Operator-based semantics for logic programs

Course at the International Master Programme for Computational Logic
Departement of Computer Science, Technische Universität Dresden

by Dr. Pascal Hitzler

Institute AIFB, Universität Karlsruhe (TH)
hitzler@aifb.uni-karlsruhe.de, www.aifb.uni-karlsruhe.de/WBS/phi

Winter Semester 2004/2005

22/10/04

Abstract

The question how knowledge can be represented by means of logic programs with negation has been a driving force for the field of non-monotonic reasoning. Implemented systems, known as answer set programming systems, have emerged recently and are currently being used in various application domains like the semantic web.

Intuitively, logic programs are being used for encoding commonsense resoning, in particular the phenomenon that human reasoning tends to "jump to conclusions" under incomplete knowledge. Formally, this kind of reaoning is described by interpreting the first-order syntax of logic programs in a more sophisticated way. This, in turn, is most easily done by means of fixed points of semantic operators.

The study of different semantic operators and their relationships gives rise to a rich theory. In this lecture, we will undertake a thorough formal study of these issues as a foundation for the study of advanced topics in theoretical or applied non-mononotonic reasoning. Particular emphasis will be on basic paradigms and on the supported, Kripke-Kleene, stable, and well-founded semantics for logic programs and their rich relationships.

Contents

1 Introduction

First: Organization. Meet 16:00 to 19:20 on the following Fridays
October 22nd, 2004
November 5th, 2004
November 12th, 2004
December 10th, 2004
January 7th, 2005
January 21st, 2005
January 28th, 2005
February 4th, 2005 (only 16:00 to 17:30)

Exams: February 11th, 14:00 to 17:00

Modules: KRAI, TCSL

Content: ASP, LP (semantic foundations)

1. Definite Programs (least model semantics)
2. normal programs (classical + default negation, stable models)
3. three-valued semantics for progs with default negation (e.g. well-founded semantics

In particular: Relations between these different semantics!

Semantics: Wikipedia

> In general, semantics (from the Greek semantikos, or "significant meaning," derived from sema, sign) is the study of meaning, in some sense of that term. Semantics is often opposed to syntax, in which case the former pertains to what something means while the latter pertains to the formal structure/patterns in which something is expressed (e.g. written or spoken).

Different semantics: declarative, procedural (in LP: via algorithm. Elsewhere: operational), denotational, operational (in LP: via operators. Elsewhere: same as procedural), fixed-point, etc.

We focus here on declarative and fixed-point semantics for logic programs (with negation). An example for the denotational approach is the following from [Hit97]

```
PROGRAM factorial(input,output);

CONST m=maxint;
TYPE  natural = 0..m;
```

```
VAR n: natural;

FUNCTION f(n:natural):natural;
    BEGIN
    IF n=0 THEN f:=1
           ELSE f:=n*f(n-1);
    END;

BEGIN
read(n);
write(n,' Fakultaet ist ',f(n));
END.
```

As a recursive function:

$$\mathsf{fact} : \mathbb{N} \longrightarrow \mathbb{N} : \mathsf{fact}(n) := \begin{cases} 1 & \text{for } n = 0 \\ n \cdot \mathsf{fact}(n-1) & \text{for } n \geq 1. \end{cases}$$

Now convert this into a functional of partial funcions on $\mathbb{N}$, as follows.

$$\mathsf{Fact} : [\mathbb{N} \to \mathbb{N}] \longrightarrow [\mathbb{N} \to \mathbb{N}] : \mathsf{Fact}(h) := \left(n \mapsto \begin{cases} 1 & \text{for } n = 0 \\ n \cdot h(n-1) & \text{for } n \geq 1 \end{cases} \right).$$

Here, $[\mathbb{N} \to \mathbb{N}]$ is the set of all partial functions on $\mathbb{N}$.

Exercise 1
Prove that fact *is a fixed point of* Fact*, i.e. show that* $\mathsf{Fact}(\mathsf{fact}) = \mathsf{fact}$*.*

Exercise 1 shows that the *meaning* (i.e. the *semantics*) of fact is encoded as fixed point of Fact. But how to obtain this fixed point (i.e. meaning)?

Identify each element of $[\mathbb{N} \to \mathbb{N}]$ with its graph. We furthermore define recursively $\mathsf{Fact} \uparrow 0 = \{(0,0)\}$ and $\mathsf{Fact} \uparrow (n+1) = \mathsf{Fact}(\mathsf{Fact} \uparrow n)$. Writing F_n for $\mathsf{Fact} \uparrow n$, we can compute

$$\begin{aligned} F_0 &:= \emptyset \\ F_1 &:= \mathsf{Fact}(F_0) = \{(0,1)\} \\ F_2 &:= \mathsf{Fact}(F_1) = \{(0,1),(1,1)\} \\ F_3 &:= \mathsf{Fact}(F_2) = \{(0,1),(1,1),(2,2)\} \\ F_4 &:= \mathsf{Fact}(F_3) = \{(0,1),(1,1),(2,2),(3,6)\} \\ &\ldots. \end{aligned}$$

We can now order $[\mathbb{N} \to \mathbb{N}]$ via set-inclusion of the graphs, and note that $\mathsf{Fact} \uparrow n \subseteq \mathsf{Fact} \uparrow (n+1)$ for all $n \in \mathbb{N}$. We define

$$\mathsf{Fact} \uparrow \omega = \bigcup_{n=1}^{\infty} \mathsf{Fact} \uparrow n.$$

Obviously, $\mathsf{Fact} \uparrow \omega = \mathsf{fact}$, and fact is a fixed point of Fact. Indeed (as we will see later), it is its *least* fixed point.

So the meaning of the factorial programme fact can be described as the least fixed point of the associated mapping Fact.

Exercise 2 ([SHLG94])
Write a naive imperative recursive algorithm for computing the greatest common divisor of any given pair of natural numbers. Write the same algorithm as a recursive function gcd*. Convert* gcd *into a functional* GCD *as above, such that its least fixed point is* gcd*. Give* $\mathsf{GCD} \uparrow n$ *for* $n = 0, 1, 2$*.*

This approach works *very well* for functional programming languages, and to a certain extent for imperative paradigms. For logic programs, the situation is much more difficult when negation is present. In general, for programs with negation least fixed points describing the semantics do not exist.

This observation is the starting point for a rich theory on the declarative reading of negation in logic programming.

2 Two-valued semantics for logic programs

Standard general references for logic programming: [Llo88, Apt97].

2.1 Definition A (*normal*) *logic program* is a finite set of (universally quantified) *clauses* of the form $\forall(A \leftarrow A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_m)$, commonly written as $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$, where A, A_i, and B_j, for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, are atoms over some given first order language. A is called the *head* of the clause, while the remaining atoms make up the *body* of the clause. We allow a body, i.e. a conjunction, to be empty, in which case it always evaluates to true. A clause with empty body is called a *unit clause* or a *fact*. A clause is called *definite*, if it contains no negation symbol. A program is called *definite* if it consists only of definite clauses. We will usually denote atoms with A or B, and literals, which may be atoms or negated atoms, by L or K.

Given a logic program P, we can extract from it the components of a first order language. The corresponding set of ground atoms, i.e. the *Herbrand base* of the program, will be denoted by B_P. For a subset $I \subseteq B_P$, we set $\neg I = \{\neg A \mid A \in B_P\}$. The set of all ground instances of P with respect to B_P will be denoted by $\mathsf{ground}(P)$.

Unless otherwise mentioned, P will in the following denote some arbitrary normal logic program.

2.2 Example For the program Even consisting of the two clauses

$$\begin{aligned} \mathtt{even}(0) &\leftarrow \\ \mathtt{even}(s(X)) &\leftarrow \neg\mathtt{even}(X), \end{aligned}$$

the underlying language has constants $\{0\}$, function symbols $\{s\}$ and predicate symbols $\{\mathtt{even}\}$. We have $B_{\mathrm{Even}} = \{\mathtt{even}(s^n(0)) \mid n \in \mathbb{N}\}$ and $\mathsf{ground}(\mathrm{Even})$ consists of the set $\{\mathtt{even}(0) \leftarrow\} \cup \{\mathtt{even}\,(s^{n+1}(0)) \leftarrow \neg\mathtt{even}\,(s^n(0)) \mid n \in \mathbb{N}\}$.

2.3 Program (Tweety1) Let Tweety1 be the program consisting of the following clauses.

$$\begin{aligned} \texttt{penguin}(\texttt{tweety}) &\leftarrow \\ \texttt{bird}(\texttt{bob}) &\leftarrow \\ \texttt{bird}(X) &\leftarrow \texttt{penguin}(X) \\ \texttt{flies}(X) &\leftarrow \texttt{bird}(X), \neg\texttt{penguin}(X) \end{aligned}$$

Tweety1 obviously represents the following knowledge: `tweety` is a penguin, `bob` is a bird, all penguins are birds, and every bird which is not a penguin can fly.

Exercise 3
Use your favorite Prolog-Interpreter (e.g. SWI-Prolog) to solve the 8-queens problem without using any built-ins. The 8-queens problem is as follows. Place 8 queens on 8 of the 64 squares of a chessboard, such that no two queens share a row, a column, or a diagonal. Under a diagonal we understand each line of corner-adjacent squares along a 45 degrees angle to the ground line (hence, there are altogether 30 different diagonals).

Logic programs are finite sets of first-order formulae (i.e. they are *theories*) and have *interpretations* and *models*, as known from predicate logic. A (Herbrand-)-*interpretation* is thus simply a mapping from ground atoms to the set $\{\mathbf{t}, \mathbf{f}\}$ of truth values. Each (Herbrand-)interpretation $I : B_P \to \{\mathbf{t}, \mathbf{f}\}$ can be interpreted as a subset of B_P by identifying it with $\{A \in B_P \mid I(A) = \mathbf{t}\}$, and vice-versa. The set of all (Herbrand-)interpretations of a logic program P is denoted by I_P. In the following, we will restrict our attention to Herbrand-interpretations.

2.4 Definition Given a (normal) logic program P, we define an operator $T_P : I_P \to I_P$ by setting $T_P(I)$ to be the set of all $A \in B_P$ for which there exists a clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$ with $I \models \texttt{body}$.

The set I_P of all interpretations of P has subset-inclusion as a natural order.

2.5 Proposition Let P be any normal logic program. Then the models of P are exactly the pre-fixed points of T_P, i.e. the models are exactly those $I \in I_P$ for which $T_P(I) \subseteq I$ holds.

Proof: Let $I \in I_P$ be a model for P and let $A \in T_P(I)$. Then there is a clause $A \leftarrow L_1, \ldots, L_n$, call it C, in $\mathsf{ground}(P)$ with $I(L_1 \wedge \ldots \wedge L_n) = \mathbf{t}$. Since I is a model for P, we also have $I(C) = \mathbf{t}$. Hence $I(A) = \mathbf{t}$, and so $A \in I$, as required.

Conversely, let $T_P(I) \subseteq I$ and let $A \leftarrow L_1, \ldots, L_n$ be a clause C in $\mathsf{ground}(P)$ with $I(L_1 \wedge \ldots \wedge L_n) = \mathbf{t}$. Then $A \in T_P(I) \subseteq I$, and hence $I(A) = \mathbf{t}$ and in consequence $I(C) = \mathbf{t}$, as required. ■

Exercise 4
Describe all models of the Program Even. Are there countably or uncountably many models?

Exercise 5
Compute $T^n_{Even}(\emptyset)$ for $n = 0, \ldots, 5$. (Note: $T^0_P(I) = I$ and $T^{n+1}_P(I) = T_P(T^n_P(I))$.)

The program consisting of the single clause $q \leftarrow p$ has $\{q\}$ as a model, but it is apparent that this is not a good way of assigning a *meaning* to the program, if considering it e.g. from a Prolog perspective.

2.1 Least model semantics for definite programs

2.6 Definition ([SHLG94]) A *poset* (partially ordered set) is a set X equipped with a *partial order*, i.e. with a binary relation which is reflexive, antisymmetric, and transitive. A poset is *linearly ordered* (or a *chain*) if each two elements are comparable in the order. A poset is an *ω-cpo*, if it has a lest element $\perp$ and every linearly ordered sequence (or *ω-chain* of elements has a supremum (least upper bound).

Let $(X, \leq)$ be an ω-cpo. A function $f : X \to X$ is *ω-continuous* if the following conditions hold.

(i) f is monotonic, i.e. $a \leq b$ implies $f(a) \leq f(b)$ for all $a, b \in X$.

(ii) f preserves suprema of ω-chains, i.e. for any ω-chain $(x_i)_{i\in\mathbb{N}}$ we have $f(\sup x_i) = \sup f(x_i)$.

Note that monotonicity implies that the image of an ω-chain is again an ω-chain.

