Level mapping characterizations for quantitative and disjunctive logic programs

Matthias Knorr

Bachelor Thesis

supervised by Prof. Dr. Steffen Hölldobler

under guidance of Dr. Pascal Hitzler

Knowledge Representation and Reasoning Group
Artificial Intelligence Instititute
Department of Computer Science
Dresden University of Technology
Dresden, Germany

Contents

Abstract

Several different approaches of logic programming semantics have been proposed during the last two decades. These semantics varied in many aspects and it was difficult to find the exact relationships between them. Hitzler and Wendt proposed a new method, based on level mappings, which allows to provide uniform characterizations of different semantics for logic programs. They gave new characterizations of different semantics, like the well-founded semantics or the Fitting semantics. We will apply this method to other classes of logic programs, namely quantitative logic programs and disjunctive logic programs. There are also different approaches of semantics for both classes and we will provide characterizations for some of them. In fact, we will consider a quantitative semantics due to van Emden and a specialization of a semantics due to Mateis where real numbers, respectively intervals of real numbers, are used as measures of uncertainty. Furthermore, we will provide a level mapping characterization of the minimal model semantics for disjunctive logic programs and a characterization for the combination of these two classes, i.e. quantitative disjunctive logic programs.

1 Introduction

A variety of different approaches to fixed-point semantics have been proposed in the field of logic programming and nonmonotonic reasoning. The different approaches were compared several times (for example the stable model semantics by Gelfond and Lifschitz [3] and the well founded semantics by van Gelder, Ross and Schlipf [17] in [16] and [17]) but no comparison covered all concepts of semantics because the rationales behind some of these semantics differ in too many aspects.
In [5] and [6], Hitzler and Wendt introduced level mappings as a tool to characterize logic programming semantics. These mappings from elements of the Herbrand base to ordinals induces an ordering on the set of all ground atoms while preventing the occurence of infinite descending chains. In [6], the focus was set on negation in logic programming, and the least model semantics for definite programs, the stable model semantics, the Fitting semantics [2], the well-founded semantics and the weakly perfect model semantics [13] were characterized by level mappings. These level mapping characterizations are comparable and so it is also possible to compare the corresponding semantics. Furthermore, Hitzler and Wendt stated that level mapping characterizations should be applicable to many semantics which are based on an operator with a least fixpoint, respectively an extension of the stable model semantics.
Quantitative logic programs were introduced because in some cases it seemed to be more useful to have probabilities or other measures of uncertainty instead of the truth values true and false. So no truth values are assigned to the atoms of a quantitative logic program but values of uncertainty and quantitative clauses assign values to the head depending on the values of the atoms in the bodies of the clauses. Several different quantitative approaches have been proposed but we will focus on the approaches by van Emden [15] and Mateis [10]. In one of the most relevant earlier works in the field of quantitative logic programs, van Emden used real numbers between 0 and 1 as measures of uncertainty and no negation occured in his programs. Furthermore, he defined a unique uncertainty calculus for applying the values to the single atoms of the program. Mateis considered negation, used intervals of real numbers between 0 and 1 as measures of uncertainty and Triangular norms (T-norms) as a class of calculi for proposing values through the clauses where the calculus defined by van Emden [15] is a special case. We will define level mappings for quantitative programs of both approaches and we will distinguish between programs with and without negation. We will therefore restrict the T-norms of Mateis' approach to the special case of van Emden's uncertainty calculus and we will then be able to show that intervals and real numbers as measures of uncertainty and the operators which are defined for these programs are closely related.
Another extension of normal logic programs are disjunctive logic programs. The difference to normal logic programs is that more than one atom may occur in the heads of the clauses. This allows to express incomplete knowledge in the sense that a disjunction of statements is a logical consequence of a statement but it is impossible to say which of these statements exactly are true. Among all proposed semantics for disjunctive logic programs, the most widely recognized is the minimal model semantics [9]. The concept of minimal models means, roughly spoken, to make as few as possible atoms true in each head of a disjunctive clause. This can also be understood as an exclusive interpretation of disjunctions. Because

the minimal model semantics was defined for disjunctive programs without negation, this concept was extended to programs including negation which was done by Przymusinski in [14] when proposing the disjunctive stable model semantics. We will define level mappings for the minimal model semantics and the disjunctive stable model semantics and we will not use a special operator for disjunctive programs but a restricted form of normal derivatives [4] and apply the results of definite programs, respectively (normal) ones.
In [10] Mateis extended his approach to quantitative disjunctive logic programs, i.e. programs including quantitativity and disjunctions in the head of the clauses. The combination of these two concepts results in quantitative minimal models and quantitative disjunctive stable models and the level mappings for the corresponding programs will also be a combination of the level mappings for quantitative, respectively disjunctive, logic programs. We will restrict the aspect of quantitativity to real numbers as uncertainty measures and omit intervals because the results would be very similar but much more difficult to read in the case of intervals.
The paper will unfold according to the following structure. In Section 2, the general syntax of all different kinds of logic programs is explained. Section 3 contains quantitative logic programs, Section 4 deals with disjunctive logic programs and quantitative disjunctive programs are discussed in Section 5. Each of these three sections is divided into a first part where the semantics of the considered kind of programs are developed and a second part which include the level mappings for the corresponding programs. We close with conclusions and a consideration of further work in Section 6.

2 Syntax

A *(normal) logic program* consists of finitely many universally quantified *clauses* of the form

$$H \leftarrow A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m,$$

usually written as

$$H \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m,$$

where H, A_i and B_j, for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, are atoms of a first order language. An *atom* is of the form $p(t_1, \ldots, t_m)$, where p is a predicate symbol of arity m and the t_i, for $i = 1, \ldots, m$ and $m \geq 0$, are terms. A *term* is a constant, a variable or $f(t_1, \ldots, t_n)$ where f is an n-nary function symbol which takes again n terms as arguments and is, for $n = 0$, equivalent to a constant. Predicate symbols, function symbols and constants all belong to a given first order language.
A clause can be divided into the *head* H and the *body* $A_1 \wedge \ldots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m$. If the body is empty, the clause is called a *fact*. We also use the notion *literal* and distinguish between *positive literals* and *negative literals*, if it is an atom, respectively a negated one. In a *definite clause* no negative literals occur and we call a program consisting only of definite clauses a *definite logic program*.
A *quantitative logic program* is a logic program extended by adding an interval $[x, y]$ to each clause with $0 < x \leq y \leq 1$. Such a *quantitative clause* is written as

$$H \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m,$$

where x, y are *components* with x being the *lower component* and y the *upper component*. Note that there also exist *quantitative definite logic programs* just if there is no negative literal in any rule and that we call a program *qualitative* if it is not a quantitative one.
We further extend these programs by allowing disjunctions to occur in the heads of the clauses and call them *quantitative disjunctive logic programs*, or just *disjunctive*, respectively *definite disjunctive*, depending on the occurence of intervals, respectively negative literals. A *quantitative disjunctive clause* is of the form

$$H_1 \vee \cdots \vee H_l \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$$

where, for $k = 1, \ldots, l$, all H_k are also atoms of the first order language.

3 Quantitative logic programming

3.1 Semantics of quantitative programs

We will start with the semantics of quantitative logic programs and deal with disjunctions in a later section. But before, we will introduce some notions which are used for all programs. The *Herbrand base* B_P of a given quantitative disjunctive logic program P (and all included special cases) is the set of all ground atoms which can be formed with the symbols of P, i.e. the constants, predicates and function symbols. The set of all ground instances of a program P with respect to B_P is called $\mathsf{ground}(P)$. Note that the Herbrand base and $\mathsf{ground}(P)$ are usually not finite because of the possible occurence of function symbols.
For qualitative logic programs a *(Herbrand) interpretation* I is a mapping $B_P \to \{false, true\}$, commonly written as a subset of B_P, consisting only of those $A \in B_P$ mapping to *true*. In the quantitative case according to [10], a *quantitative (Herbrand) interpretation* I maps elements of B_P to an interval $[x, y] \subseteq [0, 1]$ of real numbers and can therefore be written as a set of pairs $(A, [x, y])$ with $A \in B_P$ and $[x, y] = I(A)$.
For defining a semantics of quantitative programs we need to operate on intervals. In [10] Mateis proposed a way of doing this based on triangular norms and we will follow his approach.

3.1.1 Definition Let T be a, possibly infix, operator on intervals $[a_1, b_1], \ldots, [a_n, b_n]$. We define

$$T([a_1, b_1], \ldots, [a_n, b_n]) = [T(a_1, \ldots, a_n), T(b_1, \ldots, b_n)].$$

3.1.2 Example Let $\times$ be multiplication:
$\times([0.5, 0.5], [0.6, 0.8], [0.8, 0.9]) = [\times(0.5, 0.6, 0.8), \times(0.5, 0.8, 0.9)] = [0.24, 0.36]$

Given a quantitative program P, there are infinitely many quantitative interpretations since the intervals consist of real numbers. So it is useful to define an order relation on quantitative interpretations and for this purpose a relation on intervals also taken from Mateis [10].

3.1.3 Definition The relation $\preceq$ is defined on intervals such that $[a, b] \preceq [c, d]$ if and only if $a \leq c$ and $b \leq d$.

3.1.4 Definition The relation $\sqsubseteq$ is defined on quantitative interpretations such that $I_1 \sqsubseteq I_2$ if and only if $I_1(A) \preceq I_2(A)$ for all $A \in B_P$ and some quantitative interpretations I_1, I_2.

For the time being we restrict our programs to being definite and can now turn to the question when a clause is true in a given interpretation I. Starting with the qualitative case, the `body` of a clause $H \leftarrow A_1, \ldots, A_n$ is true in I if and only if, for all $i = 1, \ldots, n$, all A_i are true in I, i.e. all A_i occur in I. A clause $H \leftarrow$ `body` is true in I, if and only if `body` is false or `body` and H are both true. There also exists a quantitative version.

3.1.5 Definition A quantitative clause of the form $H \xleftarrow{[x,y]} A_1, \ldots, A_n$ is evaluated to true in I, if and only if

$$I(H) \succeq [x, y] \times \inf\{I(A_i) \mid i = 1, \ldots, n\}$$

where $\inf \emptyset$ is set to $[1, 1]$.

For any program P we call a (Herbrand) interpretation I a *(Herbrand) model* of P if every clause, i.e. every element of $\mathsf{ground}(P)$, is true in I. We also use the notion of *two-valued model* for models of definite programs and *quantitative model* for quantitative ones.
In [15] van Emden proposed many results for definite quantitative programming which we will apply. The only difference is that he used factors, real numbers r between 0 and 1, instead of intervals of real numbers, for his quantitative clauses written as

$$H \xleftarrow{r} A_1, \ldots, A_n.$$

To distinguish the two different types of quantitative programs we will call programs with factors also *scalar (quantitative) programs* consisting of *scalar clauses*. So *scalar interpretations* are sets of pairs (A, f) with $A \in B_P$ and $f = I(A)$.

3.1.6 Definition For some scalar interpretations I_1, I_2 the relation $\sqsubseteq$ becomes $I_1 \sqsubseteq I_2$ if and only if $I_1(A) \leq I_2(A)$ for all $A \in B_P$.

The evaluation of a scalar clause is very similar to that of a quantitative one.

3.1.7 Definition A scalar clause $H \xleftarrow{x} A_1, \ldots, A_n$ is evaluated to true, if and only if

$$I(H) \geq x \times \mathsf{min}\{I(A_i) \mid i = 1, \ldots, n\}$$

where $\mathsf{min}\,\emptyset$ is set to 1.

Note that a quantitative clause of the form $H \xleftarrow{[x,x]} A_1, \ldots, A_n$ corresponds to a scalar clause $H \xleftarrow{x} A_1, \ldots, A_n$ and a quantitative program consisting only of such quantitative clauses corresponds to a scalar program.
We want to define a connection between the two kinds of quantitative programs, so that afterwards it is easier to show the following results. For this purpose, we *divide* quantitative clauses, quantitative programs and quantitative interpretations.

3.1.8 Definition A quantitative clause c with an interval $[x, y]$ can be divided into two scalar clauses, such that the quantitative clause is duplicated and the interval $[x, y]$ is replaced by x at the lower scalar clause $c_\downarrow$ and by y at the upper scalar clause $c_\uparrow$.

3.1.9 Definition A quantitative program P can be divided into two scalar programs $P_\uparrow$, $P_\downarrow$, such that each clause c is divided into two scalar clauses and $P_\uparrow$ consists of all upper clauses $c_\uparrow$ and $P_\downarrow$ of all lower clauses $c_\downarrow$.

3.1.10 Definition A quantitative interpretation I can be divided into two scalar interpretations $I_\uparrow$, $I_\downarrow$, such that, for each $A \in B_P$, the pair $(A, [x, y])$ occuring in I is replaced by (A, x) in $I_\downarrow$ and by (A, y) in $I_\uparrow$.

3.1.11 Example

$$\begin{array}{ll} a & \xleftarrow{[0.5,1]} \\ b & \xleftarrow{[0.8,0.9]} a \end{array}$$

This program P with an interpretation $I = \{(a, [0.5, 1]), (b, [0.4, 0.9])\}$, which is also a model, can be divided into two scalar programs (with scalar clauses): $P_{\downarrow} = \{a \xleftarrow{0.5}, b \xleftarrow{0.8} a\}$, and $P_{\uparrow} = \{a \xleftarrow{1}, b \xleftarrow{0.9} a\}$ with the corresponding scalar interpretations, respectively models, $I_{\downarrow} = \{(a, 0.5), (b, 0.4)\}$ and $I_{\uparrow} = \{(a, 1), (b, 0.9)\}$.

