0.6]}{\longleftarrow} u .$$
$$b \overset{[0.5,0.5]}{\longleftarrow} . \qquad v \overset{[0.4,0.8]}{\longleftarrow} b . \qquad w \overset{[1,1]}{\longleftarrow} v .$$

and the interpretations I_1, I_2 and I_3,

$$I_1 = \{a : [0.9, 1], b : [0.5, 0.5], c : [0, 0], u : [0.4, 0.4], v : [0.2, 0.4], w : [0.2, 0.4]\}$$
$$I_2 = \{a : [0.9, 1], b : [0.5, 0.5], c : [0, 0], u : [0.4, 0.6], v : [0.2, 0.4], w : [0.2, 0.4]\}$$
$$I_3 = \{a : [0.9, 1], b : [0.5, 0.5], c : [0, 0], u : [0.2, 0.5], v : [0.2, 0.4], w : [0.2, 0.4]\}$$

If $p = +\infty$ (i.e., $T^{(p)} = T_3$) then $I_1, I_2 \in \mathcal{M}^{(p)}(\mathcal{P})$. $I_3 \notin \mathcal{M}^{(p)}(\mathcal{P})$ because the rule $u \overset{[0.8,0.8]}{\longleftarrow} a, b$ is not p-satisfied w.r.t. I_3. Moreover, $I_1 < I_2$ and I_1 is minimal. □

Example 2. Consider a robot which moves and changes direction according to a prefixed route and to the coordinates received from a sensor. Sensor data is subject to error and different sensors may have different reliabilities. The control mechanism of the robot can be encoded in QDLP as follows. Consider the atoms $moveToRight$, $moveToLeft$, $moveUp$, $moveDown$, $xCoord(X)$, $yCoord(Y)$, $sensorX(X)$, and $sensorY(Y)$. At regular intervals of time, the sensors return instances of the atoms $sensorX(X)$ and $sensorY(Y)$ which are used to derive the actual coordinates according to the following quantitative rules

$$xCoord(X) \overset{[0.9,1]}{\longleftarrow} sensorX(Z), |X - Z| \leq 0.5$$
$$yCoord(Y) \overset{[0.8,1]}{\longleftarrow} sensorY(Z), |Y - Z| \leq 0.5$$

where the strength of the implication of each rule represents the reliability of the corresponding sensor in normal environment conditions (e.g., good visibility, low level of usage, etc). The built-in predicates have always the maximal reliability (i.e., $[1, 1]$). The atoms $sensorX(X)$ and $sensorY(Y)$ are assigned reliabilities according to the current environment conditions. For each turning point (x, y) of the assigned route, we define a rule like

$$atom \overset{[1,1]}{\longleftarrow} xCoord(x), yCoord(y)$$

where $atom \in \{moveToRight, moveToLeft, moveUp, moveDown\}$. The robot turns to the right when the certainty degree interval of $moveToRight$ is at least $[0.75, 1]$, and so on. □

5 QDLP with Negation

Several real world situations can be represented much more naturally if negation is allowed [21]. It is therefore necessary to define a general (disjunctive) quantitative rule r which allows negative literals in its body:

$$h_1 \vee \cdots \vee h_n \overset{[x,y]}{\longleftarrow} b_1, \cdots, b_k, \neg b_{k+1}, \cdots, \neg b_{k+m}, \qquad n \geq 1, \;\; k, m \geq 0$$

where $h_1, \cdots, h_n, b_1, \cdots, b_{k+m}$ are atoms and $0 < x \leq y \leq 1$.

We show next how the definitions of p-satisfiability and (minimal) p-model change when negative literals are allowed in the rules' bodies. Moreover, we will see that the quantitative minimal model semantics is not the natural meaning to be assigned to a negative quantitative program, and we define the quantitative stable model semantics.

We have to redefine only the relation (1) which the p-satisfiability of a positive rule depends on and take into consideration the case when the body of a rule contains also negative literals; all other definitions remain unchanged.

A natural question that arises is, given $I(A) = [x, y]$, how do we evaluate the certainty degree of the negative literal $\neg A$, that is, what is $I(\neg A)$? The answer is

$$I(\neg A) = [N(y), N(x)] = [1 - y, 1 - x]$$

where N is the negation operator $N : [0, 1] \to [0, 1]$, $N(x) = 1 - x$.