Exercise 6
(a) Show that I_P is an ω-cpo, if ordered by set-inclusion.

(b) Describe a subset of I_P which has minimal elements, but no least element.

(c) Describe a linearly ordered subset of I_P which does not have a greatest element.

(d) Does there exist a linearly ordered subset of I_P without any minimal element?

Exercise 7
Let $I_{P,3}$ be the set of all pairs (A, B) with $A, B \subseteq B_P$ and $A \cap B = \emptyset$. For $(A, B), (C, D) \in I_{P,3}$ define $(A, B) \leq (C, D)$ iff $A \subseteq C$ and $B \subseteq D$. Show that $(I_{P,3}, \leq)$ is an ω-cpo.

05/11/04

2.7 Proposition ([Llo88]) T_P is ω-continuous for definite P.

Proof: We first show that T_P is monotonic. Let $I, K \in I_P$ with $I \subseteq K$, and let $A \in T_P(I)$. Then there exists a clause $A \leftarrow \mathsf{body}$ in $\mathsf{ground}(P)$ with $\mathsf{body} \subseteq I \subseteq K$, and hence $A \in T_P(K)$, as required.

Now let $\mathcal{I} = \{I_n \mid n \in \mathbb{N}\}$ be an ω-chain of interpretations, and let $I = \sup \mathcal{I} = \bigcup \mathcal{I}$. Since the order under consideration is set-inclusion and T_P is monotonic, we immediately have that $T_P(\mathcal{I})$ is an ω-chain.

We have $T_P(K) \subseteq T_P(I)$ for each $K \in \mathcal{I}$, and hence $\bigcup T_P(\mathcal{I}) \subseteq T_P(I)$.

It remains to show that $T_P(I) \subseteq \bigcup T_P(\mathcal{I})$. So suppose that A belongs to $T_P(I)$. Then there is a (definite) clause C of the form $A \leftarrow A_1, \ldots, A_n$ in $\mathsf{ground}(P)$ satisfying

$A_1, \ldots, A_n \in I$. Therefore, there exist $I_{k_1}, \ldots, I_{k_n}$ in $\mathcal{I}$ with $A_i \in I_{k_i}$ for $i = 1, \ldots, n$. Since $\mathcal{I}$ is directed, there is $I_k \in \mathcal{I}$ with $I_{k_i} \subseteq I_k$ for $i = 1, \ldots, n$. Hence, the body of C is true in I_k, and we obtain that $A \in T_P(I_k)$ and, hence, that $A \in \bigcup T_P(\mathcal{I})$, as required. ■

Exercise 8
Show that T_P is not in general ω-continuous for normal P.

Exercise 9
Give a non-definite program for which T_P is ω-continuous.

2.8 Theorem ([Llo88]) T_P has a least model by the Scott fixed-point theorem, if P is definite.

Proof: By the Scott fixed-point theorem (Theorem 2.9 below), T_P has a least fixed point which is also its least pre-fixed point. Since the latter are exactly the models of P, the proof is complete. ■

Kowalski-van Emden showed in [vEK76], that the ground atoms obtainable from a definite program P by SLD-resolution are exactly those contained in the least fixed point of T_P.

2.9 Theorem [SHLG94, AJ94, GHK+03] Let $(X, \leq)$ be an ω-cpo and $f : X \to X$ be an ω-continuous function. Then f has a least fixed point which is also its least pre-fixed point.

Proof: Define $f \uparrow 0 = \bot$ and recursively $f \uparrow (n+1) = f(f \uparrow n)$.

The sequence $(f \uparrow n)_{n \in \mathbb{N}}$ is an ω-chain. It therefore has a supremum $f \uparrow \omega = x$, say. By ω-continuity, we have $x = f \uparrow \omega = \sup\{f \uparrow (n+1) \mid n \in \mathbb{N}\} = f(\sup\{f \uparrow n \mid n \in \mathbb{N}\}) = f(x)$, and so x is a fixed point of f. If y is a pre-fixed point of f, then $\bot \leq y$ and, by monotonicity of f, we obtain $f \uparrow 1 = f(\bot) \leq f(y) \leq y$. Inductively, it follows that $f \uparrow n \leq y$ for all $n \in \mathbb{N}$, and hence $x = f \uparrow \omega \leq y$. So x is the least pre-fixed point of f, and hence also its least fixed point. ■

Exercise 10
For the following program P, describe $T_P \uparrow n$ for all $n \in \mathbb{N}$ and give its least model.

$$p(0) \leftarrow$$
$$p(s(X)) \leftarrow p(X)$$

Exercise 11
Give a naive Prolog-implementation of list membership, and describe the least model of the program.

Exercise 12
Show that fact *is the least fixed point of* Fact*, by applying Theorem 2.9.*

Ordinals

2.10 Definition A poset X is *well-ordered* (or a *well-ordering*), if each subset of X has a least element.

Exercise 13
(a) Show that the natural numbers are well-ordered (under the usual order).

(b) Show that the integers are not well-orderd (under the usual order).

(c) Is I_P well-ordered under subset-inclusion?

Exercise 14
Show the following.

(a) Every well-ordering is linearly orderd.

(b) No well-ordering contains an infinite strictly descending sequence.

Given two well-orderings $(X, \leq_X)$ and $(Y, \leq_Y)$, we call $f : X \to Y$ *monotonic* if $a \leq_X b$ implies $f(a) \leq_Y f(b)$ for all $a, b \in X$. If f is also injective, it is called an *embedding* of X into Y. If f is also bijective, it is called an *order isomorphism* between X and Y. The two well-orderings X and Y are then called *isomorphic*.

Exercise 15
Prove that your answers are correct.

(a) Give an order-isomorphism between $\mathbb{N}$ and $2\mathbb{N}$, where $2\mathbb{N}$ ist the set of all even natural numbers.

(b) Give an embedding of $\mathbb{N}$ into I_P.

(c) Give an embedding of I_P into $I_{P,3}$.

For two well-orderings $(X, \leq_X)$ and $(Y, \leq_Y)$, we write $X \leq Y$ if X is isomorphic to an *initial segment* of Y, i.e. if X is isomorphic to $\{y \mid y \leq_Y x\}$ for some $x \in Y$. We write $X < Y$ if X is isomorphic to a *proper* initial segment of Y.

2.11 Theorem For any two well-orderings X and Y, exactly one of the following holds.

(i) $X < Y$.

(ii) $X > Y$.

(iii) X and Y are isomorphic.

Proof: We first show the following.
(1) No well-ordering $(Z, \leq_Z)$ is isomorphic to a proper initial segment of itself.
In order to see this, assume $f : I \to Z$ is an isomorphism and I is a proper initial segment of Z. Then we cannot have $f(x) = x$ for all $x \in I$ because then f would not be surjective. Let x_0 be the least element of I such that $f(x) \neq x$.

We cannot have $f(x_0) <_Z x_0$ since then $f(f(x_0)) = f(x_0)$, so f were not injective. Hence $x_0 <_Z f(x_0)$. Now let $x_1 \in I$ such that $f(x_1) = x_0$. Then $x_1 \neq x_0$ (because $f(x_0) \neq x_0$). If $x_1 <_Z x_0$ then $x_0 = f(x_1) = x_1 <_Z x_0$, which is impossible. If $x_0 <_Z x_1$, then $f(x_1) = x_0 <_Z f(x_0)$ which contradicts f being monotonic. Hence statement (1) holds.

We will also need the following statement.

(2) If the well-orderings $(W, \leq_W)$ and $(Z, \leq_Z)$ are isomorphic, then the isomorphism is unique.

In order to see this, suppose $f, g : W \to Z$ are isomorphisms. We show $f = g$. Assume this is not the case, and let $w_0 \in W$ be the $\leq_W$-least w such that $f(w) \neq g(w)$, say, $f(w_0) <_Z g(w_0)$. Let w_1 be such that $g(w_1) = f(w_0)$. Then $w_1 \neq w_0$. If $w_1 <_W w_0$ then by minimality of w_0 we obtain $g(w_1) = f(w_1) <_W f(w_0) = g(w_1)$, which is impossible. If $w_0 <_W w_1$ then $f(w_0) <_Z g(w_0) <_Z g(w_1) = f(w_0)$, which is also impossible. So statement (2) holds.

We now return to the proof of the theorem.

For $x \in X$ and $y \in Y$, define $R(x, y)$ iff the initial segments $\{w \in X \mid w \leq_X x\}$ and $\{v \in Y \mid v \leq_Y y\}$ are isomorphic. First note that $R(x, y_1)$ and $R(x, y_2)$ implies $y_1 = y_2$ by statement (1). So R is a partial function. By symmetry, it is also injective.

We next show that $\mathsf{dom}(R)$ is an initial segment of X. Suppose $x_2 \in \mathsf{dom}(R)$, say $R(x_2, y_2)$, and let $x_1 <_X x_2$. Let f be an isomorphism between the initial segments corresponding to x_2 and y_2. But then the initial segments corresponding to x_1 and $f(x_1)$ are also isomorphic, so $R(x_1, f(x_1))$, hence $x_1 \in \mathsf{dom}(R)$. We have also shown that R is order-preserving.

A similar argument shows that the range of R is an initial segment of Y. Hence R is an isomorphism from an initial segment of X, say I, to an initial segment of Y, say J.

Now consider the following cases. If $I = X$, but $J \neq Y$, then case (i) holds. If $I \neq X$ but $J = Y$, then case (ii) holds. If $I = X$ and $J = Y$, then case (iii) holds. Suppose finally that $I \neq X$ and $J \neq Y$. If $I = \{w \mid w \leq_X x\}$ and $J = \{v \mid v \leq_Y y\}$, then by definition $R(x, y)$. Thus $x \in \mathsf{dom}(R) = I$, a contradiction.

Because of (2), only one of (i), (ii), (iii) holds. ■

Exercise 16
Let X be a linearly ordered poset which is not well-ordered. Show that it contains an infinite strictly descending sequence.

12/11/04

2.12 Definition An *ordinal*[1] is an equivalence class of a well-ordering under isomorphism.

2.13 Proposition Every set of ordinals is itself a well-ordering under $\leq$.

Proof: If $\alpha \leq \beta$ are ordinals, then we can assume without loss of generality, that $\alpha \subseteq \beta$. We will do this in the following.

[1] We're treating the notion of *ordinal* very lightly here, meaning that there are some set-theoretic subtleties which shall not concern us here.

Let X be a set of ordinals which is not well-ordered, i.e. by Exercise 16 it contains an infinite descending sequence $\alpha_0 > \alpha_1 > \alpha_2 > \ldots$ of ordinals. Then for all $i \in \mathbb{N}$ there exists $a_i \in (\alpha_i \setminus \alpha_{i+1})$. But then $\{a_i \mid i \in \mathbb{N}\} \subseteq \alpha_0$ is a subset of α_0 without a least element, which is impossible. ■

It is common practice, to identify any ordinal α with the set of all ordinals β such that $\beta < \alpha$. We will follow this practice in the following. In particular, when we speak of a mapping $f : X \to \alpha$, where α is an ordinal, we mean in fact a mapping $f : X \to \{\beta \mid \beta < \alpha\}$.

Ordinals fall into two classes. A *successor ordinal* is an ordinal α such that there is a greatest ordinal β with $\beta < \alpha$. In this case, α is called the *successor* of β and can be denoted by $\beta + 1$. Any ordinal which is not a successor ordinal is called a *limit ordinal*.

Exercise 17
Show that any ordinal has a successor.

2.14 Example Natural numbers as ordinals. ω as an ordinal. Etc.

The principle of transfinite induction: Suppose we want to prove that a property Q holds for all members of an ordinal α. Then it suffices to show that the following hold.

(i) $Q(0)$ holds.

(ii) For any ordinal β, if $Q(\alpha)$ holds for all ordinals $\alpha < \beta$ then $Q(\beta)$ holds as well.

When applying the proof principle of transfinite induction, it is usually convenient to split part (ii) into two cases, namely whether β is a limit or a successor ordinal.

Mention of the well-ordering principle and some of its implications.

Exercise 18
Prove that the principle of transfinite induction is correct.

2.15 Definition A *level mapping* for a program P is a mapping $l : B_P \to \alpha$, where α is some ordinal.

2.16 Theorem ([HW02, HW05]) The least model $T_P \uparrow \omega$ for a definite program P is the unique model M for P satisfying the following condition: there exists a mapping $l : B_P \to \alpha$, for some ordinal α, such that for each $A \in M$ there is a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ with $M(\mathtt{body}) = \mathbf{t}$ and $l(B) < l(A)$ for each $B \in \mathtt{body}$.

