Approximation Fixpoint Theory as a Unifying Framework for Fuzzy Logic Programming Semantics

Pascal Kettmann[1] , **Jesse Heyninck**[2,3] , **Hannes Strass**[1,4]
[1]TU Dresden, Germany
[2]Open Universiteit, The Netherlands
[3]University of Cape Town, South Africa
[4]ScaDS.AI Center for Scalable Data Analytics and Artificial Intelligence, Dresden/Leipzig, Germany
{pascal.kettmann, hannes.strass}@tu-dresden.de, jesse.heyninck@ou.nl

Abstract

Fuzzy logic programming is an established approach for reasoning under uncertainty. Several semantics from classical, two-valued logic programming have been generalized to the case of fuzzy logic programs. In this paper, we show that two of the most prominent classical semantics, namely the stable model and the well-founded semantics, can be reconstructed within the general framework of approximation fixpoint theory (AFT).

This not only widens the scope of AFT from two- to many-valued logics, but allows a wide range of existing AFT results to be applied to fuzzy logic programming. As first examples of such applications, we clarify the formal relationship between existing semantics, generalize the notion of stratification from classical to fuzzy logic programs, and devise "more precise" variants of the semantics.

1 Introduction

Logic programs [Kowalski and Kuehner, 1971; Lloyd, 1987] are an established language not only for programming, but also for knowledge representation and reasoning. However, in classical logic programs, the dichotomy of allowing propositions to only be either true or false can limit expressivity. More precisely, logic programs under two-valued semantics cannot easily represent knowledge that is imprecise, vague, or uncertain. To address this limitation, Lee [1972] proposed to generalize logic programs to enable the use of *fuzzy logic* in its semantics, which among other things means allowing to use truth values from the real unit interval $[0, 1]$.

This initial proposal of *fuzzy logic programs* (FLPs) has led to a considerable body of research that continues growing to this day [Shapiro, 1983; van Emden, 1986; Subrahmanian, 1987; Vojtáš, 2001; Medina *et al.*, 2001; Loyer and Straccia, 2002; Loyer and Straccia, 2003; Loyer and Straccia, 2009a; Cornejo *et al.*, 2018; Cornejo *et al.*, 2020], with an increasing interest in frameworks that abstract away from concrete sets of truth values and instead require only some algebraic properties on them, thus also encompassing the classical, two-valued version as a special case. However, there is no uniform approach to syntax, e.g. what kinds of connectives are allowed in rule bodies, and also no universally accepted "standard" semantics. Instead, several proposals exist, e.g. the approximate well-founded semantics by Loyer and Straccia [2009b] or the stable model semantics by Cornejo *et al.* [2018], but it is not clear how they are related.

In the present paper, we address some of these open questions within the general algebraic framework of *approximation fixpoint theory* (AFT). For classical logic programming [van Emden and Kowalski, 1976], it is known that the (standard) least-model semantics of definite logic programs can be defined using an operator on interpretations. In a series of papers, Denecker *et al.* demonstrated that similar constructions are possible for a range of knowledge representation formalisms, obtaining several profound results that uncovered fundamental similarities between semantics of different knowledge representation languages [Denecker *et al.*, 2000; Denecker *et al.*, 2003; Denecker *et al.*, 2004]. Today, approximation fixpoint theory stands as a unifying framework for studying fixpoint-based semantics, in particular in the context of languages allowing for non-monotonicity.

More concretely, in this paper we show that several fuzzy logic programming semantics from the literature can be incorporated into the framework of approximation fixpoint theory:

1. the approximate well-founded semantics by Loyer and Straccia [2009b], a generalization of the well-founded semantics for classical logic programming [van Gelder *et al.*, 1991]; and
2. the stable model semantics by Cornejo *et al.* [2018], a generalization of the classical stable model semantics [Gelfond and Lifschitz, 1988].

In fact, we will show that both semantics can be obtained *by a single operator* using standard constructions from approximation fixpoint theory. As direct results, we obtain how the two semantics are related, and that we can also apply other known techniques from AFT directly to fuzzy logic programming, which we demonstrate by generalizing stratification and ultimate approximation. We are not aware of any existing reconstructions of fuzzy logic programming semantics within approximation fixpoint theory; perhaps the closest related work is that on weighted abstract dialectical frameworks [Brewka *et al.*, 2018] (that are conceptually similar to fuzzy logic programs, but with a different area of application) with an existing treatment in AFT [Bogaerts, 2019].

The paper proceeds as follows: We next give necessary background on approximation fixpoint theory (insofar it is relevant for logic programming) and fuzzy logic programming. Afterwards, we define the concrete syntax of fuzzy logic programs and define the operator that we subsequently use to reconstruct the well-founded semantics (Section 4) and the stable model semantics (Section 5). In Section 6, we showcase some applications of AFT to fuzzy logic programming, reaping the immediate benefits of our reconstruction. Finally, we discuss some avenues for further work and conclude.

2 Background

In this section, we briefly introduce AFT, cover basic fuzzy logic principles, and give an overview of (truth-functional) fuzzy logic programming, focusing on the operator defined by Vojtáš [2001] and many-valued modus ponens.

2.1 Approximation Fixpoint Theory

A particular way of defining semantics for programming languages is to associate a program P with a *transformation* operator T_P that transforms a given input-output relation R into an updated relation $T_P(R)$. In this view, the semantics of program P is given by the least fixpoint of the operator T_P. Van Emden and Kowalski [1976] applied this way of defining semantics to the field of logic programming, developing the first one-step logical consequence operator and employing its monotonicity in conjunction with Tarski's fixpoint theorem [1955] to define the intended semantics of definite logic programs with potentially recursive predicate definitions.

More formally, an *operator* is a (total) function $O\colon V \to V$ on a set V. We are typically interested in operators on sets V with an internal structure: At the minimum, we require $(V, \leqslant)$ to be a partially ordered set. More often, $(V, \leqslant)$ will be a *complete lattice*, that is, such that every subset $S \subseteq V$ has a *least upper bound (lub)* $\bigvee S \in V$ and *greatest lower bound (glb)* $\bigwedge S \in V$; this entails that $(V, \leqslant)$ has both a least element $0 = \bigwedge V = \bigvee \emptyset$ and a greatest element $1 = \bigvee V = \bigwedge \emptyset$. An operator $O\colon V \to V$ is *monotone* iff $x \leqslant y$ implies $O(x) \leqslant O(y)$, and *antimonotone* iff $x \leqslant y$ implies $O(y) \leqslant O(x)$. An element $x \in V$ with $O(x) = x$ is a *fixpoint* of O; if $O(x) \leqslant x$ then x is a *pre-fixpoint* of O; if $x \leqslant O(x)$ then x is a *post-fixpoint* of O. A fundamental result in lattice theory, Tarski's fixpoint theorem [1955], states that whenever O is a monotone operator on a complete lattice $(V, \leqslant)$, the set $\{x \in V \mid O(x) = x\}$ of its fixpoints forms a complete lattice itself, and so has a least element $\mathit{lfp}(O)$. While this result is immensely useful, it is not constructive; however, constructive versions exist and tell us how to actually construct least fixpoints [Markowsky, 1976; Cousot and Cousot, 1979]: To this end, one defines, for any $x \in V$, the iterated (transfinite) application of O to x via $O^0(x) := x$, $O^{\alpha+1}(x) := O(O^\alpha(x))$ for successor ordinals, and $O^\beta(x) := \bigvee\{O^\alpha(x) \mid \alpha < \beta\}$ for limit ordinals. It is then known that (for monotone O) there exists an ordinal α such that $O^\alpha(0) = \mathit{lfp}(O)$ [Bourbaki, 1949].

In order to apply Tarski's fixpoint theorem [1955] (or its constructive versions), it is thus necessary to have an operator that is monotone, a condition that is no longer fulfilled when extending the syntax of logic programs from *definite* to *normal* logic programs, i.e., when allowing "negation as failure" to occur in bodies of rules. To still have a useful fixpoint theory in the case of non-monotone operators, Denecker *et al.* [2000] fundamentally generalized the theory underlying the fixpoint-based approach to semantics, thereby founding what is now known as *approximation fixpoint theory*. Its underlying idea is that when an operator of interest does not have properties that guarantee the existence of fixpoints, then this operator can be approximated in a more fine-grained algebraic structure where fixpoint existence can be guaranteed.

