Grounded fixpoints and their applications in knowledge representation[☆]

Bart Bogaerts[a,*], Joost Vennekens[a,b], Marc Denecker[a]

[a] *Department of Computer Science, KU Leuven, 3001 Heverlee, Belgium*
[b] *Department of Computer Science, Campus De Nayer, KU Leuven, 2860 Sint-Katelijne-Waver, Belgium*

Abstract

In various domains of logic, researchers have made use of a similar intuition: that facts (or models) can be derived *from the ground up*. They typically phrase this intuition by saying, e.g., that the facts should be *grounded*, or that they should not be *unfounded*, or that they should be supported by *cycle-free* arguments, et cetera. In this paper, we formalise this intuition in the context of algebraical fixpoint theory. We define when a lattice element $x \in L$ is *grounded* for lattice operator $O : L \to L$. On the algebraical level, we investigate the relationship between grounded fixpoints and the various classes of fixpoints of approximation fixpoint theory, including supported, minimal, Kripke-Kleene, stable and well-founded fixpoints. On the logical level, we investigate groundedness in the context of logic programming, autoepistemic logic, default logic and argumentation frameworks. We explain what grounded points and fixpoints mean in these logics and show that this concept indeed formalises intuitions that existed in these fields. We investigate which existing semantics are grounded. We study the novel semantics for these logics that is induced by grounded fixpoints, which has some very appealing properties, not in the least its mathematical simplicity and generality. Our results unveil a remarkable uniformity in intuitions and mathematics in these fields.

Keywords: approximation fixpoint theory, lattice operator, stable semantics, well-founded semantics, groundedness, logic programming, autoepistemic logic, abstract argumentation, abstract dialectical frameworks

1. Introduction

Motivated by structural analogies in the semantics of several non-monotonic logics, Denecker, Marek and Truszczyński (from now on abbreviated as DMT) (2000) developed an algebraic fixpoint theory that defines different types of fixpoints for a so-called approximating bilattice operator, called supported, Kripke-Kleene, stable and well-founded fixpoints. In the context of logic programming, they found that Fitting's (three- or four-valued) immediate consequence operator is an approximating operator of the two-valued immediate consequence operator and that its four different types of fixpoints correspond exactly with the four major, equally named semantics of logic programs. They also identified approximating operators for default logic (DL) and autoepistemic logic (AEL) and showed that the fixpoint theory induces all main and some new semantics in these fields (Denecker et al., 2003). Moreover, by showing that Konolige's mapping from DL to AEL (Konolige, 1988) preserves the approximating operator, they resolved an old research question regarding the nature of these two logics: AEL and DL were essentially unified in the sense that for each semantics covered by AFT, a DL theory is equivalent with its Konolige mapping in AEL. However, the original DL and AEL semantics occupy different positions in this family and do not correspond. As such AEL and DL

[☆] A short version of this paper is accepted for publication in the proceedings of the AAAI'15 conference (Bogaerts et al., 2015). This paper extends the previous work with proofs of all propositions and applications of the theory to abstract dialectical frameworks, autoepistemic logic and default logic.

[*] Corresponding author

Email addresses: bart.bogaerts@cs.kuleuven.be (Bart Bogaerts), joost.vennekens@cs.kuleuven.be (Joost Vennekens), marc.denecker@cs.kuleuven.be (Marc Denecker)

under their original semantics, are "just" two different dialects of autoepistemic reasoning (Denecker et al., 2003, 2011).

The study of these approximating operators called approximation fixpoint theory (AFT) was later used to define semantics of extensions of logic programs, such as logic programs with aggregates (Pelov et al., 2007) and HEX logic programs (Antic et al., 2013). Vennekens et al. (2006) used AFT in an algebraical modularity study for logic programming, AEL and DL. Recently, Strass (2013) showed that many semantics from Dung's argumentation frameworks (AFs) and abstract dialectical frameworks (ADFs) can be obtained by direct applications of AFT. Bi et al. (2014) extended AFT with approximators allowing for inconsistencies and used it to integrate description logics with logic programs. Bogaerts et al. (2014) defined the causal logic FO(C) as an instantiation of AFT. This suggests that fixpoint theory, in ways that are difficult to explain due to its high level of abstraction, captures certain fundamental intuitions in a range of logics and sorts of human knowledge. It is this observation that provides the basic motivation for the present study.

In this paper, we formally use fixpoint theory to investigate an intuition that is found in all aforementioned logic domains. There, researchers have made use of a similar intuition: that facts (or models) can be derived *from the ground up*. They typically phrase this intuition by saying that, e.g., the facts should be *grounded*, or that they should not be *unfounded*, or that they should be supported by *cycle-free* arguments, or by arguments that contain no vicious circles, et cetera. In several cases, great efforts were done to refine semantics which did allow ungrounded models or facts. For example, it is well-known that the completion semantics of logic programs allows ungrounded models, e.g., for the transitive closure program. The efforts to avoid these led to the development of perfect, stable and well-founded semantics. Also for AEL, it was known that Moore's expansion semantics accepted ungrounded models, e.g., for the theory $\{KP \Rightarrow P\}$ which has the ungrounded model in which P is known but this knowledge is self-supported. Examples like this motivated several attempts to refine Moore's semantics, among others by Halpern and Moses (1985) and Konolige (1988).

We formalise the intuition of groundedness in the context of algebraical fixpoint theory. We call a lattice element $x \in L$ *grounded* for lattice operator $O : L \to L$ if for all $v \in L$ such that $O(x \wedge v) \leq v$, it holds that $x \leq v$. We investigate this notion on the algebraical level in AFT and on the logical level in the context of logic programming, autoepistemic logic, default logic and abstract argumentation frameworks. We explain what grounded points and fixpoints mean in these logics and show that this concept indeed formalises intuitions that existed in these fields. We investigate which existing semantics are grounded, where we call a semantics grounded if all its models are grounded, and investigate a novel semantics for these logics that is based on grounded fixpoints. Our results unveil a remarkable uniformity in intuition and mathematics in these fields and lead to a new candidate semantics with some very appealing properties, not in the least the mathematical simplicity and generality to define it in the context of operator-based logics and logic constructs.

We can summarise the main contributions of this paper as follows. We extend AFT with the notion of a grounded fixpoint, a fixpoint closely related to stable and well-founded fixpoints. We show that if the Kripke-Kleene fixpoint is exact, then it is grounded. If the well-founded fixpoint is exact, then it is the unique grounded and the unique stable fixpoint. Otherwise, stable fixpoints are grounded but not necessarily the other way around. A useful feature of grounded fixpoints that distinguishes them from stable and well-founded fixpoints is that they are determined by O and do not require the choice of an approximator. We then apply this theory to different logical research domains. In all domains, we explain the meaning of grounded fixpoints, relate them to attempts to formalise groundedness, study which semantics are grounded and finally, we explore the semantics induced by grounded fixpoints. More specifically, *(i)* in the context of logic programming, our theory yields an intuitive, purely two-valued, semantics that is easily extensible and that formalises well-known intuitions related to unfounded sets. *(ii)* We show that two of the main semantics of AFs can be characterised as grounded fixpoints of previously defined operators and discuss grounded fixpoints in the context of ADFs. *(iii)* Applied to autoepistemic logic and default logic, groundedness turns out to provide an alternative and improved formalisation of intuitions described by Konolige (1988).

2. Preliminaries

2.1. Lattices and Operators

A *partially ordered set (poset)* $\langle L, \leq \rangle$ is a set L equipped with a partial order $\leq$, i.e., a reflexive, anti-symmetric, transitive relation. As usual, we write $x < y$ as abbreviation for $x \leq y \wedge x \neq y$. If S is a subset of L, then x is an *upper bound*, respectively a *lower bound* of S if for every $s \in S$, it holds that $s \leq x$ respectively $x \leq s$. An element x is a *least upper bound*, respectively *greatest lower bound* of S if it is an upper bound that is smaller than every other upper bound, respectively a lower bound that is greater than every other lower bound. If S has a least upper bound, respectively a greatest lower bound, we denote it $\mathrm{lub}(S)$, respectively $\mathrm{glb}(S)$. As is custom, we sometimes call a greatest lower bound a *meet*, and a least upper bound a *join* and use the related notations $\bigwedge S = \mathrm{glb}(S)$, $x \wedge y = \mathrm{glb}(\{x, y\})$, $\bigvee S = \mathrm{lub}(S)$ and $x \vee y = \mathrm{lub}(\{x, y\})$. We call $\langle L, \leq \rangle$ a *complete lattice* if every subset of L has a least upper bound and a greatest lower bound. A complete lattice has both a least element $\bot$ and a greatest element $\top$.

An operator $O : L \to L$ is *monotone* if $x \leq y$ implies that $O(x) \leq O(y)$ and *anti-monotone* if $x \leq y$ implies that $O(y) \leq O(x)$. An element $x \in L$ is a *prefixpoint*, a *fixpoint*, a *postfixpoint* of O if $O(x) \leq x$, respectively $O(x) = x$, $x \leq O(x)$. Every monotone operator O in a complete lattice has a least fixpoint, denoted $\mathrm{lfp}(O)$, which is also O's least prefixpoint and the limit (the least upper bound) of the increasing sequence $(x_i)_{i \geq 0}$ defined by

- $x_0 = \bot$,
- $x_{i+1} = O(x_i)$, for successor ordinals $i + 1$,
- $x_\lambda = \mathrm{lub}(\{x_i \mid i < \lambda\})$, for limit ordinals λ.

2.2. Logic Programming

In the following sections, we illustrate our abstract results in the context of logic programming. We recall some preliminaries. We restrict ourselves to propositional logic programs, but allow arbitrary propositional formulas in rule bodies. However, our results basically apply to all extensions of logic programming that admit an immediate consequence operator (non-propositional ones, aggregates in the body, etc.).

Let Σ be an alphabet, i.e., a collection of symbols which are called *atoms*. A *literal* is an atom p or the negation $\neg q$ of an atom q. A logic program $\mathcal{P}$ is a set of *rules* r of the form $h \leftarrow \varphi$, where h is an atom called the *head* of r, denoted $head(r)$, and φ is a propositional formula called the *body* of r, denoted $body(r)$. An interpretation I of the alphabet Σ is an element of 2^Σ, i.e., a subset of Σ. The set of interpretations 2^Σ forms a lattice equipped with the order $\subseteq$. The truth value ($\mathbf{t}$ or $\mathbf{f}$) of a propositional formula φ in a structure I, denoted φ^I is defined as usual. With a logic program $\mathcal{P}$, we associate an immediate consequence operator (van Emden and Kowalski, 1976) $T_\mathcal{P}$ that maps a structure I to

$$T_\mathcal{P}(I) = \{p \mid \exists r \in \mathcal{P} : head(r) = p \wedge body(r)^I = \mathbf{t}\}.$$

3. Grounded Fixpoints

Let $\langle L, \leq \rangle$ be a complete lattice and $O : L \to L$ a lattice operator, fixed throughout this entire section. We start by giving the most central definition of this text, namely the notion of groundedness.

Definition 3.1 (Grounded)**.** We call $x \in L$ *grounded* for O if for each $v \in L$ such that $O(x \wedge v) \leq v$, it holds that $x \leq v$. We call x a *grounded fixpoint* of O if it is a fixpoint of O and it is grounded for O.

This concept is strongly related to the following.

Definition 3.2 (Strictly grounded)**.** We call $x \in L$ *strictly grounded* for O if there is no $y \in L$ such that $y < x$ and $(O(y) \wedge x) \leq y$.

The intuition behind these concepts is very similar and is easy to explain if we assume that the elements of L are sets of "facts" of some kind and the $\leq$ relation is the subset relation between such sets. In this case, $\wedge$ is the intersection and $\vee$ the union of sets. Intuitively, a set of facts x is (strictly) grounded if it can be "built from the ground up" by O. Such sets are built in several stages. Facts that are derived in later stages depend on those of earlier stages. This means that x has a stratified internal structure. If we remove multiple strata from x, we cannot expect that applying O will reconstruct all of the removed facts, but we should expect that at least some of the removed facts of x reappear, in particular those in the lowest stratum from where facts were deleted. This idea is formalised in slightly different ways in the two definitions.

If L is a powerset lattice, this intuition directly translates to $\forall u \neq \emptyset : u \subseteq x \Rightarrow O(x \setminus u) \cap u \neq \emptyset$, i.e., whenever a set of elements u is removed from x, at least one of these elements returns. To express it in the context of lattices in general, the statement needs to be reformulated without using set subtraction. There are two ways to do this.

In the definition of *strictly grounded* point, removing strata from x corresponds to taking $y < x$ (y represents $x \setminus u$ in this case). The condition that some elements of x should come back corresponds to $(O(y) \wedge x) \not\leq y$ (at least one element of $O(y)$ is in x but not in y; thus, at least one is in u).

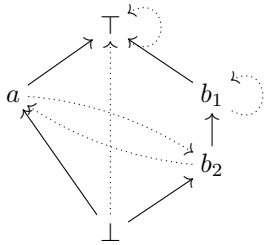
In the definition of *grounded* point, a slightly different but equivalent set-theoretic expression is formalised: $\forall u : u \cap x \neq \emptyset \Rightarrow O(x \setminus u) \cap u \neq \emptyset$. Removing strata from x corresponds to selecting a $v \in L$ and taking $x \wedge v$ (v corresponds to the complement of u, hence $x \setminus u$ to $x \wedge v$ and the fact that $u \cap x \neq \emptyset$ corresponds to $x \not\leq v$). The condition that $O(x \wedge v)$ reintroduces an element of u corresponds to $O(x \wedge v) \not\leq v$.