Proof: To start with, take M to be the least model $T_P \uparrow \omega$, choose $\alpha = \omega$, and define $l : B_P \to \alpha$ by setting $l(A) = \mathsf{min}\{n \mid A \in T_P \uparrow (n+1)\}$, if $A \in M$, and by setting $l(A) = 0$, if $A \notin M$. From the fact that $\emptyset \subseteq T_P \uparrow 1 \subseteq \ldots \subseteq T_P \uparrow n \subseteq \ldots \subseteq T_P \uparrow \omega = \bigcup_m T_P \uparrow m$, for each n, we see that l is well-defined and that the least model $T_P \uparrow \omega$ for P has the desired properties.

Conversely, if M is a model for P which satisfies the given condition for some mapping $l : B_P \to \alpha$, then it is easy to show, by induction on $l(A)$, that $A \in M$

implies $A \in T_P \uparrow (l(A)+1)$. This yields that $M \subseteq T_P \uparrow \omega$, and hence that $M = T_P \uparrow \omega$ by minimality of the model $T_P \uparrow \omega$. ■

Exercise 19
Show that "ω" is replacable by "$\mathbb{N}$" in Theorem 2.16.

2.2 Stable model semantics for normal programs

2.17 Definition ([ABW88]) An interpretation I for a program P is called *supported* if for each $A \in I$ there is a clause $A \leftarrow \texttt{body}$ in $\texttt{ground}(P)$ with $I(\texttt{body}) = \mathbf{t}$.

2.18 Example Tweety1 from Program 2.3 has supported model M, where $M = \{\texttt{penguin(tweety)}, \texttt{bird(bob)}, \texttt{bird(tweety)}, \texttt{flies(bob)}\}$, as is easily verified. Careful inspection will also convince the reader that M is the unique supported model for Tweety1.

2.19 Proposition The supported interpretations for a program P are exactly the post-fixed points of T_P. The supported models for P are exactly the fixed points of T_P.

Proof: Let I be a supported interpretation for P and suppose that $A \in I$. Then there is a clause $A \leftarrow \texttt{body}$ in $\texttt{ground}(P)$ with $I(\texttt{body}) = \mathbf{t}$. But then $A \in T_P(I)$, showing that $I \subseteq T_P(I)$, as required to see that I is a post-fixed point of T_P.

Conversely, assume that $I \subseteq T_P(I)$ is a post-fixed point of T_P and let $A \in I$. Then $A \in T_P(I)$. Therefore, there exists a clause $A \leftarrow \texttt{body}$ in $\texttt{ground}(P)$ with $I(\texttt{body}) = \mathbf{t}$, showing that I is a supported model for P.

Finally, using Proposition 2.5, we obtain that an interpretation for P is a supported model for P if and only if it is both a pre-fixed point and a post-fixed point of T_P, that is, if and only if it is a fixed point of T_P. ■

Exercise 20
Consider the program consisting of the single clause $p(X) \leftarrow \neg p(s(X))$. Give all supported models of the program.

Discussion of relation between supported models and the Clark completion (as in FLCP).

One of the drawbacks of the supported model semantics is that definite programs may have more than one supported model.

2.20 Program Let P be the program consisting of the single clause $p \leftarrow p$. Then both $\emptyset$ and $\{p\}$ are supported models for P.

Why not use least supported models? It may not exist! Consider program with $p \leftarrow \neg q$ and $q \leftarrow \neg p$.

Why not use minimal supported models? It's unintuitive! Consider program with $p \leftarrow p$ and $q \leftarrow \neg p$.

This unsatisfactory situation is resolved by the introduction of *stable models*.

2.21 Definition ([Fag94]) An interpretation I for a program P is called *well-supported* if there exists a mapping $l : B_P \to \alpha$, for some ordinal α, with the property that, for any $A \in I$, there is a clause C in $\mathsf{ground}(P)$ of the form $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_k$ such that the body of C is true in I and $l(A_i) < l(A)$ for all $i = 1, \ldots, n$. A well-supported model for P is called a *stable model* for P.

2.22 Theorem The following statements hold.

(i) Every stable model is supported but not vice-versa.

(ii) Every stable model is a minimal model but not vice-versa.

(iii) Every definite program has a unique stable model which is its least model.

Proof: (i) Supportedness of stable models follows immediately from the definition. The supported model $\{p\}$ for Program 2.20 is not stable.

(ii) Let P be a program, let M be a stable model for P, and let l be a level mapping with respect to which M is well-supported. Assume that K is a model for P with $K \subset M$. Then there exists $A \in M \setminus K$, and we can assume without loss of generality that A is also such that $l(A)$ is minimal. By the well-supportedness of M, there is a clause C of the form $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_k$ in $\mathsf{ground}(P)$ such that for all $i = 1, \ldots, n$ and $j = 1, \ldots, k$ we have $A_i \in M$, $l(A) > l(A_i)$ and $B_j \notin M$. Since $K \subset M$, we obtain, for all $j = 1, \ldots k$, that $B_j \notin K$, and by minimality of $l(A)$ we obtain $A_i \in K$ for all $i = 1, \ldots, n$. Since K is a model for P and the body of C is true with respect to K, we conclude that $A \in K$ which contradicts the assumption that $A \in M \setminus K$. Hence, M must be a minimal model.

In the opposite direction, Program 2.23 below has $\{p\}$ as its only model, and hence this is a minimal model. It is clearly not a stable model, however.

(iii) By Theorem 2.16, we see that the least model is indeed stable. Uniqueness follows from (ii) and Theorem 2.8 (iii). ■

There are programs with unique supported models which are not stable.

2.23 Program The program P consisting of the two clauses

$$p \leftarrow p$$
$$p \leftarrow \neg p$$

has unique supported model $\{p\}$, and this model is not stable.

Exercise 21
(a) Show that a unique stable model is always a least model.

(b) If a program has a least model, is this model then always stable?

10/12/04

A characterization of stable models as fixed points of an operator can be given, and we proceed with this next.

2.24 Definition ([GL88, GL91]) Let P be a normal logic program and let $I \in I_P$. The *Gelfond-Lifschitz transform* P/I of P is the set of all clauses $A \leftarrow A_1, \ldots, A_n$ for which there exists a clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_k$ in $\mathsf{ground}(P)$ with $B_1, \ldots, B_k \notin I$.

We note that the Gelfond-Lifschitz transform P/I of a program P is always definite (as a set of ground clauses) and therefore has a least model $T_{P/I} \uparrow \omega$ by Theorem 2.8. The operator $\mathsf{GL}_P : I \mapsto T_{P/I} \uparrow \omega = \bigcup_{n \in \mathbb{N}} T^n_{P/I}(\emptyset)$ is called the *Gelfond-Lifschitz operator*[2] associated with P.

A function f on a partially ordered set is called *antitonic* if $x \leq y$ implies $f(y) \leq f(x)$ for all x, y.

2.25 Theorem The following hold.

(i) The Gelfond-Lifschitz operator is antitonic, and in general is not monotonic.

(ii) An interpretation I is a stable model for a program P if and only if it is a fixed point of GL_P, that is, if and only if it satisfies $\mathsf{GL}_P(I) = I$.

Proof: Let P be a program and let I, K be interpretations for P with $I \subseteq K$. Then $P/K \subseteq P/I$, and it is a straightforward proof by induction to show that $T_{P/K} \uparrow n \subseteq T_{P/I} \uparrow n$ for all $n \in \mathbb{N}$. Hence, $\mathsf{GL}_P(K) = T_{P/K} \uparrow \omega \subseteq T_{P/I} \uparrow \omega = \mathsf{GL}_P(I)$, which shows that GL_P is antitonic. To see that it is not generally monotonic, take P to be Program 2.23. On setting $I = \emptyset$, we obtain that P/I is the definite program consisting of the clauses $p \leftarrow p$ and $p \leftarrow$, and $\mathsf{GL}_P(I) = \{p\}$; on setting $I = \{p\}$, we obtain that P/I consists of the single clause $p \leftarrow p$, and $\mathsf{GL}_P(I) = \emptyset$. This establishes (i).

For (ii), we start by supposing that $\mathsf{GL}_P(I) = T_{P/I} \uparrow \omega = I$. Then I is the least model for P/I, hence is also a model for P, and, by Proposition 2.16, is well-supported with respect to any level mapping l satisfying $l(A) = \mathsf{min}\{n \mid A \in T_{P/I} \uparrow (n+1)\}$ for each $A \in I$. Conversely, let I be a stable, hence well-supported, model for P. Then, for every $A \in I$, there is a clause C in $\mathsf{ground}(P)$ of the form $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_k$ such that the body of C is true in I and satisfying $l(A_i) < l(A)$ for $i = 1, \ldots, n$. But then, for every $A \in I$, there is a clause $A \leftarrow A_1, \ldots, A_n$ in P/I whose body is true in I and such that $l(A_i) < l(A)$ for $i = 1, \ldots, n$. By Proposition 2.16, this means that I is the least model for P/I, that is, $I = T_{P/I} \uparrow \omega = \mathsf{GL}(I)$. ■

Exercise 22
Complete the proof of Theorem 2.25 by spelling out the missing induction argument.

We will now give some examples.

[2]The Gelfond-Lifschitz operator is named after the authors of [GL88], who introduced it in defining the stable model semantics.

2.26 Example Consider again Program Tweety1 and its supported model M as given in Example 2.18. We show that M is stable. The program Tweety1/M is as follows.

$$\begin{aligned}
\texttt{penguin}(\texttt{tweety}) &\leftarrow \\
\texttt{bird}(\texttt{bob}) &\leftarrow \\
\texttt{bird}(\texttt{tweety}) &\leftarrow \texttt{penguin}(\texttt{tweety}) \\
\texttt{bird}(\texttt{bob}) &\leftarrow \texttt{penguin}(\texttt{bob}) \\
\texttt{flies}(\texttt{bob}) &\leftarrow \texttt{bird}(\texttt{bob})
\end{aligned}$$

The least model for this program turns out to be M, which shows that M is supported.

A strange feature of the supported model semantics is that the addition of clauses of the form $p \leftarrow p$ may change the semantics.

2.27 Program (Tweety2) Consider the following program Tweety2.

$$\begin{aligned}
\texttt{penguin}(\texttt{tweety}) &\leftarrow \\
\texttt{bird}(\texttt{bob}) &\leftarrow \\
\texttt{bird}(X) &\leftarrow \texttt{penguin}(X) \\
\texttt{flies}(X) &\leftarrow \texttt{bird}(X), \neg\texttt{penguin}(X) \\
\texttt{penguin}(\texttt{bob}) &\leftarrow \texttt{penguin}(\texttt{bob})
\end{aligned}$$

Tweety2 results from Tweety1 by adding the clause $\texttt{penguin}(\texttt{bob}) \leftarrow \texttt{penguin}(\texttt{bob})$. Intuitively, this addition should not change the semantics of the program. However, in addition to the supported model M from Example 2.18, Tweety2 also has

$$M' = \{\texttt{penguin}(\texttt{tweety}), \texttt{penguin}(\texttt{bob}), \texttt{bird}(\texttt{tweety}), \texttt{bird}(\texttt{bob})\}$$

as supported model. Whilst M is also a stable model for Tweety2, M' is not. This can be seen by inspecting the program Tweety2/M', as follows, which has $\{\texttt{penguin}(\texttt{tweety}), \texttt{bird}(\texttt{bob}), \texttt{bird}(\texttt{tweety})\} \neq M'$ as its least model:

$$\begin{aligned}
\texttt{penguin}(\texttt{tweety}) &\leftarrow \\
\texttt{bird}(\texttt{bob}) &\leftarrow \\
\texttt{bird}(\texttt{tweety}) &\leftarrow \texttt{penguin}(\texttt{tweety}) \\
\texttt{bird}(\texttt{bob}) &\leftarrow \texttt{penguin}(\texttt{bob}) \\
\texttt{penguin}(\texttt{bob}) &\leftarrow \texttt{penguin}(\texttt{bob})
\end{aligned}$$

We can also use the stable semantics for modelling *choice*.

2.28 Program (Tweety3) Consider the program Tweety3, as follows.

$$\begin{aligned}
\texttt{eagle}(\texttt{tweety}) &\leftarrow \neg\texttt{penguin}(\texttt{tweety}) \\
\texttt{penguin}(\texttt{tweety}) &\leftarrow \neg\texttt{eagle}(\texttt{tweety}) \\
\texttt{bird}(X) &\leftarrow \texttt{eagle}(X) \\
\texttt{bird}(X) &\leftarrow \texttt{penguin}(X) \\
\texttt{flies}(X) &\leftarrow \texttt{bird}(X), \neg\texttt{penguin}(X)
\end{aligned}$$

This program has the two stable models

$$\{\texttt{eagle}(\texttt{tweety}), \texttt{bird}(\texttt{tweety}), \texttt{flies}(\texttt{tweety})\}$$

and

$$\{\texttt{penguin}(\texttt{tweety}), \texttt{bird}(\texttt{tweety})\}.$$

Exercise 23
Compute $\mathsf{GL}^n_{Tweety1}(\emptyset)$ *and* $\mathsf{GL}^n_{Tweety3}(\emptyset)$ *for all* $n \in \mathbb{N}$.