Formally – following ideas of Belnap [1977], Ginsberg [1988], and Fitting [2002] –, Denecker *et al.* [2000] moved from a complete lattice $(V, \leqslant)$ to its associated *bilattice* on the set $V^2 := V \times V$. So where elements of V correspond to candidates for the semantics (interpretations), the pairs contained in V^2 correspond to *approximations* of such candidates. More technically, a pair $(x, y) \in V^2$ approximates all $z \in V$ with $x \leqslant z \leqslant y$. A pair $(x, y) \in V^2$ is called *consistent* iff $x \leqslant y$, in other words, if the interval $[x, y] := \{z \in V \mid x \leqslant z \leqslant y\}$ is non-empty; a pair of the form (x, x) is *exact*. There are two natural orderings on V^2:

- $(x_1, y_1) \leq_t (x_2, y_2) :\iff x_1 \leqslant x_2$ and $y_1 \leqslant y_2$, the *truth ordering* extending the (truth) lattice ordering $\leqslant$;
- $(x_1, y_1) \leq_p (x_2, y_2) :\iff x_1 \leqslant x_2$ and $y_2 \leqslant y_1$, the *precision ordering* comparing intervals by precision of approximation. The greatest lower bound induced by $\leq_p$ is denoted by $\otimes$ and the least upper bound by $\oplus$, where we have $(x_1, y_1) \otimes (x_2, y_2) = (x_1 \wedge x_2, y_1 \vee y_2)$ and $(x_1, y_1) \oplus (x_2, y_2) = (x_1 \vee x_2, y_1 \wedge y_2)$.

Just as pairs of V^2 approximate elements of V, the main idea of Denecker *et al.* [2000] was that operators $O\colon V \to V$ can be approximated by operators on V^2: An *approximator* is an operator $\mathcal{A}\colon V^2 \to V^2$ that is $\leq_p$-monotone and that maps exact pairs to exact pairs. By virtue of the latter condition, we say that $\mathcal{A}$ *approximates* the operator $O\colon V \to V$ with $O(x) = \mathcal{A}(x, x)_1$ (where $(x, y)_1 = x$ and $(x, y)_2 = y$); by virtue of the first condition, $\mathcal{A}$ is guaranteed to possess ($\leq_p$-least) fixpoints. An approximator $\mathcal{A}$ is *symmetric* iff for all $(x, y) \in V^2$, we have $\mathcal{A}(x, y)_2 = \mathcal{A}(y, x)_1$; thus symmetry allows to specify $\mathcal{A}$ by giving only $\mathcal{A}(\cdot, \cdot)_1$.

The main contribution of Denecker *et al.* [2000] was the association of the *stable approximator* $\mathcal{A}^{st}$ to an approximator $\mathcal{A}$: For a complete lattice $(V, \leqslant)$ and an approximator $\mathcal{A}\colon V^2 \to V^2$, define the *stable approximator for* $\mathcal{A}$ as $\mathcal{A}^{st}\colon V^2 \to V^2$ by $\mathcal{A}^{st}(x, y) := (\mathit{lfp}(\mathcal{A}(\cdot, y)_1), \mathit{lfp}(\mathcal{A}(x, \cdot)_2))$, where $\mathcal{A}(\cdot, y)_1$ (resp. $\mathcal{A}(x, \cdot)_2$) denotes the operator that maps any $z \in V$ to $\mathcal{A}(z, y)_1$ (resp. to $\mathcal{A}(x, z)_2$). Among other results, Denecker *et al.* [2000] also showed that this construction is well-defined because both relevant operators $\mathcal{A}(\cdot, y)_1\colon V \to V$ and $\mathcal{A}(x, \cdot)_2\colon V \to V$ are $\leqslant$-monotone (as $\mathcal{A}$ is an approximator) and thus possess least fixpoints. Furthermore, any approximator maps consistent pairs to consistent pairs, and the stable approximator $\mathcal{A}^{st}$ maps consistent post-fixpoints of $\mathcal{A}$ to consistent pairs. Consequently, not only is $\mathit{lfp}(\mathcal{A})$ consistent (and called the *Kripke-Kleene fixpoint* of $\mathcal{A}$), but so is $\mathit{lfp}(\mathcal{A}^{st})$, the *well-founded fixpoint* of $\mathcal{A}$. Moreover,

a fixpoint (x, y) of $\mathcal{A}^{st}$, called a *stable fixpoint* of $\mathcal{A}$, is always a $\leqslant$-minimal fixpoint of O (when $\mathcal{A}$ approximates O). The term "stable" is not coincidental: When using the definite logic program transformation operator T_P by van Emden and Kowalski [1976] as O and defining a suitable approximator $\mathcal{A}$ (see Section 3 for a generalization of such an approximator), then the exact fixpoints of $\mathcal{A}^{st}$ correspond one-to-one to the stable models of the logic program P [Denecker *et al.*, 2000].

2.2 Fuzzy Logic Programming

Let us first define the syntax of *positive* fuzzy formulas, a generalization of the conjunctions of literals that appear in classical definite logic programs. Given a complete lattice $(V, \leqslant)$ of truth values, a set Π of propositional atoms, and a family $\mathcal{F}$ of fuzzy connectives (where each $f \in \mathcal{F}$ has an associated *arity* $n \in \mathbb{N}$ which we denote by $f^{(n)}$ when important), we designate a finitely representable subset $V' \subseteq V$ of truth values that are allowed to be explicitly mentioned in programs,[1] and define the syntax of a *positive fuzzy formula* via

$$\varphi ::= c \mid p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid @(\varphi, \dots, \varphi) \tag{1}$$

where $c \in V'$, $p \in \Pi$, and $\wedge, \vee, @ \in \mathcal{F}$, to which we refer as conjunctors, disjunctors, and aggregators, respectively. A *positive logic program rule* is an expression of the form

$$p \overset{\vartheta}{\leftarrow} \mathcal{B} \tag{2}$$

where $p \in \Pi$, $\vartheta \in V'$, $\leftarrow$ is an implicator and $\mathcal{B}$ is a positive fuzzy formula. The atom p is called the *head* of the rule, $\mathcal{B}$ is called the *body* of the rule and ϑ its *weight* (whenever $\vartheta = 1$, we will write a rule as $p \leftarrow \mathcal{B}$). A *positive fuzzy logic program* P is then a finite set of rules (2).

Every connective $f^{(n)} \in \mathcal{F}$ has an associated truth function $f^{\cdot} : V^n \to V$ on the complete lattice $(V, \leqslant)$ of truth values. The truth functions of the (positive) connectives in our language are assumed to satisfy the following properties:

- for each $f \in \mathcal{F}$, $f^{\cdot}$ is $\leqslant$-monotone in each argument;
- for each conjunctor $\wedge \in \mathcal{F}$, its truth function $\wedge^{\cdot}$ satisfies $p_1 \wedge^{\cdot} p_2 \leqslant p_1, p_1 \wedge^{\cdot} p_2 \leqslant p_2$, and $p \wedge^{\cdot} 1 = p = 1 \wedge^{\cdot} p$;
- for each disjunctor $\vee \in \mathcal{F}$, its truth function $\vee^{\cdot}$ satisfies $p_1 \leqslant p_1 \vee^{\cdot} p_2, p_2 \leqslant p_1 \vee^{\cdot} p_2$, and $p \vee^{\cdot} 0 = p = 0 \vee^{\cdot} p$;
- for each implicator $\leftarrow \in \mathcal{F}$, $\leftarrow^{\cdot}$ is monotone in the first argument and antimonotone in the second argument.

If $\wedge^{\cdot}$ is also associative and commutative and $V = [0, 1] \subseteq \mathbb{R}$, then $\wedge^{\cdot}$ is called a *t-norm*. However, we do not make any assumptions on associativity, commutativity, or continuity of truth functions in this paper. An example of truth functions are Gödel's, with $x \wedge^{\cdot}_G y = \min\{x, y\}$, $x \vee^{\cdot}_G y = \max\{x, y\}$, and $x \leftarrow^{\cdot}_G y = x$ if $y > x$ and $x \leftarrow^{\cdot}_G y = 1$ otherwise.