Thus, the certainty degree interval of the body of r w.r.t. I and p is given by

$$I^{(p)}(B(r)) = \begin{cases} [1, 1] & \text{if } k + m = 0 \\ \mathcal{T}^{(p)}(I(b_1), \ldots, I(b_k), I(\neg b_{k+1}), \ldots, I(\neg b_{k+m})) & \text{if } k + m > 0 \end{cases}$$

Like in DLP, the quantitative minimal model semantics is applicable also to negative quantitative programs, but it does not capture the meaning of negation by failure (i.e., CWA). We define a new semantics, called *quantitative stable model semantics*.

The quantitative stable model semantics involves the notion of *stable p-model*. Before defining this new notion, we define the extended quantitative program and the quantitative version (qGL) of the Gelfond-Lifschitz transformation (GL).

An *extended quantitative program* is a quantitative program $\mathcal{P}_e$ where subintervals of the unity interval $[0, 1]$ may occur as body atoms in the rules of $\mathcal{P}_e$ and are considered like normal atoms. It is worth noting that such atoms are not in $B_{\mathcal{P}_e}$. We assume that every quantitative interpretation I of $\mathcal{P}_e$ assigns to each atom $[x, y]$ occurring in the body of a rule the certainty degree interval $[x, y]$, that is, $I([x, y]) = [x, y]$.

Given a quantitative interpretation I for $\mathcal{P}$, the qGL-transformation $\frac{\mathcal{P}}{I}$ of $\mathcal{P}$ w.r.t. I is the positive extended quantitative program obtained from $\mathcal{P}$ by replacing in the body of every rule each negative literal $\neg B_i$ by the constant interval $I(\neg B_i)$.

Let M be a p-model of $\mathcal{P}$, for some $p \in \overline{\mathbf{R}}$. M is a *stable p-model* of $\mathcal{P}$ iff M is a minimal p-model of $\frac{\mathcal{P}}{M}$. The stable p-model semantics of $\mathcal{P}$ is the set of all stable p-models of $\mathcal{P}$ and is denoted by $\mathcal{SM}^{(p)}(\mathcal{P})$. Note that if $\mathcal{P}$ is positive then $\mathcal{MM}^{(p)}(\mathcal{P}) = \mathcal{SM}^{(p)}(\mathcal{P})$ for each $p \in \overline{\mathbf{R}}$.

Example 3. Let $\mathcal{P} = \{a \overset{[0.5,0.6]}{\longleftarrow} \neg\, b\}$, $p \in \overline{\mathbf{R}}$ (the value of p is irrelevant, since the body of the single rule of $\mathcal{P}$ contains only one literal), and the minimal p-model $M = \{a : [0.5, 0.6], b : [0, 0]\}$. Note that $\frac{\mathcal{P}}{M} = \{a \overset{[0.5,0.6]}{\longleftarrow} [1, 1]\}$ and that M is a minimal p-model of $\frac{\mathcal{P}}{M}$, hence M is a stable p-model of $\mathcal{P}$.

Consider now the minimal p-model $N = \{a : [0.25, 0.36], b : [0.4, 0.5]\}$. Thus, $\frac{\mathcal{P}}{N} = \{a \overset{[0.5,0.6]}{\longleftarrow} [0.5, 0.6]\}$ and $N' = \{a : [0.25, 0.36], b : [0, 0]\}$ is a p-model of $\frac{\mathcal{P}}{N}$. Since $N' < N$, N is not minimal for $\frac{\mathcal{P}}{N}$, hence N is not a stable p-model of $\mathcal{P}$. □

6 Generalization Results

6.1 Van Emden's Approach

One of the most relevant earlier works in this field was accomplished by van Emden in [34]. There, a quantitative rule r is of the form $A \overset{\mathrm{f}}{\longleftarrow} B_1, \ldots, B_n$, where $n \geq 0$, A,

$B_1, \ldots, B_n$ are all positive atoms, f is a real number in the interval $(0, 1]$. r is true in a quantitative interpretation I iff $I(A) \geq f \times min\{\ I(B_i) \mid i \in \{1, \ldots, n\}\ \}$.