Now we can turn towards van Emden's results.

3.1.12 Definition (A, r) is true in a scalar model I if $r \leq I(A)$.

3.1.13 Definition For all programs P, all $A \in B_P$ and all $r \in [0, 1]$, $P \models (A, r)$ holds if and only if (A, r) is true in every Herbrand model M of P.

In [15] van Emden also stated that $P \models (A, r)$ implies $P \models (A, r_1)$ for any $r_1 \leq r$ so that it is the aim to make r as large as possible which in the qualitative case corresponds to $r = 1$. This is also applicable to quantitative programs with intervals.

3.1.14 Definition $(A, [x, y])$ is true in a quantitative model I if $[x, y] \preceq I(A)$.

3.1.15 Definition For all programs P, all $A \in B_P$ and all intervals $[x, y] \subseteq [0, 1]$, $P \models (A, [x, y])$ holds if and only if $(A, [x, y])$ is true in every Herbrand model M of P.

Here also $P \models (A, [x, y])$ implies $P \models (A, [x_1, y_1])$ for any interval $[x_1, y_1] \preceq [x, y]$. Returning to van Emden's approach he defines the least Herbrand model M of P:

3.1.16 Definition The least Herbrand model M of P, denoted M_P, on scalar interpretations is determined by

$$M_P(A) = \sup\{r \mid P \models (A, r)\} \text{ for each } A \in B_P$$

where sup denotes the least upper bound.

The least Herbrand model of quantitative programs can be obtained in a similar way but this will be easier to show by means of the theorem below which connect scalar and quantitative models.
With each program P we can associate an operator T mapping from interpretations to interpretations where the fixpoints of T are models of P. Following this method, we recall the operator for scalar quantitative programs taken from van Emden [15].

3.1.17 Definition Let P be a scalar program. The operator S_P on scalar quantitative interpretations is, for every $A \in B_P$, defined as

$$S_P(I)(A) = \sup\{x \times \mathsf{min}\{I(A_i) \mid i = 1, \ldots, n\} \mid A \xleftarrow{x} A_1, \ldots, A_n \in \mathsf{ground}(P)\}$$

where sup is the least upper bound with respect to $\leq$.

As the properties of this operator are already shown in [15] they will just be recalled. But before that we will define an operator for quantitative programs with intervals as a specialization of the operator proposed by Mateis [10].

3.1.18 Definition Let P be a quantitative program. The operator T_P on quantitative interpretations is, for every $A \in B_P$, defined as

$$T_P(I)(A) = \sup\{[x,y] \times \inf\{I(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{[x,y]} A_1,\ldots,A_n \in \mathsf{ground}(P)\}$$

where sup is the least upper bound with respect to $\preceq$.

Now that we have defined operators for two different kinds of quantitative programs and a way to divide quantitative programs into scalar ones, we can show the connection between these two operators.

3.1.19 Theorem Let P be a quantitative program and $P_\uparrow$, $P_\downarrow$ the scalar programs obtained from dividing P. The operator T_P is characterized by the operators of these scalar programs, for all $A \in B_P$, by

$$T_P(I)(A) = [S_{P_\downarrow}(I_\downarrow)(A), S_{P_\uparrow}(I_\uparrow)(A)].$$

Proof: The operator T_P for the quantitative program P is, by Definition 3.1.18, defined as $T_P(I)(A) = \sup\{[x,y] \times \inf\{I(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{[x,y]} A_1,\ldots,A_n \in \mathsf{ground}(P)\}$. Having divided the program P, the interpretation I can also be divided into $I_\uparrow$, $I_\downarrow$ because only the upper components are necessary for $P_\uparrow$, respectively the lower components for $P_\downarrow$. If we apply the function inf to quantitative interpretations then we have to calculate for each $A \in B_P$ the smallest interval on $\preceq$ which is done separately for each component. So we calculate for every $A \in B_P$ for each component the smallest number.

$$\begin{aligned} T_P(I)(A) \;=\; & \sup\{[x,y] \times [\mathsf{min}\{I_\downarrow(A_i) \mid i = 1,\ldots,n\}, \mathsf{min}\{I_\uparrow(A_i) \mid i = 1,\ldots,n\}] \mid \\ & A \xleftarrow{x} A_1,\ldots,A_n \in \mathsf{ground}(P_\downarrow), A \xleftarrow{y} A_1,\ldots,A_n \in \mathsf{ground}(P_\uparrow),\} \end{aligned}$$

Applying Definition 3.1.1 leads to

$$\begin{aligned} T_P(I)(A) \;=\; & \sup\{[x \times \mathsf{min}\{I_\downarrow(A_i) \mid i = 1,\ldots,n\}, y \times \mathsf{min}\{I_\uparrow(A_i) \mid i = 1,\ldots,n\}] \mid \\ & A \xleftarrow{x} A_1,\ldots,A_n \in \mathsf{ground}(P_\downarrow), A \xleftarrow{y} A_1,\ldots,A_n \in \mathsf{ground}(P_\uparrow),\} \end{aligned}$$

and the supremum is also obtained separately for each component.

$$\begin{aligned} T_P(I)(A) \;=\; & [\sup\{x \times \mathsf{min}\{I_\downarrow(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{x} A_1,\ldots,A_n \in \mathsf{ground}(P_\downarrow)\}, \\ & \sup\{y \times \mathsf{min}\{I_\uparrow(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{y} A_1,\ldots,A_n \in \mathsf{ground}(P_\uparrow)\}] \end{aligned}$$

So $S_{P_\downarrow}(I_\downarrow)(A) = \sup\{x \times \mathsf{min}\{I_\downarrow(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{x} A_1,\ldots,A_n \in \mathsf{ground}(P_\downarrow)\}$ and $S_{P_\uparrow}(I_\uparrow)(A) = \sup\{y \times \mathsf{min}\{I_\uparrow(A_i) \mid i = 1,\ldots,n\} \mid A \xleftarrow{y} A_1,\ldots,A_n \in \mathsf{ground}(P_\uparrow)\}$ by Definition 3.1.17 and because we consider all clauses occuring in $\mathsf{ground}(P_\downarrow)$, respectively $\mathsf{ground}(P_\uparrow)$. ■

This result is very useful since it simplifies the proofs for monotonicity and continuity, as we show later, of the operator T_P defined for quantitative programs. We are now able to

divide the program and apply to each of the resulting scalar programs the operator S_P defined on scalar programs and combine the results afterwards.
Now we recall the properties of the operator S_P for scalar programs from [15]. The operator S_P is monotonic, i.e. $I_1 \sqsubseteq I_2$ implies $S_P(I_1) \sqsubseteq S_P(I_2)$, which in turn implies the existence of the *least fixed point* $\mathsf{lfp}(S_P)$ of S_P by the Knaster-Tarski fixed-point theorem. This fixed point can be obtained by defining, for each monotonic operator T, that $T \uparrow 0 = \emptyset$, $T \uparrow (\alpha + 1) = T(T \uparrow \alpha)$ for any ordinal α, $T \uparrow \beta = \bigsqcup_{\gamma<\beta} T \uparrow \gamma$ for any limit ordinal β and $T \uparrow \alpha$ as the least fixed point of T for some ordinal α. Moreover, this operator is continuous [15] and for every program P and every interpretation I it follows that P is true in I if and only if $S_P(I) \sqsubseteq I$. Furthermore, van Emden showed in [15] that this operator is *finite*, in the sense that, for all $A \in B_P$, there exists a natural number n such that $M_P(A) = (S_P \uparrow n)(A)$. The consequence is that the operator always reaches, for each $A \in B_P$, the highest value $M_P(A)$, hence $M_P = \mathsf{lfp}(S_P) = S_P \uparrow \omega$.
There is a special property of scalar programs which will be helpful when defining level mappings for them.

3.1.20 Lemma Let P be a scalar program. If M is the least scalar model of P then for each $H \in M$ with $M(H) > 0$ exists a clause $H \xleftarrow{x} A_1 \ldots, A_n$ in $\mathsf{ground}(P)$ with $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$.

Proof: If M is the least scalar model of P and $M(H) > 0$ then H has to occur somewhere in $\mathsf{ground}(P)$ since otherwise M_1 with $M_1(H) = 0$ and $M_1(A) = M(A)$ for all other $A \in B_P$ is also a model of P and M is not the least one. If H occurs only in the bodies of some clauses then M_1 is again a model of these clauses hence a model for the whole program and M is not the least one. So for all $H \in M$ with $M(H) > 0$ there is at least one scalar clause in $\mathsf{ground}(P)$ with H as the head. Because M is a model for P, all these clauses with H in the head are true in M and $M(H) \geq x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ is satisfied for all clauses $H \xleftarrow{x} A_1 \ldots, A_n$, by Definition 3.1.7. Then $M(H) \geq \sup\{x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} \mid H \xleftarrow{x} A_1 \ldots, A_n\}$ is true. $M(A) = M_P(A) = S_P \uparrow \omega(A)$ holds for all $A \in B_P$ because M is the least model and for all $A \in B_P$ and some natural number m also $M_P(A) = S_P \uparrow m(A)$ is satisfied by finiteness of S_P. So there is a maximal value $(x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\})$ for all clauses $H \xleftarrow{x} A_1 \ldots, A_n$ and we can conclude $M(H) \geq \max\{x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} \mid H \xleftarrow{x} A_1 \ldots, A_n\}$. If $M(H) > \max\{x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} \mid H \xleftarrow{x} A_1 \ldots, A_n\}$ then M is not the least model of P hence $M(H) = \max\{x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} \mid H \xleftarrow{x} A_1 \ldots, A_n\}$. ■

With the help of the following definitions we are able to show even a stronger version of this lemma.

3.1.21 Definition Let P be a scalar program. A *chain* $A \xleftarrow{x} A_1, \ldots, A_n$ of $\mathsf{ground}(P)$ is defined inductively:
1. A clause $A \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ is a chain.
2. If $A \xleftarrow{y} B_1, \ldots, B_m$ is a chain of $\mathsf{ground}(P)$ and $B_j \xleftarrow{z} A_1, \ldots, A_n$ a clause in $\mathsf{ground}(P)$, for $1 \leq j \leq m$ and $x = y \times z$, then $A \xleftarrow{x} A_1, \ldots, A_n$ is a chain.

3.1.22 Definition A scalar program P is *cycle-free* if there exists no chain of $\mathsf{ground}(P)$ where an atom A occurs in the head and in the body.

Now we are able to select a subset of any scalar program with a least scalar model such that this subset is cycle-free.

3.1.23 Lemma If P is a scalar program and M is the least scalar model of P then a cycle-free subset P_{cf} of P exists such that for each $H \in M$ with $M(H) > 0$ exists exactly one scalar clause $H \xleftarrow{x} A_1 \ldots, A_n$ in $\mathsf{ground}(P_{cf})$ with $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$.

Proof: If M is the least scalar model of P then for each $H \in M$ with $M(H) > 0$ exists a clause $H \xleftarrow{x} A_1 \ldots, A_n$ with $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ by Lemma 3.1.20. So there is indeed at least one such clause and we only have to select exactly one of them for each $H \in M$ with $M(H) > 0$ and to show that the so defined subset of $\mathsf{ground}(P)$ is cycle-free. We choose the clauses according to the following conditions:
If $H \in M$ with $M(H) > 0$ and there is a clause $H \xleftarrow{x}$ with $M(H) = x$ then we choose this fact.
If $H \in M$ with $M(H) > 0$ and there is no fact $H \xleftarrow{x}$ with $M(H) = x$ then there are clauses $H \xleftarrow{x} A_1 \ldots, A_n$ with $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $n \geq 1$. $M(H)$ is greater than 0, so $x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} > 0$ is also true and x as well as $\mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ have to be greater than 0. Then $M(A_i) > 0$ for all $i = 1, \ldots, n$ and there is at least one A_l, $1 \leq l \leq n$, in each such clause with $M(A_l) = \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(H) = x \times M(A_l)$. So we select in each clause among all the A_l with minimal value $M(A_l)$ one A_{l*} with a minimal value k such that $S_P \uparrow \omega(A_{l*}) = S_P \uparrow (k)(A_{l*})$ and $S_P \uparrow k(A_i) \geq S_P \uparrow k(A_{l*})$, for all $i = 1, \ldots, n$ and $1 \leq l* \leq n$, where $H \xleftarrow{x} A_1 \ldots, A_n$ is the corresponding clause with $M(H) = x \times M(A_{l*})$. Because $M(A_i) \geq M(A_{l*})$ for all $i = 1, \ldots, n$ and, for all $A \in M$ and a natural number m, also $S_P \uparrow \omega(A) = S_P \uparrow m(A)$ is true by finiteness of S_P, this minimal value k has to exist for each of these clauses. So we can select one corresponding clause with an A_{l*} with a minimal value k_* among all values k for the A_{l*} of the corresponding clauses.
Now there is exactly one scalar clause $H \xleftarrow{x} A_1 \ldots, A_n$ in $\mathsf{ground}(P_{cf})$ for each $H \in M$ with $M(H) > 0$ where $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and additionally M is the least scalar model of P_{cf}. So we only have to show that P_{cf} is cycle-free.
Assume that P_{cf} is not cycle-free. Then there exists a chain of $\mathsf{ground}(P_{cf})$ such that an atom H occurs in the head and in the body of this chain. If there is a chain with the head H then there is a clause $H \xleftarrow{x} A_1 \ldots, A_n$ in $\mathsf{ground}(P_{cf})$ with $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$. This clause cannot be a fact $H \xleftarrow{x}$ because facts have an empty body. So there is one A_{l*}, $1 \leq l* \leq n$, in this clause with minimal value k such that $M(A_i) \geq M(A_{l*})$ and $S_P \uparrow k(A_i) \geq S_P \uparrow k(A_{l*})$ for all $i = 1, \ldots, n$, $S_P \uparrow \omega(A_{l*}) = S_P \uparrow k(A_{l*})$ and $M(H) = x \times M(A_{l*})$. If k is minimal and $M(H) = x \times M(A_{l*})$ and $M(A_{l*}) = S_P \uparrow \omega(A_{l*}) = S_P \uparrow k(A_{l*})$ then $S_P \uparrow (k+1)(H) = S_P \uparrow \omega(H) = M(H) = x \times S_P \uparrow k(A_{l*})$ and $S_P \uparrow \omega(H) > S_P \uparrow k(H)$ by definition of S_P. If $S_P \uparrow \omega(H) = S_P \uparrow k(H)$ would be satisfied then $S_P \uparrow k(H) = M(H) = x \times S_P \uparrow k_1(A_{l*})$, for $k_1 < k$ and another minimal A_{l*}, and k would not be minimal.
But H or the atom which is the connection to that H in the body of the chain equals one A_i, $1 \leq i \leq n$, and $S_P \uparrow k(A_i) \geq S_P \uparrow k(A_{l*})$ is satisfied for this A_i. If $H = A_i$, $1 \leq i \leq n$, then