Exercise 24
Obtain and install the dlv *System (http://www.dbai.tuwien.ac.at/proj/dlv/) Use it to compute the stable models for the programs Tweety1, Tweety2, and Tweety3.*

Exercise 25
Extend Tweety1 by adding ostriches and bats and rules stating "flying things normally have feathers" and "birds normally are slow walkers". Encode exceptions to these rules using negation and test the program with dlv.

Exercise 26
Encode Tweety1 using Reiter's Default Logic (see ICL). Can you guess how the stable model semantics and Default Logic relate? (You can also consult [GL91] for an answer.)

2.29 Definition ([DK89]) A *quasi-interpretation*[3] is a set of clauses of the form $A \leftarrow \neg B_1, \ldots, \neg B_m$, where A and B_i are ground atoms for all $i = 1, \ldots, m$. Given a normal logic program P and a quasi-interpretation Q, we define $T'_P(Q)$ to be the quasi-interpretation consisting of the set of all clauses

$$A \leftarrow \texttt{body}_1, \ldots, \texttt{body}_n, \neg B_1, \ldots, \neg B_m$$

for which there exists a clause

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

in $\mathsf{ground}(P)$ and clauses $A_i \leftarrow \texttt{body}_i$ in Q for all $i = 1, \ldots, n$. We explicitly allow the cases $n = 0$ or $m = 0$ in this definition.

Exercise 27
Show that the set of all quasi-interpretations is an ω*-cpo with respect to set-inclusion.*

2.30 Proposition For normal programs P, the operator T'_P is ω-continuous on the set of all quasi-interpretations.

Proof: We show first that T'_P is monotonic. So let $Q \subseteq R$ be quasi-interpretations and let $A \leftarrow \texttt{body}$ be in $T'_P(Q)$. If $A \leftarrow \texttt{body}$ results from the unfolding of some clause $A \leftarrow \texttt{body}_0$ in P with some clauses $B_i \leftarrow \texttt{body}_i$ in Q, then $B_i \leftarrow \texttt{body}_i$ is contained

[3]This notion is due to [DK89]. We stick to the old terminology, although quasi-interpretations should really be thought of as, and indeed are, programs with negative body literals only.

in R for all i by assumption, and by the existence of the clause $A \leftarrow \mathtt{body}_0$ in P we obtain $A \leftarrow \mathtt{body}$ in $T'_P(R)$ by unfolding. If $A \leftarrow \mathtt{body} \in T'_P(Q)$ does not result from some unfolding, then it is already contained in P, and hence in $T'_P(R)$.

Now let $\mathcal{Q} = \{Q_n \mid n \in \mathbb{N}\}$ be an increasing sequence of quasi-interpretations, and let $Q = \bigsqcup \mathcal{Q} = \bigcup \mathcal{Q}$. Since the order under consideration is set-inclusion and T'_P is monotonic, we immediately have that $T'_P(\mathcal{Q})$ is an increasing sequence in n. It remains to show that $T'_P(Q) \subseteq \bigcup T'_P(\mathcal{Q})$ (the other inclusion always holds for monotonic operators — see the proof of Proposition 2.7). So suppose that $A \leftarrow \mathtt{body}$ belongs to $T'_P(Q)$. If $A \leftarrow \mathtt{body}$ does not result from an unfolding, then it is already contained in P, hence also in $T'_P(\mathcal{Q})$. Otherwise, $A \leftarrow \mathtt{body}$ results from the unfolding of some $A \leftarrow \mathtt{body}_0$ in P with some $B_i \leftarrow \mathtt{body}_i$ in Q. But then there is n such that all $B_i \leftarrow \mathtt{body}_i$ are contained in Q_n, hence $A \leftarrow \mathtt{body}$ is contained in $T'_P(Q_n) \subseteq T'_P(\mathcal{Q})$ as required. ■

So we can define the *fixpoint completion* $\mathsf{fix}(P)$ of P by $\mathsf{fix}(P) = T'_P \uparrow \omega$, i.e. $\mathsf{fix}(P)$ is the least fixed point of the operator T'_P.

2.31 Theorem ([Wen02a]) For any normal program P and (two-valued) interpretation I, we have

$$\mathsf{GL}_P(I) = T_{\mathsf{fix}(P)}(I).$$

Proof: We show first that for every $A \in \mathsf{GL}_P(I)$ there exists a clause in $\mathsf{fix}(P)$ with head A whose body is true in I, which implies $A \in T_{\mathsf{fix}(P)}(I)$. We show this by induction on the powers of $T_{P/I}$; recall that $\mathsf{GL}_P(I) = T_{P/I} \uparrow \omega$.

For the base case $T_{P/I} \uparrow 0 = \emptyset$ there is nothing to show.

So assume now that for all $A \in T_{P/I} \uparrow n$ there exists a clause in $\mathsf{fix}(P)$ with head A, whose body is true in I. For $A \in T_{P/I} \uparrow (n+1)$ there exists a clause $A \leftarrow A_1, \ldots, A_n$ in P/I such that $A_1, \ldots, A_n \in T_{P/I} \uparrow n$, hence by construction of P/I there is a clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ with $B_1, \ldots, B_m \notin I$. By induction hypothesis we obain that for each $i = 1, \ldots, n$ there exists a clause $A_i \leftarrow \mathtt{body}_i$ in $\mathsf{fix}(P)$ with $I \models \mathtt{body}_i$, hence $A_i \in T_{\mathsf{fix}(P)}(I)$. So by definition of T'_P the clause $A \leftarrow \mathtt{body}_1, \ldots \mathtt{body}_n, \neg B_1, \ldots, \neg B_m$ is contained in $\mathsf{fix}(P)$. From $I \models \mathtt{body}_i$ and $B_1, \ldots, B_m \notin I$ we obtain $A \in T_{\mathsf{fix}(P)}(I)$ as desired.

This closes the induction argument and we obtain $\mathsf{GL}_P(I) \subseteq T_{\mathsf{fix}(P)}(I)$.

Now conversly, assume that $A \in T_{\mathsf{fix}(P)}(I)$. We show that $A \in \mathsf{GL}_P(I)$ by proving inductively on k that $T_{T'_P \Uparrow k}(I) \subseteq \mathsf{GL}_P(I)$ for all $k \in \mathbb{N}$.

For the base case, we have $T_{T'_P \Uparrow 0}(I) = \emptyset$ so there is nothing to show.

So assume now that $T_{T'_P \Uparrow k}(I) \subseteq \mathsf{GL}_P(I)$, and let $A \in T_{T'_P \Uparrow (k+1)}(I) \setminus T_{T'_P \Uparrow k}(I)$. Then there exists a clause $A \leftarrow \mathtt{body}_1, \ldots, \mathtt{body}_n, \neg B_1, \ldots, \neg B_m$ in $T'_P \uparrow (k+1)$ whose body is true in I. Thus $B_1, \ldots, B_m \notin I$ and for each $i = 1, \ldots, n$ there exists a clause $A_i \leftarrow \mathtt{body}_i$ in $T'_P \uparrow k$ with $\mathtt{body}_i$ true in I. So $A_i \in T_{T'_P \Uparrow k}(I) \subseteq \mathsf{GL}_P(I)$. Furthermore, by definition of T'_P there exists a clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$, and since $B_1, \ldots, B_m \notin I$ we obtain $A \leftarrow A_1, \ldots, A_n \in P/I$. Since we know that $A_1, \ldots, A_n \in \mathsf{GL}_P(I)$ we obtain $A \in \mathsf{GL}_P(I)$, and hence $T_{T'_P \Uparrow (k+1)}(I) \subseteq \mathsf{GL}_P(I)$. This closes the induction argument and we obtain $T_{\mathsf{fix}(P)}(I) \subseteq \mathsf{GL}_P(I)$. ■

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \leftarrow q$
t	**t**	**f**	**t**	**t**	**t**
t	**f**	**f**	**f**	**t**	**t**
t	**u**	**f**	**u**	**t**	**t**
f	**t**	**t**	**f**	**t**	**f**
f	**f**	**t**	**f**	**f**	**t**
f	**u**	**t**	**f**	**u**	**f**
u	**t**	**u**	**u**	**t**	**f**
u	**f**	**u**	**f**	**u**	**t**
u	**u**	**u**	**u**	**u**	**t**

Table 1: Truth tables for Kleene's strong three-valued logic.

Exercise 28
Which are the main results on semantics of logic programs and their relations which we have obtained so far? Collect them all on a single sheet of paper.

3 Three-valued semantics for logic programs

3.1 Fitting semantics

The stable model semantics is more satisfactory than the supported model semantics in that each definite program has a unique stable model which coincides with its least model. However, for normal logic programs in general, uniqueness cannot be guaranteed, as can be seen from the program $\{p \leftarrow \neg q, \quad q \leftarrow \neg p\}$ which has two stable models $\{p\}$ and $\{q\}$. It is desirable to be able to associate with each program a unique model in some natual way. One way of doing this is by means of three-valued logic, and we discuss this next. The resulting Kripke-Kleene semantics, herein called the Fitting semantics, is due to Fitting [Fit85].

We will work in a three-valued logic with the set of truth values $T_3 = \{\mathbf{t}, \mathbf{u}, \mathbf{f}\}$ for *true*, *undefined* and *false*. We will need the truth tables for negation, disjunction, conjunction, and implication — they are given in Table 3.1.

A three-valued interpretation $I : B_P \to T_3$ can be identified with an element of $I_{P,3}$ as in Exercise 7, where $(A, B) \in I_{P,3}$ is understood as mapping all elements of A to $\mathbf{t}$, all elements of B to $\mathbf{f}$, and all others to $\mathbf{u}$. Here, we will use an alternative notation using *signed sets*: Instead of (A, B) we write $A \cup \{\neg p \mid p \in B\}$. The ordering on $I_{P,3}$ introduced in Exercise 7 will be denoted by $\leq_k$ (or sometimes by $\sqsubseteq_k$) and called the *knowledge ordering* on $I_{P,3}$. It is the same as subset inclusion on signed sets.

We will also need another ordering on $I_{P,3}$ called the *truth ordering* $\leq_t$ (or $\sqsubseteq_t$): We set $(A, B) \leq_t (C, D)$ if $A \subseteq C$ and $D \subseteq B$.

We also call members of I_P *two-valued* or *total interpretations* and members of $I_{P,3}$ *three-valued* or *partial interpretations*.

Exercise 29
In what sense is I_P a subset of $I_{P,3}$?

For convenience, we often write I_P instead of $I_{P,3}$ in this chapter. We will do this only if it does not cause any confusion. 07/01/05

Given a normal logic program P, we define the following operators[4] T'_P and F_P on $I_P = I_{P,3}$. First, $T'_P(I)$ is the set of all $A \in B_P$ for which there is a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ with $\mathtt{body}$ true in I with respect to Kleene's strong three-valued logic. Second, $F_P(I)$ is the set of all $A \in B_P$ such that for all clauses $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ we have that $\mathtt{body}$ is false in I with respect to Kleene's strong three-valued logic. Finally, we define

$$\Phi_P(I) = T'_P(I) \cup \neg F_P(I)$$

for all $I \in I_P$. We will call the operator Φ_P the *Fitting operator* for P.

The development of the operator Φ_P somewhat parallels that of T_P except that there are two orderings involved, and the following result is analogous to Proposition 2.5.

3.1 Proposition Let P be a normal logic program. Then the three-valued models for P are exactly the pre-fixed points of Φ_P in the truth ordering $\sqsubseteq_t$.

Proof: Suppose that M is a three-valued interpretation for P satisfying $\Phi_P(M) \sqsubseteq_t M$, and let $A \in B_P$ be arbitrary. Suppose that $\Phi_P(M)(A) = \mathbf{u}$. Then we must have $M(A)$ equal to $\mathbf{u}$ or to $\mathbf{t}$. Since no clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ can have $M(\mathtt{body}) = \mathbf{t}$, otherwise $\Phi_P(M)(A)$ would be equal to $\mathbf{t}$, we must have $M(\mathtt{body})$ equal to $\mathbf{u}$ or to $\mathbf{f}$ for each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$. But then we get that $A \leftarrow \mathtt{body}$ is true in M. The other possible values for $\Phi_P(M)$ are handled similarly, and so M is a model for P.

The converse is also handled similarly, and we omit the details. ■

Exercise 30
Spell out the missing details of the converse of the previous proof.