Theorem 2. *The language proposed by van Emden is a particular case of the p-model semantics, where $p = +\infty$ (i.e., $T^{(p)} = T_3$).* □

There are important differences between our approach and that of van Emden. First of all, the programs considered in [34] are positive and without disjunction. Moreover, unlike in our approach, each clause implication receives a scalar and not an interval. Finally, van Emden defines a unique uncertainty calculus, based on the T-norm T_3.

6.2 Traditional Disjunctive Logic Programming

From the syntax point of view, QDLP is an extension of DLP. Each $\mathcal{P}$ in DLP can be transformed in a program $\mathcal{P}'$ in QDLP, called the quantitative version of $\mathcal{P}$, by assigning $[1, 1]$ to the strength of the implication of each rule (fact) of $\mathcal{P}$. Remember that in DLP the implications are strict logical true and the logical value **true** is regarded as $[1, 1]$ in QDLP. Thus, $\mathcal{P}$ is equivalent to $\mathcal{P}'$ from the syntax point of view.

Example 4. Consider the logic program $\mathcal{P} = \{a \leftarrow\ ;\ b \leftarrow\ ;\ c \vee d \leftarrow a, b\}$. The quantitative version of $\mathcal{P}$ is $\mathcal{P}' = \{a \xleftarrow{[1,1]}\ ;\ b \xleftarrow{[1,1]}\ ;\ c \vee d \xleftarrow{[1,1]} a, b\}$. □

We wish to see now whether QDLP is an extension of DLP also from the semantics point of view. We say that a stable p-semantics of QDLP is a generalization of the stable model semantics of DLP iff $\mathcal{SM}^{(p)}(\mathcal{P}') = \mathcal{SM}(\mathcal{P})$ for each $\mathcal{P}$ in DLP, where $\mathcal{P}'$ is the quantitative version of $\mathcal{P}$. Given p, a priori, it is not guaranteed that the p-semantics of QDLP generalizes the DLP semantics.

It is highly desirable that QDLP semantics coincides with DLP semantics on boolean quantitative programs. Whether the p-semantics of a given class of boolean quantitative programs coincides with the DLP semantics, depends strongly on the value of p and on the features (e.g., positive, stratified negative, disjunctive,etc.) of the QDLP class. We single out the classes of QDLP and the values of p for which the p-semantics on the boolean quantitative programs of these classes coincides with the DLP semantics.

	$\{\ \}$	$\{\ \neg^{\mathbf{s}}\ \}$	$\{\ \neg\ \}$	$\{\ \vee^{\mathbf{h}}\ \}$	$\{\ \vee\ \}$	$\{\ \vee^{\mathbf{h}},\ \neg^{\mathbf{s}}\ \}$	$\{\ \vee,\ \neg^{\mathbf{s}}\ \}$	$\{\ \vee,\ \neg\ \}$
$p = -\infty$	YES	YES	NO	YES	NO	YES	NO	NO
$p \in (-\infty, 0)$	YES	YES	NO	NO	NO	NO	NO	NO
$p \in [0, +\infty]$	YES	YES	NO	YES	YES	YES	YES	NO

Table 1. QDLP fragments generalizing DLP

The results on generalizations are summarized in Table 1. Each column of the table collects the results for a specific class of programs for the T-norms induced by the values of p on the rows. The symbol $\neg^s$ refers to the stratified negation, while $\vee^h$ refers to the head cycle free (HCF) disjunction. [2] For instance, the last column of the table refers to the (unstratified) negative (non-HCF) disjunctive programs.

A box of the table contains the answer YES if the class of quantitative programs given by the corresponding column header is a generalization for the values of p given by the header of the corresponding row of the table, and NO otherwise.

From the non-disjunctive programs class, for positive and stratified programs, every $p \in \overline{\mathbf{R}}$ induces a quantitative extension of DLP.

From the class of disjunctive programs, like in the non-disjunctive case, for positive and stratified programs there are values of p which induce quantitative extensions of DLP, but unlike in the non-disjunctive case, where $p \in \overline{\mathbf{R}}$, p is reduced to $\{-\infty\} \cup [0, +\infty]$ for the HCF case and to $[0, +\infty]$ for the non-HCF case. The generalizations of the HCF and non-HCF programs are not supported by other values of p.