$S_P \uparrow k(H) \geq S_P \uparrow k(A_{l*})$. If A_i, $1 \leq i \leq n$, is the atom which connects the clause to that H in the body of the chain and $S_P \uparrow k(A_i) \geq S_P \uparrow k(A_{l*})$ then also $S_P \uparrow k(H) \geq S_P \uparrow k(A_{l*})$ because $x \in [0,1]$ for all factors x and $M(H) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ for each clause $H \xleftarrow{x} A_1 \ldots, A_n$ in $\mathsf{ground}(P_{cf})$.
If $S_P \uparrow k(H) \geq S_P \uparrow k(A_{l*})$ then $S_P \uparrow (k+1)(H) \geq S_P \uparrow k(A_{l*})$ by monotonicity of S_P. Since $S_P \uparrow (k+1)(H) = x \times S_P \uparrow k(A_{l*})$ and $x \in [0,1]$ we conclude that $S_P \uparrow (k+1)(H) > S_P \uparrow k(A_{l*})$ is impossible because the multiplication of $S_P \uparrow k(A_{l*})$ with a value less then 1 cannot be greater than $S_P \uparrow k(A_{l*})$ and $S_P \uparrow (k+1)(H) = S_P \uparrow k(A_{l*})$ has to hold. If $S_P \uparrow k(H) \geq S_P \uparrow k(A_{l*})$ and $S_P \uparrow (k+1)(H) = S_P \uparrow k(A_{l*})$ then $S_P \uparrow k(H) = S_P \uparrow k(A_{l*})$ by monotonicity of S_P. Therefore $S_P \uparrow (k+1)(H) = S_P \uparrow k(H)$ and because $S_P \uparrow \omega(H) = S_P \uparrow (k+1)(H)$ also $S_P \uparrow \omega(H) = S_P \uparrow k(H)$. We conclude by contradiction that P_{cf} is cycle-free. ■

3.1.24 Example Let P be a scalar program consisting of the following clauses:

$$
\begin{aligned}
q &\xleftarrow{0.5} \\
r &\xleftarrow{0.5} \\
w &\xleftarrow{0.6} \\
z &\xleftarrow{1} \\
p &\xleftarrow{1} p, q \\
p &\xleftarrow{1} w, r \\
r &\xleftarrow{0.5} z \\
w &\xleftarrow{0.8} z
\end{aligned}
$$

The least scalar model is obtained as $\{(p, 0.5), (q, 0.5), (r, 0.5), (w, 0.8), (z, 1)\}$ and P_{cf} consists of the following clauses.

$$
\begin{aligned}
q &\xleftarrow{0.5} \\
r &\xleftarrow{0.5} \\
z &\xleftarrow{1} \\
p &\xleftarrow{1} w, r \\
w &\xleftarrow{0.8} z
\end{aligned}
$$

Note that $S_P \uparrow 1(q) = S_P \uparrow \omega(q)$ but $S_P \uparrow 1(p) < S_P \uparrow 1(q)$ so that the clause $p \xleftarrow{1} p, q$ cannot be chosen because $S_P \uparrow 1(r) = S_P \uparrow \omega(r)$ and $S_P \uparrow 1(w) > S_P \uparrow 1(r)$.

We are now able to define least models of quantitative programs and to show the properties of the operator T_P for quantitative programs.

3.1.25 Theorem The operator T_P is monotonic, i.e. $I_1 \sqsubseteq I_2$ implies $T_P(I_1) \sqsubseteq T_P(I_2)$, for quantitative interpretations I_1, I_2.

Proof: Suppose $I_1 \sqsubseteq I_2$ then $I_1(A) \preceq I_2(A)$ for all $A \in B_P$ by Definition 3.1.4 and in turn $I_{1\downarrow}(A) \leq I_{2\downarrow}(A)$ and $I_{1\uparrow}(A) \leq I_{2\uparrow}(A)$ for all $A \in B_P$ by Definition 3.1.3. Then also $I_{1\downarrow} \sqsubseteq I_{2\downarrow}$ and $I_{1\uparrow} \sqsubseteq I_{2\uparrow}$ by Definition 3.1.6. We already know that S_P is monotonic, so $S_{P_\downarrow}(I_{1\downarrow}) \sqsubseteq S_{P_\downarrow}(I_{2\downarrow})$ and $S_{P_\uparrow}(I_{1\uparrow}) \sqsubseteq S_{P_\uparrow}(I_{2\uparrow})$. By Definition 3.1.6, we obtain $S_{P_\downarrow}(I_{1\downarrow})(A) \leq S_{P_\downarrow}(I_{2\downarrow})(A)$ and $S_{P_\uparrow}(I_{1\uparrow})(A) \leq S_{P_\uparrow}(I_{2\uparrow})(A)$ for all $A \in B_P$. Therefore, we conclude by Definition 3.1.1 and 3.1.3 that

$$[S_{P_\downarrow}(I_{1\downarrow})(A), S_{P_\uparrow}(I_{1\uparrow})(A)] \preceq [S_{P_\downarrow}(I_{2\downarrow})(A), S_{P_\uparrow}(I_{2\uparrow})(A)].$$

Using the characterization of T_P (Theorem 3.1.19) we deduce $T_P(I_1)(A) \preceq T_P(I_2)(A)$ for all $A \in B_P$ hence $T_P(I_1) \sqsubseteq T_P(I_2)$ by Definition 3.1.4. ■

Again, the monotonicity implies the existence of a least fixed point $\mathsf{lfp}(T_P)$ of T_P.

3.1.26 Theorem The least Herbrand model M of a quantitative program P, denoted M_P, on quantitative interpretations is determined by

$$M_P(A) = \sup\{[x, y] \mid P \models (A, [x, y])\} \text{ for each } A \in B_P$$

where sup denotes the least upper bound with respect to $\preceq$ for intervals.

Proof: If P is a quantitative program, then we can divide P into the scalar programs $P_\downarrow$, $P_\uparrow$. By Definition 3.1.16, we have $M_{P_\downarrow}(A) = \sup\{x \mid P_\downarrow \models (A, x)\}$ and $M_{P_\uparrow}(A) = \sup\{y \mid P_\uparrow \models (A, y)\}$ for each $A \in B_P$. By Theorem 3.1.19 $T_P(I)(A) = [S_{P_\downarrow}(I_\downarrow)(A), S_{P_\uparrow}(I_\uparrow)(A)]$ holds for all $A \in B_P$. Then also $T_P \uparrow \omega(A) = [S_{P_\downarrow} \uparrow \omega(A), S_{P_\uparrow} \uparrow \omega(A)]$ and $M_P(A) = [M_{P_\downarrow}(A), M_{P_\uparrow}(A)]$ for all $A \in B_P$. We can substitute $M_{P_\downarrow}(A)$ and $M_{P_\uparrow}(A)$ and obtain $M_P(A) = [\sup\{x \mid P_\downarrow \models (A, x)\}, \sup\{y \mid P_\uparrow \models (A, y)\}]$ for all $A \in B_P$.
The supremum is also applicable to intervals in the ordering $\preceq$ and this leads to $M_P(A) = \sup\{[x, y] \mid P \models (A, [x, y])\}$. ■

3.1.27 Theorem The operator T_P is continuous, i.e. it is monotonic and for every directed set of interpretations I, we have $\bigsqcup T_P(I) = T_P(\bigsqcup I)$.

Proof: We have already shown in Theorem 3.1.25 that T_P is monotonic, so we have only to show that $\bigsqcup T_P(I) = T_P(\bigsqcup I)$ is true. $P_\downarrow$ and $P_\uparrow$ are obtained from dividing P and we can divide the quantitative interpretations. S is applicable to this programs and because we know that S is a continuous operator, $\bigsqcup S_{P_\downarrow}(I_\downarrow) = S_{P_\downarrow}(\bigsqcup I_\downarrow)$ and $\bigsqcup S_{P_\uparrow}(I_\uparrow) = S_{P_\uparrow}(\bigsqcup I_\uparrow)$ are true. So we have $\bigsqcup S_{P_\downarrow}(I_\downarrow)(A) = S_{P_\downarrow}(\bigsqcup I_\downarrow)(A)$ and $\bigsqcup S_{P_\uparrow}(I_\uparrow)(A) = S_{P_\uparrow}(\bigsqcup I_\uparrow)(A)$ for all $A \in B_P$. We combine the two operators $S_{P_\downarrow}$ and $S_{P_\uparrow}$ and obtain

$$\left[\bigsqcup S_{P_\downarrow}(I_\downarrow)(A), \bigsqcup S_{P_\uparrow}(I_\uparrow)(A)\right] = \left[S_{P_\downarrow}\left(\bigsqcup I_\downarrow\right)(A), S_{P_\uparrow}\left(\bigsqcup I_\uparrow\right)(A)\right]$$

for all $A \in B_P$. Applying Definition 3.1.1 to the left part of the equation leads to

$$\bigsqcup [S_{P_\downarrow}(I_\downarrow)(A), S_{P_\uparrow}(I_\uparrow)(A)] = \left[S_{P_\downarrow}\left(\bigsqcup I_\downarrow\right)(A), S_{P_\uparrow}\left(\bigsqcup I_\uparrow\right)(A)\right]$$

for all $A \in B_P$. By Theorem 3.1.19, we conclude $\bigsqcup T_P(I)(A) = T_P(\bigsqcup I)(A)$ for all $A \in B_P$ and therefore $\bigsqcup T_P(I) = T_P(\bigsqcup I)$. ■

So T_P is monotonic and continuous and for every definite quantitative program P and every interpretation I this program P is true in I if and only if $T_P(I) \sqsubseteq I$. We follow the method used for S_P and show now that the operator T_P is finite.

3.1.28 Theorem The operator T_P is finite, i.e., for all $A \in B_P$, there exists a natural number n such that $M_P(A) = (T_P \uparrow n)(A)$.

Proof: We can divide the model (interpretation) M_P and the associated program P and apply Theorem 3.1.19 to the right part of the equation:

$$[M_{P_\downarrow}(A), M_{P_\uparrow}(A)] = [(S_{P_\downarrow} \uparrow n_1)(A), (S_{P_\uparrow} \uparrow n_2)(A)]$$

We have to distinguish n_1, n_2 because they are not necessarily equal. We know that S_P is finite, i.e. $M_{P_\downarrow}(A) = (S_{P_\downarrow} \uparrow n_1)(A)$ and $M_{P_\uparrow}(A) = (S_{P_\uparrow} \uparrow n_2)(A)$, for some natural numbers n_1, n_2. But either $n_1 = n_2$ is true or one of the two n_i is greater as the other one, so that the operator T_P reaches the highest value $M_P(A) = (T_P \uparrow n_1)(A)$ for $n_1 \geq n_2$ and otherwise $M_P(A) = (T_P \uparrow n_2)(A)$. ■

The operator also reaches the highest value $M_P(A)$ for each $A \in B_P$, so that we conclude $M_P = \mathsf{lfp}(T_P) = T_P \uparrow \omega$.

3.1.29 Example Let P be a quantitative program consisting of the following clauses.

$$\begin{array}{lll} a & \overset{[0.8,0.9]}{\longleftarrow} & \\ b & \overset{[0.3,0.9]}{\longleftarrow} & a \\ c & \overset{[0.7,0.8]}{\longleftarrow} & a \\ b & \overset{[0.6,0.7]}{\longleftarrow} & c \\ c & \overset{[0.5,1]}{\longleftarrow} & b \end{array}$$

If we apply the operator T_P we obtain the following results:

$$\begin{array}{lcl} T_P \uparrow 0 & = & \emptyset \\ T_P \uparrow 1 & = & \{(a, [0.8, 0.9]), (b, [0, 0]), (c, [0, 0])\} \\ T_P \uparrow 2 & = & \{(a, [0.8, 0.9]), (b, [0.24, 0.81]), (c, [0.56, 0.72])\} \\ T_P \uparrow 3 & = & \{(a, [0.8, 0.9]), (b, [0.336, 0.81]), (c, [0.56, 0.81])\} \\ T_P \uparrow 4 & = & \{(a, [0.8, 0.9]), (b, [0.336, 0.81]), (c, [0.56, 0.81])\} \end{array}$$

The least model M_P is obtained as $T_P \uparrow 3$.