3.2 Program Consider the following program P.

$$p \leftarrow \neg q$$
$$p \leftarrow \neg r$$
$$q \leftarrow q$$
$$r \leftarrow r$$

Define M as follows: $M(p) = \mathbf{f}$, $M(q) = \mathbf{u}$, $M(r) = \mathbf{t}$. Then M is a three-valued interpretation for P satisfying $\Phi_P(M) \sqsubseteq_k M$, and yet M is not a model for P.

Exercise 31
Find a program P with a model $I \in I_{P,3}$ which is not a $\leq_k$-pre-fixed point of Φ_P.

[4]The notation T'_P was already used in the context of Definition 2.29. From here on, we will mean by T'_P always the new notion.

Therefore, neither implication of Proposition 3.1 holds in the case of the knowledge ordering.

The following fact about Φ_P is fundamental.

3.3 Proposition ([Fit85]) Let P be a program. Then Φ_P is monotonic on $I_{P,3}$ in the knowledge ordering $\sqsubseteq_k$.

Proof: Let $I, K \in I_{P,3}$ with $I \subseteq K$. We show $\Phi_P(I) \subseteq \Phi_P(K)$. Let $A \in \Phi_P(I)$ be an atom. Then $A \in T'_P(I)$. Therefore, there is a ground clause $A \leftarrow \texttt{body}$ such that `body` is true in I. From Table 3.1, each literal in `body` must be true and therefore must belong to I. Hence, each literal in `body` belongs to K since $I \subseteq K$ and is therefore true in K. Hence `body` is also true in K and we obtain that $A \in T'_P(K) \subseteq \Phi_P(K)$. Now let $\neg A \in \Phi_P(I)$ be a negated atom. Then $A \in F_P(I)$, and so, for all ground clauses $A \leftarrow \texttt{body}$, we have that `body` is false in I. So, given such a clause, from Table 3.1 we see that at least one literal L_j, say, in `body` is false. Hence, we have $\neg L_j \in I$. But $I \subseteq K$ and hence $\neg L_j \in K$. Therefore, L_j is also false in K and consequently `body` is false in K, and we obtain $A \in F_P(K)$ and hence $\neg A \in \Phi_P(K)$, as required. ■

Exercise 32
Show that Φ_P is in general not monotonic with respect to the truth ordering.

3.4 Theorem (Tarski) Let $(D, \sqsubseteq)$ denote am ω-cpo, let $f : D \to D$ be monotonic and let $x \in D$ be such that $x \sqsubseteq f(x)$. Then f has a least fixed point a above x, which is also the least pre-fixed point of f above x, and there exists a least ordinal α such that $a = f^\alpha(x)$. In particular, f has a least fixed point a which is also its least pre-fixed point.

Proof: We just sketch the proof.

Let γ be an ordinal whose cardinality exceeds that of D, and form the set $\{f^\beta(x) \mid \beta \leq \gamma\}$. By cardinality considerations, there must be ordinals $\alpha < \beta \leq \gamma$ with $f^\alpha(x) = f^\beta(x)$, and we can assume without loss of generality that α is least with this property. Since $f^\alpha(x) \sqsubseteq f(f^\alpha(x)) \sqsubseteq f^\beta(x) = f^\alpha(x)$, we obtain that $f^\alpha(x) = f(f^\alpha(x))$, and so $a = f^\alpha(x)$ is a fixed point of f. Clearly, we have $x \sqsubseteq a$. Furthermore, if b is any pre-fixed point of f with $x \sqsubseteq b$, then by monotonicity of f and the fact that $f(b) \sqsubseteq b$ we obtain $f^\beta(x) \sqsubseteq b$ for all ordinals β. Hence, $a \sqsubseteq b$ and so a is both the least pre-fixed point and the least fixed point of f above x.

To obtain the final conclusion, we simply set $x = \bot$ and note then that $x \sqsubseteq f(x)$. ■

Some remarks about the missing details in the proof of Theorem 3.4: Notion of cardinality for (possibly) infinite sets. $|\mathbb{N}| = |2\mathbb{N}| = |\mathbb{Q}| < |\mathbb{R}|$. $|A| < |\mathcal{P}(A)|$ for all sets A. Continuum hypothesis: Is there some A with $|\mathbb{N}| < |A| < |\mathbb{R}|$? The Well-Ordering Principle. The Axiom of Choice.

Since the operator Φ_P is monotonic relative to the ordering $\sqsubseteq_k$, it has a least fixed point by the Knaster-Tarski theorem, Theorem 3.4, and this least fixed point is an ordinal power $\Phi_P \uparrow \alpha$ for some ordinal α. The least fixed point of Φ_P is called

the *Kripke-Kleene model* or *Fitting model* for P. It turns out, as we show later in Program 3.11, that Φ_P is not ω-continuous relative to $\sqsubseteq_k$, and so Theorem 2.9 is not generally applicable to Φ_P.

Exercise 33
Compute the Fitting models of the programs Tweety1, Tweety3, and Even. Compare the results with the stable and supported models of the programs.

3.5 Proposition Let P be a program. Then every fixed point M of Φ_P is a model for P with the following properties. (i) If $A \in B_P$ is such that $M(A) = \mathbf{t}$, then there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ with $M(\mathtt{body}) = \mathbf{t}$. (ii) If $A \in B_P$ is such that for all clauses $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ we have $M(\mathtt{body}) = \mathbf{f}$, then $M(A) = \mathbf{f}$.

Proof: Let $A \leftarrow \mathtt{body}$ be a clause in $\mathsf{ground}(P)$. If $M(\mathtt{body}) = \mathbf{t}$, then $M(A) = \Phi_P(M)(A) = M(\mathtt{body}) = \mathbf{t}$. If $M(A) = \mathbf{f}$, then $\Phi_P(M)(A) = M(A) = \mathbf{f}$, hence $M(\mathtt{body}) = \mathbf{f}$. Finally, if $M(A) = \mathbf{u}$, then $\Phi_P(M)(A) = M(A) = \mathbf{u}$ and therefore $M(\mathtt{body}) = \mathbf{f}$ or $M(\mathtt{body}) = \mathbf{u}$. By definition of the truth value given to $\leftarrow$, we see that this suffices to show that M is a model for P.

In order to show (i), assume $M(A) = \mathbf{t}$ for some $A \in B_P$. Then $\Phi_P(M)(A) = M(A) = \mathbf{t}$ and there is a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ with $M(\mathtt{body}) = \mathbf{t}$ by definition of Φ_P.

To show (ii), let $A \in B_P$ and assume that for all clauses $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ we have $M(\mathtt{body}) = \mathbf{f}$. Then $M(A) = \Phi_P(M)(A) = \mathbf{f}$, again by definition of Φ_P. ■

Proposition 3.5 shows that fixed points of Φ_P are *three-valued supported models* for P, meaning that they satisfy (i) and (ii) of Proposition 3.5.

Exercise 34
Show that total three-valued supported models are two-valued supported models and vice-versa.

3.6 Proposition Let P be a program. Then the fixed points of Φ_P are exactly the three-valued supported models for P.

Proof: Certainly, every fixed point of Φ_P is a three-valued supported model for P by Proposition 3.5. Conversely, let M be a three-valued supported model for P, and let $A \in B_P$. If $M(A) = \mathbf{t}$, then there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $M(\mathtt{body}) = \mathbf{t}$, hence $\Phi_P(M)(A) = M(\mathtt{body}) = \mathbf{t} = M(A)$. If $M(A) = \mathbf{f}$, then for all clauses $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ we have that $M(\mathtt{body}) = \mathbf{f}$, since M is a model for P. Hence, $\Phi_P(M)(A) = M(\mathtt{body}) = \mathbf{f} = M(A)$. It follows that M is a fixed point of Φ_P, as required. ■

Before discussing further properties of the Fitting model, we give an alternative characterization of it.

For a program P and a three-valued interpretation $I \in I_{P,3}$, an *I-partial level mapping* for P is a partial mapping $l : B_P \to \alpha$ with domain $\mathsf{dom}(l) = \{A \mid A \in I \text{ or } \neg A \in I\}$, where α is some (countable) ordinal. We extend every such mapping to literals by setting $l(\neg A) = l(A)$ for all $A \in \mathsf{dom}(l)$.

3.7 Definition Let P be a normal logic program, let I be a model for P, and let l be an I-partial level mapping for P. We say that P *satisfies* (F) *with respect to* I *and* l if each $A \in \mathsf{dom}(l)$ satisfies one of the following conditions.

(Fi) $A \in I$ and there is a clause $A \leftarrow L_1, \ldots, L_n$ in $\mathsf{ground}(P)$ such that $L_i \in I$ and $l(A) > l(L_i)$ for all i.

(Fii) $\neg A \in I$ and for each clause $A \leftarrow L_1, \ldots, L_n$ in $\mathsf{ground}(P)$ there exists i with $\neg L_i \in I$ and $l(A) > l(L_i)$.

If $A \in \mathsf{dom}(l)$ satisfies (Fi), then we say that A *satisfies* (Fi) *with respect to* I *and* l, and similarly if $A \in \mathsf{dom}(l)$ satisfies (Fii).

3.8 Theorem ([HW02, HW05]) Let P be a normal logic program with Fitting model M. Then, in the knowledge ordering $\sqsubseteq_k$, M is the greatest model amongst all three-valued models I for which there exists an I-partial level mapping l for P such that P satisfies (F) with respect to I and l.

Proof: Let M_P be the Fitting model for P and define the M_P-partial level mapping l_P as follows: $l_P(A) = \alpha$, where α is the least ordinal such that A is not undefined in $\Phi_P \uparrow (\alpha + 1)$. The proof will be established by showing the following facts. (1) P satisfies (F) with respect to M_P and l_P. (2) If I is a three-valued model for P and l is an I-partial level mapping such that P satisfies (F) with respect to I and l, then $I \subseteq M_P$.

(1) Let $A \in \mathsf{dom}(l_P)$ and suppose that $l_P(A) = \alpha$. We consider the two cases corresponding to (Fi) and (Fii).

Case (Fi). If $A \in M_P$, then $A \in T'_P(\Phi_P \uparrow \alpha)$. Hence, there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $\mathtt{body}$ is true in $\Phi_P \uparrow \alpha$. Therefore, for all $L_i \in \mathtt{body}$, we have that $L_i \in \Phi_P \uparrow \alpha$, and hence $l_P(L_i) < \alpha$, and also that $L_i \in M_P$ for all i. Consequently, A satisfies (Fi) with respect to M_P and l_P.

Case (Fii). If $\neg A \in M_P$, then $A \in F_P(\Phi_P \uparrow \alpha)$. Hence, for each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$, there is a literal $L \in \mathtt{body}$ with $\neg L \in \Phi_P \uparrow \alpha$. But then $l_P(L) < \alpha$ and $\neg L \in M_P$. Consequently, A satisfies (Fii) with respect to M_P and l_P, and we have established that fact (1) holds.

(2) We show via transfinite induction on $\alpha = l(A)$ that, whenever $A \in I$ (respectively, $\neg A \in I$), we have $A \in \Phi_P \uparrow (\alpha + 1)$ (respectively, $\neg A \in \Phi_P \uparrow (\alpha + 1)$). For the base case, note that if $l(A) = 0$, then $A \in I$ implies that A occurs as the head of a fact in $\mathsf{ground}(P)$, hence $A \in \Phi_P \uparrow 1$, and $\neg A \in I$ implies that there is no clause with head A in $\mathsf{ground}(P)$, hence $\neg A \in \Phi_P \uparrow 1$. So assume now that the induction hypothesis holds for all $B \in B_P$ with $l(B) < \alpha$, and that $l(A) = \alpha$. We consider two cases.

Case i. If $A \in I$, then it satisfies (Fi) with respect to I and l. Hence, there is a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $\mathtt{body} \subseteq I$ and $l(K) < \alpha$ for all $K \in \mathtt{body}$. Hence, $\mathtt{body} \subseteq M_P$ by the induction hypothesis, and since M_P is a model for P we obtain $A \in M_P$.

Case ii. If $\neg A \in I$, then A satisfies (Fii) with respect to I and l. Hence, for each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$, there is $K \in \mathtt{body}$ with $\neg K \in I$ and $l(K) < \alpha$. But

then, by the induction hypothesis, we have $\neg K \in M_P$, and consequently for each clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$ we obtain that $\texttt{body}$ is false in M_P. Since $M_P = \Phi_P(M_P)$ is a fixed point of the Φ_P-operator, we obtain $\neg A \in M_P$. This establishes fact (2) and concludes the proof. ■

Exercise 35
Display Theorem 3.8 on the examples in Exercise 33.

21/01/05

The following corollary follows immediately as a special case of the previous result.