For a complete lattice $(V, \leqslant)$ of truth values and Π a set of atoms, an *interpretation* is a total function $I \colon \Pi \to V$. The set of all interpretations is denoted by $\mathcal{I}$. It can be shown (and is known) that $(\mathcal{I}, \leq)$ again is a complete lattice, where $I \leq J$ iff $I(p) \leqslant J(p)$ for all $p \in \Pi$. We denote the least and greatest elements of this lattice by $\mathbf{0} := \{p \mapsto 0 \mid p \in \Pi\}$ and $\mathbf{1} := \{p \mapsto 1 \mid p \in \Pi\}$, respectively. Later, we will work on the associated bilattice $\mathcal{I}^2$ with $\leq_p$-least element $(\mathbf{0}, \mathbf{1})$; glb $\otimes$ and lub $\oplus$ carry over from V^2 to $\mathcal{I}^2$ in a pointwise manner, e.g. $((L, U) \otimes (L', U'))(p) := (L, U)(p) \otimes (L', U')(p)$.

We work with truth-functional logic in a narrow sense, whence the truth value of a formula is uniquely determined by the truth value of its constituents. Given an interpretation I, we thus extend the interpretation to arbitrary formulas φ via structural induction, and denote the truth value of φ under I by $\hat{I}(\varphi)$. More precisely, for connective $f^{(n)} \in \mathcal{F}$, we define $\hat{I}(f(\varphi_1, \dots, \varphi_n)) := f^{\cdot}(\hat{I}(\varphi_1), \dots, \hat{I}(\varphi_n))$. For example, for Gödel logic we obtain $\hat{I}(x \wedge_G y) = I(x) \wedge^{\cdot}_G I(y)$.

We next define how the truth value $\hat{I}(\mathcal{B})$ of a body of a rule is propagated towards the head p. The inference rule proposed by Vojtáš [2001] for fuzzy logic programming is based on many-valued modus ponens (Hájek [1998] gives details):

$$\frac{p \leftarrow \varphi \quad \varphi}{p} \tag{MP}$$

(MP) should infer a truth value z of the conclusion p when given a truth value x of φ and a truth value $y = z \leftarrow^{\cdot} x$ of the implication $p \leftarrow \varphi$. Let mp denote a candidate function, i.e. a way to compute $mp(x, y) = z$. To ensure soundness of the inference, mp should be monotone in both of its arguments (increasing truth values of the premises lead to higher truth in the conclusion) and satisfy the boundary conditions $mp(0, 1) = mp(1, 0) = 0$ and $mp(1, 1) = 1$. These are the properties of a truth function of a conjunctor $\wedge$. If in addition the truth value of $z \leftarrow^{\cdot} x$ is as large as possible, we get

$$x \wedge^{\cdot} y \leqslant z \text{ iff } y \leqslant z \leftarrow^{\cdot} x. \tag{3}$$

A pair $(\leftarrow^{\cdot}, \wedge^{\cdot})$ that satisfies (3) for all $x, y, z \in V$ of a partially ordered set $(V, \leqslant)$ is then called an *adjoint pair* [Medina *et al.*, 2001]. In order to designate adjoint pairs, we will write them with the same index, i.e. as $(\leftarrow_i, \wedge_i)$. Intuitively, adjoint pairs guarantee correctness of inference whenever we pair "matching" implicators and conjunctors, e.g. $(\leftarrow_G, \wedge_G)$.

Given an interpretation $I \in \mathcal{I}$, a weighted rule $p \overset{\vartheta}{\leftarrow}_i \mathcal{B}$ is *satisfied* by I iff $\vartheta \leqslant I(p) \leftarrow^{\cdot}_i \hat{I}(\mathcal{B})$. An interpretation I is a *model* of P iff it satisfies all rules in P. Intuitively, I satisfies a rule if it makes the rule "at least as true" as its weight ϑ. By (3), this translates to: the head is "at least as true" as the body limited by the rule's weight, or formally $\vartheta \wedge^{\cdot}_i \hat{I}(\mathcal{B}) \leqslant I(p)$.

Vojtáš [2001] and Medina *et al.* [2001] define a fuzzy generalization of the transformation operator given by van Emden and Kowalski [1976]: For a fuzzy logic program P, the immediate consequence operator $T_P \colon \mathcal{I} \to \mathcal{I}$ is defined by

$$T_P(I)(p) = \bigvee \left\{ \vartheta \wedge^{\cdot}_i \hat{I}(\mathcal{B}) \;\middle|\; p \overset{\vartheta}{\leftarrow}_i \mathcal{B} \in P \right\} \tag{4}$$

[1] Allowing truth values to occur explicitly in formulas is with the intended usage of being able to represent *some* truth values in rule bodies as we do in Example 2. This does not necessarily work for arbitrary sets of truth values (e.g. it does not work for $V = [0, 1] \subseteq \mathbb{R}$ by sheer cardinality) – in those cases we want to allow to restrict the syntax to a meaningful subset of truth values that can be finitely represented, e.g. the rational unit interval $V' = [0, 1] \cap \mathbb{Q}$.

The semantics of a positive FLP P can then be characterized by the pre-fixpoints of T_P [Vojtáš, 2001, Theorem 2.2], i.e. an interpretation $I \in \mathcal{I}$ is a model of P iff $T_P(I) \leq I$. This generalizes a classical result by van Emden and Kowalski [1976].

Example 1. Consider the positive fuzzy logic program $P = \{r \leftarrow_G 0.3 \vee_G (s \wedge_G 0.6), s \leftarrow_G s\}$ over $([0,1], \leqslant)$. Any interpretation I with $0.3 \leqslant I(r)$ is a model of P. The *least* model of P can be obtained as the least fixpoint of T_P via transfinitely iterating T_P on $\mathbf{0}$, leading to the interpretation $T_P(\mathbf{0}) = \{r \mapsto 0.3, s \mapsto 0\} = I_1 = T_P(I_1) = \mathit{lfp}(T_P)$.

3 An Approximator for FLPs

The main challenge in defining semantics for (fuzzy) logic programs lies in the treatment of "negation as failure". Syntactically, we now move to *normal* fuzzy logic programs, that is, we extend the syntax of positive fuzzy formulas by *default negation* $\sim$, a unary connective of which we only require that its truth function $\sim\colon V \to V$ is an antimonotone involution [Ovchinnikov, 1983]. ($\sim$ being an involution means that $\sim\sim v = v$ for all $v \in V$; it follows that $\sim 0 = 1$ and $\sim 1 = 0$ whence $\sim$ is a generalization of two-valued negation.) For $V = [0,1] \subseteq \mathbb{R}$, the typical choice is $\sim x = 1 - x$, which we will use in all subsequent examples.

Formally, a *normal fuzzy formula* is of the form

$$\varphi ::= c \mid p \mid {\sim}p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid @(\varphi, \ldots, \varphi) \qquad (5)$$

where $c \in V'$, $p \in \Pi$, and $\wedge, \vee, @ \in \mathcal{F}$ as before. A *normal fuzzy logic program* is then a finite set of normal fuzzy rules, and the remaining notions involving "normal" carry over from "positive" as expected. In particular, to define the evaluation of a normal fuzzy formula φ under interpretation $I \in \mathcal{I}$, we have the additional inductive case $\hat{I}(\sim\varphi) = \sim \hat{I}(\varphi)$.

The one-step consequence operator T_P [Vojtáš, 2001; Medina *et al.*, 2001] is likewise extended to the case of normal FLPs, with models of P and pre-fixpoints of T_P still being in one-to-one correspondence. However, the resulting operator T_P is no longer $\leq$-monotone and so cannot directly be used to define a least-model semantics for normal programs.

Example 2. Consider the following normal fuzzy logic program P over $V = [0,1]$ (due to Loyer and Straccia [2009b]):

$$p \leftarrow_G {\sim}q \vee_G r, \qquad q \leftarrow_G {\sim}p \vee_G s,$$
$$r \leftarrow_G 0.3, \vee_G (s \wedge_G 0.6), \qquad s \leftarrow_G s.$$

Any fuzzy interpretation I with $0.3 \leqslant I(r) \leqslant I(p)$, $I(s) \leqslant I(q)$, and $1 - I(p) \leqslant I(q)$ is a model of P. Whenever $I(p) + I(q) = 1$, $0.3 \leqslant I(p)$, $I(r) = 0.3$, and $I(s) = 0$, then I is also a *minimal* model. Consequently, P has an infinite number of minimal models.

Approximation fixpoint theory fortunately provides an elegant solution to this problem – all we have to do is define an approximator to T_P, and AFT does the rest.