Thus, generalization is guaranteed in most cases where recursion through negation and disjunction is forbidden (stratified and HCF programs). This is a nice result because stratified HCF programs have a very clear and intuitive declarative meaning (while unstratification and recursion through disjunction can be confusing).

Intuitively, the fact that a fragment QF of QDLP is not a generalization of the corresponding fragment F of DLP is due to (i) the disjunctive rules' heads, and (ii) that some values of p induce T-conorms for which, when applied to a disjunction of atoms, it is not absolutely necessary that the certainty degree interval of all atoms be $[1, 1]$ or $[0, 0]$ in order to derive $[1, 1]$ as certainty degree interval for the disjunction. For these values of p, the quantitative version $\mathcal{P}'$ in QF of a program $\mathcal{P}$ in F has pure quantitative stable p-models in QDLP which clearly cannot be accepted as stable models in DLP for $\mathcal{P}$. Only the T-conorms and not also the T-norms corresponding to these values of p were reasons for not obtaining generalizations of DLP.

7 Complexity Results

As for traditional DLP, four main decisional problems arise in the context of QDLP. In particular, given a quantitative program $\mathcal{P}$ and $p \in \overline{\mathbf{R}}$,

1. is a given quantitative interpretation I of $\mathcal{P}$ a p-model for $\mathcal{P}$? (*p-Model Checking*)
2. is a given p-model M of $\mathcal{P}$ a stable p-model for $\mathcal{P}$? (*Stable p-Model Checking*)
3. does there exist a stable p-model for $\mathcal{P}$? (*p-Consistency*)
4. given an atom $A \in B_{\mathcal{P}}$ and a certainty interval $[x, y]$, does there exist a stable p-model M for $\mathcal{P}$ such that $M(A) \geq [x, y]$? (*Brave p-Reasoning*)

We have analyzed the complexity of the above decisional problems for the classes of QDLP which are generalizations of the corresponding DLP classes, the other fragments

[2] The notion of Stratified Negation [1] and of Head Cycle Free Disjunction [4, 5] are extended from traditional DLP to QDLP in a straightforward manner. Their formal definitions are given in Appendix A.

being of low interest from the practical point of view. The results for non-disjunctive and disjunctive quantitative programs are summarized in Table 2 and 3, respectively.

A box in the tables contains the complexity of the decisional problem given by the corresponding column header for the fragment of QDLP given by the corresponding row header.

	p-Model Checking	Stable p-Model Checking	p-Consistency	Brave p-Reasoning
$\{\ \}$	P	P	Ensured	P
$\{\ \neg^{\mathbf{s}}\ \}$	P	P	Ensured	P

Table 2. Complexity of non-disjunctive QDLP Fragments for $p \in \overline{\mathbf{R}}$

The results in Table 2 for the non-disjunctive fragments are valid for $p \in \overline{\mathbf{R}}$. For both positive and stratified classes, all decisional QDLP problems, apart from p-Consistency which is $\mathcal{O}(1)$, are polynomial.

	p-Model Checking	Stable p-Model Checking	p-Consistency	Brave p-Reasoning
$\{\ \vee^{\mathbf{h}}\ \}$	P	P	Ensured	NP-complete
$\{\ \vee\ \}$	P	coNP-complete	Ensured	Σ_2^P-complete
$\{\ \vee^{\mathbf{h}},\ \neg^{\mathbf{s}}\ \}$	P	P	Ensured	NP-complete
$\{\ \vee,\ \neg^{\mathbf{s}}\ \}$	P	coNP-complete	Ensured	Σ_2^P-complete

Table 3. Complexity of QDLP Fragments for $p = +\infty$ (T-norm T_3)

Determining precisely the complexity of disjunctive QDLP is much more difficult. In this paper, we have concentrated our attention on the QDLP fragments relative to the T-norm T_3 ($p = +\infty$). This T-norm is of particular interest, as it is the norm for which QDLP generalizes also the quantitative language of van Emden (see section 6.1).

The results for the disjunctive QDLP are shown in Table 3. The first column reports about the complexity of p-Model Checking for the various disjunctive fragments of QDLP. In all cases the complexity is polynomial.