3.9 Corollary A normal logic program P has a total Fitting model if and only if there is a total model I for P and a (total) level mapping l for P such that P satisfies (F) with respect to I and l.

3.10 Example Tweety2 (Program 2.27) has Fitting model

$$\{\texttt{penguin(tweety)}, \texttt{bird(bob)}, \texttt{bird(tweety)}, \neg\texttt{flies(tweety)}\}.$$

Thus, we cannot decide whether or not $\texttt{bob}$ is a penguin. Hence, the Fitting semantics suffers from the same deficiency as the supported model semantics, see our discussion of Program 2.27.

The Fitting operator is not ω-continuous in general, not even for definite programs, as shown by the next example.

3.11 Program Consider the program P consisting of the following clauses.

$$\begin{aligned} p(s(X)) &\leftarrow p(X) \\ q &\leftarrow p(X) \end{aligned}$$

Then $\Phi_P \uparrow n = \left\{\neg p\left(s^k(0)\right) \mid k < n\right\}$ for all $n \in \mathbb{N}$ and $\Phi_P \uparrow \omega = \{\neg p(s^n(0)) \mid n \in \mathbb{N}\}$. However, $\Phi_P \uparrow (\omega + 1) = \{\neg q, \neg p(s^n(0)) \mid n \in \mathbb{N}\}$ is the least fixed point of the operator.

The Fitting operator can be thought of as an approximation to the immediate consequence operator, in the sense of the following proposition. For $I \in I_{P,3}$ we will in the following use the notations $I^+ = B_P \cap I$ and $I^- = \{A \in B_P \mid \neg A \in I\}$.

3.12 Proposition Let P be a program. Then for all $I \in I_{P,3}$ we have that $\Phi_P(I)^+ \subseteq T_P(I^+) \subseteq B_P \setminus \Phi_P(I)^-$. Furthermore, the Fitting operator maps total interpretations to total interpretations, and coincides with the immediate consequence operator on these.

Proof: Let $I = I^+ \cup \neg I^-$ be a three-valued interpretation.

Let $A \in \Phi_P(I)^+$. Then there is a clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$, where $\texttt{body}$ equals $A_1 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \cdots \wedge \neg B_k$, say, and is true in the three-valued interpretation I. Therefore, for all i and j, we have $A_i \in I^+$ and $B_j \in I^-$ so that $A_i \in I^+$ and $B_j \notin I^+$. Therefore, $\texttt{body}$ is true in the two-valued interpretation I^+, and so $A \in T_P(I^+)$.

Conversely, if I is total, then $B_j \notin I^+$ means that $B_j \in I^-$, and hence whenever $A \in T_P(I^+)$ we have $A \in \Phi_P(I)^+$, and this deals with the first inclusion.

For the second inclusion, $A \in \Phi_P(I)^-$ if and only if for all clauses $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ we have $\mathtt{body}$ false in the three-valued interpretation I. But then one of the literals in $\mathtt{body}$ is false and so, using the notation already established for $\mathtt{body}$, either some $A_i \in I^-$ or some $B_j \in I^+$, that is, either some $A_i \notin I^+$ or some $B_j \in I^+$. Therefore, $\mathtt{body}$ is also false in the two-valued interpretation I^+ leading to $A \notin T_P(I^+)$. We thus obtain $\Phi_P(I)^- \subseteq B_P \setminus T_P(I^+)$ so that $T_P(I^+) \subseteq B_P \setminus \Phi_P(I)^-$. If I is total, then $B_P \setminus \Phi_P(I)^- = \Phi_P(I)^+ = T'_P(I) = T_P(I^+)$. ■

From Proposition 3.12, we immediately obtain that total Fitting models are always supported. They are in fact also stable in general, as we will see later in Section 3.2. However, if a program has a unique stable model, then it does not necessarily have a total Fitting model.

3.13 Program The program consisting of the three clauses

$$p \leftarrow \neg q$$
$$q \leftarrow \neg p$$
$$p \leftarrow \neg p$$

has unique (two-valued) supported model $\{p\}$, which is also stable. However, its (three-valued) Fitting model is everywhere equal to $\mathbf{u}$.

3.2 Well-founded semantics

The well-founded semantics is due to [vGRS91].

Motivate with stratification idea.

3.14 Definition Let P be a normal logic program, let I be a model for P, and let l be an I-partial level mapping for P. We say that P *satisfies* (WF) *with respect to* I *and* l if each $A \in \mathsf{dom}(l)$ satisfies one of the following conditions.

(WFi) $A \in I$ and there is a clause $A \leftarrow L_1, \ldots, L_n$ in $\mathsf{ground}(P)$ such that $L_i \in I$ and $l(A) > l(L_i)$ for all i.

(WFii) $\neg A \in I$ and for each clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ one (at least) of the following conditions holds:

(WFiia) There exists i with $\neg A_i \in I$ and $l(A) \geq l(A_i)$.

(WFiib) There exists j with $B_j \in I$ and $l(A) > l(B_j)$.

If $A \in \mathsf{dom}(l)$ satisfies (WFi), then we say that A *satisfies* (WFi) *with respect to* I *and* l, and similarly if $A \in \mathsf{dom}(l)$ satisfies (WFii).

We note that conditions (Fi) and (WFi) are identical. However, replacing (WFi) by a "stratified version" such as the following is not satisfactory.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \leftarrow q$
t	**b**	**f**	**b**	**t**	**t**
f	**b**	**t**	**f**	**b**	**f**
b	**t**	**b**	**b**	**t**	**f**
b	**f**	**b**	**f**	**b**	**t**
b	**b**	**b**	**b**	**b**	**t**
b	**u**	**b**	**f**	**t**	**t**
u	**b**	**u**	**f**	**t**	**t**

Table 2: A four-valued logic.

(SFi) $A \in I$ and there is a clause $A \leftarrow A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m$ in $\mathsf{ground}(P)$ such that $A_i, \neg B_j \in I$, $l(A) \geq l(A_i)$, and $l(A) > l(B_j)$ for all i and j.

Indeed, if we do replace condition (WFi) by condition (SFi), then it is not guaranteed that, for a given program, there is a greatest model satisfying the desired properties: consider the program consisting of the two clauses $p \leftarrow p$ and $q \leftarrow \neg p$, the two (total) models $\{p, \neg q\}$ and $\{\neg p, q\}$, and the level mapping l with $l(p) = 0$ and $l(q) = 1$. These models are incomparable, yet in both cases the conditions obtained by replacing (WFi) by (SFi) in (WF) are satisfied.

So, in the light of Theorem 3.8, Definition 3.14 provides a natural "stratified version" of the Fitting semantics, Furthermore, the resulting semantics coincides with another well-known semantics, called the *well-founded semantics*, which is a very satisfactory result. To establish this claim, we need to introduce well-founded models, and this we do next.

For a proper treatment, we will temporarily need a fourth truth value **b**, called *both*. For the truth table we extend Table 3.1 by Table 3.2. Interpretations in this logic can be written as follows. Let the set of *signed sets* $I_{P,4}$ be the set of all subsets of $B_P \cup \neg B_P$. We consider $I_{P,4}$ to be ordered by subset-inclusion. Elements of $I_{P,4}$ are called *four-valued interpretations*.

Given a normal logic program P and $I \in I_{P,4}$, we say that $U \subseteq B_P$ is an *unfounded set (of P) with respect to I* if each atom $A \in U$ satisfies the following condition: for each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ at least one of the following holds.

(Ui) Some (positive or negative) literal in `body` is false in I.

(Uii) Some (non-negated) atom in `body` occurs in U.

3.15 Proposition Let P be a program and let $I \in I_{P,4}$. Then there exists a greatest unfounded set of P with respect to I.

Proof: If $(U_i)_{i \in \mathcal{I}}$ is a family of sets each of which is an unfounded set of P with respect to I, then it is easy to see that $\bigcup_{i \in \mathcal{I}} U_i$ is also an unfounded set of P with respect to I. ■

Let P be a program and recall the definition of the operator T'_P from Section 3.1. It is straightforward to lift T'_P to an operator on $I_{P,4}$, namely, by defining $T'_P(I)$,

for $I \in I_{P,4}$, to be the set of all $A \in B_P$ for which there is a clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$ with body true in $I \in I_{P,4}$. For all $I \in I_{P,4}$, define $U_P(I)$ to be the greatest unfounded set (of P) with respect to I. Finally, define[5]

$$W_P(I) = T'_P(I) \cup \neg U_P(I)$$

for all $I \in I_{P,4}$.

We note that W_P does not restrict to a function on $I_{P,3}$, which necessitates using $I_{P,4}$ instead.

3.16 Example Consider Program 2.20 and $I = \{p\} \in I_{P,3}$. Then $T'(I) = \{p\}$ and $U_P(I) = \{p\}$, so $W_P(I) = \{p, \neg p\} \notin I_{P,3}$.

3.17 Proposition ([vGRS91]) Let P be a program. Then W_P is monotonic on $I_{P,4}$.

Proof: Let $I, K \in I_{P,4}$ with $I \subseteq K$. Then we obtain $T'_P(I) \subseteq T'_P(K)$ as in the proof of Proposition 3.3. So it suffices to show that every unfounded set of P with respect to I is also an unfounded set of P with respect to K, and this fact follows immediately from the definition. ■

Since W_P is monotonic, it has a least fixed point by the Knaster-Tarski theorem, Theorem 3.4. The least fixed point of W_P is called the *well-founded model* for P, giving the *well-founded semantics* of P.

Exercise 36
Compute the well-founded model for the programs Tweety1, Tweety2, Tweety3, and Even. Compare with the other semantics of these programs.

We will show shortly that the well-founded model is always in $I_{P,3}$, but let us remark first that the operator W_P is not order continuous in general nor even ω-continuous, as the following example shows.

3.18 Program Let P be the following program.

$$\begin{aligned} p(0) &\leftarrow \\ p(s(X)) &\leftarrow p(X) \\ q(s(X)) &\leftarrow \neg p(X) \\ r &\leftarrow \neg q(X) \end{aligned}$$

Then $W_P \uparrow n = \{p(s^k(0)), \neg q(s^k(0)) \mid k < n\}$, and

$$\begin{aligned} W_P \uparrow \omega &= \{p(s^n(0)), \neg q(s^n(0)) \mid n \in \mathbb{N}\} \\ &\neq \{p(s^n(0)), \neg q(s^n(0)), \neg r \mid n \in \mathbb{N}\} = W_P \uparrow (\omega + 1). \end{aligned}$$

3.19 Theorem ([vGRS91]) Let P be a program. Then $W_P \uparrow \alpha \in I_{P,3}$ for all ordinals α. In particular, the well-founded model for P is in $I_{P,3}$.

[5]The operator W_P and the well-founded semantics are due to van Gelder, Ross, and Schlipf, see [vGRS91].

Proof: We first need some notation. Let M be the least fixed point of W_P and for each atom $A \in M^+$ let $l(A)$ be the least ordinal β such that $A \in W_P \uparrow (\beta + 1)$.

Now assume that there is an ordinal γ which is least under the condition that $W_P \uparrow \gamma \notin I_{P,3}$. Then γ must be a successor ordinal, so let $I = W_P \uparrow (\gamma - 1) \in I_{P,3}$. Now consider the set $U = T'_P(I) \cap U_P(I)$, which means that for each $A \in U$ and each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $\mathtt{body}$ is true in I we have that some (non-negated) atom B in $\mathtt{body}$ occurs in $U_P(I)$. We obtain $B \in U_P(I) \cap I$ and since $I \subseteq T'_P(I)$ we get $B \in U$. Now let $A \in U$ be chosen such that it is minimal with respect to $l(A) = \beta$, and notice that necessarily $\beta < \gamma$. Then there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ with $\mathtt{body}$ being true in $W_P \uparrow \beta \subseteq I$, and in particular $B \in I$ and $l(B) < l(A)$ for all (non-negated) atoms B which occur in $\mathtt{body}$. But we have just shown that then $B \in U$ which contradicts minimality of $l(A)$. ■

28/01/05

3.20 Proposition Let P be a program and let $I \in I_{P,3}$. Then $\Phi_P(I) \subseteq W_P(I)$. Furthermore, the three-valued fixed points of W_P are three-valued supported models for P with respect to Kleene's strong three-valued logic.

Proof: Let $A \in F_P(I)$. Then for each clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$, we have that $I(\mathtt{body}) = \mathbf{f}$ and so there is a literal $L \in \mathtt{body}$ with $I(L) = \mathbf{f}$. But then A is in the greatest unfounded set of P with respect to I, and so $A \in U_P(I)$. This shows that $\Phi_P(I) \subseteq W_P(I)$.

The remaining statement is proven in Exercise 37. ■

Exercise 37
Complete the proof of Proposition 3.20.