Definition 1 (Approximator). For a normal FLP P over complete lattice $(V, \leqslant)$, we associate the symmetric approximator $\mathcal{T}_P\colon \mathcal{I}^2 \to \mathcal{I}^2$ with

$$\mathcal{T}_P(L,U)_1(p) = \bigvee \left\{ \vartheta \wedge_i \widehat{(L,U)}(\mathcal{B}) \;\middle|\; p \xleftarrow{\vartheta}_i \mathcal{B} \in P \right\}$$

where $\widehat{(L,U)}(\mathcal{B})$, the evaluation of $\mathcal{B}$ under interpretation pair $(L,U) \in \mathcal{I}^2$, is defined via structural induction with notable base cases $\widehat{(L,U)}(p) := L(p)$ and $\widehat{(L,U)}(\sim p) := \sim U(p)$, and straightforward inductive case $\widehat{(L,U)}(f(\varphi_1, \ldots, \varphi_n)) := f\left(\widehat{(L,U)}(\varphi_1), \ldots, \widehat{(L,U)}(\varphi_n)\right)$.

Thus $\mathcal{T}_P(L,U) = (\mathcal{T}_P(L,U)_1, \mathcal{T}_P(U,L)_1)$ gives the overall approximator. Figure 1 (p. 6) shows how the approximator maps (pairs of) interpretations to others for Example 2.

It is easy to see that $\mathcal{T}_P$ is well-defined to serve its purpose:

Proposition 1. *$\mathcal{T}_P$ approximates T_P and is $\leq_p$-monotone.*

Thus $\mathcal{T}_P$ also approximates the operator by Vojtáš [2001] and generalizes its semantics. Additionally, using the standard definitions of approximation fixpoint theory, Definition 1 also indirectly yields the *well-founded semantics* of P via $\mathit{lfp}(\mathcal{T}_P^{st})$, as well as the *stable model semantics* via the exact fixpoints of $\mathcal{T}_P^{st}$. In the remainder of the paper, we will relate these semantics (as obtained by AFT directly) to similar semantics that have been manually defined in the literature.

4 Well-Founded Semantics

In this section we show that AFT captures the well-founded semantics for fuzzy logic programs as defined by Loyer and Straccia [2009a]. The first (minor) technical hurdle is that where AFT uses pairs (L,U) of fuzzy interpretations, Loyer and Straccia [2009a] use interpretations that assign a *pair* of truth values to each atom $p \in \Pi$. More formally, an *approximate interpretation* is a total function $X\colon \Pi \to V \times V$; the set of all approximate interpretations is denoted by $\mathcal{C}$. Loyer and Straccia [2009a] then extend approximate interpretations to normal fuzzy formulas: Firstly, they extend the truth functions of connectives $f^{(n)} \in \mathcal{F}$ from V to $V \times V$ via $f^{\cdot\cdot}((\ell_1,u_1), \ldots, (\ell_n,u_n)) := (f^{\cdot}(\ell_1, \ldots, \ell_n), f^{\cdot}(u_1, \ldots, u_n))$, and set $\sim^{\cdot\cdot}((\ell,u)) := (\sim^{\cdot} u, \sim^{\cdot} \ell)$; secondly, they define $\hat{X}(\varphi)$ by structural induction on formulas φ (including default negation $\sim$), with base cases $\hat{X}(p) = X(p)$ and $\hat{X}(c) = (c,c)$ for $c \in V'$. Loyer and Straccia [2009a] then define an operator on approximate interpretations via $\mathcal{S}_P(X)(p) = \hat{X}(\mathcal{B}_p)$ where they assume that every $p \in \Pi$ has a unique rule $p \leftarrow \mathcal{B}_p \in P$ (which is not a restriction because several rules for p can be joined with disjunction $\vee$). It is clear that the set $\mathcal{C}$ of approximate interpretations and the set $\mathcal{I}^2$ of pairs (L,U) of fuzzy interpretations are isomorphic (whence $\otimes$ and $\oplus$ carry over to $\mathcal{C}$); for a given $X \in \mathcal{C}$, we use $\zeta(X)$ to denote (L,U) such that $X(p) = (L(p), U(p))$ for all $p \in \Pi$. As our first result in this section, it then follows that modulo this isomorphism on the underlying structures, the operator $\mathcal{S}_P$ by Loyer and Straccia [2009a] and our approximator $\mathcal{T}_P$ of Definition 1 coincide.

Lemma 2. *For any FLP P and any approximate interpretation X over Π, we have $\zeta(\mathcal{S}_P(X)) = \mathcal{T}_P(\zeta(X))$.*

This result paves the way for our subsequent formal comparison of the well-founded semantics defined by Loyer and Straccia [2009a] and the well-founded semantics given by approximation fixpoint theory [Denecker *et al.*, 2000]. The

next definition makes use of two special interpretations: $X_{\mathtt{f}}$ is such that $\zeta(X_{\mathtt{f}}) = (\mathbf{0}, \mathbf{0})$ (it assigns false everywhere), and $X_{\bot}$ is such that $\zeta(X_{\bot}) = (\mathbf{0}, \mathbf{1})$ (it assigns unknown everywhere). The *closed world operator* s_P is used to construct the well-founded semantics. We will use the following characterization [Loyer and Straccia, 2009a, Theorem 4]: Given an approximate interpretation X, $s_P(X)$ is defined as the least fixpoint of the function $F_{P,X}(Y) := X_{\mathtt{f}} \otimes \mathcal{S}_P(X \oplus Y)$, that is, $s_P(X) := lfp_{\leq_t}(F_{P,X})$. The *approximate well-founded operator* is then defined as: $\mathcal{AW}_P(X) = \mathcal{S}_P(X) \oplus s_P(X)$. Finally, the *approximate well-founded model* is $lfp_{\leq_p}(\mathcal{AW}_P)$.

Lemma 3. *For any FLP P and any approximate interpretation X over Π with $\zeta(X) = (L, U)$, we have that $\zeta(s_P(X)) = (\mathbf{0}, lfp(\mathcal{T}_P(L, \cdot)_2))$.*

We now show that the approximate well-founded model is identical to the well-founded fixpoint of $\mathcal{T}_P$. Intuitively, we prove this by relating the paths that the operators $\mathcal{AW}_P$ and $\mathcal{T}_P^{st}$ traverse when applied iteratively to $X_{\bot} \hat{=} (\mathbf{0}, \mathbf{1})$: every pair visited by $\mathcal{T}_P^{st}$ is also visited by $\mathcal{AW}_P$, but not necessarily vice versa. (This can also be seen in Figure 1.)

Theorem 4. *For any FLP P with approximate well-founded model X_{SWF}, we have $\zeta(X_{\mathsf{SWF}}) = lfp(\mathcal{T}_P^{st})$.*

Proof. *In what follows, we will denote the sequence used to construct $lfp_{\leq_p}(\mathcal{AW}_P)$ by $\langle X_i \rangle_{i \leq \alpha}$, and $\zeta(X_i)$ by (L_i, U_i) (for any $i \leq \alpha$).*[2] *We prove the theorem by showing that $\langle (L_i, U_i) \rangle_{i \leq \alpha}$ is a* terminal monotone induction *of $\mathcal{T}_P^{st}$, i.e. a sequence $\langle (L_i, U_i) \rangle_{i \leq \alpha}$ such that:*

1. *$(L_0, U_0) = (\mathbf{0}, \mathbf{1})$,*
2. *$(L_i, U_i) <_p (L_{i+1}, U_{i+1}) \leq_p \mathcal{T}_P^{st}(L_i, U_i)$ for all $i < \alpha$,*
3. *$(L_\sigma, U_\sigma) = \bigoplus(\{(L_i, U_i) \mid i < \sigma\})$, for every limit ordinal $\sigma < \alpha$,*
4. *there is no $(L_{\alpha+1}, L_{\alpha+1})$ such that $\langle (L_i, U_i) \rangle_{i \leq \alpha+1}$ is a monotone induction.*

It then follows with [Denecker and Vennekens, 2007, Proposition 1] that $\zeta(lfp_{\leq_p}(\mathcal{AW}_P)) = lfp(\mathcal{T}_P^{st})$.We denote $lfp(\mathcal{T}_P(L, \cdot)_2)$ by $c\mathcal{T}_P^{st}(L)$ to avoid clutter in what follows.