In what follows, we study some properties of (strictly) grounded (fix)points. We start by showing the tight relationship between the two concepts. In general, every strictly grounded point is grounded but not necessarily the other way around. Unsurprisingly, in the context of powerset lattices — the context in which we explained the intuitions — the two notions coincide.

Proposition 3.3. *If x is strictly grounded for O, then x is grounded for O.*

Proof. Assume x is strictly grounded and $v \in L$ is such that $O(x \wedge v) \leq v$. Let y denote $x \wedge v$. Then $O(y) \leq v$, and hence $(O(y) \wedge x) \leq (x \wedge v) = y$. Since $(O(y) \wedge v) \leq y$ and x is strictly grounded, it cannot be the case that $y < x$. Since $(x \wedge v) = y \leq x$, equality holds, i.e. $(x \wedge v) = x$ and $x \leq v$. This shows that x is indeed grounded. □

Example 3.4. The converse of proposition 3.3 does not hold. Consider lattice and operator represented by the following graph, where full edges express the order relation (to be precise, the $\leq$ relation is the reflexive transitive closure of these edges) and the dotted edges represent the operator:



In this case, b_1 is grounded but not strictly grounded, as can be seen by taking $y = b_2$.

Proposition 3.5. *Let L be a powerset lattice $\langle 2^{\top}, \subseteq \rangle$. Then a point $x \in L$ is grounded if and only if it is strictly grounded.*

Proof. We recall that in the context of powerset lattices the greatest lower bound is the intersection and least upper bound is the union.

Proposition 3.3 guarantees that we only need to show that all grounded points are strictly grounded. Hence, suppose x is grounded. Assume towards contradiction that x is not strictly grounded, i.e., that for some $y \subsetneq x$, $O(y) \cap x \subseteq y$. Take $v = y \cup (\top \setminus x)$. It holds that $v \cap x = y$. We have that $x \not\subseteq v$. Also, it holds that $O(y) \subseteq v$ if and only if $O(y) \cap x \subseteq y$. Since we assumed the latter, it holds that $O(v \cap x) = O(y) \subseteq v$. Hence, we obtain a contradiction with the groundedness of x, which proves our claim. □

Proposition 3.6. *Let O be a monotone operator. If x is grounded for O then x is a postfixpoint of O that is less than or equal to* $\mathrm{lfp}(O)$*, i.e., $x \leq O(x)$ and $x \leq \mathrm{lfp}(O)$.*

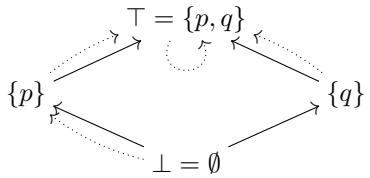
Proof. First, we show that $x \leq \mathrm{lfp}(O)$. Since O is monotone, we have $O(\mathrm{lfp}(O) \wedge x) \leq O(\mathrm{lfp}(O)) = \mathrm{lfp}(O)$. Hence, groundedness of x with $v = \mathrm{lfp}(O)$ indeed yields that $x \leq \mathrm{lfp}(O)$.

In order to show that x is a postfixpoint of O, take $v = O(x)$. Again using monotonicity of O, we find $O(v \wedge x) \leq O(x) = v$. Hence, groundedness yields that $x \leq v = O(x)$, and indeed x is a postfixpoint of O. □

Example 3.7. The converse of Proposition 3.6 does not hold. Consider the following logic program $\mathcal{P}$:

$$\left\{ \begin{array}{l} p. \\ q \leftarrow p \vee q. \end{array} \right\}$$

Its immediate consequence operator $T_\mathcal{P}$ is represented by the following graph:



$T_\mathcal{P}$ is a monotone operator with least fixpoint $\top$. Also, $\{q\}$ is a postfixpoint of $T_\mathcal{P}$ since $T_\mathcal{P}(\{q\}) = \top \geq \{q\}$. However, $\{q\}$ is not grounded since $T_\mathcal{P}(\{q\} \wedge \{p\}) = T_\mathcal{P}(\bot) = \{p\}$, while $\{q\} \not\leq \{p\}$.

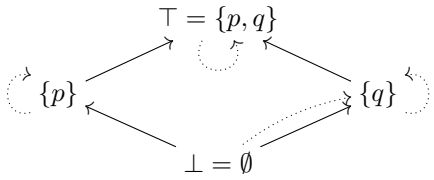
Proposition 3.8. *All grounded fixpoints of O are minimal fixpoints of O.*

Proof. Suppose x is grounded and y and x are fixpoints of O with $y \leq x$. In this case, $O(x \wedge y) = O(y) = y \leq y$. Thus, since x is grounded, we conclude that $x \leq y$, which yields $x = y$. We find that indeed, all grounded fixpoints are minimal fixpoints. □

Example 3.9. The converse of Proposition 3.8 does not hold. Consider the logic program $\mathcal{P}$:

$$\left\{ \begin{array}{l} p \leftarrow p. \\ q \leftarrow \neg p \vee q. \end{array} \right\}$$

This logic program has as immediate consequence operator $T_\mathcal{P}$:



In this case, $\{p\}$ is a minimal fixpoint of $T_\mathcal{P}$, but $\{p\}$ is not grounded since $T_\mathcal{P}(\{p\} \wedge \{q\}) = T_\mathcal{P}(\bot) = \{q\}$, while $\{p\} \not\leq \{q\}$.

Proposition 3.10. *A monotone operator has exactly one grounded (and strictly grounded) fixpoint, namely its least fixpoint.*

Proof. Proposition 3.8 guarantees that grounded fixpoints are minimal, hence a monotone operator O can have at most one grounded fixpoint $\mathrm{lfp}(O)$. Now we show that $\mathrm{lfp}(O)$ is indeed strictly grounded. Since O is monotone, for every $y \leq \mathrm{lfp}(O)$, it holds that $O(y) \leq O(\mathrm{lfp}(O)) = \mathrm{lfp}(O)$ and hence that $O(y) \wedge \mathrm{lfp}(O) = O(y)$. Now suppose that for some $y \leq \mathrm{lfp}(O)$, $O(y) \wedge \mathrm{lfp}(O) \leq y$. Then by the previous also $O(y) \leq y$. Thus y is a prefixpoint of O. However, $\mathrm{lfp}(O)$ is the least prefixpoint of O, hence $\mathrm{lfp}(O) \leq y$, and thus $y = \mathrm{lfp}(O)$. We conclude that $\mathrm{lfp}(O)$ is indeed strictly grounded. From Proposition 3.3, it then follows that x is grounded as well. □

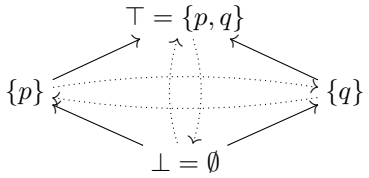
Proposition 3.11. *Every postfixpoint of an anti-monotone operator is strictly grounded.*

Proof. Suppose x is a postfixpoint of an anti-monotone operator O, i.e., $x \leq O(x)$ and that $y \leq x$. In that case $O(y) \geq O(x) \geq x$. If $O(y) \wedge x \leq y$, then it follows that $x \leq y$. Thus indeed $x = y$ and x is strictly grounded. □

Example 3.12. The converse of Proposition 3.11 does not hold. Consider the logic program $\mathcal{P}$:

$$\left\{ \begin{array}{l} p \leftarrow \neg p. \\ q \leftarrow \neg q. \end{array} \right\}$$

This logic program corresponds has as immediate consequence operator $T_{\mathcal{P}}$:



The operator $T_{\mathcal{P}}$ is anti-monotone, and $\{p\}$ is strictly grounded for $T_{\mathcal{P}}$. However, $\{p\}$ is not a postfixpoint of $T_{\mathcal{P}}$.

4. Grounded Fixpoints and Approximation Fixpoint Theory

4.1. Preliminaries: AFT

Given a lattice L, approximation fixpoint theory makes use of the bilattice L^2. We define two *projection* functions for pairs as usual: $(x, y)_1 = x$ and $(x, y)_2 = y$. Pairs $(x, y) \in L^2$ are used to approximate all elements in the interval $[x, y] = \{z \mid x \leq z \wedge z \leq y\}$. We call $(x, y) \in L^2$ *consistent* if $x \leq y$, that is, if $[x, y]$ is non-empty. We use L^c to denote the set of consistent elements. Elements $(x, x) \in L^c$ are called *exact*; they constitute the embedding of L in L^2. We sometimes abuse notation and use the tuple (x, y) and the interval $[x, y]$ interchangeably. The *precision ordering* on L^2 is defined as $(x, y) \leq_p (u, v)$ if $x \leq u$ and $v \leq y$. In case (u, v) is consistent, this means that (x, y) approximates all elements approximated by (u, v), or in other words that $[u, v] \subseteq [x, y]$. If L is a complete lattice, then $\langle L^2, \leq_p \rangle$ is also a complete lattice.

AFT studies fixpoints of lattice operators $O : L \to L$ through operators approximating O. An operator $A : L^2 \to L^2$ is an *approximator* of O if it is $\leq_p$-monotone, and has the property that for all x, $O(x) \in A(x, x)$. Approximators are internal in L^c (i.e., map L^c into L^c). As usual, we restrict our attention to *symmetric* approximators: approximators A such that for all x and y, $A(x, y)_1 = A(y, x)_2$. DMT (2004) showed that the consistent fixpoints of interest (supported, stable, well-founded) are uniquely determined by an approximator's restriction to L^c, hence, sometimes we only define approximators on L^c.

AFT studies fixpoints of O using fixpoints of A.

- The *A-Kripke-Kleene fixpoint* is the $\leq_p$-least fixpoint of A and has the property that it approximates all fixpoints of O.
- A *partial A-stable fixpoint* is a pair (x, y) such that $x = \text{lfp}(A(\cdot, y)_1)$ and $y = \text{lfp}(A(x, \cdot)_2)$, where $A(\cdot, y)_1$ denotes the operator $L \to L : x \mapsto A(x, y)_1$ and analogously for $A(x, \cdot)_2$.
- The *A-well-founded fixpoint* is the least precise partial A-stable fixpoint.
- An *A-stable fixpoint* of O is a fixpoint x of O such that (x, x) is a partial A-stable fixpoint. This is equivalent with the condition that $x = \text{lfp}(A(\cdot, x)_1)$.

The A-Kripke-Kleene fixpoint of O can be constructed by iteratively applying A, starting from $(\bot, \top)$. For the A-well-founded fixpoint, a similar constructive characterisation has been worked out by Denecker and Vennekens (2007):

Definition 4.1. An *A-refinement* of (x, y) is a pair $(x', y') \in L^2$ satisfying one of the following two conditions:

- $(x, y) \leq_p (x', y') \leq_p A(x, y)$, or
- $x' = x$ and $A(x, y')_2 \leq y' \leq y$.

An A-refinement is *strict* if $(x, y) \neq (x', y')$.

Definition 4.2. A *well-founded induction* of A is a sequence $(x_i, y_i)_{i \leq \beta}$ with β an ordinal such that

- $(x_0, y_0) = (\bot, \top)$;
- (x_{i+1}, y_{i+1}) is an A-refinement of (x_i, y_i), for all $i < \beta$;
- $(x_\lambda, y_\lambda) = \text{lub}_{\leq_p} \{(x_i, y_i) \mid i < \lambda\}$ for each limit ordinal $\lambda \leq \beta$.

A well-founded induction is *terminal* if its limit (x_β, y_β) has no strict A-refinements.

A well-founded induction is an algebraical generalisation of the well-founded model construction defined by Van Gelder et al. (1991). The first type of refinements correspond to making a partial structure more precise by applying Fitting's immediate consequence operator; the second type of refinement corresponds to making a structure more precise by eliminating an unfounded set.

For a given approximator A, there are many different terminal well-founded inductions of A. Denecker and Vennekens (2007) showed that they all have the same limit, which equals the A-well-founded fixpoint of O. Furthermore, if A is symmetric, the A-well-founded fixpoint of O (and in fact, every tuple in a well-founded induction of A) is consistent. Well-founded inductions that only use the first sort of refinement converge to the A-Kripke-Kleene fixpoint.

The precision order can be pointwise extended to the family of approximators of O. It then follows that more precise approximators have a more precise well-founded fixpoint and more precise approximators have more stable fixpoints. DMT (2004) showed that there exists a most precise approximator, U_O, called the ultimate approximator of O. This operator is defined by

$$U_O : L^c \to L^c : (x, y) \mapsto (\bigwedge O([x, y]), \bigvee O([x, y])).$$

Here, we used the notation $O(X) = \{O(x) \mid x \in X\}$ for a set $X \subseteq L$. It then follows that for every approximator A, all A-stable fixpoints are U_O-stable fixpoints, and the U_O-well-founded fixpoint is always more precise than the A-well-founded fixpoint. We refer to U_O-stable fixpoints as ultimate stable fixpoints of O and to the U_O-well-founded fixpoint as the ultimate well-founded fixpoint of O. Semantics defined using the ultimate approximator have as advantage that they only depend on O since the approximator can be derived from O.