Now we return to programs where also negation is allowed. At first, we have to define how to evaluate intervals in the case of negative literals.

3.1.30 Definition Let $I(A) = [x, y]$, then $I(\neg A) = [1 - y, 1 - x]$.

Like in the definite case we can turn to the question when a clause is true in a given interpretation I and we start again with the qualitative case. The `body` of a clause $H \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_n$ is true in I if and only if, for all $i = 1, \ldots, n$ and all $j = 1, \ldots, m$, all A_i are true in I and all B_j are false, i.e. all A_i but no B_j occur in I. A clause $H \leftarrow$ `body` is true in I, if and only if `body` is false or `body` and H are both true. This definition is similar to the corresponding definite one and this holds also for the following definition:

3.1.31 Definition A quantitative clause of the form $H \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ is evaluated to true in I, if and only if

$$I(H) \succeq [x, y] \times \inf(\{I(A_i) \mid i = 1, \ldots, n\} \cup \{I(\neg B_j) \mid j = 1, \ldots, m\})$$

where $\mathsf{min}\,\emptyset$ is set to $[1, 1]$.

Several different logic programming semantics for programs containing negation have been proposed, many of them for three-valued logic. But the stable model semantics due to Gelfond and Lifschitz [3] is defined on two-valued logic and we will use it because it is also well applicable to quantitative programs with negation. We will now repeat the concept of the stable model semantics in the qualitatitve case as it was proposed in [3]. For a normal program P and a set of atoms M with $M \subseteq B_P$, we define P/M to be the subprogram of $\mathsf{ground}(P)$ where all clauses $H \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_n$ with, for $j = 1, \ldots, m$, at least one $B_j \in M$ are removed. The negative literals in the remaining clauses are also removed. Obviously, the resulting program P/M contains no negation and has therefore a least two-valued model $T_{P/M} \uparrow \omega$. We define for any two-valued interpretation I the operator $GL_P(I) = T_{P/I} \uparrow \omega$ and call M a *stable model* of P if it is a fixed point of the operator GL_P, i.e. if $M = GL_P(M) = T_{P/M} \uparrow \omega$.

3.1.32 Example Let P be the program consisting of the following clauses:

$$\begin{array}{lcl} p & \leftarrow & q, \neg r \\ q & \leftarrow & \neg s, \neg r \\ r & \leftarrow & \neg p, \neg r \end{array}$$

Let $M = \{p, q\}$. The resulting program P/M consists only of the clauses

$$\begin{array}{lcl} p & \leftarrow & q \\ q & \leftarrow & \end{array}$$

and has M as its least two-valued model. Otherwise, if $M = \{q, s\}$ then P/M consists of the clauses

$$\begin{array}{lcl} p & \leftarrow & q \\ r & \leftarrow & \end{array}$$

and has $\{r\}$ as its least two-valued model, so that $M = \{q, s\}$ is not stable.

As already mentioned, we will use this concept also for quantitative programs. So we define the quantitative stable model semantics in the way it has been done by Mateis in [10]. Therefore, we need extended quantitative programs.

3.1.33 Definition An extended quantitative program is a definite quantitative program P_e where subintervals of the unit interval $[0, 1]$ may occur as body atoms in the rules of P_e and, though not contained in B_P, are treated like normal atoms. Furthermore, for every quantitative interpretation I of P_e, each atom $[x, y]$ occuring in the body of a rule is evaluated as $I([x, y]) = [x, y]$.

With the help of extended quantitative programs it is now possible to recall the definition of the quantitative stable model semantics from [10].

3.1.34 Definition For a quantitative program P and a quantitative interpretation M we define P/M to be that extended quantitative program where, for all $j = 1, \ldots, m$, each occuring negative literal $\neg B_j$ in a given clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ is substituted by the interval $M(\neg B_j)$. The operator T_P is applicable to P/M and M is a quantitative stable model if and only if M is the least quantitative model $T_{P/M} \uparrow \omega$ of P/M.

We now want to emphasize a special property of stable models which holds also for quantitative stable models.

3.1.35 Theorem If M is the least quantitative model for the extended quantitative program P/M then M is also a quantitative model for P.

Proof: If M is the least quantitative model of P/M then every clause in P/M is true in M. We have obtained P/M from P by substituting all negative literals $\neg B_j$ in each clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m \in \mathsf{ground}(P)$ by the intervals $M(\neg B_j)$ for all $j = 1, \ldots, m$. Consider two cases:
a) There are no negative literals in the clause in $\mathsf{ground}(P)$. Then this clause occurs also in $\mathsf{ground}(P/M)$ and M is a model of this clause.
b) There is at least one negative literal in the clause in $\mathsf{ground}(P)$. We substitute all negative literals $\neg B_j$ by intervals $M(\neg B_j)$ and the obtained clause is true in M. According to Definition 3.1.7, if this clause $H \leftarrow A_1, \ldots, A_n, A_{n+1}, \ldots, A_m$ is true in M then $M(H) \succeq [x, y] \times \inf\{M(A_k) \mid k = 1, \ldots, m\}$ and each A_i is a positive literal and each A_j an interval representing $M(\neg B_j)$ for $i = 1, \ldots, n$ and $j = n + 1, \ldots, m$. Then also $M(H) \succeq [x, y] \times \inf(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(A_j) \mid j = n + 1, \ldots, m\})$ is true. By $M(A_j) = M(\neg B_j)$ we conclude $M(H) \succeq [x, y] \times \inf(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(\neg B_j) \mid j = 1, \ldots, m\})$ and, by Definition 3.1.31 that the original clause is true in M. ■

3.1.36 Example Let P be the quantitative program consisting of the following clauses

$$\begin{aligned} a &\xleftarrow{[0.5,0.8]} \\ b &\xleftarrow{[0.8,0.9]} a, \neg c \\ c &\xleftarrow{[0.5,0.8]} \neg a \end{aligned}$$

and M be the interpretation $\{(a,[0.5,0.8]),(b,[0.4,0.72]),(c,[0.1,0.3])\}$ so that P/M consists of the clauses

$$\begin{array}{lll} a & \overset{[0.5,0.8]}{\longleftarrow} & \\ b & \overset{[0.8,0.9]}{\longleftarrow} & a,[0.7,0.9] \\ c & \overset{[0.5,0.8]}{\longleftarrow} & [0.2,0.5] \end{array}$$

and $T_{P/M} \uparrow \omega = M$.

3.2 Level mappings for quantitative programs

In [6] Hitzler and Wendt used level-mappings to characterize different semantics for logic programs to make them comparable to each other. For our purposes, we only need total level mappings, recalled in the following. For an interpretation I and a program P a *(total) level mapping* for P is a total mapping $l : B_P \to \alpha$ for some ordinal α. We start by recalling the result for definite logic programs from [6].

3.2.1 Theorem Let P be a definite program. Then there is a unique two-valued model M of P for which there exists a (total) level mapping $l : B_P \to \alpha$ such that for each atom $A \in M$ there exists a clause $A \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $A_i \in M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$. Furthermore, M is the least two-valued model of P.

We will continue with scalar programs, because the result will later simplify the proof in the case of level mappings for definite quantitative programs.

3.2.2 Theorem Let P be a scalar program. Then there is a unique scalar model M of P, consisting of pairs $(A, M(A))$, $A \in B_P$, $M(A) \in [0,1]$, for which there exists a total level mapping $l : B_P \to \alpha$ such that for each $A \in M$ with $M(A) > 0$ exists a clause $A \overset{x}{\longleftarrow} A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$. Furthermore, M is the least scalar model of P.

Proof: If M is the least scalar model $S_P \uparrow \omega$ of P then, by Lemma 3.1.23, there exists a cycle-free subset P_{cf} of P such that for each $A \in M$ with $M(A) > 0$ there is exactly one scalar clause $A \overset{x}{\longleftarrow} A_1 \ldots, A_n$ in P_{cf} with A as the head and $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$. Because there is only one clause for each $A \in M$ and $M(A) > 0$ also $x > 0$ holds all factors x and all the A_i in each clause of P_{cf} satisfy $M(A_i) > 0$. So it does not matter which values $x > 0$ occur as factors for the clauses. All A are contained in M with $M(A) > 0$ and the factors only influence the exact values $M(A)$ for all $A \in M$. Thus we can define P_{cf}^+ consisting of all clauses occuring in P_{cf} where all factors are set to 1 and $M^+ = \{(A, 1) \mid A \in M \text{ and } M(A) > 0\}$ is the least scalar model of P_{cf}^+. If we ignore all factors then P_{cf}^+ is a definite program and M^+ where we ignore the values $M(A) = 1$ for all A is the least model of P_{cf}^+. Now we can apply Theorem 3.2.1 so that there exists a (total) level mapping $l : B_P \to \alpha$ such that for each atom $A \in M^+$ exists a clause $A \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(P_{cf}^+)$ with $A_i \in M^+$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$. This level mapping exists also for the corresponding scalar clauses in P_{cf} thus for clauses in $\mathsf{ground}(P)$.

Conversely, if M is a scalar model for P which satisfies the given condition for some mapping $l : B_P \to \alpha$, then we can show by induction on $l(A)$ that $A \in M$ with $M(A) > 0$ implies $M(A) = S_P \uparrow (l(A) + 1)(A)$.
If $l(A) = 0$ we have to show that $A \in M$ with $M(A) > 0$ implies $M(A) = S_P \uparrow 1(A)$. If $A \in M$ with $M(A) > 0$ then there exists a clause $A \xleftarrow{x} A_1, \ldots, A_n \in \mathsf{ground}(P)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$, according to the hypothesis. Since $l(A) = 0$ and there is no ordinal less than 0, this clause has to be a fact and $M(A) = x$. This implies directly that $M(A) = S_P \uparrow 1(A)$.
Suppose for all $i = 1, \ldots, n$ that $l(A_i) = k_i$ and $A_i \in M$ with $M(A_i) > 0$ implies $M(A_i) = S_P \uparrow (k_i + 1)(A_i)$. Let $l(A) = k + 1$, then we have to show that $A \in M$ with $M(A) > 0$ implies $M(A) = S_P \uparrow ((k + 1) + 1)(A)$. If $A \in M$ with $M(A) > 0$ then there exists a clause $A \xleftarrow{x} A_1, \ldots, A_n \in \mathsf{ground}(P)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ according to the condition. If $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ then $l(A_i) \leq k$ and $M(A_i) = S_P \uparrow (k + 1)(A)$. Consequently, the $M(A_i)$ will not grow further and the calculation of the next step of S_P, i.e. $S_P \uparrow ((k + 1) + 1)$, leads to $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\} = S_P \uparrow ((k + 1) + 1)(A)$. ■

3.2.3 Example Recalling the program from Example 3.1.24, we obtain the following level mapping: $l(q) = l(r) = l(z) = 0$, $l(w) = 1$ and $l(p) = 2$. Furthermore, we can see that if we take P_{cf} and obtain P_{cf}^+ by cancelling all factors then the application of T_P^+ results in $M^+ = \{p, q, r, w, z\}$.

As we can see, the clause $p \xleftarrow{1} w, r$ is the only clause in P_{cf} with p as the head. The interesting thing is that the level is no longer equivalent to the least number k such that $M(p) = S_P \uparrow (k + 1)(p)$ because in this example we can see that $M(p) = S_P \uparrow (1 + 1)$ but $l(p) = 2$.
We can formulate a similar result for definite quantitative programs using the result for scalar programs.

3.2.4 Theorem Let P be a definite quantitative program. Then there is a unique quantitative model M of P, consisting of pairs $(A, M(A))$, $A \in B_P$, $M(A) \subseteq [0, 1]$ for which there exist two total level mappings $l_1, l_2 : B_P \to \alpha$ such that for each A with $M_\downarrow(A) > 0$ exists a clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $M_\downarrow(A) = x \times \mathsf{min}\{M_\downarrow(A_i) \mid i = 1, \ldots, n\}$ and $l_1(A) > l_1(A_i)$ for all $i = 1, \ldots, n$, and a clause $A \xleftarrow{[u,v]} C_1, \ldots, C_p$ in $\mathsf{ground}(P)$ with $M_\uparrow(A) = v \times \mathsf{min}\{M_\uparrow(A_h) \mid h = 1, \ldots, p\}$ and $l_2(A) > l_2(C_h)$ for all $h = 1, \ldots, p$. Furthermore, M is the least quantitative model of P.