The 2nd column reports about the complexity of Stable p-Model Checking. The "hardest" QDLP fragments for this problem are proved to be the classes of positive and stratified negative (non-HCF) disjunctive programs whose complexity is coNP-complete. In the other two considered cases the complexity is polynomial.

The 3rd column reports about the complexity of p-Consistency. In all considered cases the complexity is $\mathcal{O}(1)$ because the existence of a stable p-model is ensured.

Finally, the 4th column reports about the complexity of Brave p-Reasoning. We can note an increment of complexity from NP-complete for the HCF case to Σ_2^P-complete for the non-HCF case.

Note that the here considered classes of QDLP with stratified negation do not increase the complexity of any of the four decisional problems w.r.t. the corresponding positive classes. This result is shown in Table 3 by the rows pairs $(1, 3)$ and $(2, 4)$ which store the same complexity results in all columns.

Remark that our results for QDLP coincide precisely with the results for DLP obtained by Eiter et al. in [13, 14], i.e., reasoning under multiple-valued logics is more general but not harder than reasoning under boolean logics. That is, uncertainty comes for free!

Acknowledgments

I am very grateful to Georg Gottlob and Nicola Leone for their useful criticism and numerous fruitful discussions on the manuscript.

References

1. K.R. Apt, H.A. Blair, and A. Walker. *Towards a Theory of Declarative Knowledge*. Foundations of Deductive Databases and Logic Programming, Minker, J. (ed.), Morgan Kaufmann, Los Altos, 1987.
2. F. Bacchus. *Representing and Reasoning with Probabilistic Knowledge*. Research Report CS-88-31, University of Waterloo, 1988.
3. Baral, C., Gelfond, M. Logic Programming and Knowledge Representation. *Journal of Logic Programming*, Vol. 19/20, May/July, pp. 73–148, 1994.
4. R. Ben-Eliyahu and R. Dechter. Propositional Semantics for Disjunctive Logic Programs. *Annals of Mathematics and Artificial Intelligence*, 12:53–87, 1994.
5. R. Ben-Eliyahu and L. Palopoli. Reasoning with Minimal Models: Efficient Algorithms and Applications. In *Proc. KR-94*, pp. 39–50, 1994.
6. H.A. Blair and V.S. Subrahmanian. *Paraconsistent Logic Programming*. Theoretical Computer Science, 68, pp. 35–54, 1987.
7. P. Bonissone. Summarizing and Propagating Uncertain Information with Triangular Norms. *International Journal of Approximate Reasoning*, 1:71–101,1987.
8. P.R. Cohen and M.R. Grinberg. A Framework for Heuristic Reasoning about Uncertainty. In *Proc. IJCAI '83*, pp. 355–357, Karlsruhe, Germany, 1983.
9. P.R. Cohen and M.R. Grinberg. A Theory of Heuristics Reasoning about Uncertainty. *AI Magazine*, 4(2):17–23, 1983.
10. A.P. Dempster. *A Generalization of Bayesian Inference*. J. of the Royal Statistical Society, Series B, 30, pp. 205–247, 1968.
11. J. Dix. Semantics of Logic Programs: Their Intuitions and Formal Properties. An Overview. In *Logic, Action and Information*, pp. 241–329. DeGruyter, 1995.
12. J. Doyle. Methodological Simplicity in Expert System Construction: the Case of Judgements and Reasoned Assumptions. *AI Magazine*, 4(2):39–43, 1983.
13. T. Eiter, G. Gottlob, and H. Mannila. Disjunctive Datalog. *ACM Transaction on Database System*, 22(3):364–417, September 1997.