Approximation Fixpoint Theory and Logic Programming. In the context of logic programming, elements of the bilattice $\left(2^\Sigma\right)^2$ are four-valued interpretations, pairs $\mathcal{I} = (I_1, I_2)$ of interpretations. The pair (I_1, I_2) approximates all interpretations I' with $I_1 \subseteq I' \subseteq I_2$. We often identify an interpretation I with the four-valued interpretation (I, I). If $\mathcal{I} = (I_1, I_2)$ is a (four-valued) interpretation, and $U \subseteq \Sigma$, we write $\mathcal{I}[U : \mathbf{f}]$ for the (four-valued) interpretation that equals $\mathcal{I}$ on all elements not in U and that interprets all elements in U as $\mathbf{f}$, i.e., the interpretation $(I_1 \setminus U, I_2 \setminus U)$. We are mostly concerned with consistent (also called partial or three-valued) interpretations: tuples $\mathcal{I} = (I_1, I_2)$ with $I_1 \subseteq I_2$. For such an interpretation, the atoms in I_1 are *true* ($\mathbf{t}$) in $\mathcal{I}$, the atoms in $I_2 \setminus I_1$ are *unknown* ($\mathbf{u}$) in $\mathcal{I}$ and the other atoms are *false* ($\mathbf{f}$) in $\mathcal{I}$. If $\mathcal{I}$ is a three-valued interpretation, and φ a formula, we write $\varphi^{\mathcal{I}}$ for the standard three-valued valuation based on the Kleene truth tables (see Figure 1). An alternative valuation is the *supervaluation*; with this valuation, the value of a formula φ is $\mathbf{t}$ (respectively $\mathbf{f}$) in partial interpretation $\mathcal{I}$ if and only if it is $\mathbf{t}$ (respectively $\mathbf{f}$) in all interpretations approximated by $\mathcal{I}$; it is unknown otherwise.

We call two formulas 3-*equivalent* if they have the same truth value in all three-valued interpretations and 2-*equivalent* if they have the same truth value in all (two-valued) interpretations. Several approximators

$A \wedge B$		B		
		t	**f**	**u**
A	**t**	**t**	**f**	**u**
	f	**f**	**f**	**f**
	u	**u**	**f**	**u**

$A \vee B$		B		
		t	**f**	**u**
A	**t**	**t**	**t**	**t**
	f	**t**	**f**	**u**
	u	**t**	**u**	**u**

		$\neg A$
A	**t**	**f**
	f	**t**
	u	**u**

Figure 1: The Kleene truth tables (Kleene, 1938).

have been defined for logic programs. The most common is Fitting's immediate consequence operator $\Psi_{\mathcal{P}}$ (Fitting, 2002), a direct generalisation of $T_{\mathcal{P}}$ to partial interpretations. DMT (2000) showed that the well-founded fixpoint of $\Psi_{\mathcal{P}}$ is the well-founded model of $\mathcal{P}$ as defined by Van Gelder et al. (1991) and that $\Psi_{\mathcal{P}}$-stable fixpoints are exactly the stable models of $\mathcal{P}$ as defined by Gelfond and Lifschitz (1988). In this case, the operator $\Psi_{\mathcal{P}}(\cdot, y)_1$ coincides with the immediate consequence operator of the Gelfond-Lifschitz reduct (Gelfond and Lifschitz, 1988). The most precise approximator is the ultimate approximator $U_{\mathcal{P}}$.

Replacing the body of a rule by a 3-equivalent formula obviously preserves $\Psi_{\mathcal{P}}$; replacing the body of a rule by a 2-equivalent formula preserves $T_{\mathcal{P}}$ and hence also $U_{\mathcal{P}}$. Thus, transformations that preserve 3-equivalence, preserve standard Kripke-Kleene, stable and well-founded semantics, and transformations preserving 2-equivalence preserve all ultimate semantics (ultimate Kripke-Kleene, ultimate stable, ultimate well-founded). The ease with which this can be proven demonstrates the power of AFT.

Preserving 2-equivalence is not enough to preserve standard semantics. For example, consider programs $\mathcal{P} = \{p \leftarrow p \vee \neg p\}$ and $\mathcal{P}' = \{p.\}$. Even though the body of the rule defining p in $\mathcal{P}$ is a tautology, $\{p\}$ is not a stable model of $\mathcal{P}$ while it is a stable model of $\mathcal{P}'$. But ultimate semantics treat these two programs identically. For instance, $\{p\}$ is the unique ultimate stable model of both programs.

While substituting formulas for 2-equivalent formulas in rule bodies preserves the ultimate but not necessarily the standard versions of semantics, a weaker equivalence property can still be guaranteed. It holds that the standard Kripke-Kleene and well-founded models of both programs are compatible with each other: no atom is true in the model of one and false in the model of the other program. This follows from the fact that both the ultimate Kripke-Kleene and the ultimate well-founded model are preserved by these substitutions and that they are consistent and more precise than the standard Kripke-Kleene and well-founded model respectively.

The nice property that ultimate semantics only depend on the operator comes at a cost. DMT (2004) showed that deciding whether $\mathcal{P}$ has an ultimate stable model is Σ_2^P-complete, while that same task is only NP-complete for classical stable models.

4.2. Grounded Fixpoints and Approximation Fixpoint Theory

In this section, we discuss how groundedness relates to AFT. More concretely, we show that all (ultimate) stable fixpoints are grounded and that all grounded fixpoints are approximated by the (ultimate) well-founded fixpoint.

Proposition 4.3. *All ultimate stable fixpoints of O are (strictly) grounded fixpoints.*

Proof. Let x be an ultimate stable fixpoint of O. Thus, (x, x) is a fixpoint of the ultimate stable operator, i.e., $x = U_O(x)_1 = \text{lfp}(\bigwedge O([\cdot, x]))$. Since $O(x) = \bigwedge O([x, x])$, it follows that x is also a fixpoint of O. Now suppose for some $y \leq x$, $O(y) \wedge x \leq y$; we show that $x = y$. We know that

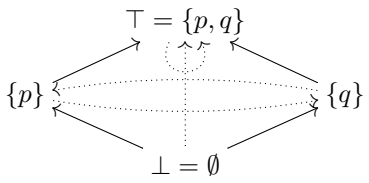
$$\bigwedge O([y, x]) \leq O(y) \wedge O(x) = O(y) \wedge x \leq y.$$

Thus, y is a prefixpoint of the monotone operator $\bigwedge O([\cdot, x])$. Since x is the least fixpoint (and also the least prefixpoint) of that same operator, we find that $x \leq y$, and thus $x = y$, which shows that x is strictly grounded indeed. □

Example 4.4. The converse of Proposition 4.3 does not always hold. Consider the logic program $\mathcal{P}$:

$$\left\{ \begin{array}{l} p \leftarrow \neg p \vee q. \\ q \leftarrow \neg q \vee p. \end{array} \right\}$$

This logic program has as immediate consequence operator $T_{\mathcal{P}}$:



$\top$ is grounded for $T_{\mathcal{P}}$, since the only v with $T_{\mathcal{P}}(\top \wedge v) = T_{\mathcal{P}}(v) \leq v$ is $\top$ itself. However, since $T_{\mathcal{P}}([\bot, \top]) = L \setminus \{\bot\}$ and $\{p\} \wedge \{q\} = \bot$, it follows that $\bigwedge(T_{\mathcal{P}}[\bot, \top]) = \bot$. Thus, $\mathrm{lfp}(\bigwedge T_{\mathcal{P}}([\cdot, \top])) = \bot$. Therefore, $\top$ is not an ultimate stable fixpoint of $T_{\mathcal{P}}$.

The fact that all A-stable fixpoints are ultimate stable fixpoints yields:

Corollary 4.5. *If A is an approximator of O, then all A-stable fixpoints are (strictly) grounded fixpoints of O.*

Theorem 4.6. *The well-founded fixpoint (u, v) of a symmetric approximator A of O approximates all grounded fixpoints of O.*

Proof. Let $(a_i, b_i)_{i \leq \beta}$ be a well-founded induction of A and let x be a grounded fixpoint of O. We show by induction that for every $i \leq \beta$, $a_i \leq x \leq b_i$. The result trivially holds for $i = 0$ since $(a_0, b_0) = (\bot, \top)$. It is also clear that the property is preserved in limit ordinals. Hence, all we need to show is that the property is preserved by A-refinements. Suppose (a', b') is an A-refinement of (a, b) and (a, b) approximates x. We show that also (a', b') approximates x, i.e., that $a' \leq x \leq b'$. We distinguish two cases.

First, assume that $(a, b) \leq_p (a', b') \leq_p A(a, b)$. Since x is a fixpoint of O and A an approximator of O, we find that $x = O(x) \in A(x, x) \subseteq A(a, b) \subseteq (a', b')$.

Second, assume that $a' = a$ and $A(a, b')_2 \leq b' \leq b$. Since every tuple in a well-founded induction of a symmetric approximator is consistent, we know that $b' \geq a$. Since also $x \geq a$, we see that $a \leq x \wedge b' \leq b'$. Hence $x \wedge b' \in [a, b']$, thus $O(x \wedge b') \in A(a, b')$, and we see that $O(x \wedge b') \leq A(a, b')_2 \leq b'$. Since x is grounded, this implies that $x \leq b'$; we conclude that also in this case $x \in [a', b']$.

We have thus shown that every step in a well-founded induction of A preserves all grounded fixpoints. □

Corollary 4.7. *If the well-founded fixpoint of a symmetric approximator A of O is exact, then this point is the unique (strictly) grounded fixpoint of O.*

Since the Kripke-Kleene fixpoint approximates the well-founded fixpoint, we also get the following property.

Corollary 4.8. *If the Kripke-Kleene fixpoint of a symmetric approximator A of O is exact, then it is the unique fixpoint of O and it is (strictly) grounded.*

5. Grounded Fixpoints of Logic Programs

In this section, we discuss grounded fixpoints in the context of logic programming. It follows immediately from the algebraical results (Corollary 4.5 and Theorem 4.6) that stable models are grounded fixpoints of the immediate consequence operator and that all grounded fixpoints are minimal fixpoints approximated by the well-founded model. Furthermore, if the Kripke-Kleene or well-founded model is exact, then it is the unique grounded fixpoint of $T_{\mathcal{P}}$.

Grounded fixpoints can be explained in terms of unfounded sets, a notion that was first defined by Van Gelder, Ross and Schlipf (1991) in their seminal paper introducing the well-founded semantics. Unfounded sets of three-valued interpretations are a key concept in the construction of the well-founded model. Intuitively, an unfounded set is a set of atoms that might circularly support themselves, but have no support from outside. Stated differently, an unfounded set of a logic program $\mathcal{P}$ with respect to a (partial) interpretation $\mathcal{I}$ is a set U of atoms such that $\mathcal{P}$ does not provide support for any atom in U if the atoms in U are assumed false.

Below, we define the concept of unfounded set in the context of two-valued interpretations. For clarity, we refer to our unfounded sets as "2-unfounded sets" and to the original definition by Van Gelder, Ross and Schlipf as "GRS-unfounded sets".

Definition 5.1 (2-Unfounded set)**.** Let $\mathcal{P}$ be a logic program and $I \subseteq \Sigma$ an interpretation.

A set $U \subseteq \Sigma$ is a *2-unfounded set* of I (with respect to $\mathcal{P}$) if for each rule $r \in \mathcal{P}$ with $head(r) \in U$, $body(r)^{I[U:\mathbf{f}]}$ is false. A 2-unfounded set U of I is called *proper* if U is a nonempty subset of I.

Thus, U is a 2-unfounded set of I if after revising I by setting atoms of U to false, no atom in U can be derived.

All interpretations I admit 2-unfounded sets, in particular the empty set $\emptyset$ and every set U consisting of atoms p that are false in I and for which every rule $r \in \mathcal{P}$ with $head(r) = p$ has a false body in I. Indeed, for such sets, it holds that $I[U : \mathbf{f}] = I$ and no rule derives an element of U. However, not every interpretation I admits a proper 2-unfounded set. If I has a proper 2-unfounded set, then this means that I cannot be built up from the ground.

In Section 5.1, we investigate the relationship between this formalisation of unfounded sets and the original one. In particular, we extend the above definition to three-valued interpretations and show that the different notions of unfounded set are equivalent in the context of the well-founded model construction. First, we show how unfounded sets are related to the algebraical notion of groundedness.

The definition of 2-unfounded set can be easily rephrased in terms of the operator $T_{\mathcal{P}}$, as shown in the following proposition.

Proposition 5.2. *U is a 2-unfounded set of I with respect to $\mathcal{P}$ if and only if $T_{\mathcal{P}}(I[U : \mathbf{f}]) \cap U = \emptyset$.*

Proof. Follows immediately from the fact that $p \in T_{\mathcal{P}}(I[U : \mathbf{f}])$ if and only if for some rule $r \in \mathcal{P}$ with $head(r) = p$, it holds that $body(r)^{I[U:\mathbf{f}]} = \mathbf{t}$. □

Example 5.3. Let $\mathcal{P}$ be the following program:

$$\left\{ \begin{array}{l} p \leftarrow q. \\ q \leftarrow p. \\ r \leftarrow \neg p. \end{array} \right\}$$

Let I be the interpretation $\{p, q\}$. Then $U_1 = \{p, q\}$ is a 2-unfounded set of I since $I[U_1 : \mathbf{f}] = \{r\}$ and in this structure, the bodies of rules defining p and q are false. Alternatively, we notice that $T_{\mathcal{P}}(I[U_1 : \mathbf{f}]) \cap U_1 = \emptyset \cap U_1 = \emptyset$.

The set $U_2 = \{p\}$ is not a 2-unfounded set of I since the rule body for p evaluates to true in $I[U_2 : \mathbf{f}]$.

In what follows, we use $\overline{U}$ for the set complement of U, i.e., $\overline{U} = \Sigma \setminus U$.