Ad 1: Clear from the definition of $\zeta(lfp_{\leq_p}(\mathcal{AW}_P))$.

Ad 2: We first show by induction that (†) $\langle (L_{i+1}, U_{i+1}) \rangle = (\mathcal{T}_P(L_i, U_i)_1, c\mathcal{T}_P^{st}(L_i))$. The base case is trivial. For the inductive case, suppose that $\langle (L_{i+1}, U_{i+1}) \rangle = (\mathcal{T}_P(L_i, U_i)_1, c\mathcal{T}_P^{st}(L_i))$. We show that $\langle (L_{i+2}, U_{i+2}) \rangle = (\mathcal{T}_P(L_{i+1}, U_{i+1})_1, c\mathcal{T}_P^{st}(L_{i+1}))$. Indeed as $\mathcal{AW}_P(X_{i+2}) = \mathcal{S}_P(X_{i+1}) \otimes s_P(X_{i+1})$ and $\zeta(s_P(X_{i+1})) = (0, c\mathcal{T}_P^{st}(L_{i+1}))$ (Lemma 3), it is easy to see that $L_{i+2} = \mathcal{T}_P(L_{i+1}, U_{i+1})_1$. It remains to show that $U_{i+2} = c\mathcal{T}_P^{st}(L_{i+1})$. We do this by showing that $c\mathcal{T}_P^{st}(L_{i+1}) \leqslant \mathcal{T}_P(L_{i+1}, U_{i+1})$. As $L_i \leqslant L_{i+1}$ (as $\mathcal{AW}_P$ is a $\leq_p$-monotone operator [Loyer and Straccia, 2009a, Theorem 7]) and $\mathcal{T}_P^{st}$ is $\leq_p$-monotone (which means that $c\mathcal{T}_P^{st}$ is $\leqslant$-anti monotone), $c\mathcal{T}_P^{st}(L_{i+1}) \leqslant c\mathcal{T}_P^{st}(L_i) = U_{i+1}$. As $\mathcal{T}_P$ is $\leq_p$-monotone, $\mathcal{T}_P(L_{i+1}, \cdot)_2$ is $\leqslant$-monotone, and thus, $\mathcal{T}_P(L_{i+1}, c\mathcal{T}_P^{st}(L_{i+1}))_2 \leq \mathcal{T}_P(L_{i+1}, U_{i+1})_2$. As $\mathcal{T}_P(L_{i+1}, c\mathcal{T}_P^{st}(L_{i+1}))_2 = c\mathcal{T}_P^{st}(L_{i+1})$ (after all, $c\mathcal{T}_P^{st}(L_{i+1})$ is the least fixed point of $c\mathcal{T}_P^{st}(L_{i+1}, \cdot)$, this concludes the proof of †.

Back to the proof of the main claim. That $X_i <_p X_{i+1}$ follows immediately from that fact that $\mathcal{AW}_P$ is a $\leq_p$-monotone operator [Loyer and Straccia, 2009a, Theorem 7]. We show that $\zeta(\mathcal{S}_P(X_i)) \leq_p \mathcal{T}_P^{st}(\zeta(X_i))$, by induction on i. The base case is clear. For the inductive case, suppose that $X_i \leq_p \mathcal{T}_P^{st}(\zeta(X_{i-1}))$. As $X_{i-1} \leq_p X_i$ (as we just showed), and $\mathcal{T}_P^{st}$ is a $<_p$-monotone operator, $\mathcal{T}_P^{st}(\zeta(X_{i-1})) \leq_p \mathcal{T}_P^{st}(\zeta(X_i))$, and thus we have that $\zeta(X_i) \leq_p \mathcal{T}_P^{st}(\zeta(X_i))$. This implies that $L_i \leqslant \mathcal{T}_P^{st}(X_i)_1$. As $\mathcal{T}_P(\cdot, U_i)_1$ is $\leqslant$-monotone, $\mathcal{T}_P(L_i, U_i)_1 \leqslant \mathcal{T}_P(\mathcal{T}_P^{st}(X_i)_1, U_i)_1$. As $\mathcal{T}_P^{st}(X_i)_1$ is the least fixpoint of $\mathcal{T}_P(\cdot, U_i)_1$, $\mathcal{T}_P(\mathcal{T}_P^{st}(X_i)_1, U_i)_1 = \mathcal{T}_P^{st}(X_i)_1$. We infer that $\mathcal{T}_P(L_i, U_i)_1 \leqslant \mathcal{T}_P^{st}(X_i)_1$, concluding the proof.

Ad 3: This follows immediately from the definition of the iterated application of an operator.

Ad 4: This is clear as we take the least fixpoint of $\zeta(\mathcal{AW}_P)$, which means that for any extension of the sequence, the condition $X_i <_p X_{i+1}$ is not satisfied. □

However, this result does not generalize to arbitrary partial stable interpretations (arbitrary fixpoints of $\mathcal{T}_P^{st}$):

Example 3. Consider the normal fuzzy logic program $P = \{p \leftarrow_G q, p \leftarrow_G p, q \leftarrow_G \sim r, r \leftarrow_G \sim q\}$. From Theorem 4, it follows that the least fixpoints of $\mathcal{T}_P^{st}$ and $\mathcal{AW}_P$ coincide (modulo isomorphism). However, the approximate interpretation $X = \{p \mapsto (1, 1), q \mapsto (0, 1), r \mapsto (0, 1)\}$ is a fixpoint of $\mathcal{AW}_P$, but $\zeta(X)$ is *not* a fixpoint of $\mathcal{T}_P^{st}$. Intuitively, $\mathcal{AW}_P$ cannot identify the positive cycle $p \leftarrow_G p$ that is needed to keep p true in X, while $\mathcal{T}_P^{st}$ *can* identify it.

5 Stable Model Semantics

We now turn our attention to the stable model semantics for normal fuzzy logic programs as defined by Cornejo *et al.* [2018]. As for classical (two-valued) normal logic programs, the notion of a stable model of a fuzzy normal logic program is associated with the least model of a positive program (whose consequence operator is monotone and thus has a unique least fixpoint). Given a normal FLP P and an interpretation I, Cornejo *et al.* [2018] introduce a mechanism to obtain the positive program P_I by substituting each rule $r = p \overset{\vartheta}{\leftarrow}_i \mathcal{B}$ in P by a rule $r_I := p \overset{\vartheta}{\leftarrow}_i \mathcal{B}_I$, where $\mathcal{B}_I$ is formally defined as follows: Firstly, to indicate which atoms are used positively or negatively in the body formula $\mathcal{B}$, we write $\mathcal{B}[p_1, \ldots, p_m, \sim p_{m+1}, \ldots, \sim p_n]$ as used by Cornejo *et al.* [2018]. Secondly, every formula $\mathcal{B}[p_1, \ldots, p_n]$ of our language leads to an n-ary aggregator $\mathcal{B}[\cdot, \ldots, \cdot]$; for example, the formula $\varphi[p, q, r] = p \wedge (q \vee r)$ leads to the aggregator $\varphi[\cdot, \cdot, \cdot]$ (that could also produce $\varphi[r, p, q] = r \wedge (p \vee q)$) with associated truth function $\varphi^{\cdot}(x_1, x_2, x_3) = x_1 \wedge^{\cdot} (x_2 \vee^{\cdot} x_3)$. This allows to define the associated truth function of $\mathcal{B}_I$ via:

$$\mathcal{B}_I^{\cdot}(x_1, \ldots, x_m) = \mathcal{B}^{\cdot}(x_1, \ldots, x_m, \sim^{\cdot} I(p_{m+1}), \ldots, \sim^{\cdot} I(p_n))$$

Intuitively, $\mathcal{B}_I$ evaluates all and only the atoms that appear *negatively* in $\mathcal{B}$ by I, and keeps the atoms that occur positively. The program $P_I := \{r_I \mid r \in P\}$ is the *reduct of P*

[2] Notice that Loyer and Straccia [2009a] do not define the approximate well-founded semantics in a constructive way, but refer to Tarski [1955] to guarantee the existence of a least fixpoint. However, as explained in Section 2.1 we can (without loss of generality) assume that this least fixpoint was constructed using an iterated (possibly transfinite) application of $\mathcal{AW}_P$.