We will now show formally that the well-founded model can be characterized using Definition 3.14[6].

3.21 Theorem ([HW02, HW05]) Let P be a normal logic program with well-founded model M. Then, in the knowledge ordering, M is the greatest model amongst all models I for which there exists an I-partial level mapping l for P such that P satisfies (WF) with respect to I and l.

Proof: Let M_P be the well-founded model for P and define the M_P-partial level mapping l_P as follows: $l_P(A) = \alpha$, where α is the least ordinal such that A is not undefined in $W_P \uparrow (\alpha + 1)$. The proof will proceed by establishing the following facts: (1) P satisfies (WF) with respect to M_P and l_P. (2) If I is a model for P and l is an I-partial level mapping such that P satisfies (WF) with respect to I and l, then $I \subseteq M_P$.

(1) Let $A \in \mathsf{dom}(l_P)$ and suppose that $l_P(A) = \alpha$. We consider the two cases corresponding to (WFi) and (WFii).

Case i. $A \in M_P$. Then $A \in T'_P(W_P \uparrow \alpha)$. Hence, there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $\mathtt{body}$ is true in $W_P \uparrow \alpha$. Thus, for all $L_i \in \mathtt{body}$, we have that

[6] A different characterization using level mappings, which is nevertheless in the same spirit, can be found in [LMPS95].

$L_i \in W_P \uparrow \alpha$. Hence, $l_P(L_i) < \alpha$ and $L_i \in M_P$ for all i. Consequently, A satisfies (WFi) with respect to M_P and l_P.

Case ii. $\neg A \in M_P$. Then $A \in U_P(W_P \uparrow \alpha)$, and so A is contained in the greatest unfounded set of P with respect to $W_P \uparrow \alpha$. Hence, for each clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$, either (Ui) or (Uii) holds for this clause with respect to $W_P \uparrow \alpha$ and the unfounded set $U_P(W_P \uparrow \alpha)$. If (Ui) holds, then there exists some literal $L \in \texttt{body}$ with $\neg L \in W_P \uparrow \alpha$. Hence, $l_P(L) < \alpha$ and condition (WFiia) holds relative to M_P and l_P if L is an atom, or condition (WFiib) holds relative to M_P and l_P if L is a negated atom. On the other hand, if (Uii) holds, then some (non-negated) atom B in body occurs in $U_P(W_P \uparrow \alpha)$. Hence, $l_P(B) \leq l_P(A)$ and A satisfies (WFiia) with respect to M_P and l_P. Thus, we have established that the statement (1) holds.

(2) We show via transfinite induction on $\alpha = l(A)$ that, whenever $A \in I$ (respectively, $\neg A \in I$), then $A \in W_P \uparrow (\alpha + 1)$ (respectively, $\neg A \in W_P \uparrow (\alpha + 1)$). For the base case, note that if $l(A) = 0$, then $A \in I$ implies that A occurs as the head of a fact in $\mathsf{ground}(P)$. Hence, $A \in W_P \uparrow 1$. If $\neg A \in I$, then consider the set U of all atoms B with $l(B) = 0$ and $\neg B \in I$. We show that U is an unfounded set of P with respect to $W_P \uparrow 0$, and this suffices since it implies $\neg A \in W_P \uparrow 1$ by the fact that $A \in U$. So let $C \in U$ and let $C \leftarrow \texttt{body}$ be a clause in $\mathsf{ground}(P)$. Since $\neg C \in I$, and $l(C) = 0$, we have that C satisfies (WFiia) with respect to I and l, and so condition (Uii) is satisfied showing that U is an unfounded set of P with respect to I. Assume now that the induction hypothesis holds for all $B \in B_P$ with $l(B) < \alpha$. We consider two cases.

Case i. $A \in I$. Then A satisfies (WFi) with respect to I and l. Hence, there is a clause $A \leftarrow \texttt{body}$ in $\mathsf{ground}(P)$ such that $\texttt{body} \subseteq I$ and $l(K) < \alpha$ for all $K \in \texttt{body}$. Hence, $\texttt{body} \subseteq W_P \uparrow \alpha$, and we obtain $A \in T'_P(W_P \uparrow \alpha)$, as required.

Case ii. $\neg A \in I$. Consider the set U of all atoms B with $l(B) = \alpha$ and $\neg B \in I$. We show that U is an unfounded set of P with respect to $W_P \uparrow \alpha$, and this suffices since it implies $\neg A \in W_P \uparrow (\alpha + 1)$ by the fact that $A \in U$. So let $C \in U$ and let $C \leftarrow \texttt{body}$ be a clause in $\mathsf{ground}(P)$. Since $\neg C \in I$, we have that C satisfies (WFii) with respect to I and l. If there is a literal $L \in \texttt{body}$ with $\neg L \in I$ and $l(L) < l(C)$, then by the induction hypothesis we obtain $\neg L \in W_P \uparrow \alpha$, so condition (Ui) is satisfied for the clause $C \leftarrow \texttt{body}$ with respect to $W_P \uparrow \alpha$ and U. In the remaining case, we have that C satisfies condition (WFiia), and there exists an atom $B \in \texttt{body}$ with $\neg B \in I$ and $l(B) = l(C)$. Hence, $B \in U$ showing that condition (Uii) is satisfied for the clause $C \leftarrow \texttt{body}$ with respect to $W_P \uparrow \alpha$ and U. Hence, U is an unfounded set of P with respect to $W_P \uparrow \alpha$. ■

Exercise 38
Illustrate Theorem 3.21 on the examples Tweety1, Tweety2, Tweety3, and Even.

As a special case, we immediately obtain the following corollary.

3.22 Corollary A normal logic program P has a total well-founded model if and only if there is a total model I for P and a (total) level mapping l such that P satisfies (WF) with respect to I and l.

An alternative way of characterizing the well-founded semantics is via the Gelfond-Lifschitz operator from Section 2.2. Recall from Theorem 2.25 that the Gelfond-Lifschitz operator is antitonic. In particular, this means that, for any program P, the operator GL_P^2, obtained by applying GL_P twice, is monotonic, and by the Knaster-Tarski theorem has a least fixed point, L_P. Note further that $I_{P,2}$ is a complete lattice in the dual of the truth ordering on $I_{P,2}$. So, on applying the Knaster-Tarski theorem again, we also obtain that GL_P^2 has a greatest fixed point, G_P. Since $L_P \subseteq G_P$, we obtain that $L_P \cup \neg(B_P \setminus G_P)$ is a three-valued interpretation for P, and is in fact a model for P, as we show next, called the *alternating fixed point model* for P.

We are going to show that the alternating fixed point model coincides with the well-founded model. Let us first introduce some temporary notation, where P is an arbitrary program.

$$\begin{array}{rrl} L_0 = \emptyset & G_0 = B_P & \\ L_{\alpha+1} = \mathsf{GL}_P(G_\alpha) & G_{\alpha+1} = \mathsf{GL}_P(L_\alpha) & \text{for any ordinal } \alpha \\ L_\alpha = \bigcup_{\beta<\alpha} L_\beta & G_\alpha = \bigcap_{\beta<\alpha} G_\beta & \text{for limit ordinal } \alpha. \end{array}$$

Since $\emptyset \subseteq B_P$, we obtain $L_0 \subseteq L_1 \subseteq G_1 \subseteq G_0$ and, by transfinite induction, it can easily be shown that $L_\alpha \subseteq L_\beta \subseteq G_\beta \subseteq G_\alpha$ whenever $\alpha \leq \beta$.

3.23 Theorem ([vG89]) Let P be a program. Then the following hold.

(a) $L_P = \mathsf{GL}_P(G_P)$ and $G_P = \mathsf{GL}_P(L_P)$.

(b) For every stable model S for P, we have $L_P \subseteq S \subseteq G_P$.

(c) $M = L_P \cup \neg(B_P \setminus G_P)$ is the well-founded model for P.

Proof: The proof is due to [HW05].

(a) We obtain $\mathsf{GL}_P^2(\mathsf{GL}_P(L_P)) = \mathsf{GL}_P(\mathsf{GL}_P^2(L_P)) = \mathsf{GL}_P(L_P)$, so $\mathsf{GL}_P(L_P)$ is a fixed point of GL_P^2, and hence $L_P \subseteq \mathsf{GL}_P(L_P) \subseteq G_P$. Similarly, $L_P \subseteq \mathsf{GL}_P(G_P) \subseteq G_P$. Since $L_P \subseteq G_P$, we get from the antitonicity of GL_P that $L_P \subseteq \mathsf{GL}_P(G_P) \subseteq \mathsf{GL}_P(L_P) \subseteq G_P$. Similarly, since $\mathsf{GL}_P(L_P) \subseteq G_P$, we obtain $\mathsf{GL}_P(G_P) \subseteq \mathsf{GL}_P^2(L_P) = L_P \subseteq \mathsf{GL}_P(G_P)$, so $\mathsf{GL}_P(G_P) = L_P$, and hence $G_P = \mathsf{GL}_P^2(G_P) = \mathsf{GL}_P(L_P)$.

(b) It suffices to note that S is a fixed point of GL_P, by Theorem 2.25, and hence is a fixed point of GL_P^2.

(c) We prove this statement by applying Theorem 3.21. First, we define an M-partial level mapping l. For convenience, we will take as image set of l, pairs (α, n) of ordinals, where $n \leq \omega$, with the lexicographic ordering. This can be done without loss of generality because any set of pairs of ordinals, lexicographically ordered, is certainly well-ordered and therefore order-isomorphic to an ordinal. For $A \in L_P$, let $l(A)$ be the pair (α, n), where α is the least ordinal such that $A \in L_{\alpha+1}$, and n is the least ordinal such that $A \in T_{P/G_\alpha} \uparrow (n+1)$. For $B \notin G_P$, let $l(B)$ be the pair (β, ω), where β is the least ordinal such that $B \notin G_{\beta+1}$. We show next by transfinite induction that P satisfies (WF) with respect to M and l.

Let $A \in L_1 = T_{P/B_P} \uparrow \omega$. Since P/B_P consists of exactly all clauses from $\mathsf{ground}(P)$ which contain no negation, we have that A is contained in the least two-valued model for a definite subprogram of P, namely, P/B_P, and (WFi) is satisfied, by Theorem 2.8. Now let $\neg B \in \neg(B_P \setminus G_P)$ be such that $B \in (B_P \setminus G_1) = B_P \setminus T_{P/\emptyset} \uparrow \omega$. Since $P/\emptyset$ contains all clauses from $\mathsf{ground}(P)$ with all negative literals removed, we obtain that each clause in $\mathsf{ground}(P)$ with head B must contain a positive body literal $C \notin G_1$, which, by definition of l, must have the same level as B, hence (WFiia) is satisfied.

Assume now that, for some ordinal α, we have shown that A satisfies (WF) with respect to M and l for all $n \leq \omega$ and all $A \in B_P$ with $l(A) \leq (\alpha, n)$.

Let $A \in L_{\alpha+1} \setminus L_\alpha = T_{P/G_\alpha} \uparrow \omega \setminus L_\alpha$. Then $A \in T_{P/G_\alpha} \uparrow n \setminus L_\alpha$ for some $n \in \mathbb{N}$; note that all (negative) literals which were removed by the Gelfond-Lifschitz transformation from clauses with head A have level less than $(\alpha, 0)$. Then the assertion that A satisfies (WF) with respect to M and l follows again by Theorem 2.8.

Let $A \in (B_P \setminus G_{\alpha+1}) \cap G_\alpha$. Then $A \notin T_{P/L_\alpha} \uparrow \omega$. Let $A \leftarrow A_1, \ldots, A_k, \neg B_1, \ldots, \neg B_m$ be a clause in $\mathsf{ground}(P)$. If $B_j \in L_\alpha$ for some j, then $l(A) > l(B_j)$. Otherwise, since $A \notin T_{P/L_\alpha} \uparrow \omega$, we have that there exists A_i with $A_i \notin T_{P/L_\alpha} \uparrow \omega$, and hence $l(A) \geq l(A_i)$, and this suffices.

This finishes the proof that P satisfies (WF) with respect to M and l. It therefore only remains to show that M is greatest with this property.

So assume that $M_1 \neq M$ is the greatest model such that P satisfies (WF) with respect to M_1 and some M_1-partial level mapping l_1.

Assume $L \in M_1 \setminus M$ and, without loss of generality, let the literal L be chosen such that $l_1(L)$ is minimal. We consider the following two cases.

(Case i) If $L = A$ is an atom, then there exists a clause $A \leftarrow \mathtt{body}$ in $\mathsf{ground}(P)$ such that $l_1(L) < l_1(A)$ for all literals L in $\mathtt{body}$, and such that $\mathtt{body}$ is true in M_1. Hence, $\mathtt{body}$ is true in M and $A \leftarrow \mathtt{body}$ transforms to a clause $A \leftarrow A_1, \ldots, A_n$ in P/G_P with $A_1, \ldots, A_n \in L_P = T_{P/G_P} \uparrow \omega$. But this implies $A \in M$, contradicting $A \in M_1 \setminus M$.