Proposition 5.4. *Let $\mathcal{P}$ be a logic program, $I \in 2^{\Sigma}$ an interpretation, $U \subseteq \Sigma$. The following statements are equivalent:*

- *U is a 2-unfounded set of I,*
- *$T_{\mathcal{P}}(I \cap \overline{U}) \subseteq \overline{U}$, and*
- *$T_{\mathcal{P}}(I \setminus U) \cap I \subseteq I \setminus U$.*

Proof. The equivalence of the first two follows immediately from proposition 5.2 since $I[U : \mathbf{f}] = I \setminus U = I \cap \overline{U}$ and for every set X, $X \subseteq \overline{U}$ if and only if $X \cap U = \emptyset$. The equivalence of the second and third follows from the fact that $I \setminus U = I \cap \overline{U}$ and for all subsets X, Y and Z of Σ it holds that $X \cap Y \subseteq Y \setminus Z$ if and only if $X \subseteq (\Sigma \setminus Z)$. □

Proposition 5.4 shows that U is a 2-unfounded set if and only if its complement satisfies the condition on v in Definition 3.1 if and only if $I \setminus U$ satisfies the condition on y in Definition 3.2. This allows us to reformulate the condition that I is grounded as follows.

Proposition 5.5. *A structure I is (strictly) grounded for $T_\mathcal{P}$ if and only if I does not contain atoms that belong to a 2-unfounded set U of I with respect to P.*

Proof. First, suppose I is grounded for $T_\mathcal{P}$ and U is a 2-unfounded set of I. Let $V = \overline{U}$ denote the set complement of U. Since U is a 2-unfounded set, $T_\mathcal{P}(I \cap V) \subseteq V$. Thus, the definition of groundedness yields $I \subseteq V$, and hence that $U \cap I = \emptyset$. We conclude that I is indeed disjoint from any 2-unfounded set.

The reverse direction is analogous. Suppose every 2-unfounded set is disjoint from I. Let V be such that $T_\mathcal{P}(I \cap V) \subseteq V$ and let $U = \overline{V}$ denote the complement of V. Then again $I[U : \mathbf{f}] = I \cap V$ and the result follows.

We already established in Proposition 3.5 that groundedness is equivalent with strict groundedness in this context. □

Corollary 5.6. *A structure I is a grounded fixpoint of $T_\mathcal{P}$ if and only if it is a fixpoint of $T_\mathcal{P}$ and it has no proper 2-unfounded sets.*

We call grounded fixpoints of $T_\mathcal{P}$ *grounded models of* $\mathcal{P}$. Similarly to ultimate semantics, grounded models are insensitive to 2-equivalence-preserving rewritings in the bodies of rules: if $\mathcal{P}$ and $\mathcal{P}'$ are such that $T_\mathcal{P} = T_{\mathcal{P}'}$, then the grounded models of $\mathcal{P}$ and $\mathcal{P}'$ coincide. Also similar to ultimate semantics, the above property comes at a cost.

Theorem 5.7. *The problem "given a finite propositional logic program $\mathcal{P}$, decide whether $\mathcal{P}$ has a grounded model" is Σ_2^P-complete.*

The proof of this theorem is heavily inspired by the proof of a similar property for ultimate stable models (Theorem 6.12) by DMT (2004); we use the same reduction of a Σ_2^P-hard problem to our problem.

Proof. Given an interpretation I, the task of verifying that I is a grounded model can be done by calculating $T_\mathcal{P}(I[U : \mathbf{f}])$ for all candidate proper 2-unfounded sets, i.e., non-empty sets U with $U \subseteq I$. Hence this task is in co-NP. Thus the task of deciding whether there exists grounded model certainly is in the class Σ_2^P.

We now show Σ_2^P-hardness of the problem of existence of a grounded model of a program $\mathcal{P}$. Let φ be a propositional formula in DNF over propositional symbols $x_1, \ldots, x_m, y_1, \ldots, y_n$. For an interpretation $I \subseteq \{x_1, \ldots, x_m\}$, we define φ_I as the formula obtained from φ by replacing all atoms $x_i \in I$ by $\mathbf{t}$ and all atoms $x_i \notin I$ by $\mathbf{f}$. Recall that the problem of deciding whether there exists an interpretation $I \subseteq \{x_1, \ldots, x_m\}$ such that φ_I is a tautology is Σ_2^P-hard. We now reduce this problem to our problem. For each x_i, we introduce a new variable x_i'; we will use x_i' to represent the negation of x_i. Let φ' be the formula obtained from φ by replacing all literals $\neg x_i$ by x_i'. We define a program $\mathcal{P}(\varphi)$ consisting of the following clauses

1. $x_i \leftarrow \neg x_i'$ and $x_i' \leftarrow \neg x_i$ for each $i \in \{1, \ldots, m\}$,
2. $y_i \leftarrow \varphi'$ for each $i \in \{1, \ldots, n\}$,
3. $p \leftarrow \varphi'$,
4. $q \leftarrow \neg p \wedge \neg q$.

We now show that there is an $I \subseteq \{x_1, \ldots, x_m\}$ such that φ_I is a tautology if and only if $\mathcal{P}(\varphi)$ has a grounded model. It is easy to see that in each fixpoint M of $T_{\mathcal{P}(\varphi)}$ the following properties hold:

1. q is false in M (if q is true $T_{\mathcal{P}(\varphi)}$ does not derive q),
2. p is true in M (otherwise $T_{\mathcal{P}(\varphi)}$ derives q),

3. $y_1, \ldots, y_n$ are true in M (since their rules have the same body as p),
4. for each $i \in \{1, \ldots, m\}$, exactly one of x_i and x'_i is true in M.

Given a set $I \subseteq \{x_1, \ldots, x_m\}$, we define $\check{I} = I \cup \{x'_i \mid x_i \notin I\}$. It follows from the above properties that for each fixpoint M of $T_{\mathcal{P}(\varphi)}$, there exists an I such that

$$M = \check{I} \cup \{p, y_1, \ldots, y_n\}.$$

Thus it suffices to show that if $I \subseteq \{x_1, \ldots, x_m\}$, then $M = \check{I} \cup \{p, y_1, \ldots, y_n\}$ is a grounded model of $\mathcal{P}(\varphi)$ if and only if φ_I is a tautology.

In order to prove this, we fix I and $M = \check{I} \cup \{p, y_1, \ldots, y_n\}$. Now, M is *not* a grounded model if and only if there exists a non-empty $U \subseteq M$ such that $T_{\mathcal{P}}(M \setminus U) \cap U = \emptyset$. Since $T_{\mathcal{P}}$ is anti-monotone when restricted to the x_i and x'_i, each such U has the property that $x_i \notin U$ and $x'_i \notin U$. Hence $U \subseteq \{p, y_1, \ldots, y_n\}$. Hence, such an U has the property that φ' is false in $M \setminus U$. But φ' is false in $M \setminus U$ if and only if φ_I is false in $\{y_1, \ldots, y_n\} \setminus U$. Thus, we conclude that M is *not* a grounded model if and only if there exists a truth assignment to $J \subseteq \{y_1, \ldots, y_n\}$ such that φ_I is false in J. Thus, M is *not* a grounded model if and only if φ_I is *not* a tautology, which is exactly what we needed to show. □

Let us briefly compare grounded model semantics with the two most frequently used semantics of logic programming: well-founded and stable semantics. Firstly, we observe that these three semantics tend to prefer a subclass of the minimal models. We called this criterion groundedness and indeed found that stable semantics and two-valued well-founded semantics have the property that they only accept grounded interpretations.

Secondly, since these three semantics are closely related, it is to be expected that they often coincide. We established that for programs with a two-valued Kripke-Kleene model, the Kripke-Kleene semantics coincides with the supported model semantics and with the three semantics mentioned above. Also for programs with a two-valued well-founded model, the three semantics coincide. This sort of programs is common in applications for deductive databases (Datalog and extensions (Abiteboul and Vianu, 1991)) and for representing inductive definitions (Denecker and Ternovska, 2008; Denecker and Vennekens, 2014). In contrast, well-founded semantics only rarely coincides with stable semantics in the context of answer set programming (ASP). In the context where the well-founded model is three-valued, the question arises when stable and grounded models coincide. We illustrated in Example 4.4 that in this case, stable and grounded model semantics may disagree (since $\{p, q\}$ is not an ultimate stable model, it certainly is no stable model either). However, we observe that this example is quite extraordinary, and this is the case for all such programs that we found. It leads us to expect that both semantics coincide for large classes of ASP programs. It is therefore an interesting topic for future research to search for characteristics of programs that guarantee that both semantics agree. If such properties can be identified, then within those classes the grounded model semantics gives an equivalent reformulation of the stable semantics. If these classes cover the pragmatically important classes of ASP programs (that is, if the ASP programs written for practical problems fall inside them), then the grounded model semantics is an *elegant*, *intuitive* and *concise* variant of the standard stable semantics, which in practice coincides with it. And if pragmatically important classes of programs are discovered for which both semantics disagree, the question then is if other properties than groundedness can be identified that are possessed by stable but not by grounded models.

Grounded model semantics is, to the best of our knowledge, the first *purely two-valued and algebraical* semantics for logic programs that satisfies the desirable property of groundedness. The well-founded semantics explicitly uses three-valued interpretations in the well-founded model construction. Stable semantics uses three-valued logic implicitly in the sense that, as we showed, the Gelfond-Lifschitz reduct corresponds to an evaluation in a partial interpretation. One of the main advantages of grounded model semantics is that it is so easily definable for language extensions. All it takes is to define the (two-valued) immediate consequence operator. Typically this is quite easy (see the next paragraph). Note that the ultimate versions of the well-founded and stable semantics are purely algebraical as well but they are mathematically more involved since they still refer to three-valued interpretations (replacing Kleene valuation by supervaluation).

Grounded Fixpoints for Logic Programs with Abstract Constraint Atoms. The fact that grounded model semantics is two-valued and algebraical makes it not only easier to understand, but also to *extend* the semantics. To illustrate this, we consider logic programs with abstract constraint atoms as defined by Marek et al. (2008). An *abstract constraint* is a collection $C \subseteq 2^{\Sigma}$. A *constraint atom* is an expression of the form $C(X)$, where $X \subseteq \Sigma$ and C is an abstract constraint. The goal of such an atom is to model constraints on subsets of X. The truth value of $C(X)$ in interpretation I is $\mathbf{t}$ if $I \cap X \in C$ and $\mathbf{f}$ otherwise. Abstract constraints are a generalisation of pseudo-Boolean constraints, cardinality constraints, containment constraints, and much more. A *deterministic* logic program with abstract constraint atoms (Marek et al., 2008) is a set of rules of the form[1]

$$p \leftarrow a_1 \wedge \cdots \wedge a_n \wedge \neg b_1 \wedge \cdots \wedge \neg b_m,$$

where p is an atom and the a_i and b_i are constraint atoms. Having defined the truth value of a constraint atom $C(X)$ in an interpretation I, an immediate consequence operator can be defined in the standard way:

$$T_{\mathcal{P}}(I) = \{p \mid \exists r \in \mathcal{P} : head(r) = p \wedge body(r)^I = \mathbf{t}\}.$$

Grounded models of this operator still represent the same intuitions: an interpretation I is grounded if it admits no unfounded sets, or said differently, if it contains no atoms without external support. Thus, I is grounded if it contains no non-empty set U of atoms such that $body(r)^{I[U:\mathbf{f}]}$ for each rule r with $head(r) \in U$.

Example 5.8. Let Σ be the alphabet $\{a, b, c, d\}$. For every i, let $C_{\geq i}$ be the cardinality constraint $\{X \subseteq \Sigma \mid |X| \geq i\}$. Consider the following logic program $\mathcal{P}$ over Σ:

$$\left\{ \begin{array}{ll} a. & b \leftarrow C_{\geq 1}(\Sigma). \\ c \leftarrow \neg C_{\geq 4}(\Sigma). & d \leftarrow C_{\geq 4}(\Sigma). \end{array} \right\}$$

Any interpretation in which d holds is *not* grounded since, taking $U = \{d\}$ yields for every I that $C_{\geq 4}(\Sigma)^{I[U:\mathbf{f}]} = \mathbf{f}$ and thus $d \notin T_{\mathcal{P}}(I[U : \mathbf{f}])$. It can easily be verified that $\{a, b, c\}$ is the only grounded model of $\mathcal{P}$.

This example illustrates that even for complex, abstract extensions of logic programs, groundedness is an intuitive property. Groundedness easily extends to these rich formalisms: the lattice always is the space of interpretations, the immediate consequence is defined in the standard way and defining grounded models takes only one line given this immediate consequence operator. This is in sharp contrast with more common semantics of logic programming (such as stable and well-founded semantics) which are often hard(er) to extend to richer formalisms, as can be observed by the many different versions of those semantics that exist for logic programs with aggregates (Ferraris, 2005; Son et al., 2006; Pelov et al., 2007; Faber et al., 2011; Gelfond and Zhang, 2014).

Furthermore, groundedness is closely related to one of the most popular semantics for logic programs with aggregates, namely the FLP-stable semantics defined by Faber et al. (2011). Given an interpretation I, Faber, Pfeifer and Leone (2011) defined the reduct of $\mathcal{P}$ with respect to I as the program $\mathcal{P}^I = \{r \mid r \in \mathcal{P} \wedge I \models body(r)\}$. I is an *FLP-stable model* of $\mathcal{P}$ if it is a subset-minimal model of $T_{\mathcal{P}^I}$. For a large class of programs, this semantics is equivalent with qrounded model semantics, as the following theorem shows.