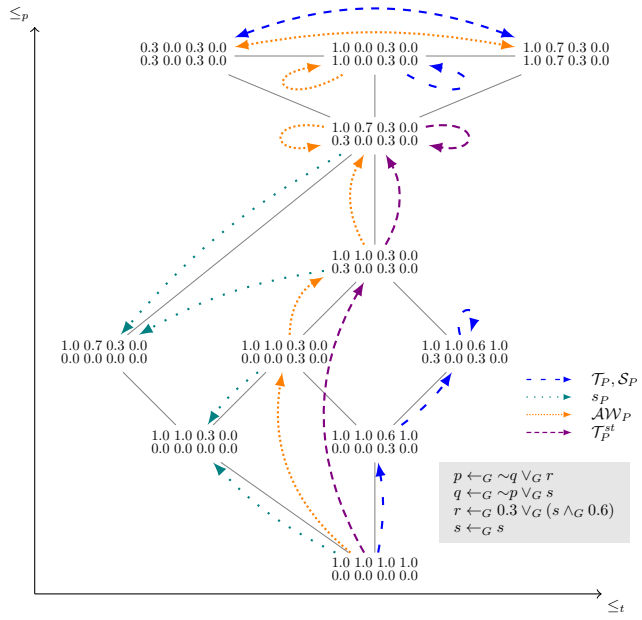

Figure 1: Parts of the bilattice of interpretations for Example 2. The bilattice elements are represented by giving lower and upper truth value bounds for p, q, r, and s, respectively; so for example $\begin{smallmatrix}1.0\ 1.0\ 0.3\ 0.0\\ 0.3\ 0.0\ 0.3\ 0.0\end{smallmatrix}$ represents the pair (L, U) with lower bound $L = \{p \mapsto 0.3, q \mapsto 0.0, r \mapsto 0.3, s \mapsto 0.0\}$ and respective upper bound $U = \{p \mapsto 1.0, q \mapsto 1.0, r \mapsto 0.3, s \mapsto 0.0\}$. An operator O is visualized by an arrow from X to $O(X)$. Thus, the well-founded model of P is found by starting from $(\mathbf{0}, \mathbf{1})$ and then following the $\mathcal{T}_P^{st}$ (or $\mathcal{AW}_P$) arrows until the least fixpoint is reached.

w.r.t. I. An $I \in \mathcal{I}$ is then a *stable model* of a normal FLP if and only if I is the least model of P_I [Cornejo *et al.*, 2018].

In the remainder of this section, we show how to reconstruct the stable model definition of Cornejo *et al.* [2018] within approximation fixpoint theory. The first result relates our approximator with Cornejo *et al.*'s definition of reduct.

Lemma 5. *For any FLP P and $I \in \mathcal{I}$, $\mathcal{T}_P(\cdot, I)_1 = T_{P_I}$.*

It is clear that equal operators have equal fixpoints:

Corollary 6. *For any $I \in \mathcal{I}$, $lfp(\mathcal{T}_P(\cdot, I)_1) = lfp(T_{P_I})$.*

This result now allows to prove the main result of this section, namely the correspondence of the two variants of definining the stable model semantics for fuzzy logic programs.

Theorem 7. *An exact interpretation (I, I) is a (AFT) stable model of P iff I is a (Cornejo* et al.*) stable model of P.*

Proof. *(I, I) is an AFT stable model of P iff $\mathcal{T}_P^{st}(I, I) = (I, I)$ iff $lfp(\mathcal{T}_P(\cdot, I)_1) = I$ iff $lfp(T_{P_I}) = I$ iff I is a Cornejo* et al. *stable model of P.* □

6 Applying AFT to Fuzzy Logic Programming

In this section, we derive several useful results that follow as straightforward corollaries from the fact that we used AFT as a unifying framework for representing the semantics of FLPs. In more detail, we investigate the relation between well-founded and stable model semantics (Section 6.1), provide results on stratification (Section 6.2), and introduce the ultimate semantics for fuzzy logic programming (Section 6.3).

6.1 Relationship Between Semantics

One of the benefits of the operator-based characterization given in this paper is that it allows us to straightforwardly derive results on the relationship between different semantics considered here. Firstly we consider the fuzzy well-founded and stable model semantics. In AFT, it is well-known that the well-founded fixpoint is an approximation of every exact stable fixpoint. Combining this insight with the characterizations developed in this paper, we obtain the following result, relating the semantics of Loyer and Straccia [2009a] and Cornejo *et al.* [2018].

Theorem 8. *For any (AFT) stable model (I, I) of P, we have $lfp(\mathcal{T}_P^{st}) \leq_p (I, I)$. Equivalently, for any (Cornejo* et al.*) stable model I of P, we have $\zeta(X_{\mathsf{SWF}}) \leq_p (I, I)$.*

Secondly, we show that the fuzzy semantics are generalizations of the classical, two-valued semantics. This is not obvious from their original definitions in the literature, but becomes obvious after our reconstruction in AFT.

Proposition 9. *Assume the complete lattice of truth values $V = \{0, 1\}$ with $0 < 1$. For every classical normal logic program P: (1) its well-founded model according to Loyer and Straccia [2009a] coincides with its well-founded model according to van Gelder* et al. *[1991]; (2) its stable models according to Cornejo* et al. *[2018] coincide one-to-one with its stable models according to Gelfond and Lifschitz [1988].*

6.2 Stratification for Fuzzy Logic Programs

We first recall some necessary preliminaries [Vennekens *et al.*, 2006], adapted to our setting.[3] Given an interpretation $I \in \mathcal{I}$, and a set of atoms $\Pi' \subseteq \Pi$, $I_{|\Pi'} : \Pi' \to V$ is defined by $I_{|\Pi'}(p) = I(p)$ for any $p \in \Pi'$. For elements (L, U) of the bilattice V^2, we denote $(L_{|\Pi'}, U_{|\Pi'})$ by $(L, U)_{|\Pi'}$. The order $\leqslant$ can likewise be restricted, denoted by $\leqslant_{|\Pi'}$. Given an operator O over $\mathcal{I}$ relative to a set of atoms Π, where Π_1, Π_2 forms a partition of Π, O is *stratifiable* over Π_1, Π_2 iff for every $I^1, I^2 \in \mathcal{I}$, if $I^1_{|\Pi_1} = I^2_{|\Pi_1}$ then $O(I^1)_{|\Pi_1} = O(I^2)_{|\Pi_1}$. Some of the main results of Vennekens *et al.* [2006] include that the the construction of (or search for) all major AFT semantics can be split up along the strata an operator is stratifiable over.

We now recall a syntactical criterion for identifying strata based on dependency graphs, again inspired by the work of Vennekens *et al.* [2006]. This criterion generalizes the well-known definition known from normal logic programs [Apt *et al.*, 1988; van Gelder, 1988]. We restrict attention to stratifications consisting of two strata for simplicity, but these results can be extended straightforwardly to arbitrary numbers of strata. We first need some additional preliminaries. Given a program P and $p, q \in \Pi$, we define $q \preceq_P p$ iff there is a $p \overset{\vartheta}{\leftarrow}_i \mathcal{B} \in P$ such that $q \in \mathcal{B}$. We then say that P is *stratifiable* over the partition Π_1, Π_2 of Π iff $q \preceq_P p$, $q \in \Pi_i$, and $p \in \Pi_j$ imply that $i \leq j$. Informally, if two atoms depend on each other, they either occur in the same stratum or the dependent atom occurs in the

[3] In the general algebraic setting of Vennekens *et al.* [2006], these definitions are introduced using the notion of a product lattice, but we could simplify these notions as done above.

higher stratum. This is an immediate generalization (at least for the case of two strata) from normal to fuzzy logic programs of the definition of splitting by Vennekens *et al.* [2006]. The traditional notion of stratification [Apt *et al.*, 1988; van Gelder, 1988] additionally requires that negatively occuring body atoms come from a *strictly lower* stratum.