14. T. Eiter and G. Gottlob. On the Computational Cost of Disjunctive Logic Programming: Propositional Case. *Annals of Mathematics and Artificial Intelligence*, 15(3/4):289–323, 1995.
15. M.C. Fitting. *Bilattices and the Semantics of Logic Programming*. J. Logic Programming, 11, pp. 91–116, 1991.
16. M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.
17. M. Ishizuka. *Inference Methods Based on Extended Dempster-Shafer Theory for Problems with Uncertainty/Fuzziness*. New Generation Computing, 1, 2, pp. 159–168, 1983.
18. M. Kifer and A. Li. *On the Semantics of Rule-Based Expert Systems with Uncertainty*. In 2-nd International Conference on Database Theory, Springer Verlag LNCS 326, pp. 102–117, 1988.
19. M. Kifer and V.S. Subrahmanian. *Theory of the Generalized Annotated Logic Programming and its Applications*. J. Logic Programming, 12, pp. 335–367, 1992.
20. L.V.S. Lakshmanan, N. Leone, R. Ross, and V.S. Subrahmanian. *ProbView: A Flexible Probabilistic Database System*. ACM Transaction on Database Systems, 22, 3, pp. 419–469, 1997.
21. J.W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, 1987.
22. J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming*. MIT Press, Cambridge, MA, 1992.
23. C. Mateis. *A Quantitative Extension of Disjunctive Logic Programming*. Technical Report, available on the web as: **http://www.dbai.tuwien.ac.at/staff/mateis/gz/qdlp.ps**.
24. R.T. Ng and V.S. Subrahmanian. Probabilistic Logic Programming. *Information and Computation*, 101:150–201, 1992.
25. R.T. Ng and V.S. Subrahmanian. Empirical Probabilities in Monadic Deductive Databases. In *Proc. Eighth Conf. Uncertainty in AI*, pp. 215–222, Stanford, 1992.
26. R.T. Ng and V.S. Subrahmanian. *A Semantical Framework for Supporting Subjective and Conditional Probabilities in Deductive Databases*. J. of Automated Reasoning, 10, 2, pp. 191–235, 1993.
27. R.T. Ng and V.S. Subrahmanian. Stable Semantics for Probabilistic Deductive Databases. *Information and Computation*, 110:42–83, 1994.
28. R.T. Ng and V.S. Subrahmanian. *Non-monotonic Negation in Probabilistic Deductive Databases*. In Proc. 7-th Conf. Uncertainty in AI, pp. 249–256, Los Angeles, 1991.
29. N.J. Nilsson. Probabilistic Logic. *Artificial Intelligence*, vol. 28, pp. 71–87, 1986.
30. J. Pearl. Probabilistic Reasoning in Intelligent Systems – Networks of Plausible Inference. *Morgan Kaufmann*, 1988.
31. B. Schweizer and A. Sklar. Associative Functions and Abstract Semi-Groups. *Publicationes Mathematicae Debrecen*, 10:69–81, 1963.
32. G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.
33. E. Shapiro. *Logic Programs with Uncertainties: A Tool for Implementing Expert Systems*. In Proc. IJCAI '83, pp. 529–532, 1983.
34. M.H. van Emden. Quantitative Deduction and its Fixpoint Theory. *The Journal of Logic Programming*, 1:37–53, 1986.
35. L.A. Zadeh. Fuzzy Sets. *Inform. and Control*, 8:338–353, 1965.

A Stratified and Head Cycle Free QDLP

The stratified and head-cycle-free (HCF) quantitative programs are important classes of the quantitative programs and, as we will see, they have nice properties.

The stratified quantitative programs are defined in the classical way, as introduced by Apt et al. in [1]. A quantitative program $\mathcal{P}$ is (locally) stratified iff it is possible to partition the set of its atoms into strata $\langle S_1, \ldots, S_r \rangle$, such that for every rule $h_1 \vee \cdots \vee h_k \overset{[x,y]}{\longleftarrow} b_1, \cdots, b_l, \neg b_{l+1}, \cdots, \neg b_{l+m}$ in $\mathcal{P}$ the following holds,

(i) $Strat(a) = i$ iff $a \in S_i$ and
(ii) $Strat(b_s) \leq Strat(h_t)$ for all $1 \leq s \leq l, 1 \leq t \leq k$ and
(iii) $Strat(b_s) < Strat(h_t)$ for all $l+1 \leq s \leq l+m, 1 \leq t \leq k$.

Note that if $\mathcal{P}$ is stratified then there is a partition $\mathcal{P} = \mathcal{P}_1 \cup \ldots \cup \mathcal{P}_r$, where r is the number of strata and $\mathcal{P}_i$ contains the rules of $\mathcal{P}$ defining the atoms of $S_i, 1 \leq i \leq r$.

In the sequel, if a negative program is not explicitly said to be stratified, it is assumed to be unstratified.