Theorem 5.9. *Let $\mathcal{P}$ be a logic program with abstract constraint atoms. If for each $p \in \Sigma$, there is at most one rule $r \in \mathcal{P}$ with $head(r) = p$, then I is an FLP-stable model of $\mathcal{P}$ if and only if I is a grounded model of $\mathcal{P}$.*

Proof. In this case, for all I and J, $T_{\mathcal{P}^I}(J) = T_{\mathcal{P}}(J) \cap T_{\mathcal{P}}(I)$ since $\mathcal{P}^I$ is obtained from $\mathcal{P}$ by removing all rules with body false in I and there is at most one rule defining each atom. If I is a supported model, we find that $T_{\mathcal{P}^I}(J) = T_{\mathcal{P}}(J) \cap I$. It is easy to see that FLP-stable models are supported models, i.e., fixpoints of $T_{\mathcal{P}}$. Assume I is a supported model. In this case I is FLP-stable if and only if there is no $J \subsetneq I$ with $T_{\mathcal{P}}(J) \cap I = T_{\mathcal{P}^I}(J) \subseteq J$, i.e., if and only if I is strictly grounded. Now, we know from Proposition 3.5 that in the context of logic programming, groundedness and strict groundedness are equivalent, which proves our claim. □

[1]The approach by Marek et al. (2008) also includes nondeterministic programs. We come back to this issue in Section 5.1.

5.1. Discussion

Unfounded Sets. Unfounded sets were first defined by Van Gelder et al. (1991) in their seminal paper introducing the well-founded semantics. Their definition slightly differs from Definition 5.1.

Definition 5.10 (GRS-Unfounded set)**.** Let $\mathcal{P}$ be a logic program and $\mathcal{I}$ a three-valued interpretation. A set $U \subseteq \Sigma$ is a *GRS-unfounded set* of $\mathcal{I}$ (with respect to $\mathcal{P}$) if for each rule r with $head(r) \in U$, $body(r)^{\mathcal{I}} = \mathbf{f}$ or $body(r)^{\mathcal{I}[U:\mathbf{f}]} = \mathbf{f}$.

The first difference between 2-unfounded sets and GRS-unfounded sets is that GRS-unfounded sets are defined for three-valued interpretations, while we restricted our attention to (two-valued) interpretations. Our definition easily generalises to three-valued interpretations as well.

Definition 5.11 (3-Unfounded set)**.** Let $\mathcal{P}$ be a logic program and $\mathcal{I}$ a three-valued interpretation. A set $U \subseteq \Sigma$ is a 3-*unfounded set* of $\mathcal{I}$ (with respect to $\mathcal{P}$) if for each rule r with $head(r) \in U$, $body(r)^{\mathcal{I}[U:\mathbf{f}]} = \mathbf{f}$.

Lemma 5.12. *Let $\mathcal{P}$ be a logic program and I an interpretation. A set $U \subseteq \Sigma$ is a 2-unfounded set of I with respect to $\mathcal{P}$ if and only if it is a 3-unfounded set of I with respect to $\mathcal{P}$.*

Proof. Follows immediately from the definitions. □

This definition formalises the same intuitions as Definition 5.1: U is a 3-unfounded set if making all atoms in U false in $\mathcal{I}$ results in a state where none of them can be derived. This definition easily translates to algebra as well.

Proposition 5.13. *Let $\mathcal{P}$ be a logic program, $\Psi_{\mathcal{P}}$ Fitting's immediate consequence operator and $\mathcal{I}$ a three-valued interpretation. A set $U \subseteq \Sigma$ is a 3-unfounded set if and only if $\Psi_{\mathcal{P}}(\mathcal{I}[U : \mathbf{f}])_2 \cap U = \emptyset$.*

Proof. Recall that Fitting's operator is defined by

$$\Psi_{\mathcal{P}}(\mathcal{I})_1 = \{a \in \Sigma \mid body(r)^{\mathcal{I}} = \mathbf{t} \text{ for some rule } r \in \mathcal{P} \text{ with } head(r) = a\}$$
$$\Psi_{\mathcal{P}}(\mathcal{I})_2 = \{a \in \Sigma \mid body(r)^{\mathcal{I}} \neq \mathbf{f} \text{ for some rule } r \in \mathcal{P} \text{ with } head(r) = a\}$$

The claim now follows immediately from the definition of $\Psi_{\mathcal{P}}(\mathcal{I})_2$. □

The following proposition relates the two notions of unfounded sets.

Proposition 5.14. *Let $\mathcal{P}$ be a logic program, $\mathcal{I}$ a three-valued interpretation and $U \subseteq \Sigma$. The following properties hold.*

- *If U is a 3-unfounded set, then U is a GRS-unfounded set.*
- *If $\mathcal{I}[U : \mathbf{f}]$ is more precise than $\mathcal{I}$, then U is a GRS-unfounded set if and only if U is a 3-unfounded set.*

Proof. The first claim follows directly from the definitions.

If $\mathcal{I}$ and U are chosen such that $\mathcal{I}[U : \mathbf{f}]$ is more precise than $\mathcal{I}$, for every formula φ, the condition $\varphi^{\mathcal{I}} = \mathbf{f}$ or $\varphi^{\mathcal{I}[U:\mathbf{f}]} = \mathbf{f}$ is equivalent with $\varphi^{\mathcal{I}[U:\mathbf{f}]} = \mathbf{f}$. Thus we conclude that in this case the notions of 3-unfounded set and GRS-unfounded set are indeed equivalent, which proves the second claim. □

Thus, for a certain class of interpretations, the two notions of unfounded sets coincide. Furthermore, Van Gelder et al. only use unfounded sets to define the well-founded model construction. It follows immediately from Lemma 3.4 by Van Gelder et al. (1991) that every partial interpretation $\mathcal{I}$ in that construction with GRS-unfounded set U satisfies the condition in the second claim in Proposition 5.14. This means that 3-unfounded sets and GRS-unfounded sets are equivalent for all interpretations that are relevant in the original work! Essentially, we provided a new formalisation of unfounded sets that coincides with the old definition on all interpretations used in the original work.

Corollary 5.6, which states that grounded models of $\mathcal{P}$ are fixpoints of $T_{\mathcal{P}}$ that permit no proper 2-unfounded sets, might sound familiar. Indeed, it has been shown that an interpretation is a *stable model* of a logic program if and only if it is a fixpoint of $T_{\mathcal{P}}$ and it permits no proper GRS-unfounded sets (Lifschitz, 2008).

Groundedness and Nondeterminism. In Section 5, we restricted our attention to logic programs with abstract constraint atoms in the *bodies* of rules, and we did not allow them in heads of rules. As argued by Marek et al. (2008), allowing them as well in heads gives rise to a *nondeterministic* generalisation of the immediate consequence operator. A consistent nondeterministic operator maps every point $x \in L$ to a non-empty set $O(x) \subseteq L$. Extending the notion of groundedness to this nondeterministic setting is out of the scope of this paper.

6. Grounded Fixpoints in Dung's Argumentation Frameworks and Abstract Dialectical Frameworks

Abstract argumentation frameworks (AFs) (Dung, 1995) are simple and abstract systems to deal with contentious information and draw conclusions from it. An AF is a directed graph where the nodes are arguments and the edges encode a notion of attack between arguments. In AFs, we are not interested in the actual content of arguments; this information is abstracted away. In spite of their conceptual simplicity, there exist many different semantics with different properties in terms of characterisation, existence and uniqueness.

Abstract dialectical frameworks (ADFs) (Brewka and Woltran, 2010; Brewka et al., 2013) are a generalisation of AFs in which not only attack, but also support, joint attack and joint support can be expressed.

Recently, Strass (2013) has showed that many of the existing semantics of AFs and ADFs can be obtained by direct applications of AFT. In this section we use the aforementioned study to relate grounded fixpoints to AFs and ADFs. We first do so for the case of AFs and afterwards generalise to ADFs.

6.1. Abstract Argumentation Frameworks

An *abstract argumentation framework* Θ is a directed graph (A, R) in which the nodes A represent arguments and the edges in R represent attacks between arguments. We say that a *attacks* b if $(a, b) \in R$. A set $S \subseteq A$ *attacks* a if some $s \in S$ attacks a. A set $S \subseteq A$ *defends* a if it attacks all attackers of a. An *interpretation* of an AF $\Theta = (A, R)$ is a subset S of A. The intended meaning of such an interpretation is that all arguments in S are accepted (or believed) and all arguments not in S are rejected. Interpretations are ordered according to the acceptance relation: $S_1 \leq S_2$ iff $S_1 \subseteq S_2$, i.e., if S_2 accepts more arguments than S_1. There exist many different semantics of AFs which each define different sets of acceptable arguments according to different standards or intuitions. The major semantics for argumentation frameworks can be formulated using two operators: the *characteristic function* F_Θ, which maps an interpretation S to

$$F_\Theta(S) = \{a \in A \mid S \text{ defends } a\}$$

and the operator U_Θ (U stands for unattacked), which maps an interpretation S to

$$U_\Theta(S) = \{a \in A \mid a \text{ is not attacked by } S\}.$$

An interpretation S is *conflict-free* if it is a postfixpoint of U_Θ ($S \leq U_\Theta(S)$), i.e., if no argument in S is attacked by S. The characteristic function is a monotone operator; its least fixpoint is called the *grounded extension* of Θ. The operator U_Θ is an anti-monotone operator; its fixpoints are called *stable extensions* of Θ. Many more semantics, such as *admissible interpretations*, *complete extensions*, *semi-stable extensions*, *stage extensions* and *preferred extensions* can be characterised using the above operators as well (Dung, 1995).

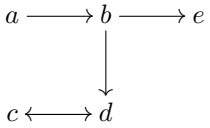
The following proposition shows that grounded extensions as defined in argumentation theory are indeed grounded in the sense defined in this paper.

Proposition 6.1. *The grounded extension of Θ is the unique grounded fixpoint of F_Θ.*

Proof. Follows immediately from Proposition 3.10 which states that a monotone operator has exactly one grounded fixpoint, namely its least fixpoint. □

The grounded extension S consists of all arguments a that should definitely be accepted: all arguments that are globally unattacked, defended by globally unattacked arguments, and so on (recursively). As such, the intuition regarding the grounded extension is similar to intuitions regarding grounded fixpoints: we only accept arguments with a good, non self-supporting defence.

Example 6.2. Consider the following framework:



In this example a is unattacked, hence should be accepted; b is attacked by a, hence should not be accepted. The argument e is defended by a, hence can safely be accepted. c and d mutually attack each other and hence, defend themselves. Since we have already established that b is rejected, the only remaining argument that defends c is c itself. The grounded extension rejects self-defending arguments (i.e., rejects both c and d) and hence is $\{a, e\}$.

Proposition 6.3. *An interpretation S is a stable extension of Θ if and only if it is a grounded fixpoint of U_Θ.*

Proof. Follows immediately from Proposition 3.11 which states that all postfixpoints of an antimonotone operator are grounded. Indeed, U_Θ is anti-monotone and stable extensions are exactly the fixpoints of U_Θ. □

Example 6.4 (Example 6.2 continued)**.** Stable extensions are more liberal in accepting arguments than the grounded extension. In stable extensions, arguments are "by default" accepted, unless another accepted argument contradicts them. The framework considered in this example has two stable extensions: $\{a, e, c\}$ and $\{a, e, d\}$.

6.2. Abstract Dialectical Frameworks

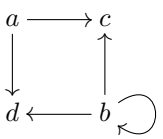
We now extend our theory to the more general case of ADFs. In the context of AFs, grounded fixpoints characterise two existing semantics, when applied to two previously defined operators. In the context of ADFs, however, this does not hold. Here, grounded fixpoints yield a new semantics.

An *abstract dialectical framework* is a triple $\Xi = (S, L, C)$, where

- S is a set of *arguments*
- $L \subseteq S \times S$ is a set of *links*; we define the parents of $s \in S$ as $par(s) = \{r \in S \mid (r, s) \in L\}$,
- $C = \{C_s^{in}\}_{s \in S}$ is a collection of sets C_s^{in} where for every s, $C_s^{in} \subseteq 2^{par(s)}$.

Intuitively, for every s, $par(s)$ is the set of arguments that influence whether or not S should be accepted. This influence can be positive (support), negative (attack) or a combination of both. An argument s should be accepted if for some set $A \in C_s^{in}$, all arguments in A are accepted and all arguments in $par(s) \setminus A$ are not. An argumentation framework $\Theta = (A, R)$ is an ADF in which all links are attack relations, i.e., for every s, $par(s) = \{s' \mid (s', s) \in R\}$ and $C_s^{in} = \{\emptyset\}$ (the only way to accept an argument is if none of its attackers is accepted).

Example 6.5. Let S be the set of arguments $\{a, b, c, d\}$ and L the following graph



Furthermore, $C_a^{in} = \{\emptyset\}$, $C_b^{in} = \{\{b\}\}$, $C_c^{in} = \{\emptyset, \{a\}, \{b\}\}$ and $C_d^{in} = \{\{a, b\}\}$. The following observations provide an intuitive reading of the ADF $\Xi = (S, L, C)$.

- a is a valid argument since it has trivial support.
- b supports itself: the only "reason" to believe b is b itself.
- a and b jointly attack c: since $C_c^{in} = \{\emptyset, \{a\}, \{b\}\}$, c is only rejected if a and b are both present
- a and b jointly support d: d is only acceptable if a and b both hold.