Theorem 10. *For any FLP P, if P is stratifiable over Π_1, Π_2, then $\mathcal{T}_P$ is stratifiable over Π_1, Π_2.*

Proof (Sketch). Suppose that P is stratifiable over Π_1, Π_2. We show that (†): for any $(L^1, U^1), (L^2, U^2) \in V^2$, if $(L^1_{|\Pi_1}, U^1_{|\Pi_1}) = (L^2_{|\Pi_1}, U^2_{|\Pi_1})$, then $\mathcal{T}_P(L^1, U^1)_{|\Pi_1} = \mathcal{T}_P(L^2, U^2)_{|\Pi_1}$. Consider some $p \in \Pi_1$, some $p \overset{\vartheta}{\leftarrow}_i \mathcal{B} \in P$ and some $q \in \mathcal{B}$. Thus $q \preceq_P p$ and since P is stratifiable, $p \in \Pi_1$ implies $q \in \Pi_1$. Thus, we have established that $p \in \Pi_1$ implies $q \in \Pi_1$, for any $p \overset{\vartheta}{\leftarrow}_i \mathcal{B} \in P$ and $q \in \mathcal{B}$. (†) now follows immediately from the definition of $\mathcal{T}_P$. □

It follows now from the results of Vennekens *et al.* [2006] (Theorem 3.11 and Corollary 3.12), that the construction of the well-founded fixpoint and the search for stable fixpoints can be split up along the strata formed in a dependency graph. In practice, this is done by first doing the necessary computations for the lower stratum Π_1, and then transforming the program as to take into account the computed values. More technically, given an interpretation (L^1, U^1) for Π_1 and a formula $\mathcal{B}$, let $\mathcal{B}_{(L^1,U^1)}$ be the formula obtained by replacing every positive $p \in \Pi_1$ by $L^1(p)$ and every negative $\sim p$ by $\dot{\sim} U^1(p)$, and let $P_{(L^1,U^1)} = \{p \overset{\vartheta}{\leftarrow}_i \mathcal{B}_{(L^1,U^1)} \mid p \overset{\vartheta}{\leftarrow}_i \mathcal{B} \in P\}$.

Lemma 11. *Suppose that P is stratifiable over Π_1, Π_2. Given an interpretation (L, U) over $\Pi_1 \cup \Pi_2$ and an interpretation (L^1, U^1) over Π_1, if $(L, U)_{|\Pi_1} = (L^1, U^1)$ then $\mathcal{T}_{P_{(L^1,U^1)}}(L, U)_{|\Pi_2} = (\mathcal{T}_P(L, U))_{|\Pi_2}$.*

Theorem 12. *For any FLP P, if P is stratifiable over Π_1, Π_2, then (L, U) is a stable [the well-founded] fixpoint of $\mathcal{T}_P$ iff $(L_{|\Pi_1}, U_{|\Pi_1})$ is a stable [the well-founded] fixpoint of $(\mathcal{T}_P)_{|\Pi_1}$ and $(L_{|\Pi_2}, U_{|\Pi_2})$ is a stable [the well-founded] fixpoint of $\mathcal{T}_{P_{(L^1,U^1)}}$.*

6.3 Ultimate Semantics

For a given operator $O\colon V \to V$, there will typically be many different approximators $\mathcal{A}$ for O, and choosing among them might not be trivial. Denecker *et al.* [2004] devised a way to construct the most precise approximator possible: The *ultimate* approximator of O is given by $\mathcal{U}_O\colon V^2 \to V^2$ with $\mathcal{U}_O(x, y) = (\bigwedge O([x, y]), \bigvee O([x, y]))$ where we denote $O([x, y]) := \{O(z) \mid x \leqslant z \leqslant y\}$.[4] The usual AFT definitions of semantics relying on fixpoints of $\mathcal{U}^{st}_O$ then yield *ultimate* versions of the semantics, e.g. the *ultimate well-founded semantics* of $\mathcal{U}_O$ is given by $\mathit{lfp}(\mathcal{U}^{st}_O)$.

The ultimate approximation of the operator T_P on normal FLPs (Section 3) is thus given by $\mathcal{U}_{T_P}$, which we denote by $\mathcal{U}_P$ for ease of notation. The following example illustrates the potential advantages of $\mathcal{U}_P$ in comparison to $\mathcal{T}_P$.

[4]Most precise here means that for all approximators $\mathcal{A}$ of O and pairs $(x, y) \in V^2$, it holds that $\mathcal{A}(x, y) \leq_p \mathcal{U}_O(x, y)$.

Example 4. Consider for $\Pi = \{p\}$ the fuzzy logic program

$$P = \{ \quad p \leftarrow_G p, \qquad p \leftarrow_G {\sim} p \quad \}.$$

While we have $\mathcal{T}_P(\mathbf{0}, \mathbf{1}) = (\mathbf{0}, \mathbf{1})$ for our approximator from Definition 1, the ultimate approximator obtains the strictly better bounds $\mathcal{U}_P(\mathbf{0}, \mathbf{1}) = (\{p \mapsto 0.5\}, \mathbf{1}) = \mathit{lfp}(\mathcal{U}_P)$. Consequently, for the well-founded semantics, we obtain $\mathcal{T}_P(\mathbf{0}, \mathbf{1})_1 = \mathbf{0}$ and $\mathcal{T}_P(\mathbf{0}, \mathbf{0})_2 = \mathbf{1}$ whence $(\mathbf{0}, \mathbf{1})$ is the well-founded fixpoint of $\mathcal{T}_P$; in contrast, using the ultimate approximator we get $\mathcal{U}_P(\mathbf{0}, \mathbf{1})_1 = \{p \mapsto 0.5\}$ and $\mathcal{U}_P(\mathbf{0}, \mathbf{0})_2 = \mathbf{1}$ whence $(\{p \mapsto 0.5\}, \mathbf{1})$ is the well-founded fixpoint of $\mathcal{U}_P$.

For the semantics characterized by least fixpoints, it is known in AFT that the ultimate approximator leads to the most precise version of the semantics [Denecker *et al.*, 2004, Theorem 5.2]. For fuzzy logic programming, we thus get:

Proposition 13. *For any FLP P, $\mathit{lfp}(\mathcal{T}^{st}_P) \leq_p \mathit{lfp}(\mathcal{U}^{st}_P)$.*

Similarly, for semantics characterized by exact fixpoints, ultimate approximators preserve all exact (stable) models of other approximators, and may in general add new models [Denecker *et al.*, 2004, Theorem 5.3].

Proposition 14. *For any FLP P and $I \in \mathcal{I}$:*

1. *If (I, I) is a fixpoint of $\mathcal{T}_P$, it is a fixpoint of $\mathcal{U}_P$;*
2. *if (I, I) is a stable fixpoint of $\mathcal{T}_P$, it is a stable fixpoint of $\mathcal{U}_P$.*

Thus, having the semantics reconstructed within AFT, we could not only easily define fuzzy logic programming semantics that are more precise than those in the literature [Loyer and Straccia, 2009a; Cornejo *et al.*, 2018], but also got results on how they relate to the less precise semantics.

7 Discussion and Conclusion

In this paper, we have reconstructed two specific semantics for fuzzy logic programming within approximation fixpoint theory. This reconstruction enabled several novel results on fuzzy logic programming: It allowed us to (1) establish a formal relationship between the two semantics, (2) generalize stratification to the case of fuzzy logic programs, and (3) define *ultimate* (stable/well-founded) semantics, which provide more information than their ordinary counterparts.

In future work, we want to analyze what current restrictions we could lift while keeping the positive, unifying aspects of our reconstruction. Firstly, we intend to allow for a more general syntax of fuzzy logic programs, thus moving towards (A) *extended* logic programs, where both classical negation $\neg$ and default negation $\sim$ can be used in rules [Gelfond and Lifschitz, 1991], where Cornejo *et al.* [2020] proposed a (fuzzy) stable model semantics; and/or (B) *nested* logic programs [Lifschitz *et al.*, 1999], where negation can occur freely in program bodies. Next, we also want to study lifting our requirements on negation; in particular, the condition that negations $\dot{\sim}$ must be involutions, so that we can also allow Gödel negation. Finally, we are interested in the (ordinal) number of steps that are needed to reach the least fixpoint of relevant operators such as $\mathcal{T}_P$ and $\mathcal{T}^{st}_P$. We are fairly confident that for finite programs with only rational explicit truth values and connectives that stay within the rationals, the fixpoint can be reached within $\omega = |\mathbb{N}|$ steps.

Acknowledgements

This work is partly supported by BMBF (Federal Ministry of Education and Research) in DAAD project 57616814 (SECAI, School of Embedded Composite AI). We also acknowledge funding from BMBF within projects KIMEDS (grant no. GW0552B), MEDGE (grant no. 16ME0529), and SEMECO (grant no. 03ZU1210B).