Example 5. Consider the program $\mathcal{P}$ consisting of the following rules:

$$a \vee b \overset{[0.6,0.6]}{\longleftarrow} \neg\, c. \qquad a \overset{[0.4,0.4]}{\longleftarrow}.$$
$$e \overset{[0.5,0.5]}{\longleftarrow} b,\ \neg\, d. \qquad c \vee d \overset{[0.8,0.8]}{\longleftarrow}.$$

$\mathcal{P}$ is stratified. A partition of $B_\mathcal{P}$ into strata is $\langle S_1, S_2 \rangle$ with $S_1 = \{c,\ d\}$ and $S_2 = \{a,\ b,\ e\}$. The partition of $\mathcal{P}$ corresponding to the partition of $B_\mathcal{P}$ is $\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2$ with $\mathcal{P}_1 = \{c \vee d \overset{[0.8,0.8]}{\longleftarrow}\}$ and $\mathcal{P}_2 = \{a \overset{[0.4,0.4]}{\longleftarrow};\ a \vee b \overset{[0.6,0.6]}{\longleftarrow} \neg\, c;\ e \overset{[0.5,0.5]}{\longleftarrow} b,\ \neg\, d\}$. □

At every program $\mathcal{P}$, we associate a directed graph $DG_\mathcal{P} = (N, E)$, called the *dependency graph* of $\mathcal{P}$, in which (i) each predicate of $\mathcal{P}$ is a node in N, and (ii) there is an arc in E directed from a node a to a node b iff there is a rule in $\mathcal{P}$ such that b and a are the predicates of a positive literal appearing in $H(r)$ and $B(r)$, respectively. $DG_\mathcal{P}$ singles out the dependencies of the head predicates of a rule r from the positive predicates in its body. [3]

Example 6. Consider the program $\mathcal{P}_1$ consisting of the following rules: [4]

$$a \vee b \overset{[0.6,0.6]}{\longleftarrow}. \qquad c \overset{[0.6,0.6]}{\longleftarrow} a. \qquad c \overset{[0.6,0.6]}{\longleftarrow} b.$$

$DG_{\mathcal{P}_1}$ is depicted in Figure 2a. (Note that, since the sample program is propositional, the nodes of the graph are atoms, as atoms coincide with predicates in this case.)

Consider now the program $\mathcal{P}_2$, obtained by adding to $\mathcal{P}_1$ the rules

$$d \vee e \overset{[0.8,0.8]}{\longleftarrow} a. \qquad d \overset{[0.4,0.4]}{\longleftarrow} e. \qquad e \overset{[0.5,0.5]}{\longleftarrow} d,\ \neg\, b.$$

The dependency graph $DG_{\mathcal{P}_2}$ is shown in Figure 2b. □

The HCF quantitative programs are an important class of the quantitative programs with disjunction in the head and are defined in the classical way, as defined in [4, 5].

A program $\mathcal{P}$ is HCF iff there is no clause r in $\mathcal{P}$ such that two predicates occurring in the head of r are in the same cycle of $DG_\mathcal{P}$. In the sequel, if a disjunctive program is not explicitly said to be HCF, it is assumed to be non-HCF.

[3] Note that negative literals cause no arc in $DG_\mathcal{P}$.

[4] We point out again that we use propositional programs for simplicity, but the results are valid for the general case of (function-free) programs with variables.

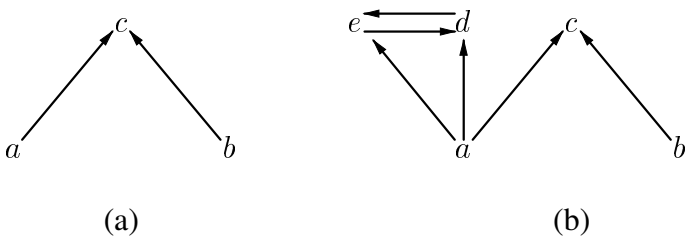

Fig. 2. Dependency Graph ($DG_{\mathcal{P}}$)

Example 7. The dependency graphs given in Figure 2 reveal that program $\mathcal{P}_1$ of Example 6 is HCF and that $\mathcal{P}_2$ is not HCF, as rule $d \vee e \overset{[0.8,0.8]}{\longleftarrow} a$ contains in its head two predicates belonging to the same cycle of $DG_{\mathcal{P}_2}$. □