Intuitively, we should accept a. Whether or not to accept b, depends on which semantics for ADFs is used. Argument c can only be accepted in case we reject b, d should be accepted if we accept b.

With an ADF Ξ, we associate an operator G_Ξ on the lattice $\langle 2^S, \subseteq \rangle$ as follows (Strass, 2013):

$$G_\Xi(X) = \{s \in S \mid X \cap par(s) \in C_s^{in}\}.$$

This operator generalises the operator U_Θ for AFs.

Strass (2013) showed that many of the existing semantics for ADFs can be characterised with AFT. For example, *models* of Ξ are fixpoints of G_Ξ, *stable models* (Brewka et al., 2013) of Ξ are ultimate stable fixpoints of G_Ξ, etcetera. He also showed that there is a one to one correspondence between ultimate semantics for ADFs and for logic programs, in the sense that every ADF Ξ can be transformed to a logic program $\mathcal{P}$ such that G_Ξ and $T_\mathcal{P}$ coincide and vice versa. It is out of the scope of this paper to discuss all of the different semantics for ADFs. Here, we restrict ourselves to discussing the intuitions regarding grounded fixpoints. The intuitions underlying grounded fixpoints of ADFs are of course similar to those in other domains where AFT is applied. Groundedness serves to eliminate "ungrounded" reasoning: if the only reason for accepting an argument is that the argument itself holds, then this argument should be rejected.

Example 6.6 (Example 6.5 continued)**.** The operator G_Ξ from this example has two fixpoints: $\{a, b, d\}$ and $\{a, c\}$. The first of the two is not a grounded fixpoint because b itself is the only reason to accept b. Formally

$$G_\Xi(\{a, b, d\} \wedge \{a, c, d\}) = G_\Xi(\{a, d\}) = \{a, c\} \leq \{a, c, d\}$$

thus indeed $\{a, b, d\}$ is ungrounded. On the other hand, $\{a, c\}$ is grounded; it is the unique grounded fixpoint of G_Ξ.

We now study groundedness in the context of ADFs.

Definition 6.7 (Support)**.** Let $\Xi = (S, L, C)$ be an ADF and $X \subseteq S$. We say that $s \in S$ *has support* in X if $X \cap par(s) \in C_s^{in}$.

Definition 6.8 (Grounded model)**.** Let $\Xi = (S, L, C)$ be an ADF and $X \subseteq L$ a model of Ξ. We say that X is a *grounded model* of Ξ if for every set U with $\emptyset \subsetneq U \subseteq X$, at least one $u \in U$ has support in $X \setminus U$.

Thus, a grounded model is one without self-supporting arguments, i.e., without arguments that no longer have support once they are removed.

Below, $\overline{U}$ denotes the set complement of U, i.e., $\overline{U} = S \setminus U$.

Proposition 6.9. *Let $\Xi = (S, L, C)$ be an ADF. Then $X \subseteq S$ is a grounded fixpoint of G_Ξ if and only if X is a grounded model of Ξ.*

Proof. The proof is analogous to the proof of Proposition 5.5.

First, suppose X is a grounded fixpoint of G_Ξ. Since X is a fixpoint of G_Ξ, by definition it is a model of Ξ. If U is a set $\emptyset \subsetneq U \subseteq X$, we need to show that at least one $u \in U$ has support in $X \setminus U$. Suppose this condition is not satisfied, i.e., that no u has support in $X \setminus U$. This means that $G_\Xi(X \setminus U) \cap U = \emptyset$. Let $V = \overline{U}$; the previous equation translates to $G_\Xi(X \wedge V) \leq V$. Thus, the definition of groundedness yields $X \leq V$, and hence that $U \cap X = \emptyset$, which contradicts with the assumption that $\emptyset \subsetneq U \subseteq X$, hence X is indeed a grounded model of Ξ..

The reverse direction is analogous. Suppose X is a grounded model of Ξ.. Let V be such that $G_\Xi(X \wedge V) \leq V$ and let $U = \overline{V} \cap X$. Thus $U \subseteq X$ and (since $X \setminus U = X \wedge V$) $G_\Xi(X \setminus U) \cap U = \emptyset$. Thus, by our assumption, U is empty, i.e., $X \leq V$ and we conclude that X is indeed a grounded fixpoint in this case. □

It is worth noting that the set U in Definition 6.8 corresponds to a proper unfounded set in the case of logic programming.

Many semantics have been defined for ADFs. Most of these semantics are three-valued. The only two-valued semantics are *conflict-free* sets, *supported models* and *two-valued stable models*. Our algebraic results immediately yield that the two-valued stable semantics has the property that it only accepts grounded interpretations. Grounded fixpoints are a new element in the family of two-valued semantics of ADFs. We believe that this is an interesting new member: as illustrated above, it formalises simple and clear intuitions. Two-valued stable semantics and grounded fixpoint semantics formalise related ideas. As with logic programs, we conjecture that for large classes of ADFs these two semantics coincide; it remains an open research question to define those classes (or classes on which they differ). Since Strass (2013) defined transformations between logic programs and ADFs that preserve the operator, solving this research question will also solve the related open question from Section 5 and vice versa.

6.3. Discussion

Complexity. Strass has showed that there is a one to one correspondence between ultimate semantics for ADFs and for logic programs, in the sense that every ADF Ξ can be transformed to a logic program $\mathcal{P}$ such that G_Ξ and $T_\mathcal{P}$ coincide and vice versa. These results allow us to port complexity results from the field of logic programming to ADFs and vice versa. Hence, Theorem 5.7 yields that checking existence of a grounded fixpoint of an ADF is Σ_2^P-complete.

In the context of ADFs, one is often also interested in other forms of reasoning such as *credulous* or *sceptical* reasoning (Strass and Wallner, 2014). Analysing complexity of grounded fixpoint semantics for more forms of reasoning is a topic for future work.

7. Grounded Fixpoints of Autoepistemic and Default Theories

In this section, we study groundedness in the context of Moore's autoepistemic logic (AEL) (Moore, 1985) and Reiter's default logic (DL) (Reiter, 1980).

In the late seventies, the field of knowledge representation and more particularly, the field of non-monotonic reasoning, increasingly became concerned with the representation of and reasoning on default statements "most P's are Q's". The idea grew to interpret such statements as defeasible inference rules "if x is a P and it is not known that x is not a Q then (derive that) x is a Q". This idea was developed independently in default logic by Reiter (1980) and nonmonotonic logic I and II (McDermott and Doyle, 1980; McDermott, 1982). Not much later, Moore (1985) identified the latter sort of statements as *autoepistemic* statements and developed autoepistemic logic (AEL) for it.

In Moore's view, an autoepistemic theory $\mathcal{T}$ is the representation of the knowledge of a perfect, rational, introspective agent. The agent is *introspective* in the sense that propositions in its theory may refer to its own knowledge, through the modal operator K. The informal interpretation of this operator is "I (the agent) know that ...". The agent is a *perfect* reasoner in the sense that its knowledge is closed under entailment. It is *rational* in the sense that it only believes propositions contained in or entailed by its knowledge base $\mathcal{T}$. Thus, $\mathcal{T}$ expresses, directly or indirectly, all the agent knows. Levesque (1990) called this assumption about T the "All I Know Assumption". It is this assumption that distinguishes autoepistemic logic from the standard modal logic of knowledge S5. The challenge in defining such a logic lies in the fact that autoepistemic theories are *self-referential*: what is known by $\mathcal{T}$ is made up from what is expressed by its statements, but what is expressed by a statement depends on what is known by $\mathcal{T}$.

Moore (1985) formalised these ideas as follows. Let $\mathcal{L}$ be the language of propositional logic based on the vocabulary Σ. Extending this language with a modal operator K, yields the language $\mathcal{L}_K$ of modal propositional logic. An *autoepistemic theory* (over Σ) is a set of formulas in $\mathcal{L}_K$. A *modal formula* is a formula of the form $K\psi$, with ψ a formula. An *objective formula* is a formula without modal subformulas.

AEL uses the semantical concepts of standard modal logic. As before, an *interpretation* I is a subset of Σ. It formally represents a potential state of affairs of the world. A *possible world structure* is a set of interpretations. It can be seen as a Kripke structure with the total accessibility relation. The set of all

possible world structures, $\mathcal{W}_\Sigma$, is thus $2^{(2^\Sigma)}$. It forms a complete lattice under $\subseteq$ as well as under $\supseteq$. A possible world structure Q formally expresses a potential belief state of an agent by providing all the states of the world that the agent considers to be possible. Interpretations $I \in Q$ are formal representations of possible states of affairs and satisfy the propositions known by the agent. Interpretations $I \notin Q$ represent impossible states of affairs in the sense that they violate some of the agent's propositions.

The semantics of AEL is based on the standard S5 truth assignment. For arbitrary formula φ in $\mathcal{L}_K$, Q a possible world structure and I an interpretation, we define that φ is satisfied with respect to Q and I (denoted $Q, I \models \varphi$) by the standard recursive rules of propositional satisfaction, augmented with one additional rule:

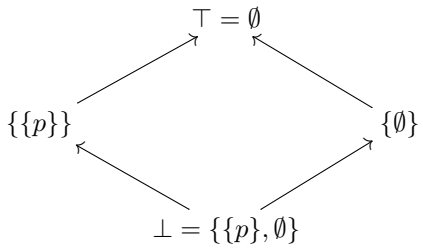
$$Q, I \models K\varphi \text{ if } Q, I' \models \varphi \text{ for every } I' \in Q.$$

For objective formulas φ, it holds that $Q, I \models \varphi$ if and only if $I \models \varphi$. We define $Q \models K\varphi$ (φ is known in Q) if $Q, I \models \varphi$ for every $I \in Q$. As can be seen from the definition of satisfaction, modal formulas are evaluated with respect to the possible world structure Q, while objective formulas are evaluated with respect to the world I.

Example 7.1. Consider a formula $\varphi = \neg p \wedge \neg Kp$. Let Q be the possible world structure $\{\{p\}, \emptyset\}$ and let $I = \emptyset$. Then, $Q, I \not\models p$, and $Q, I \not\models Kp$, hence $Q, I \models \varphi$.

The class $\mathcal{W}_\Sigma$ of possible world structures exhibits a natural knowledge order. Intuitively Q contains less knowledge than Q' if it has more possible worlds. Formally we define $Q \leq_k Q'$ if $Q \supseteq Q'$. The intuition underlying this order is clarified by considering the concept of the *objective theory* of a possible world structure Q. This is the set of objective formulas that are known in Q. Formally, it is defined as $Th_{obj}(Q) = \{\varphi \in \mathcal{L} \mid Q \models K\varphi\} = \{\varphi \in \mathcal{L} \mid \forall I \in Q : I \models \varphi\}$. Moore (1984) proved that the function Th_{obj} induces a one-to-one correspondence between possible world structures and sets of objective formulas closed under logical consequence. An obvious property is that $Q \leq_k Q'$ if and only $Th_{obj}(Q) \subseteq Th_{obj}(Q')$. Thus, if $Q \leq_k Q'$ then indeed Q possesses less knowledge than Q'.

With the order $\leq_k$, $\mathcal{W}_\Sigma$ forms a complete lattice. For example, if $\Sigma = \{p\}$, the associated lattice is:



Moore proposed to formalise the intuition that an AEL theory $\mathcal{T}$ expresses "all the agent knows" in semantical terms, as a condition on the possible world structure Q representing the agent's belief state. The condition is as follows: a world I is possible according to Q if and only if I satisfies $\mathcal{T}$ given Q, that is if $Q, I \models \mathcal{T}$. Equivalently, I is impossible if and only if I violates $\mathcal{T}$ given Q, or $Q, I \not\models \mathcal{T}$. Formally, Moore defines that Q is an *autoepistemic expansion* of $\mathcal{T}$ if for every world I, it holds that $I \in Q$ if and only if $Q, I \models \mathcal{T}$.

The above definition is essentially a fixpoint characterisation. The underlying operator $D_\mathcal{T}$ is:

$$D_\mathcal{T}(Q) = \{I \mid Q, I \models \mathcal{T}\}.$$

Clearly, Q is an autoepistemic expansion of T if and only if Q is a fixpoint of $D_\mathcal{T}$. These autoepistemic expansions are the possible world structures that, according to (Moore, 1985) express candidate belief states of an autoepistemic agent with knowledge base $\mathcal{T}$. Moore called such structures *grounded*.

Soon, researchers such as Halpern and Moses (1985) and Konolige (1988) pointed out certain "anomalies" in the expansion semantics. The simplest example is the theory $\mathcal{T} = \{Kp \Rightarrow p\}$. One of its expansions is $Q_2 = \{\{p\}\}$. The problem with Q_2 is that it is *self-supporting*: Q_2's assumption that p is known to be

true, is essential for deriving p. Even from Moore's perspective there might be a problem with such self-supporting belief states. In the first part of his work (Moore, 1985), he argues that sets of inference rules such as the theories that arise in nonmonotonic reasoning, correspond to autoepistemic theories. Viewed from this perspective, $\mathcal{T}$ is the singleton set consisting of one inference rule:

$$\frac{\vdash p}{p}$$

Surely, such an inference rule should be of no value, as it can only derive something that has been derived before! Therefore, the only acceptable belief state for $\mathcal{T}$ seems to be Q_1, the state of total ignorance.