Proof: Let M be the least quantitative model of P then $M = M_P = T_P \uparrow \omega$ and $M(A) = T_P \uparrow \omega(A)$ for each $A \in B_P$. Then $T_P \uparrow \omega(A) = [S_{P_\downarrow} \uparrow \omega(A), S_{P_\uparrow} \uparrow \omega(A)]$ for each $A \in B_P$ by Theorem 3.1.19 and $M(A) = [M_\downarrow(A), M_\uparrow(A)]$ for each $A \in B_P$. Then $M_\downarrow = S_{P_\downarrow} \uparrow \omega$ and $M_\uparrow = S_{P_\uparrow} \uparrow \omega$, i.e. $M_\downarrow$ is the least scalar model of $P_\downarrow$ and $M_\uparrow$ is the least scalar model of $P_\uparrow$. Thus we can apply Theorem 3.2.2 to $P_\downarrow$ and $P_\uparrow$ and obtain exactly the two desired level mappings for clauses in $\mathsf{ground}(P_\downarrow)$, respectively $\mathsf{ground}(P_\uparrow)$. If we add the missing components again to the clauses in $\mathsf{ground}(P_\downarrow)$ and $\mathsf{ground}(P_\uparrow)$ then we obtain the clauses

in $\mathsf{ground}(P)$ and the level mappings exist also for them thus for M as the least quantitative model of P.
Conversely, let M be a quantitative model of P which satisfies the given condition for some mappings l_1, l_2. Then we can divide the definite quantitative program P and obtain two scalar programs $P_\downarrow, P_\uparrow$. The quantitative model M of P can also be divided into $M_\downarrow$ and $M_\uparrow$ so that $M_\downarrow$ is a model of $P_\downarrow$ and $M_\uparrow$ is a model of $P_\uparrow$. $M_\downarrow$ consists of pairs $(A, M_\downarrow(A))$ and $M_\uparrow$ consists of pairs $(A, M_\uparrow(A))$ with $M_\downarrow, M_\uparrow \in [0,1]$. We can also divide the clauses $A \xleftarrow{[x,y]} A_1, \ldots, A_n$ and $A \xleftarrow{[u,v]} C_1, \ldots, C_p$ and conclude that there exist clauses $A \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(P_\downarrow)$ and $A \xleftarrow{v} C_1, \ldots, C_p$ in $\mathsf{ground}(P_\uparrow)$. Now we have two scalar quantitative programs $P_\downarrow$, $P_\uparrow$ and for each one a scalar quantitative model and a total level mapping which satisfies the conditions. We can apply Theorem 3.2.2 to each scalar program and thereby conclude that $M_\downarrow$ is the least scalar model of $P_\downarrow$ and $M_\uparrow$ is the least scalar model for $P_\uparrow$, i.e. $M_\downarrow(A) = S_{P_\downarrow} \uparrow \omega(A)$ and $M_\uparrow(A) = S_{P_\uparrow} \uparrow \omega(A)$ for each $A \in B_P$. Then by Theorem 3.1.19 $M(A) = T_P \uparrow \omega(A) = [S_{P_\downarrow} \uparrow \omega(A), S_{P_\downarrow} \uparrow \omega(A)]$ for all $A \in B_P$ hence M is the least quantitative model. ■

3.2.5 Example Recall the program from Example 3.1.29. We obtain the following two level mappings. $l_1(a) = 0$, $l_1(b) = 2$ and $l_1(c) = 1$ and $l_2(a) = 0$, $l_2(b) = 1$ and $l_2(c) = 2$.

The question might arise, whether it is possible to combine the two level mappings, so that we can compare the result more easily with all the other level mappings proposed for many logic programming semantics. If we take a closer look at the simple example above, it seems to be a bit difficult. By $l_1(b) = 2$, $l_2(b) = 1$, $l_1(c) = 1$ and $l_2(c) = 2$ it becomes obvious that the dependencies are different for each level mapping as $l_1(b) < l_1(c)$ and $l_2(b) > l_2(c)$ which is in general caused by the two scalar operators which are independent from each other. It is therefore not surprising that any attempt to combine these two level mappings results in a loss of structure, i.e. the dependencies become invisible, no matter if we use the maximum, the minimum, the sum or anything similar.
We end this section with a proposal for (normal) quantitative programs, i.e. including negation, and start therefore by recalling the qualitative version due to Fages [1].

3.2.6 Theorem Let P be normal. Then a two-valued model $M \subseteq B_P$ of P is a stable model of P if and only if there exists a (total) level mapping $l : B_P \to \alpha$ such that for each $A \in M$ there exists a clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ with $A_i \in M$, $B_j \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

We now take Theorem 3.2.4 and can extend the level mapping for definite quantitative programs in a similar way to quantitative programs including negation.

3.2.7 Theorem Let P be a (normal) quantitative logic program. Then a quantitative model M of P consisting of pairs $(A, M(A))$, $A \in B_P$, $M(A) \subseteq [0,1]$ is a quantitative stable model of P if and only if there exist two total level mappings $l_1, l_2 : B_P \to \alpha$ such that for each A with $M_\downarrow(A) > 0$ exists a clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ with $M_\downarrow(A) = x \times \mathsf{min}(\{M_\downarrow(A_i) \mid i = 1, \ldots, n\} \cup \{M_\downarrow(\neg B_j) \mid j = 1, \ldots, m\})$ and $l_1(A) > l_1(A_i)$, for all $i = 1, \ldots, n$, and a clause $A \xleftarrow{[u,v]} C_1, \ldots, C_p, \neg D_1, \ldots, \neg D_q$ in

$\mathsf{ground}(P)$ with $M_\uparrow(A) = v \times \mathsf{min}(\{M_\uparrow(C_h) \mid h = 1, \ldots, p\} \cup \{M_\uparrow(\neg D_g) \mid g = 1, \ldots, q\})$ and $l_2(A) > l_2(C_h)$ for all $h = 1, \ldots, p$.

Proof: Let M be a quantitative stable model of P, i.e. $T_{P/M} \uparrow \omega = M$. Then M is the least quantitative model for P/M and also a model for P according Theorem 3.1.35. P/M is an extended quantitative program and therefore a definite quantitative program. If P/M is a definite quantitative program and M the least quantitative model of P/M we can apply Theorem 3.2.4 and so there exist two level mappings such that for each A with $M_\downarrow(A) > 0$ exists a clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n, A_{n+1} \ldots, A_m$ in $\mathsf{ground}(P/M)$ with $M_\downarrow(A) = x \times \mathsf{min}\{M_\downarrow(A_f) \mid f = 1, \ldots, m\}$ and $l_1(A) > l_1(A_f)$ for all $f = 1, \ldots, m$, and a clause $A \xleftarrow{[u,v]} C_1, \ldots, C_p, C_{p+1}, \ldots, C_q$ in $\mathsf{ground}(P/M)$ with $M_\uparrow(A) = v \times \mathsf{min}\{M_\uparrow(C_z) \mid z = 1, \ldots, q\}$ and $l_2(A) > l_2(C_z)$ for all $z = 1, \ldots, q$. Note that the levels of the intervals $A_{n+1}, \ldots, A_m, C_{p+1}, \ldots, C_q$ do not matter and are therefore considered to be 0. Then we can conclude directly $l_1(A) > l_1(A_i)$ and $l_2(A) > l_2(C_h)$ for all $i = 1, \ldots, n$ and all $h = 1, \ldots, p$. Furthermore, the intervals in these clauses represented by the A_f and C_z for $f = n+1, \ldots, m$ and $z = p+1, \ldots, q$ are equal to $M_\downarrow(\neg B_j)\}$, respectively $M_\uparrow(\neg D_g)\}$ and $M_\downarrow(A) = x \times \mathsf{min}(\{M_\downarrow(A_i) \mid i = 1, \ldots, n\} \cup \{M_\downarrow(\neg B_j) \mid j = 1, \ldots, m\})$ and $M_\uparrow(A) = v \times \mathsf{min}(\{M_\uparrow(C_h) \mid h = 1, \ldots, p\} \cup \{M_\uparrow(\neg D_g) \mid g = 1, \ldots, q\})$.
Conversely, let M be a model which satisfies the required condition. Then for every $A \in B_P$ there exist two total level mappings $l_1, l_2 : B_P \to \alpha$ such that for each A with $M_\downarrow(A) > 0$ exists a clause $A \xleftarrow{[x,y]} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ with $M_\downarrow(A) = x \times \mathsf{min}(\{M_\downarrow(A_i) \mid i = 1, \ldots, n\} \cup \{M_\downarrow(\neg B_j) \mid j = 1, \ldots, m\})$ and $l_1(A) > l_1(A_i)$ for all $i = 1, \ldots, n$, and a clause $A \xleftarrow{[u,v]} C_1, \ldots, C_p, \neg D_1, \ldots, \neg D_q$ in $\mathsf{ground}(P)$ with $M_\uparrow(A) = v \times \mathsf{min}(\{M_\uparrow(C_h) \mid h = 1, \ldots, p\} \cup \{M_\uparrow(\neg D_g) \mid g = 1, \ldots q\})$ and $l_2(A) > l_2(C_h)$ for all $h = 1, \ldots, p$. But then, for every $A \in B_P$, there are also clauses $A \xleftarrow{[x,y]} A_1, \ldots, A_n, A_{n+1}, \ldots, A_m$ and $A \xleftarrow{[u,v]} C_1, \ldots, C_p, C_{p+1}, \ldots, C_q$ in $\mathsf{ground}(P/M)$ with $l_1(A) > l_1(A_f)$ and $l_2(A) > l_2(C_z)$ for all $f = 1, \ldots, m$ and $z = 1, \ldots, q$ and $M_\downarrow(A) = x \times \mathsf{min}\{M_\downarrow(A_f) \mid f = 1, \ldots, m\}$ and $M_\uparrow(A) = v \times \mathsf{min}\{M_\uparrow(C_z) \mid z = 1, \ldots, q\}$ where the levels of intervals $A_{n+1}, \ldots, A_m, C_{p+1}, \ldots, C_q$ do not matter and are therefore considered to be 0. By Theorem 3.2.4, M is the least model for P/M, i.e. $M = T_{P/M} \uparrow \omega$, and therefore a quantitative stable model of P. ■

3.2.8 Example Recalling the program of Example 3.1.36, we obtain the following two level mappings. $l_1(a) = l_2(a) = 0$, $l_1(b) = l_2(b) = 1$ and $l_1(c) = l_2(c) = 0$.

If a quantitative program P consists of clauses such that for each clause the lower component equals the upper component then we can substitute the intervals in all clauses by the lower component and obtain a *(normal) scalar program.*

3.2.9 Corollary Let P be a (normal) scalar logic program. Then a scalar model M of P consisting of pairs $(A, M(A))$, $A \in B_P$, $M(A) \in [0, 1]$ is a scalar stable model of P if and only if there exists a total level mapping $l : B_P \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \xleftarrow{x} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ with $M(A) =$

$x \times \mathsf{min}(\{M(A_i) \mid i = 1, \ldots n\} \cup \{M(\neg B_j) \mid j = 1, \ldots, m\})$ and $l_1(A) > l_1(A_i)$, for all $i = 1, \ldots, n$.

This result is an immidiate consequence of the proof of Theorem 3.2.7 where the two level mappings are just equal.
We will now conclude the section with another example to emphasize that these results are really applicable in the case of programs with an infinite Herbrand base.

3.2.10 Example Let P be the program

$$\begin{aligned} p(a) &\overset{[0.7,0.8]}{\longleftarrow} \\ p(s(X)) &\overset{[1,1]}{\longleftarrow} \neg p(X) \end{aligned}$$

and $M = \{(p(s^n(a)), [0.7, 0.8]) \mid n \geq 0 \text{ and } n \text{ even } \} \cup \{(p(s^n(a)), [0.2, 0.3]) \mid n \geq 0 \text{ and } n \text{ uneven}\}$. We consequently obtain the level mappings $l_1(p(s^n(a))) = l_2(p(s^n(a))) = 0$ for all natural numbers n.

4 Disjunctive logic programming

4.1 Semantics of disjunctive programs

Now we turn to disjunctive logic programs, restricted in this section to the qualitative case. We thereby follow the structure of the previous section und start by recalling the concepts which are also used for disjunctive programs. We usually distinguish between disjunctive programs Π and (normal) programs P.
The Herbrand base B_Π and $\mathsf{ground}(\Pi)$ of a disjunctive logic program Π are defined in the same way as for a quantitative program and have the same properties. As already stated for qualitative logic programs a (Herbrand) interpretation I is a subset of B_Π consisting only of those atoms $A \in B_\Pi$ which are mapped to *true*. The `body` of a disjunctive clause of the form $H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ is evaluated to true in an interpretation I if and only if, for all $i = 1, \ldots, n$ and all $j = 1, \ldots, m$, all A_i are true in I and all B_j are false in I. The whole clause $H_1 \vee \cdots \vee H_l \leftarrow$ `body` is true in I if and only if `body` is false or `body` and at least one H_k, $1 \leq k \leq l$, are true in I.
Like in section 3, for any disjunctive program Π the interpretation I is a (Herbrand) model M of Π if every clause in $\mathsf{ground}(\Pi)$ is true in I. But there are different opinions about the intended meaning of a disjunction in the head of a clause. Most widely preferred is the approach proposed by Minker [11] where in a model for a clause $H_1 \vee \cdots \vee H_l \leftarrow$ `body` the disjunction should be exlcusive, i.e. as few as possible of the H_k, for $k = 1, \ldots, l$, should be true. This concept is therefore called *minimal model semantics* and we will recall it in the following, restricted to definite disjunctive programs.

4.1.1 Definition Given a definite disjunctive program Π, a model I of Π is a *minimal model* if there is no model I_1 of Π with $I_1 \subset I$. The set of all minimal models of Π is denoted by $\mathcal{MM}_\Pi$.

If there is one unique minimal model it corresponds to the known least Herbrand model. Such defined minimal models have an important property and we will show this property in the following.

4.1.2 Theorem If M is a minimal model of a definite disjunctive program Π then for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$ and $H_k \notin M$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.