Jesse Heyninck was partially supported by the project LogicLM: Combining Logic Programs with Language Model with fle number NGF.1609.241.010 of the research programme NGF AiNed XS Europa 2024-1 which is (partly) financed by the Dutch Research Council (NWO).

We thank Bart Bogaerts for discussions on this paper, including suggesting the proof structure of Theorem 4 (inspired by his own work [Bogaerts and Cruz-Filipe, 2024]), and coming up with Example 3.

References

[Apt *et al.*, 1988] Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann, 1988.

[Belnap, 1977] Nuel D. Belnap. A useful four-valued logic. In *Modern uses of multiple-valued logic*, pages 5–37. Springer, 1977.

[Bogaerts and Cruz-Filipe, 2024] Bart Bogaerts and Luís Cruz-Filipe. Approximation fixpoint theory in Coq – with an application to logic programming. In Venanzio Capretta, Robbert Krebbers, and Freek Wiedijk, editors, *Logics and Type Systems in Theory and Practice - Essays Dedicated to Herman Geuvers on The Occasion of His 60th Birthday*, volume 14560 of *Lecture Notes in Computer Science*, pages 84–99. Springer, 2024.

[Bogaerts, 2019] Bart Bogaerts. Weighted abstract dialectical frameworks through the lens of approximation fixpoint theory. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 2686–2693. AAAI Press, 2019.

[Bourbaki, 1949] Nicolas Bourbaki. Sur le théorème de Zorn. *Archiv der Mathematik*, 2(6):434–437, 1949.

[Brewka *et al.*, 2018] Gerhard Brewka, Hannes Strass, Johannes Peter Wallner, and Stefan Woltran. Weighted abstract dialectical frameworks. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1779–1786. AAAI Press, 2018.

[Cornejo *et al.*, 2018] M Eugenia Cornejo, David Lobo, and Jesús Medina. Syntax and semantics of multi-adjoint normal logic programming. *Fuzzy Sets and Systems*, 345:41–62, 2018.

[Cornejo *et al.*, 2020] M. Eugenia Cornejo, David Lobo, and Jesús Medina. Extended multi-adjoint logic programming. *Fuzzy Sets and Systems*, 388:124–145, 2020.

[Cousot and Cousot, 1979] Patrick Cousot and Radhia Cousot. Constructive versions of Tarski's fixed point theorems. *Pacific Journal of Mathematics*, 82(1):43–57, 1979.

[Denecker and Vennekens, 2007] Marc Denecker and Joost Vennekens. Well-founded semantics and the algebraic theory of non-monotone inductive definitions. In *Logic Programming and Nonmonotonic Reasoning: 9th International Conference, LPNMR 2007, Tempe, AZ, USA, May 15-17, 2007. Proceedings 9*, pages 84–96. Springer, 2007.

[Denecker *et al.*, 2000] Marc Denecker, Victor Marek, and Mirosław Truszczyński. Approximations, Stable Operators, Well-Founded Fixpoints and Applications in Nonmonotonic Reasoning. In *Logic-Based Artificial Intelligence*, pages 127–144. Kluwer Academic Publishers, 2000.

[Denecker *et al.*, 2003] Marc Denecker, V. Wiktor Marek, and Mirosław Truszczyński. Uniform Semantic Treatment of Default and Autoepistemic Logics. *Artificial Intelligence*, 143(1):79–122, 2003.

[Denecker *et al.*, 2004] Marc Denecker, Victor W. Marek, and Mirosław Truszczyński. Ultimate approximation and its application in nonmonotonic knowledge representation systems. *Information and Computation*, 192(1):84–121, 2004.

[Fitting, 2002] Melvin Fitting. Fixpoint Semantics for Logic Programming: A Survey. *Theoretical Computer Science*, 278(1–2):25–51, 2002.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, pages 1070–1080. MIT Press, 1988.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Gener. Comput.*, 9(3/4):365–386, 1991.

[Ginsberg, 1988] Matthew L. Ginsberg. Multivalued logics: a uniform approach to reasoning in artificial intelligence. *Comput. Intell.*, 4:265–316, 1988.

[Hájek, 1998] Petr Hájek. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic*. Springer Netherlands, 1998.

[Kowalski and Kuehner, 1971] Robert A. Kowalski and Donald Kuehner. Linear resolution with selection function. *Artif. Intell.*, 2(3/4):227–260, 1971.

[Lee, 1972] Richard C. T. Lee. Fuzzy logic and the resolution principle. *J. ACM*, 19(1):109–119, 1972.

[Lifschitz *et al.*, 1999] Vladimir Lifschitz, Lappoon R. Tang, and Hudson Turner. Nested expressions in logic programs. *Ann. Math. Artif. Intell.*, 25(3-4):369–389, 1999.

[Lloyd, 1987] John W. Lloyd. *Foundations of Logic Programming, 2nd Edition*. Springer, 1987.

[Loyer and Straccia, 2002] Yann Loyer and Umberto Straccia. The well-founded semantics in normal logic programs with uncertainty. In Zhenjiang Hu and Mario Rodríguez-Artalejo, editors, *Functional and Logic Programming, 6th International Symposium, FLOPS 2002, Aizu, Japan, September 15-17, 2002, Proceedings*, volume 2441 of *Lecture Notes in Computer Science*, pages 152–166. Springer, 2002.

[Loyer and Straccia, 2003] Yann Loyer and Umberto Straccia. The approximate well-founded semantics for logic programs with uncertainty. In *International Symposium on Mathematical Foundations of Computer Science*, pages 541–550. Springer, 2003.

[Loyer and Straccia, 2009a] Yann Loyer and Umberto Straccia. Approximate well-founded semantics, query answering and generalized normal logic programs over lattices. *Ann. Math. Artif. Intell.*, 55(3-4):389–417, 2009.

[Loyer and Straccia, 2009b] Yann Loyer and Umberto Straccia. Approximate well-founded semantics, query answering and generalized normal logic programs over lattices. *Annals of Mathematics and Artificial Intelligence*, 55:389–417, 2009.

[Markowsky, 1976] George Markowsky. Chain-complete posets and directed sets with applications. *Algebra Universalis*, 6:53–68, 1976.

[Medina *et al.*, 2001] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtás. Multi-adjoint logic programming with continuous semantics. In Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński, editors, *Logic Programming and Nonmonotonic Reasoning, 6th International Conference, LPNMR 2001, Vienna, Austria, September 17-19, 2001, Proceedings*, volume 2173 of *Lecture Notes in Computer Science*, pages 351–364. Springer, 2001.

[Ovchinnikov, 1983] S.V Ovchinnikov. General negations in fuzzy set theory. *Journal of Mathematical Analysis and Applications*, 92(1):234–239, 1983.

[Shapiro, 1983] Ehud Y. Shapiro. Logic programs with uncertainties: A tool for implementing rule-based systems. In Alan Bundy, editor, *Proceedings of the 8th International Joint Conference on Artificial Intelligence. Karlsruhe, FRG, August 1983*, pages 529–532. William Kaufmann, 1983.

[Subrahmanian, 1987] V. S. Subrahmanian. On the semantics of quantitative logic programs. In *Proceedings of the 1987 Symposium on Logic Programming, San Francisco, California, USA, August 31 - September 4, 1987*, pages 173–182. IEEE-CS, 1987.

[Tarski, 1955] Alfred Tarski. A Lattice-Theoretical Fixpoint Theorem and Its Applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955.

[van Emden and Kowalski, 1976] Maarten H. van Emden and Robert A. Kowalski. The semantics of predicate logic as a programming language. *J. ACM*, 23(4):733–742, 1976.

[van Emden, 1986] Maarten H. van Emden. Quantitative deduction and its fixpoint theory. *J. Log. Program.*, 3(1):37–53, 1986.

[van Gelder *et al.*, 1991] Allen van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991.

[van Gelder, 1988] Allen van Gelder. Negation as failure using tight derivations for general logic programs. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 149–176. Morgan Kaufmann, 1988.

[Vennekens *et al.*, 2006] Joost Vennekens, David Gilis, and Marc Denecker. Splitting an operator: Algebraic modularity results for logics with fixpoint semantics. *ACM Transactions on computational logic (TOCL)*, 7(4):765–797, 2006.

[Vojtáš, 2001] Peter Vojtáš. Fuzzy logic programming. *Fuzzy Sets Syst.*, 124(3):361–370, 2001.