Several attempts were done to strengthen Moore's semantics. Halpern and Moses (1985) proposed an alternative possible world semantics in which the model of an an AEL theory $\mathcal{T}$ is the $\leq_p$-least prefixpoint of $D_\mathcal{T}$, if it exists. Unfortunately, many simple and natural AEL theories have no model in this semantics. An example is $\{\neg Kp \Leftrightarrow q\}$ for which several prefixpoints of $D_\mathcal{T}$ exist but no least one. Here, Moore's semantics makes sense. The unique expansion $\{\{q\}, \{p, q\}\}$ captures the idea that there is no objective information about p, hence p is unknown; therefore, q holds.

Also Konolige (1988) attempted to refine Moore's semantics. He called expansions *weakly grounded* and proposed alternative definitions for so called *moderately grounded* and *strongly grounded* expansions. Intuitively, a possible world structure is moderately grounded if all the information it contains can be derived from $\mathcal{T}$ using only ignorance statements from Q. Thus, Q is moderately grounded if all knowledge in Q follows from $\mathcal{T}$ augmented with all statements of the form $\neg K\varphi$ such that $Q \not\models K\varphi$. Konolige proved that moderate grounded expansions are exactly the minimal fixpoints of $D_\mathcal{T}$. However, he pointed out that even moderately grounded expansions can give rise to ungrounded reasoning. He illustrated this with the following example.

Example 7.2. Consider the following theory

$$\mathcal{T} = \{\neg Kp \Rightarrow q, Kp \Rightarrow p\}.$$

This theory has two moderately grounded possible world structures, namely $Q_1 = \{\{p\}, \{p, q\}\}$ and $Q_2 = \{\{q\}, \{q, p\}\}$. Q_1 is the possible world structure in which p is known ($Kp, \neg K\neg p$) and q is not known ($\neg Kq, \neg K\neg q$).

Again, Konolige argued that Q_1 from Example 7.2 should not be grounded. Indeed, in Q_1, the knowledge of p is self-supported. The intended model here is $Q_2 = \{\{q\}, \{q, p\}\}$.

This motivated him to propose the strengthened notion of *strongly grounded* expansion. The disadvantage of this notion, as recognised by Konolige, is that it is only defined for theories in a normal form where every sentence is of the form

$$K\alpha \wedge \neg K\beta_1 \wedge \cdots \wedge \neg K\beta_n \Rightarrow \gamma,$$

where α, γ and the β_i are objective. Furthermore, whether or not a possible world structure is strongly grounded depends on which transformation to the normal form is used. In other words, strong groundedness is syntactically defined and may hold for one theory and not for an equivalent theory.

Example 7.3. Consider theories $\mathcal{T}_1 = \{p\}$ and $\mathcal{T}_2 = \{Kp \Rightarrow p, \neg Kp \Rightarrow p\}$. These theories are equivalent in the modal logic S5. However, $\{\{p\}\}$ is a strongly grounded expansion of $\mathcal{T}_1$, while it is not a strongly grounded expansion of $\mathcal{T}_2$.

We now investigate how the algebraical concept of grounded fixpoint translates to the setting of AEL and how it relates to the above ideas.

Definition 7.4 (Unfounded set of impossible worlds)**.** Let $\mathcal{T}$ be an AEL theory and Q a possible world structure. A non-empty set U of worlds is an *unfounded set of impossible worlds* of Q if $U \cap Q = \emptyset$ and for all $I \in U : (Q \cup U), I \models \mathcal{T}$.

If Q admits such a U, it contains unsupported knowledge. In particular, the knowledge that the worlds in U are impossible is unsupported, since if we weaken Q by accepting U as possible worlds, then none of the worlds of U can be dismissed as impossible. All of them satisfy $\mathcal{T}$ in the revised belief state $Q \cup U$. The desired property that an agent's knowledge will satisfy is thus that Q does not admit such a U.

Definition 7.5 (Grounded expansion)**.** Let $\mathcal{T}$ be an AEL theory. A possible world structure Q is *grounded for* $\mathcal{T}$ if Q does not admit an unfounded set of impossible worlds.

A possible world structure Q is a *grounded expansion* if it is an expansion and grounded.

As it turns out, this notion is again equivalent with the algebraical notion of groundedness.

Proposition 7.6. *A possible world structure Q is grounded for $\mathcal{T}$ if and only if Q is (strictly) grounded for $D_{\mathcal{T}}$ in the lattice $\langle \mathcal{W}_\Sigma, \leq_k \rangle$.*

Proof. The definition can be rephrased by focussing on $Q' = Q \cup U$. Q is grounded if there is no $Q' \supsetneq Q$ such that $Q' \not\subseteq D_{\mathcal{T}}(Q') \cup Q$. Then the proposition follows immediately from the fact that $\supsetneq$ is $<_k$, $\cup$ is $\bigwedge$ and $\subseteq = \geq_k$ in the lattice $\langle \mathcal{W}_\Sigma, \leq_k \rangle$. Then Definition 7.5 equals the definition of a strictly grounded lattice element. Furthermore, Proposition 3.5 guarantees that the notion of strict groundedness coincides with groundedness in powerset lattices. □

The intuitions expressed above correspond closely to those written down by Konolige (1988). On all the examples he gave, groundedness as we defined it, achieves the desired result. Furthermore, since grounded fixpoints are always minimal in $\leq_k$, our notion of groundedness is indeed stronger than the notion of moderate groundedness. We show below (in Corollary 7.10) that our notion of groundedness is slightly weaker than strong groundedness. Furthermore, groundedness is defined for every AEL theory (not just for a given normal form) and it is defined purely semantically: two equivalent theories have the same grounded expansions.

Example 7.7. Consider the following autoepistemic theory:

$$T = \{p, \neg Kp \Rightarrow q, Kq \Rightarrow q\}.$$

The intended possible world structure is clear here: p follows from T using the first sentence, hence p is known. The second sentence cannot be used to derive q, since Kp holds. Furthermore, the last sentence cannot be used to derive q since it first requires Kq. This theory has two autoepistemic expansions, namely $Q_1 = \{\{p\}, \{p, q\}\}$ (which corresponds to knowing p, and not knowing whether q holds or not) and $Q_2 = \{\{p, q\}\}$ (knowing both p and q). The first one is grounded, while the second is not (it is not even a minimal fixpoint since $Q_1 \leq_k Q_2$). Indeed, if we remove the knowledge that q holds from Q_2 (i.e., we turn the previously impossible world $\{p\}$ into a possible world by adding it to Q_2), then $\{p\}$ remains possible; that is, the belief that $\{p\}$ is impossible is not derived anymore. Hence Q_2 is not grounded.

7.1. Groundedness of the AFT family of semantics for AEL

As we saw, the problem of ungrounded expansions remained unsolved for several years. A new take at it was obtained when DMT applied AFT to AEL. We explain this approach.

The bilattice of $\mathcal{W}_\Sigma$ consists of pairs (P, C) of possible worlds. Intuitively, such pairs approximate possible world structures Q such that $C \subseteq Q \subseteq P$, i.e., $P \leq_k Q \leq_k C$: therefore, C is to be understood as a set of *certainly possible worlds* and P as a set of *possibly possible worlds.*

For such pairs, the standard 3- and 4-valued Kleene truth valuation of propositional logic can be extended to a truth function $\varphi^{(P,C),I}$ by adding the following rules for modal formulas:

- $K\varphi^{(P,C),I} = \mathbf{t}$ if for all $I' \in P$, $\varphi^{(P,C),I'} = \mathbf{t}$.
- $K\varphi^{(P,C),I} = \mathbf{f}$ if for some $I' \in C$, $\varphi^{(P,C),I'} = \mathbf{f}$.
- Otherwise, $K\varphi^{(P,C),I} = \mathbf{u}$

That is, φ is known in (P, C) if it holds in all possibly possible worlds, it is not known if does not hold in at least one certainly possible world. Otherwise, it cannot be determined if φ is known.

This truth valuation induces a bilattice operator $A_{\mathcal{T}}$ that maps pairs (P, C) to (P', C') where

$$C' = \{I \mid \mathcal{T}^{(P,C),I} = \mathbf{t}\} \text{ and}$$
$$P' = \{I \mid \mathcal{T}^{(P,C),I} \neq \mathbf{f}\}$$

Intuitively, the derived certainly possible worlds are those in which $\mathcal{T}$ evaluates to true, and the derived possibly possible worlds are those in which $\mathcal{T}$ does not evaluate to false.

DMT showed that $A_{\mathcal{T}}$ is an approximator of $D_{\mathcal{T}}$. Hence, it induces a class of existing and new semantics for AEL: Moore's expansion semantics (supported fixpoints), Kripke-Kleene expansion semantics (Denecker et al., 1998) (Kripke-Kleene fixpoints), stable extension semantics (stable fixpoints) and well-founded extension semantics (well-founded fixpoints) (Denecker et al., 2003). The latter two were new semantics induced by AFT. As a corollary of Theorem 4.6 and Proposition 4.3, we obtain the following analysis of groundedness.

Corollary 7.8. *Stable and two-valued well-founded extensions of $\mathcal{T}$ are grounded (in the sense of Definition 7.5). If the well-founded extension is two-valued, it is the unique stable extension and the unique grounded expansion. If the Kripke-Kleene expansion is two-valued, it is the well-founded extension, the unique expansion, the unique stable extension and it is grounded.*

For the AEL theory $\{Kp \Rightarrow p\}$, the possible world structure $\{\emptyset, \{p\}\}$ is the well-founded and the unique stable extension. It is also the unique grounded expansion.

An example of an AEL theory with a grounded expansion that is not a stable extension is $\{Kp \Rightarrow p, \neg Kp \Rightarrow p\}$. Its unique grounded expansion is $\{\{p\}\}$ but it has no stable extensions and the well-founded extension is three-valued.

7.2. Default logic

Similar to McDermott and Doyle (1980), Reiter (1980) proposed to implement defaults "most P's are Q's" by their defeasible inference rule "If x is known to be a P and it is consistent to believe that it is a Q, then (infer that) x is a Q". Note that, through the standard duality of modal logic, the second condition is equivalent to "it is not known that x is not a Q". A default logic theory consists of sentences of propositional calculus and default expressions of the form:

$$\frac{\alpha : M\beta_1, \ldots, M\beta_n}{\gamma}$$

where $\alpha, \beta_1, \ldots, \beta_n, \gamma$ are expressions of propositional logic. The informal semantics of such an expression is "if α is known, and it is consistent to believe β_1, ..., and β_n, then γ holds". For this logic, Reiter developed his extension semantics (see below). It soon became clear that Reiter's extension semantics had some exquisite features. For example, consider the following default theory:

$$\left\{\frac{p:}{p}\right\}$$

It has one extension, namely the theory $Th_{obj}(\{\emptyset, \{p\}\})$. Clearly, in this example default logic avoids the ungrounded model that the related AEL theory $\{Kp \Rightarrow p\}$ has. The sort of ungrounded models that existed for AEL were never discovered in DL.

There is an obvious correspondence on the informal level between default expressions and AEL formulas. This connection was explicated by Konolige (1988) who proposed to translate a default theory $\mathcal{T}$ to the AEL theory $Kon(\mathcal{T})$ consisting of AEL formulas:

$$K\alpha \wedge \neg K\neg\beta_1 \wedge \cdots \wedge \neg K\neg\beta_n \Rightarrow \gamma$$

However Kon is not equivalence preserving. Indeed, $\left\{\frac{p:}{p}\right\}$ is a counterexample, as it translates to the AEL theory $\{Kp \Rightarrow p\}$ which under Moore's expansion semantics is not equivalent. In fact, Gottlob (1995) showed

that no modular translations exist from DL to AEL (but non-modular transformations exist). For a while, it was believed that AEL and DL were quite different logics. Later, DMT (2003) showed that, similar as for AEL, also with a default theory $\mathcal{T}$ it is possible to associate an approximator $A_{\mathcal{T}}$. This induced again the family of AFT semantics for DL which, just like for AEL, included several existing and some new semantics for DL: weak extensions (Marek and Truszczynski, 1989) (supported fixpoints), Kripke-Kleene extensions (Kripke-Kleene fixpoints), Reiter's extensions (Reiter, 1980) (stable fixpoints) and well-founded extension semantics (Baral and Subrahmanian, 1993) (well-founded fixpoint). Only Kripke-Kleene extensions were a new semantics induced by AFT.

Interestingly, it then appeared that $A_{\mathcal{T}} = A_{Kon(\mathcal{T})}$ for every default logic theory $\mathcal{T}$. Thus, a default theory and Konolige's (modular) translation to AEL have identical approximators. Therefore, they induce the same family of semantics. E.g., the extensions of a default theory correspond to the stable extensions of its AEL translation. DMT (2011) argued that the different semantics of AEL and DL induced by AFT correspond to different *dialects* of autoepistemic reasoning. The mismatch found between AEL and DL was due to the fact that Reiter's and Moore's semantics formalised different dialects of autoepistemic reasoning. However, Konolige's translation is correct on a deeper level: it preserves equivalence under every dialect of autoepistemic reasoning!

The above exposition not only tells the story of the link between AEL and DL but also provides the formal material for an analysis of groundedness in the context of DL. Definition 7.5 also defines groundedness in DL and using the fact that AFT characterises all main semantics of DL we obtain the following corollary of Theorem 4.6 and Proposition 4.3.

Corollary 7.9. *Reiter's extensions are grounded. If the well-founded extension of a DL theory is two-valued then it is the unique extension and the unique grounded weak extension. If the Kripke-Kleene extension of a DL theory is two-valued then it is grounded (and also the well-founded extension, the unique weak extension, the unique Reiter extension).*

An example of a DL theory that has no Reiter extensions but has a grounded weak extension is:

$$\left\{ \frac{p:}{p}, \frac{:M\neg p}{p} \right\},$$

which corresponds to the AEL theory $\{Kp \Rightarrow p, \neg Kp \Rightarrow p\}$. Its unique grounded weak extension is $Th_{obj}(\{\{p\}\})$.