Proof: If M is a minimal model then any $A \in M$ has to occur somewhere in the program, i.e. in $\mathsf{ground}(\Pi)$. Suppose A occurs only in the bodies of some clauses. Then these clauses are also true in a model $M_1 = M \backslash \{A\}$ and M is no minimal model in contradiction to the hypothesis. So A has to occur in the head of at least one clause $A \vee H_1 \vee \cdots \vee H_k \leftarrow A_1, \ldots, A_n$. If one of the A_i , $i = 1, \ldots, n$, in each of these clauses is false in M then M_1 is again a model for these clauses leading to the same contradiction, so all A_i have to be true in M for at least one such clause. Suppose there is no such clause with A as the only true atom in the head, i.e. at least one H_k, for $1 \leq k \leq l$, is true. But then A does not need not to be true to make these clauses true, i.e. there exists again a model M_1 of Π with $M_1 = M \backslash \{A\}$ and $M_1 \subset M$, so M is no minimal model which is a contradiction to

the hypothesis. ■

The opposite does in general not hold: If Π is the simple program $a \vee b \leftarrow a$ and $M = \{a\}$ a model of Π then the condition is satisfied but M is obviously not a minimal model of Π.

4.1.3 Example Let Π be the following program:

$$\begin{aligned} a \vee b &\leftarrow \\ b \vee c &\leftarrow \end{aligned}$$

We obtain $\mathcal{MM}_\Pi = \{\{a, c\}, \{b\}\}$, so there is no least model of Π. By Definition 4.1.2 it is now easy to see that neither $\{a, b\}$ nor $\{b, c\}$ can be minimal models of Π, also because $\{b\}$ is a subset of each of them.

We now want to apply the minimal model semantics also to disjunctive programs including negation and introduce the *disjunctive stable model semantics* proposed by Przymusinski [14].

4.1.4 Definition Let Π be a (normal) disjunctive program. A model I of Π is a *disjunctive stable model* if it coincides with a minimal model of the definite disjunctive program Π/I. The set of all stable models of Π is denoted by $\mathcal{ST}_\Pi$.

Of course, Π/I is obtained by cancelling all negative literals in the way we mentioned in the section before. There is also a comparable result of Theorem 4.1.2 for disjunctive logic programs:

4.1.5 Theorem If Π is a disjunctive program and M a disjunctive stable model of Π then for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$ and $B_j, H_k \notin M$ for all $i = 1, \ldots, n$, all $j = 1, \ldots, m$ and all $k = 1, \ldots, l$.

Proof: If M is a disjunctive stable model of Π, then M is also a minimal model of the definite disjunctive program Π/M by Definition 4.1.4 and for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi/M)$ with $A_i \in M$ and $H_k \notin M$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$, by Theorem 4.1.2. If $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ occur in Π/M all B_j have to be false since otherwise we would have cancelled the whole clause in Π/M and not only the B_j. ■

Again, the opposite does not hold since a definite disjunctive program is also a disjunctive one, so that the little example above is also a counterexample in this case.

4.1.6 Example Let Π be the following disjunctive program:

$$\begin{aligned} a \vee b &\leftarrow \\ b &\leftarrow a \\ c &\leftarrow \neg a \end{aligned}$$

We obtain $\mathcal{ST}_\Pi = \{\{b, c\}\}$ as the set of all disjunctive stable models of Π. Note that a occurs only in the head of the first clause so that a and b cannot be in the same disjunctive stable model. Together with the fact that c occurs only in the third clause so that a and c cannot be in the same disjunctive stable model either, this is a helpful restriction to the search space for possible disjunctive stable models.

If we follow the structure of the previous section then we have to define a monotonic and continuous operator for disjunctive programs with a least fixed point like it has been done for example by Minker and Rajasekar in [12], but we choose another way. We will try to separate the information given in the disjunctions and then use the well known operator T_P^+ for definite programs[1], respectively the stable model semantics [3]. Therefore, we recall at first the properties of the operator T_P^+ for definite logic programs from [6]. The operator T_P^+ maps interpretations, i.e. subsets of B_P, to interpretations and is monotonic on the set of all subsets of B_P, with respect to subset inclusion. It is Scott-continuous [8] and achieves its least fixed point as the supremum of the iterates $T_P^+ \uparrow n$. So $M = \mathsf{lfp}(T_P^+) = T_P^+ \uparrow \omega$ is the least two-valued model of a definite program P.
Now the notion of a normal derivative is introduced, which is in general taken from [4] where it has also been used in the context of disjunctive logic programs and offers a way to split the disjunctive information contained in disjunctive programs. We only have to make an aditional restriction and change some notions.

4.1.7 Definition Suppose that Π is a disjunctive logic program.
A *normal derivative* P of Π is defined to be a (ground) normal logic program P consisting of possibly infinitely many clauses which satisfies the following conditions.
(a) For every clause $H \leftarrow \texttt{body}$ in Π exists exactly one clause $A \leftarrow \texttt{body}$ in P such that A belongs to H.
(b) For every clause $A \leftarrow \texttt{body}$ in P there is a clause $H \leftarrow \texttt{body}$ in Π such that A belongs to H.
The set of all normal derivatives $\{P_1, \ldots, P_n\}$ of Π is denoted $\mathcal{S}^\Pi$.

So we obtain a set of normal logic programs, respectively definite logic programs if there are no negative literals in Π, and we can apply the stable model semantics to these programs, respectively the operator T_P^+.
Now we only need a connection between the minimal model semantics, the disjunctive stable model semantics respectively, and these normal derivatives to use this concept when defining level mappings for different disjunctive logic programs.

4.1.8 Lemma If M is a model of a normal derivative of a definite disjunctive program Π then it is also a model for Π.

Proof: The definite normal derivative consists of clauses of the form $A \leftarrow A_1, \ldots, A_n$. So if M is a model for all of these clauses then we have to show that it also models the original disjunctive clauses. We have to consider two cases:

[1] In [6], Hitzler and Wendt used this special notion of T_P^+ to distinguish between the operator T_P for normal logic programs and T_P^+ for definite programs. In our case, we distinguish between the operator T_P for definite quantitative programs and T_P^+ for definite qualitative programs.

a) Suppose at least one A_i, $1 \leq i \leq n$, is false in M then the whole body of the clause is false in M and so it does not matter of which atoms the disjunctive head of the original clause consists, the clause is always true in M.
b) If all A_i. $i = 1, \ldots, n$, are true in M then the head A has also to be true in M since M is a model of the normal derivative. Then the head of the original disjunctive clause is also true in M, independently from the truth values of the other atoms in the original disjunctive head. ■

With the help of Lemma 4.1.8 we are now able to show the following theorem.

4.1.9 Theorem Let Π be a definite disjunctive program and M be a minimal model of Π, i.e. $M \in \mathcal{MM}_\Pi$. Then there exists a corresponding normal derivative $P \in \mathcal{S}^\Pi$ for which M is the least two-valued model.

Proof: If M is a minimal model of the definite disjunctive program Π then according to Theorem 4.1.2 for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$ and $H_k \notin M$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. If all the H_k are false in M we can cancel them in each such clause without changing the semantics and obtain definite clauses $A \leftarrow A_1, \ldots, A_n$ which are also true in M. The set of normal derivatives $\mathcal{S}^\Pi$ consists of all normal derivatives obtained from Π. So there has to exist at least one normal derivative P where exactly these obtained clauses are included in $\mathsf{ground}(P)$. For the rest of the definite disjunctive clauses of $\mathsf{ground}(\Pi)$ consider the two cases:
a) If the disjunctive head is false in M then all atoms in the head are false in M and one of the atoms in the body has to be false in M because M is a model of Π. So it does not matter which atom in the head we select because the clause is still true in M.
b) If the disjunctive head is true in M then at least one atom in the head has to be true and we choose just that normal derivative with an atom in the head of the clause which is true in M, so that the resulting definite clause is also true in M.
So M is a model for P and we only have to show that it is the least model. Assume that M is not the least model for P, i.e. there exists a model M_1 of P with $M_1 \subset M$. If M_1 is a model of a normal derivative then, by Lemma 4.1.8, it is also a model of Π and M is no minimal model of Π and we conclude by contradiction that M is the least model of P. ■

Note that the opposite does in general not hold, i.e. there may be least models of normal derivatives which are not contained in $\mathcal{MM}_\Pi$. But with a certain restriction we can formulate a statement which works in both directions.

4.1.10 Theorem Let Π be a definite disjunctive program. Then a model M of Π is minimal if and only if it is minimal in the set of the least models of all normal derivatives of Π.

Proof: If M is a minimal model of Π then, by Theorem 4.1.9, there exists a corresponding normal derivative for which M is the least model. Hence M is contained in the set of the least models of all normal derivatives of Π. Assume there is a model M_1 which is smaller than M, i.e. $M_1 \subset M$. Then, by Lemma 4.1.8, M_1 is also a model for Π. But if $M_1 \subset M$ is a model of Π then M is no minimal model of Π which contradicts the hypothesis.

Conversely, suppose M is minimal in the set of the least models of all normal derivatives of Π, i.e. M is a model of a normal derivative and there is no other least model in this set which is smaller than M. According to Lemma 4.1.8, M is also a model of Π. If M is a model of Π and not minimal then a model M_1 has to exist which is smaller than M, i.e. $M_1 \subset M$ and M_1 or a model which is even smaller than M_1 has to be minimal. Then M_1 or the model smaller than M_1 is also contained in the set of the least models as we just have shown and M cannot be minimal in the set of the least models of all normal derivatives of Π. By contradiction we conclude that M is a minimal model of Π. ■

4.1.11 Example Recall the program from Example 4.1.3. The set of normal derivatives $\mathcal{S}^{\Pi}$ consists of these four normal derivatives with the corresponding least models:

$$\begin{array}{lll} P_1 : \{a \leftarrow, b \leftarrow\} & M_{P_1} = & \{a, b\} \\ P_2 : \{b \leftarrow, b \leftarrow\} & M_{P_2} = & \{b\} \\ P_3 : \{a \leftarrow, c \leftarrow\} & M_{P_3} = & \{a, c\} \\ P_4 : \{b \leftarrow, c \leftarrow\} & M_{P_4} = & \{b, c\} \end{array}$$

As already stated in Example 4.1.3, $\mathcal{MM}_{\Pi}$ consists of the two minimal models $\{a, c\}$ and $\{b\}$. So P_3 is the corresponding normal derivative for $\{a, c\}$ and P_2 for $\{b\}$ and the least models of P_1 and P_4 are not equal to any minimal model of P, also because they are not minimal in the set of all least models of the normal derivatives.

There is also a connection between normal derivatives and disjunctive stable models.

4.1.12 Theorem Let Π be a disjunctive program and M be a disjunctive stable model of Π, i.e. $M \in \mathcal{ST}_{\Pi}$. Then there exists a corresponding normal derivative $P \in \mathcal{S}^{\Pi}$ for which M is a stable model.

Proof: If M is a disjunctive stable model of Π then it is also a minimal model of the definite disjunctive program Π/M. Consequently, by Theorem 4.1.9, there is a corresponding definite normal derivative P/M for which M is the least two-valued model, hence M is a stable model for the corresponding program P. ■

This is not true for the other direction, but we can formulate a similar weaker version as we have done for the definite case.

4.1.13 Lemma If M is a stable model for a normal derivative of P then it is also a model for Π.

Proof: The normal derivative consists of clauses $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$. So if M is a model for all these clauses then we have to show that it also models the original disjunctive clauses. We have to consider two cases:
a) Suppose at least one A_i is false in M or one B_j is true in I then the whole body of the clause is false in M and so it does not matter of which atoms the disjunctive head of the original clause consists, the clause is always true in M.
b) If all A_i are true in M and all B_j are false in M then the head A has also to be true in

M since M is a model of the normal derivative. Then the head of the original disjunctive clause is also true in M, independently from the truth values of the other atoms in the original disjunctive head. ■

The proof is very similar to the one for definite disjunctive programs but we are at present not in the position to define also a connection for programs including negation that works in both directions. The following example can be used as an example for that.

4.1.14 Example Recall the program Π from Example 4.1.6. The set of normal derivatives $\mathcal{S}^{\Pi}$ consists of two normal derivatives with the following stable models:

$$P_1 : \{a \leftarrow, b \leftarrow a, c \leftarrow \neg a\} \quad M = \{a, b\}$$
$$P_2 : \{b \leftarrow, b \leftarrow a, c \leftarrow \neg a\} \quad M = \{b, c\}$$

Both stable models are minimal in the set of the models of all normal derivatives but only $\{b, c\}$ is a disjunctive stable model so that $\mathcal{ST}_{\Pi} = \{b, c\}$ and P_1 is the corresponding normal derivative.

4.2 Level mappings for disjunctive programs

We want to apply level mappings to disjunctive programs by means of the level mapping characterization for definite programs and we restrict at first to definite disjunctive programs.

4.2.1 Theorem Let Π be a definite disjunctive program. Then a model M of Π is a minimal model of Π if and only if there exists a total level mapping $l : B_{\Pi} \rightarrow \alpha$ such that for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $H_k \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.