To end the discussion of groundedness in AEL and DL, we return to AEL and the strongly grounded expansions defined by Konolige. He defined them for AEL theories consisting of formulas in the following canonical form:

$$K\alpha \wedge \neg K\neg\beta_1 \wedge \cdots \wedge \neg K\neg\beta_n \Rightarrow \gamma$$

Such formulas are exactly the AEL formulas in the range of Kon. Hence, every AEL theory in this canonical form is $Kon(\mathcal{T})$ for some DL theory $\mathcal{T}$. Konolige showed that the strongly grounded expansions of $Kon(\mathcal{T})$ are exactly the Reiter extensions of $\mathcal{T}$. Combining this result with the previous corollary yields the following.

Corollary 7.10. *If Q is a strongly grounded AEL expansion of $\mathcal{T}$, then Q is a grounded expansion of $\mathcal{T}$.*

8. Conclusion

The concept of groundedness is widespread in various logic and knowledge representation domains. In this paper, we formalised this as a property of fixpoints of a lattice operator. In a first step, we analysed grounded fixpoints and their relation to other notions of fixpoints, in particular minimal fixpoints and the different sorts of fixpoints classified in approximation fixpoint theory. The main results here are: given an operator O and an approximator A of O, all A-stable fixpoints are grounded for O and all grounded fixpoints of O are minimal fixpoints approximated by the A-well-founded fixpoint.

We then investigated groundedness in logic programming, abstract argumentation frameworks, autoepistemic logic and default logic. In each of these fields, the concept of groundedness had appeared before,

although not always under the same name. We showed links with the notion of unfounded set in logic programming, with groundedness in Dung's argumentation frameworks and various notions of groundedness in autoepistemic logic and default logic. In each logic, we investigated groundedness of the existing semantics and the new semantics induced by grounded fixpoints. For example, in the context of argumentation frameworks, we discovered that grounded fixpoints recover two existing semantics. In the more general abstract dialectical frameworks, grounded fixpoint yield a new semantics with a clear informal semantics: a fixpoint is grounded if it contains no self-supporting arguments. In autoepistemic logic and default logic, groundedness captures intuitions written down by Konolige.

In summary, the main contributions of the presented work are: we presented a generic formalisation of the concept of groundedness, an analysis of groundedness of existing semantics in a range of logics, and last but not least, the definition of the new semantics for operator based logics induced by grounded fixpoints. In comparison with approximation fixpoint theory, grounded fixpoints are defined directly in terms of the original lattice operator and do not require the invention of an approximator. Grounded fixpoint semantics is compact, intuitive and applicable to all operator based logics. Moreover, it is easily extensible. When adding new language constructs, it suffices to extend the operator for them.

Acknowledgements. We would like to thank the anonymous reviewers, Maurice Bruynooghe and Hannes Strass for their feedback on this work. This work was supported by the KU Leuven under project GOA 13/010 and by the Research Foundation - Flanders (FWO-Vlaanderen).

References

Abiteboul, S., Vianu, V., 1991. Datalog extensions for database queries and updates. J. Comput. Syst. Sci. 43 (1), 62–124.
URL http://dx.doi.org/10.1016/0022-0000(91)90032-Z

Antic, C., Eiter, T., Fink, M., 2013. Hex semantics via approximation fixpoint theory. In: Cabalar, P., Son, T. C. (Eds.), LPNMR. Vol. 8148 of LNCS. Springer, pp. 102–115.
URL http://dx.doi.org/10.1007/978-3-642-40564-8_11

Baral, C., De Giacomo, G., Eiter, T. (Eds.), 2014. Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014. AAAI Press.
URL http://www.aaai.org/Library/KR/kr14contents.php

Baral, C., Subrahmanian, V., 1993. Dualities between alternative semantics for logic programming and nonmonotonic reasoning. Journal of Automated Reasoning 10 (3), 399–420.
URL http://dx.doi.org/10.1007/BF00881799

Bi, Y., You, J.-H., Feng, Z., 2014. A generalization of approximation fixpoint theory and application. In: Kontchakov, R., Mugnier, M.-L. (Eds.), Web Reasoning and Rule Systems. Vol. 8741 of LNCS. Springer International Publishing, pp. 45–59.
URL http://dx.doi.org/10.1007/978-3-319-11113-1_4

Bogaerts, B., Vennekens, J., Denecker, M., 2015. Grounded fixpoints. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25–29, 2015 Austin, Texas, USA. Accepted.
URL https://lirias.kuleuven.be/handle/123456789/471671

Bogaerts, B., Vennekens, J., Denecker, M., Van den Bussche, J., 2014. FO(C): A knowledge representation language of causality. TPLP 14 (4-5-Online-Supplement), 60–69.
URL https://lirias.kuleuven.be/handle/123456789/459436

Brewka, G., Strass, H., Ellmauthaler, S., Wallner, J. P., Woltran, S., 2013. Abstract dialectical frameworks revisited. In: Rossi, F. (Ed.), IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013. IJCAI/AAAI.
URL http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6551

Brewka, G., Woltran, S., 2010. Abstract dialectical frameworks. In: Lin, F., Sattler, U., Truszczyński, M. (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13, 2010. AAAI Press, pp. 102–111.
URL http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1294

Denecker, M., Marek, V., Truszczyński, M., July 26-30 1998. Fixpoint 3-valued semantics for autoepistemic logic. In: AAAI'98. MIT Press, Madison, Wisconsin, pp. 840–845.
URL http://www.aaai.org/Papers/AAAI/1998/AAAI98-119.pdf

Denecker, M., Marek, V., Truszczyński, M., 2000. Approximations, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning. In: Minker, J. (Ed.), Logic-Based Artificial Intelligence. Vol. 597 of The Springer International Series in Engineering and Computer Science. Springer US, pp. 127–144.
URL http://dx.doi.org/10.1007/978-1-4615-1567-8_6

Denecker, M., Marek, V., Truszczyński, M., 2003. Uniform semantic treatment of default and autoepistemic logics. Artif. Intell. 143 (1), 79–122.
URL http://dx.doi.org/10.1016/S0004-3702(02)00293-X

Denecker, M., Marek, V., Truszczyński, M., Jul. 2004. Ultimate approximation and its application in nonmonotonic knowledge representation systems. Information and Computation 192 (1), 84–121.
URL https://lirias.kuleuven.be/handle/123456789/124562

Denecker, M., Marek, V., Truszczyński, M., 2011. Reiter's default logic is a logic of autoepistemic reasoning and a good one, too. In: Brewka, G., Marek, V., Truszczyński, M. (Eds.), Nonmonotonic Reasoning – Essays Celebrating Its 30th Anniversary. College Publications, pp. 111–144.
URL http://arxiv.org/abs/1108.3278

Denecker, M., Ternovska, E., Apr. 2008. A logic of nonmonotone inductive definitions. ACM Trans. Comput. Log. 9 (2), 14:1–14:52.
URL http://dx.doi.org/10.1145/1342991.1342998

Denecker, M., Vennekens, J., 2007. Well-founded semantics and the algebraic theory of non-monotone inductive definitions. In: Baral, C., Brewka, G., Schlipf, J. S. (Eds.), LPNMR. Vol. 4483 of LNCS. Springer, pp. 84–96.
URL http://dx.doi.org/10.1007/978-3-540-72200-7_9

Denecker, M., Vennekens, J., 2014. The well-founded semantics is the principle of inductive definition, revisited. In: (Baral et al., 2014), pp. 22–31.
URL https://lirias.kuleuven.be/handle/123456789/448356

Dung, P. M., 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77 (2), 321 – 357.
URL http://dx.doi.org/10.1016/0004-3702(94)00041-X

Faber, W., Pfeifer, G., Leone, N., 2011. Semantics and complexity of recursive aggregates in answer set programming. Artif. Intell. 175 (1), 278–298.
URL http://dx.doi.org/10.1016/j.artint.2010.04.002

Ferraris, P., 2005. Answer sets for propositional theories. In: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). pp. 119–131.
URL http://dx.doi.org/10.1007/11546207_10

Fitting, M., 2002. Fixpoint semantics for logic programming — A survey. Theoretical Computer Science 278 (1-2), 25–51.
URL http://dx.doi.org/10.1016/S0304-3975(00)00330-3

Gelfond, M., Lifschitz, V., 1988. The stable model semantics for logic programming. In: Kowalski, R. A., Bowen, K. A. (Eds.), ICLP/SLP. MIT Press, pp. 1070–1080.
URL http://dx.doi.org/10.1.1.24.6050

Gelfond, M., Zhang, Y., 2014. Vicious circle principle and logic programs with aggregates. TPLP 14 (4-5), 587–601.
URL http://dx.doi.org/10.1017/S1471068414000222

Gottlob, G., 1995. Translating default logic into standard autoepistemic logic. J. ACM 42 (4), 711–740.
URL http://dx.doi.org/10.1145/210332.210334

Halpern, J. Y., Moses, Y., 1985. Towards a theory of knowledge and ignorance: Preliminary report. In: Apt, K. R. (Ed.), Logics and Models of Concurrent Systems. Vol. 13 of NATO ASI Series. Springer Berlin Heidelberg, pp. 459–476.
URL http://dx.doi.org/10.1007/978-3-642-82453-1_16

Kleene, S. C., 1938. On notation for ordinal numbers. The Journal of Symbolic Logic 3 (4), 150–155.
URL http://www.jstor.org/stable/2267778

Konolige, K., 1988. On the relation between default and autoepistemic logic. Artif. Intell. 35, 343–382.
URL http://dx.doi.org/10.1016/0004-3702(88)90021-5

Levesque, H. J., 1990. All I know: A study in autoepistemic logic. Artif. Intell. 42 (2-3), 263–309.
URL http://dx.doi.org/10.1016/0004-3702(90)90056-6

Lifschitz, V., 2008. Twelve definitions of a stable model. In: García de la Banda, M., Pontelli, E. (Eds.), ICLP. Vol. 5366 of LNCS. Springer, pp. 37–51.
URL http://dx.doi.org/10.1007/978-3-540-89982-2_8

Marek, V., Niemelä, I., Truszczyński, M., 2008. Logic programs with monotone abstract constraint atoms. TPLP 8 (2), 167–199.
URL http://dx.doi.org/10.1017/S147106840700302X

Marek, V., Truszczynski, M., 1989. Relating autoepistemic and default logics. In: Brachman, R. J., Levesque, H. J., Reiter, R. (Eds.), Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR'89). Toronto, Canada, May 15-18 1989. Morgan Kaufmann, pp. 276–288.
URL http://dl.acm.org/citation.cfm?id=112950

McDermott, D., 1982. Nonmonotonic logic II: Nonmonotonic modal theories. Journal of the ACM 29 (1), 33–57.
URL http://dl.acm.org/citation.cfm?id=322293

McDermott, D., Doyle, J., 1980. Nonmonotonic logic I. Artif. Intell. 13 (1-2), 41–72.
URL http://hdl.handle.net/1721.1/6303

Moore, R. C., 1984. Possible-world semantics for autoepistemic logic. In: Proceedings of the Workshop on Non-Monotonic Reasoning. pp. 344–354, reprinted in: M. Ginsberg, ed., *Readings on Nonmonotonic Reasoning*, pages 137–142, Morgan Kaufmann, 1990.
URL http://www.sri.com/sites/default/files/uploads/publications/pdf/616.pdf

Moore, R. C., 1985. Semantical considerations on nonmonotonic logic. Artif. Intell. 25 (1), 75–94.
URL http://dx.doi.org/10.1016/0004-3702(85)90042-6

Pelov, N., Denecker, M., Bruynooghe, M., 2007. Well-founded and stable semantics of logic programs with aggregates. TPLP 7 (3), 301–353.

URL http://dx.doi.org/10.1017/S1471068406002973
Reiter, R., 1980. A logic for default reasoning. Artif. Intell. 13 (1-2), 81–132.
URL http://dx.doi.org/10.1016/0004-3702(80)90014-4
Son, T. C., Pontelli, E., Elkabani, I., 2006. An unfolding-based semantics for logic programming with aggregates. CoRR abs/cs/0605038.
URL http://arxiv.org/abs/cs/0605038
Strass, H., 2013. Approximating operators and semantics for abstract dialectical frameworks. Artif. Intell. 205, 39–70.
URL http://dx.doi.org/10.1016/j.artint.2013.09.004
Strass, H., Wallner, J. P., 2014. Analyzing the computational complexity of abstract dialectical frameworks via approximation fixpoint theory. In: (Baral et al., 2014), pp. 101–110.
URL http://www.aaai.org/ocs/index.php/KR/KR14/paper/view/7917
van Emden, M. H., Kowalski, R. A., 1976. The semantics of predicate logic as a programming language. J. ACM 23 (4), 733–742.
URL http://dx.doi.org/10.1145/321978.321991
Van Gelder, A., Ross, K. A., Schlipf, J. S., 1991. The well-founded semantics for general logic programs. J. ACM 38 (3), 620–650.
URL http://dx.doi.org/10.1145/116825.116838
Vennekens, J., Gilis, D., Denecker, M., 2006. Splitting an operator: Algebraic modularity results for logics with fixpoint semantics. ACM Trans. Comput. Log. 7 (4), 765–797.
URL http://dx.doi.org/10.1145/1182613.1189735