Proof: Let M be a minimal model of Π. Then, by Theorem 4.1.2, for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$ and $H_k \notin M$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. Furthermore, by Theorem 4.1.9, there exists a normal derivative $P \in \mathcal{S}^{\Pi}$ for which M is the least two-valued model. Because all the H_k are false in these clauses for each $A \in M$ the clause $A \leftarrow A_1, \ldots, A_n$ has to occur in $\mathsf{ground}(P)$. So we now have for each $A \in M$ a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ and a clause $A \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $A_i \in M$ and $H_k \notin M$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$ and M is the least model of P and a minimal model of Π. If M is the least model for the definite program P then, by Theorem 3.2.1, there exists a (total) level mapping $l : B_P \rightarrow \alpha$ such that for each atom $A \in M$ exists a clause $A \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $A_i \in M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and a corresponding clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $H_k \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.
Conversely, let M be a model of Π that satisfies the given condition then we have to show that M is also a minimal model of Π. Assume that there is a model M_1 of Π with $M_1 \subset M$. Therefore, we show by induction on $l(A)$ that there is no $A \in M$ which is not in M_1.
Suppose $l(A) = 0$. Then according to the condition there exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow$

$A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. Because $l(A) = 0$ and there is no ordinal smaller than 0 this clause has to be the fact $A \vee H_1 \vee \cdots \vee H_k \leftarrow$. If $M_1 \subset M$ then all H_k also have to be false in M_1. Hence A has to be true in M_1 since otherwise M_1 would be no model of Π.
Suppose for all $A_i \in M$ with $l(A_i) \leq k$, $i = 1, \ldots, n$, that they are contained in M_1 and let $l(A)$ be $k + 1$.
Then according to the hypothesis there exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. Again, all H_k, $k = 1, \ldots, l$, have to be false in M_1 because otherwise M_1 could be smaller than M. So A has to be true also in M_1 since all A_i, $i = 1, \ldots, n$, are true in M_1.
Altogether, there is no smaller model of Π than M, so it is a minimal model of Π. ■

4.2.2 Example Consider the definite disjunctive program Π:

$$\begin{array}{rcl} a \vee b & \leftarrow & \\ c & \leftarrow & \\ a \vee e & \leftarrow & c \\ d & \leftarrow & e \end{array}$$

Then we can obtain four normal derivatives each one with a least model:

$$\begin{array}{ll} P_1 = \{a \leftarrow, c \leftarrow, a \leftarrow c, d \leftarrow e\} & M_{P_1} = \{a, c\} \\ P_2 = \{b \leftarrow, c \leftarrow, a \leftarrow c, d \leftarrow e\} & M_{P_2} = \{a, b, c\} \\ P_3 = \{a \leftarrow, c \leftarrow, e \leftarrow c, d \leftarrow e\} & M_{P_3} = \{a, c, d, e\} \\ P_4 = \{b \leftarrow, c \leftarrow, e \leftarrow c, d \leftarrow e\} & M_{P_4} = \{b, c, d, e\} \end{array}$$

M_{P_1} is a subset of M_{P_2} and M_{P_3} so that these two least models cannot be minimal models of Π. So we obtain $\mathcal{MM}_\Pi = \{\{a, c\}, \{b, c, d, e\}\}$ and the level mappings $l(a) = l(b) = l(c) = l(d) = l(e) = 0$ for $\{a, c\}$ and $l(a) = l(b) = l(c) = 0$, $l(e) = 1$ and $l(d) = 2$ for $\{b, c, d, e\}$.

Like in the non-disjunctive and the quantitative case, there is also a result for disjunctive programs containing negation.

4.2.3 Theorem Let Π be a disjunctive program. Then a model M of Π is a disjunctive stable model of Π if and only if there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $B_j, H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$, all $j = 1, \ldots, m$ and all $k = 1, \ldots, l$.

Proof: Let M be a disjunctive stable model of Π. Then, by Definition 4.1.4, M is a minimal model of the definite disjunctive program Π/M. According to Theorem 4.2.1, there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi/M)$ with $A_i \in M$, $H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. All B_j, $j = 1, \ldots, m$, in each such original

disjunctive clause have to be false in M because otherwise these clauses could not occur in $\mathsf{ground}(\Pi/M)$ - they would have been cancelled completely. So there exists for each $A \in M$ such a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ satisfying the condition.
Conversely, let M be a model of Π which satisfies the given condition. Then there exists a total level mapping $l : B_\Pi \rightarrow \alpha$ such that for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ with $A_i \in M$, $B_j, H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$, all $j = 1, \ldots, m$ and all $k = 1, \ldots, l$. If all B_j in these clauses are false then there exists also a total level mapping $l : B_\Pi \rightarrow \alpha$ such that for each $A \in M$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi/M)$ with $A_i \in M$, $H_l \notin M$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. Then M is a minimal model of Π/M by Theorem 4.2.1 and therefore a disjunctive stable model of Π by Definition 4.1.4. ■

We conclude this section with an example of level mappings for disjunctive programs.

4.2.4 Example Let Π be the following disjunctive program:

$$\begin{aligned} a \vee b &\leftarrow \neg c \\ c &\leftarrow \neg a, \neg b \\ d \vee e &\leftarrow a \\ f &\leftarrow d, \neg e \end{aligned}$$

We obtain the following four normal derivatives and for each program one or more stable models:

$$\begin{aligned} P_1 &= \{a \leftarrow \neg c, c \leftarrow \neg a \wedge \neg b, d \leftarrow a, f \leftarrow d \wedge \neg e\} \quad \mathcal{M}_{P_1} = \{\{c\}, \{a, d, f\}\} \\ P_2 &= \{a \leftarrow \neg c, c \leftarrow \neg a \wedge \neg b, e \leftarrow a, f \leftarrow d \wedge \neg e\} \quad \mathcal{M}_{P_2} = \{\{b\}, \{a, e\}\} \\ P_3 &= \{b \leftarrow \neg c, c \leftarrow \neg a \wedge \neg b, d \leftarrow a, f \leftarrow d \wedge \neg e\} \quad M_{P_3} = \{b\} \\ P_4 &= \{b \leftarrow \neg c, c \leftarrow \neg a \wedge \neg b, e \leftarrow a, f \leftarrow d \wedge \neg e\} \quad M_{P_4} = \{b\} \end{aligned}$$

Surprisingly all these models are disjunctive stable models and the level mappings are $l(a) = l(b) = l(c) = l(d) = l(e) = l(f) = 0$ for $\{b\}$ and $\{c\}$, $l(a) = l(b) = l(c) = l(e) = 0$, $l(d) = 1$ and $l(f) = 2$ for $\{a, d, f\}$ and $l(a) = l(b) = l(c) = l(d) = l(f) = 0$ and $l(e) = 1$ for $\{a, e\}$.

5 Quantitative disjunctive logic programming

5.1 Semantics of quantitative disjunctive programs

In this section we want to combine the results of the last two sections, i.e. we will deal with quantitative disjunctive logic programs. In section 3 we detected that quantitative programs are just the combination of two scalar programs and that all results for quantitative programs can easily be obtained from the results for scalar programs. In order to make the following section more readable we restrict quantitative disjunctive programs to the special case of scalar disjunctive programs. So the following section deals with *scalar disjunctive logic programs* as an extension of scalar programs defined in section 3, i.e. quantitative disjunctive logic programs where the intervals are substituted by factors.
Since we have already defined a lot of notions and properties for scalar and disjunctive programs, most of them need only to be recalled. We use the notion Π for any disjunctive program and the Herbrand base B_Π and $\mathsf{ground}(\Pi)$ remain unchanged. We again consider scalar interpretations which map elements of B_Π to factors written as sets of pairs $(A, I(A))$ for each $A \in B_P$.

5.1.1 Definition Let Π be a (normal) scalar disjunctive program and I an interpretation for Π then $I(\neg A) = 1 - I(A)$.

Now we can define when a scalar quantitative clause is true.

5.1.2 Definition A scalar disjunctive clause $H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ is evaluated to true in an interpretation I if and only if

$$\max\{I(H_k) \mid k = 1, \ldots, l\} \geq x \times \mathsf{min}(\{I(A_i) \mid i = 1, \ldots, n\} \cup \{I(\neg B_j) \mid j = 1, \ldots, m\}$$

where $\mathsf{min}\,\emptyset$ is set to 1.

This is a specialization of the definition for T-norms taken from [10]. A model of a program is an interpretation which makes all the clauses of the program true and we call it a *scalar disjunctive model* if the program is a scalar disjunctive program. Now we will follow the same structure as in the previous section and define a semantics for scalar disjunctive programs which follows the same concept of minimality as in section 4 and it is called *scalar minimal model semantics*.

5.1.3 Definition Given a (definite) scalar disjunctive program Π a model I of Π is a *scalar minimal model* if there is no model I_1 of Π with $I_1 \sqsubset I$, i.e. for all $A \in I$ holds not $I(A) > I_1(A)$. The set of scalar minimal models of Π is denoted by $\mathcal{SMM}_\Pi$.

The relation $\sqsubseteq$, respectively $\sqsubset$ as a special case, can be found in Definition 3.1.6. If there is only one element in $\mathcal{SMM}_\Pi$ then this is the known least scalar model. Like in the previous section, there is a property of these minimal models which will be important later on.

5.1.4 Theorem If M is a scalar minimal model of a scalar disjunctive program Π then for each $A \in M$ with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(A) > M(H_k)$ for all $k = 1, \ldots, l$.

Proof: If M is a model of Π then any clause in $\mathsf{ground}(\Pi)$ is true in M. If $M(A) > 0$ for some A then A has to occur somewhere in the program, i.e. in $\mathsf{ground}(\Pi)$. Suppose A occurs only in the bodies of some clauses. Then these clauses are also true in a model M_1 with $M_1(A) = 0$ and $M_1(B) = M(B)$ for all other $B \in B_P$ and M is not minimal. So A has to occur in the head of at least one clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$. If for each such clause $M(H_k) > M(A)$ for some k, $1 \leq k \leq l$, then M_1 is also a model for Π and M is not minimal. So there is at least one such clause with $M(A) > M(H_k)$ for all $k = 1, \ldots, l$. By Definition 5.1.2, the maximal value $M(H)$ among all atoms H in the head of such a clause, i.e. M(A), satisfies the following property: $M(A) \geq x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$. There is a maximum among all these values $x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ which satisifies the property for each such clause, so $M(A) \geq \max\{x \times \mathsf{min}\{I(A_i) \mid i = 1, \ldots, n\} \mid A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(A) > M(H_k)$ for all $k = 1, \ldots, l\}$. If $M(A) > \max\{x \times \mathsf{min}\{I(A_i) \mid i = 1, \ldots, n\}\}$ then there is a model M_1 with $M_1(A) < M(A)$ such that $M_1(A) \geq \max\{x \times \mathsf{min}\{I(A_i) \mid i = 1, \ldots, n\}\}$ for all such clauses and M is not minimal which contradicts the hypothesis. Hence there is at least one clause with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$. ■

5.1.5 Example Let P be the following scalar disjunctive program.

$$\begin{aligned} p \vee q &\xleftarrow{0.8} \\ r &\xleftarrow{0.9} \\ p &\xleftarrow{0.6} r \\ w &\xleftarrow{0.6} q \\ w &\xleftarrow{0.7} p, r \end{aligned}$$

The set of scalar minimal models consists of two models
$M_1 = \{(p, 0.8), (q, 0), (r, 0.9), (w, 0.72)\}$ and $M_2 = \{(p, 0.54), (q, 0.8), (r, 0.9), (w, 0.54)\}$. We can see that in model M_2 to both atoms in the head of the only real disjunctive clause is assigned a value greater 0.

There is a quantitative analogue to the disjunctive stable model semantics proposed by Mateis [10] and we specialize it to scalar programs.

5.1.6 Definition Let Π be a (normal) scalar disjunctive program. A model I of Π is a *scalar disjunctive stable model* if it coincides with a scalar minimal model of the (definite) scalar disjunctive program Π/I. The set of all scalar stable models of Π is denoted by $\mathcal{SST}_\Pi$.

5.1.7 Example The (normal) scalar disjunctive program consists of the following clauses.

$$\begin{aligned} a \vee b &\xleftarrow{0.7} \\ c &\xleftarrow{0.8} \neg a, d \\ d &\xleftarrow{0.9} \\ e &\xleftarrow{0.1} \neg d, c \end{aligned}$$

The set of scalar disjunctive stable models consists of the following two disjunctive stable models. The two models are $M_1 = \{(a,0),(b,0.7),(c,0.72),(d,0.9),(e,0.1)\}$ and $M_2 = \{(a,0.7),(b,0),(c,0.56),(d,0.9),(e,0.1)\}$.

The definition of normal derivatives is also applicable to scalar disjunctive programs so that we obtain a set of scalar disjunctive programs $\mathcal{S}^\Pi$ and we can apply the operator S_P to these programs. Therefore, we also need a connection between scalar disjunctive programs and the normal derivatives.

5.1.8 Lemma If M is a model of a normal derivative P of a (definite) scalar disjunctive program Π then it is also a model for Π.

Proof: The scalar normal derivative consists of clauses of the form $A \overset{x}{\longleftarrow} A_1,\ldots,A_n$. If M is a model for these clauses then we have to show that it also models the original scalar disjunctive clauses. We consider two cases:
a) Suppose $M(A_i) = 0$ for at least one A_i with $i = 1,\ldots,n$, then $\mathsf{min}\{I(A_i) \mid i = 1,\ldots,n\} = 0$ and it does not matter of which atoms the disjunctive head of the original clause consists, the clause is always true in M.
b) If $M(A_i) > 0$ for all $i = 1,\ldots,n$ then $M(A) \geq x \times \mathsf{min}\{I(A_i) \mid i = 1,\ldots,n\}$ because M is a model for P. But then also $\max\{M(H_k) \mid k = 1,\ldots,l\} \geq x \times \mathsf{min}\{M(A_i) \mid i = 1,\ldots,n\}$ for the original scalar disjunctive clause $H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1,\ldots,A_n$ because A is one of the H_k and $\max\{M(H_k) \mid k = 1,\ldots,l\} \geq M(A)$. ■

By means of this lemma the following theorem is easy to show.

5.1.9 Theorem Let Π be a scalar disjunctive program and M be a scalar minimal model of Π, i.e. $M \in \mathcal{SMM}_\Pi$. Then there exists a corresponding normal derivative $P \in \mathcal{S}^\Pi$ for which M is the least scalar model.

Proof: If M is a minimal model of the scalar disjunctive program Π then according to Theorem 5.1.4 for each $A \in M$ with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1,\ldots,A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1,\ldots,n\}$ and $M(A) > M(H_k)$ for all $k = 1,\ldots,l$. If $M(A) > M(H_k)$ then we can cancel all H_k in each such clause and obtain scalar clauses $A \overset{x}{\longleftarrow} A_1,\ldots,A_n$ which are also true in M. The set of normal derivatives S_Π consists of all normal derivatives obtained from Π. So there has to exist at least one normal derivative P where exactly these obtained clauses are included in $\mathsf{ground}(P)$. For the rest of the scalar disjunctive clauses $H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1,\ldots,A_n$ consider two cases:
a) If $\max\{M(H_k) \mid k = 1,\ldots,l\} = 0$ then $M(H_k) = 0$ for all $k = 1,\ldots,l$ and it does not matter which H_k occur in P because $x \times \mathsf{min}\{M(A_i) \mid i = 1,\ldots,n\}$ has to be also 0 and the clause in P is true in M.
b) If $\max\{M(H_k) \mid k = 1,\ldots,l\} > 0$ then there is at least one H_k, $1 \leq k \leq l$ with $M(H_k) = \max\{M(H_k) \mid k = 1,\ldots,l\}$. Then there is a normal derivative in S^Π with one such H_k in the head of this clause and the clause is true in M.
So M is a model for P and we only have to show that it is the least scalar model. Assume there is a model M_1 of P which is smaller than M, ie. $M_1 \sqsubset M$. If M_1 is a model of P then

it is also a model of Π by Lemma 5.1.8. Thus M is no scalar minimal model of Π which contradicts the hypothesis and we conclude that M is the least model of P. ■

5.2 Level mappings for quantitative disjunctive programs

We start again with that programs containing no negation, i.e. scalar disjunctive programs.

5.2.1 Theorem Let Π be a (definite) scalar disjunctive program. Then a model M of Π, consisting of pairs $(A, M(A))$, $A \in B_\Pi$, $M(A) \in [0,1]$ is a scalar minimal model of Π if and only if there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.

Proof: Let M be a scalar minimal model of Π. Then, by Theorem 5.1.4, for each $A \in M$ with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(A) > M(H_k)$ for all $k = 1, \ldots, l$. Furthermore, by Theorem 5.1.9, there exists a normal derivative $P \in S^\Pi$ for which M is the least scalar model. If $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(A) > M(H_k)$ for all $k = 1, \ldots, l$ in these clauses then $A \xleftarrow{x} A_1, \ldots, A_n$ has to occur in $\mathsf{ground}(P)$. So we now have for each $A \in M$ a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ and a clause $A \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $M(A) > M(H_k)$ for all $k = 1, \ldots, l$ and M is the least scalar model of P and a minimal model of Π. If M is the least scalar model for the scalar program P then, by Theorem 3.2.2, there exists a total level mapping $l : B_P \to \alpha$ such that for each $A \in M$ exists a clause $A \xleftarrow{x} A_1, \ldots, A_n$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and therefore a corresponding clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.
Conversely, let M be a scalar model of Π which satisfies the given condition. Assume that $M_1 \sqsubset M$ and we show by induction on $l(A)$ that $M_1 = M$, i.e. there is no $A \in M$ with $M(A) > M_1(A)$.
Suppose $l(A) = 0$. Then according to the condition there exists a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. Since $l(A) = 0$ and there is no ordinal smaller than 0 this clause has to be the fact $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x}$ and $M(A) = x$. If $M_1 \sqsubset M$ then $M_1(H_k) \leq M(H_k)$ for all $k = 1, \ldots, l$ and by $M(A) > M(H_k)$ we conclude $M(A) > M_1(H_k)$ and $x > M_1(H_k)$ for all $k = 1, \ldots, l$ hence $M_1(A) = x$ by Definition 5.1.2.
Suppose for all $A_i \in M$ with $l(A_i) \leq m$ that $M(A_i) = M_1(A_i)$ and let $l(A)$ be $m + 1$. Then according to the hypothesis there exists a clause $A \vee H_1 \vee \cdots \vee H_l \xleftarrow{x} A_1, \ldots, A_n$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. If $M_1 \sqsubset M$ then $M_1(H_k) \leq M(H_k)$ for all $k = 1, \ldots, l$ and by $M(A) > M(H_k)$ we conclude $M(A) > M_1(H_k)$ hence $M_1(H_k) < x \times \mathsf{min}\{M_1(A_i) \mid i = 1, \ldots, n\}$ and $M_1(A) = x \times \mathsf{min}\{M(A_i) \mid i = 1, \ldots, n\}$ by Definition 5.1.2. ■

5.2.2 Example Recall the program from Example 5.1.5 and the two minimal models. The level mapping for M_1 is the following: $l(p) = l(q) = l(r) = 0$ and $l(w) = 1$ and for M_2 we obtain $l(q) = l(r) = 0$ and $l(p) = l(w) = 1$.

The definition of level mappings for scalar disjunctive programs including negation follow the same proceedings as in the previous sections. We apply a stable model semantics, obtain a negation-free program and can apply the result of programs containing no negation.

5.2.3 Theorem Let Π be a (normal) scalar disjunctive program. Then a model M of Π, consisting of pairs $(A, M(A))$, $A \in B_\Pi$, $M(A) \in [0,1]$ is a scalar disjunctive stable model of Π if and only if there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(\neg B_j) \mid j = 1, \ldots, m\})$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$.

Proof: Let M be a scalar disjunctive stable model of Π. Then, by Definition 5.1.3 , M is a scalar minimal model of the scalar disjunctive program Π/M. According to Theorem 5.2.1 there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1, \ldots, A_n, A_{n+1}, \ldots, A_m$ in $\mathsf{ground}(\Pi/M)$ with $M(A) = x \times \mathsf{min}\{M(A_j) \mid j = 1, \ldots, m\}$, $M(A) > M(H_k)$ and $l(A) > l(A_j)$ for all $j = 1, \ldots, m$ and all $k = 1, \ldots, l$. If $l(A) > l(A_j)$ for all $j = 1, \ldots, n, n+1, \ldots, m$ where the $l(A_{n+1}), \ldots, l(A_m)$ of the values are considered to be 0 then also $l(A) > l(A_i)$ for all $i = 1, \ldots, n$. Furthermore, as the $A_{n+1}, \ldots, A_m$ are real numbers representing the $M(\neg B_j)$, we can conclude $M(A) = x \times \mathsf{min}(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(A_j) \mid j = n+1, \ldots, m\})$ and $M(A) = x \times \mathsf{min}(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(\neg B_j) \mid j = 1, \ldots, m\})$..
Conversely, let M be a model of Π which satisfies the given condition. Then there exists a total level mapping $l : B_\Pi \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}(\{M(A_i) \mid i = 1, \ldots, n\} \cup \{M(\neg B_j) \mid j = 1, \ldots, m\})$, $M(A) > M(H_k)$ and $l(A) > l(A_i)$ for all $i = 1, \ldots, n$ and all $k = 1, \ldots, l$. If we substitute all B_j in each clause by the real numbers $M(\neg B_j)$ then there exists also a total level mapping $l : B_\Pi \to \alpha$ such that for each A with $M(A) > 0$ exists a clause $A \vee H_1 \vee \cdots \vee H_l \overset{x}{\longleftarrow} A_1, \ldots, A_n, A_{n+1}, \ldots, A_m$ in $\mathsf{ground}(\Pi)$ with $M(A) = x \times \mathsf{min}\{M(A_j) \mid j = 1, \ldots, m\}$, $M(A) > M(H_k)$ and $l(A) > l(A_j)$ for all $j = 1, \ldots, m$ and all $k = 1, \ldots, l$ where the levels of the values $A_{n+1}, \ldots, A_m$ equal 0. Therefore, M is a scalar minimal model of Π/M by Theorem 5.2.1 and therefore a scalar disjunctive stable model of Π by Definition 5.1.3. ■

5.2.4 Example Recall the program from Example 5.1.7. The obtained level mappings are the following: In both cases we get $l(a) = l(b) = l(d) = 0$, $l(c) = 1$ and $l(e) = 2$. Note that the values $M(e)$ and $M(c)$ in particular depend on the values of negative literals but the level mappings are only results of the dependencies between positive literals.

6 Conclusions

We have characterized quantitative logic programs, disjunctive logic programs and quantitative disjunctive logic programs by means of level mappings and thereby enlarged the applicability of these alternative characterizations for logic programming semantics from [6]. But we have not covered all possible approaches in each case, so a lot of work still needs to be done.

If we compare the proof of level mappings for definite programs [6] and for scalar programs, i.e. quantitative programs based on van Emdens approach [15], then it becomes obvious that the effort necessary to show the result for scalar programs is much higher although both characterizations apart from some quantitative specialities are really similar. This is due to the different behavior of the operators defined for definite programs, respectively scalar programs. We can avoid the difficulties in case of quantitative programs by using an operator which for example multiplies the values of all body atoms of a clause and assigns the product to the atom in the head. A so defined operator represents another special case of triangular norms and should be part of the further studies concerning level mappings for quantitative programs where besides T-norms other quantitative operators shall be discussed. The obtained characterization for quantitative programs using intervals as uncertainty measures depends strongly on the characterization for scalar programs. In fact, such a quantitative program consists of two scalar programs so that all results for scalar programs are also applicable to quantitative programs. Further studies will reveal whether this also holds for other operators so that we maybe can reject the idea of representing uncertainty in logic programs by intervals altogether. By applying the quantitative stable model semantics we have defined a level mapping characterization for (normal) quantitative logic programs similar to the one defined for (normal) qualitative programs. In both cases the levels depend only on positive literals; but in a quantitative interpretation the value of an atom depends equally on the values of positive and negative literals while in the qualitative case the negative literals have to be false in the corresponding model.

The concept of minimality is illustrated by the level mapping characterization for disjunctive programs, respectively the minimal model semantics, because for each atom occuring in a minimal model there is a clause in which only this atom is true in the head. Apart from this disjunctive part there is no difference to the level mapping characterization for definite logic programs. Besides the level mapping characterization we have investigated some restrictions to the search space of minimal models for a given disjunctive logic program by means of normal derivatives. Further studies should include alternative approaches of disjunctive programming semantics, for example the possible model semantics by Sakama and Inoue [7] where also inclusive interpretations of disjunctions are possible.

We have combined quantitative logic programs and disjunctive logic programs to quantitative disjunctive logic programs and assigned level mapping characterizations to this combinations of programs. Thereby, we have seen that the level mapping characterizations for quantitative disjunctive programs are also just a combination of the characterizations of the two basic programs. Further studies will have to show whether this property remains valid when semantic approaches different from the ones used in this paper are applied.

References

[1] Fages, F.: Consistency of Clark's completion and existence of stable models. Journal of Methods of Logic in Computer Science 1 (1994) pp. 51–60

[2] Fitting, M.: A Kripke-Kleene-semantics for general logic programs. The Journal of Logic Programming 2 (1985) pp. 295–312

[3] Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In Kowalski, R.A., Bowen, K.A., eds.: Logic Programming. Proceedings of the 5th International Conference and Symposium on Logic Programming, MIT Press (1988) pp. 1070–1080

[4] Hitzler, P., Seda, A.K.: Multivalued mappings, fixed-point theorems and disjunctive databases. 3rd Irish Workshop on Formal Methods (IWFM-99), EWIC, British Computer Society (1999)

[5] Hitzler, P., Wendt, M.: The well-founded semantics is a stratified Fitting semantics. In Jarke, M., Koehler, J., Lakemeyer, G., eds.: Proceedings of the 25th Annual German Conference on Artificial Intelligence, KI2002, Aachen, Germany, September 2002. Volume 2479 of Lecture Notes in Artificial Intelligence, Springer, Berlin (2002) pp. 205–221

[6] Hitzler, P., Wendt, M.: A uniform approach to logic programming semantics. Technical Report WV–02-14, Knowledge Representation and Reasoning Group, Department of Computer Science, Dresden University of Technology (2002)

[7] Inoue, K., Sakama, C.: An alternative approach to the semantics of disjunctive logic programs and deductive databases. Journal of Automated Reasoning 13 (1994) pp. 145–172

[8] Lloyd, J.W.: Foundations of logic programming. 2nd edition, Springer, Berlin (1988)

[9] Lobo, J., Minker, J., Rajasekar, A.: Foundations of disjunctive logic programming. MIT Press, Cambridge (1992)

[10] Mateis, C.: Extending disjunctive logic programming by T-norms. In Proceedings of the Fifth International Conference on Logic Programming and Nonmonotonic Reasoning, LPNMR 99 (1999) pp. 290–304

[11] Minker, J.: On indefinite databases and the closed world assumption. In Lecture Notes in Computer Science 138, Springer-Verlag (1982) pp. 292–308

[12] Minker, J., Rajasekar A.: A fixpoint semantics for disjunctive logic programs. Journal of Logic Programming, 9(1) (1990) pp. 45–74

[13] Przymusinska, H., Przymusinski, T.C.: Weakly stratified logic programs. Fundamenta Inforamticae 13 (1990) pp. 51–65

[14] Przymusinski, T.C.: Stable semantics for disjunctive programs. New Generation Computing 9 (3&4) (1991) pp. 401–424

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

[16] van Gelder, A.: The alternating fixpoint of logic programs with negation. In Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, ACM Press (1989) pp. 1–10

[17] van Gelder, A., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. Journal of the ACM 38(3)(1991) pp. 620–650