(Case ii) If $L = \neg A \in M_1 \setminus M$ is a negated atom, then $\neg A \in M_1$ and $A \in G_P = T_{P/L_P} \uparrow \omega$, so $A \in T_{P/L_P} \uparrow n$ for some $n \in \mathbb{N}$. We show by induction on n that this leads to a contradiction, to finish the proof.

If $A \in T_{P/L_P} \uparrow 1$, then there is a unit clause $A \leftarrow$ in P/L_P, and any corresponding clause $A \leftarrow \neg B_1, \ldots, \neg B_k$ in $\mathsf{ground}(P)$ satisfies $B_1, \ldots, B_k \notin L_P$. Since $\neg A \in M_1$, we also obtain by Theorem 3.21 that there is $i \in \{1, \ldots, k\}$ such that $B_i \in M_1$ and $l_1(B_i) < l_1(A)$. By minimality of $l_1(A)$, we obtain $B_i \in M$, and hence $B_i \in L_P$, which contradicts $B_i \notin L_P$.

Now assume that there is no $\neg B \in M_1 \setminus M$ with $B \in T_{P/L_P} \uparrow k$ for any $k < n+1$, and let $\neg A \in M_1 \setminus M$ with $A \in T_{P/L_P} \uparrow (n+1)$. Then there is a clause $A \leftarrow A_1, \ldots, A_m$ in P/L_P with $A_1, \ldots, A_m \in T_{P/L_P} \uparrow n \subseteq G_P$, and we note that we cannot have $\neg A_i \in M_1 \setminus M$ for any $i \in \{1, \ldots, m\}$, by our current induction hypothesis. Furthermore, it is also impossible for $\neg A_i$ to belong to M for any i, otherwise we would have $A_i \in B_P \setminus G_P$. Thus, we conclude that we cannot have $\neg A_i \in M_1$ for any i. Moreover, there is a corresponding clause $A \leftarrow A_1, \ldots, A_m, \neg B_1, \ldots, \neg B_{m_1}$ in $\mathsf{ground}(P)$ with $B_1, \ldots, B_{m_1} \notin L_P$. Hence, by Theorem 3.21, we know that there is $i \in \{1, \ldots, m_1\}$ such that $B_i \in M_1$ and $l_1(B_i) < l_1(A)$. By minimality of $l_1(A)$, we conclude that

$B_i \in M$, so that $B_i \in L_P$, and this contradicts $B_i \notin L_P$.

■

Exercise 39
Compute the iterates of the Gelfond-Lifschitz operator from $\emptyset$ and B_P for the programs Tweety1, Tweety2, Tweety3, and Even.

It follows from Theorem 3.23 (b) that total well-founded models are unique stable models. The converse, however, does not hold. Indeed, Program 3.13 has well-founded model $\emptyset$, as can easily be seen by noting that $\mathsf{GL}_P(\emptyset) = B_P$ and $\mathsf{GL}_P(B_P) = \emptyset$.

3.24 Theorem Let P be a program with total Fitting model. Then P has a total well-founded model. Moreover, P also has a unique stable and a unique supported model. Furthermore, all these models coincide.

Proof: By Proposition 3.20, P has a total well-founded model which coincides with the Fitting model. By Theorem 3.23 (b), P has a unique stable model and this coincides with the well-founded model by (c). Finally, by Proposition 3.12, P has a unique supported model which coincides with its Fitting model. ■

Exercise 40
Redo Exercise 28, taking all material into account.

4 Outlook

04/02/05

Timeline

1965	Robinson Resolution [Rob65]
1974	Kowalski SLD-Resolution [Kow74]
ca. 1975	Colmerauer, Prolog [CR93]
1978	McCarthy Circumscription [McC77, McC80]
1980	Reiter Default Logic [Rei80]
1984	Moore Autoepistemic Logic [Moo84, Moo85]
1985	Fitting Semantics [Fit85]
1988–91	Stable and well-founded semantics [GL88, GL91, vGRS91]
ca. 1993	XSB Prolog [CW93, CSW93]
ca. 1997	ASP systems dlv, smodels [ELM+97, SNS02]
currently	Syntactic extensions (see below)
currently	Applications (see below)

Some syntactic extensions

- two negations (default and *classical*) [GL91]
- paraconsistency [Ari02]
- disjunctive heads/databases [GL91, LMR92, Min97, MS02]

- preferences [BE99]
- updates of logic programs [ALP$^+$98, ABLP02, Lei03]
- etc.

Some applications

- NMR is a standard paradigm in KR&R and in AI
- Reasoning about action and change [Lif99, Lif02]
- Semantic Web [BH95, ELST04]
- etc.

Own interests

- Neural-symbolic integration: How to represent (or extract) logic programs under different semantics by (from) artificial neural networks [HHS04, BH04].
- Semantic Web applications.
- Decidability questions.
- Uniform approach using level mappings. E.g. disjunctive cases unclear in particular for well-founded semantcs.
- How to control iterations of T_P (metrics, topology) [Sed95, Hit01, HS03].
- etc.

Some things we have learned

- The foundations of NMR with LPs.
- The most important semantics of LPs.
- Relationships between these semantics.
- The prominent role of negation in NMR.
- Standard construction methods for semantics of LPs.
- Ordinals and transfinite induction.
- How much effort and formal considerations a correct mathematical treatment requires.
- etc.

References

[ABLP02] José J. Alferes, Antonio Brogi, Joao A. Leite, and Luís M. Pereira. Evolving logic programs. In Sergio Flesca, Sergio Greco, Nicola Leone, and Giovambattista Ianni, editors, *JELIA*, volume 2424 of *Lecture Notes in Computer Science*, pages 50–61. Springer, 2002.

[ABW88] Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann, Los Altos, CA, 1988.

[AJ94] Samson Abramsky and Achim Jung. Domain theory. In Samson Abramsky, Dov Gabbay, and Thomas S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 3. Clarendon, Oxford, 1994.

[ALP+98] José J. Alferes, Joao A. Leite, Luís M. Pereira, Halina Przymusinska, and Teodor C. Przymusinski. Dynamic logic programming. In Anthony G. Cohn, Lenhard K. Schubert, and Stuart C. Shapiro, editors, *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 98–111. Morgan Kaufmann, 1998.

[Apt97] Krzysztof R. Apt. *From Logic Programming to Prolog.* International Series in Computer Science. Prentice Hall, 1997.

[Ari02] Ofer Arieli. Paraconsistent declarative semantics for extended logic programs. *Annals of Mathematics and Artificial Intelligence*, 36(4):381–417, 2002.

[BE99] Gerhard Brewka and Thomas Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109:297–356, 1999.

[BH95] F. Baader and B. Hollunder. Embedding defaults into terminological representation systems. *J. Automated Reasoning*, 14:149–180, 1995.

[BH04] Sebastian Bader and Pascal Hitzler. Logic programs, iterated function systems, and recurrent radial basis function networks. *Journal of Applied Logic*, 2(3):273–300, 2004.

[CR93] Alain Colmerauer and Philippe Roussel. The birth of Prolog. In *ACM SIGPLAN Notices*, volume 28(3), pages 37–52. ACM Press, 1993.

[CSW93] Weidong Chen, Terrance Swift, and David S. Warren. Efficient top-down computation of queries under the well-founded semantics. *The Journal of Logic Programming*, 24(3):161–199, 1993.

[CW93] Weidong Chen and David S. Warren. Query evaluation under the well founded semantics. In *The Twelfth ACM Symposium on Principles of Database Systems*, 1993.

[DK89] Phan Minh Dung and Kanchana Kanchanasut. A fixpoint approach to declarative semantics of logic programs. In Ewing L. Lusk and Ross A. Overbeek, editors, *Logic Programming, Proceedings of the North American Conference 1989, NACLP'89, Cleveland, Ohio*, pages 604–625. MIT Press, 1989.

[ELM+97] Thomas Eiter, Nicola Leone, Christinel Mateis, Gerald Pfeifer, and Francesco Scarcello. A deductive system for nonmonotonic reasoning. In Jürgen Dix, Ulrich Furbach, and Anil Nerode, editors, *Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'97*, volume 1265 of *Lecture Notes in Artificial Intelligence*. Springer, Berlin, 1997.

[ELST04] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the Semantic Web. In *Prof. KR-2004*, pages 141–151, 2004.

[Fag94] François Fages. Consistency of Clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, 1:51–60, 1994.

[Fit85] Melvin Fitting. A Kripke-Kleene-semantics for general logic programs. *The Journal of Logic Programming*, 2:295–312, 1985.

[GHK+03] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, and D.S. Scott. *Continuous Lattices and Domains*, volume 93 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2003.

[GL88] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming. Proceedings of the 5th International Conference and Symposium on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[GL91] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[HHS04] Pascal Hitzler, Steffen Hölldobler, and Anthony K. Seda. Logic programs and connectionist networks. *Journal of Applied Logic*, 3(2):245–272, 2004.

[Hit97] Pascal Hitzler. Fixpunktsemantik. In Martin Grimm and Gudrun Kalmbach, editors, *Begabtenförderung im MINT-Bereich 1*, pages 57–61. Aegis-Verlag, Ulm, 1997.

[Hit01] Pascal Hitzler. *Generalized Metrics and Topology in Logic Programming Semantics*. PhD thesis, Department of Mathematics, National University of Ireland, University College Cork, 2001.

[Hit04] Pascal Hitzler. Nichtmonotone, neuro-symbolische und begriffliche Wissensverarbeitung. Habilitation/Postdoctoral thesis, Department of Computer Science, Dresden University of Technology, December 2004. Submitted.

[HS03] Pascal Hitzler and Anthony K. Seda. Generalized metrics and uniquely determined logic programs. *Theoretical Computer Science*, 305(1–3):187–219, 2003.

[HS0x] Pascal Hitzler and Anthony K. Seda. Mathematical foundations of logic programming and non-monotonic reasoning (tentative title). Research monograph, 200x. In preparation.

[HW02] Pascal Hitzler and Matthias Wendt. The well-founded semantics is a stratified Fitting semantics. In Matthias Jarke, Jana Koehler, and Gerhard Lakemeyer,

editors, *Proceedings of the 25th Annual German Conference on Artificial Intelligence, KI2002, Aachen, Germany, September 2002*, volume 2479 of *Lecture Notes in Artificial Intelligence*, pages 205–221. Springer, Berlin, 2002.

[HW05] Pascal Hitzler and Matthias Wendt. A uniform approach to logic rogramming semantics. *Theory and Practice of Logic Programming*, 5(1–2):123–159, 2005.

[Kow74] Robert A. Kowalski. Predicate logic as a programming language. In *Proceedings IFIP'74*, pages 569–574. North-Holland, 1974.

[Lei03] Joao A. Leite. *Evolving Knowledge Bases*, volume 81 of *Frontiers of Artificial Intelligence and Applications*. IOS Press, 2003.

[Lif99] Vladimir Lifschitz. Answer set planning. In Danny De Schreye, editor, *Logic Programming. Proceedings of the 1999 International Conference on Logic Programming*, pages 23–37, Cambridge, Massachusetts, 1999. MIT Press.

[Lif02] Vladimir Lifschitz. Answer set programming and plan generation. *Artificial Intelligence*, 138:39–54, 2002.

[Llo88] John W. Lloyd. *Foundations of Logic Programming*. Springer, Berlin, 1988.

[LMPS95] Vladimir Lifschitz, Norman McCain, Teodor C. Przymusinski, and Robert F. Stärk. Loop checking and the well-founded semantics. In V. Wiktor Marek and Anil Nerode, editors, *Logic Programming and Non-monotonic Reasoning, Proceedings of the 3rd International Conference, LPNMR'95, Lexington, KY, USA, June 1995*, volume 928 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 1995.

[LMR92] Jorge Lobo, Jack Minker, and Arcot Rajasekar. *Foundations of disjunctive logic programming*. MIT Press, 1992.

[McC77] John McCarthy. Epistemological problems of artificial intelligence. In *Proceedings of IJCAI-77*, pages 1038–1044, 1977.

[McC80] John McCarthy. Circumscription — a form of non-monotonic reasoning. *Artificial Intelligence*, 13(1):27–39, 1980.

[Min97] Jack Minker. Logic and databases: Past, present, and future. *AI Magazine*, 18(3):21–47, 1997.

[Moo84] Robert Moore. Possible-worlds semantics for autoepistemic logic. In *Proceedings of the 1984 Non-monotonic Reasoning Workshop*. AAAI, Menlo Park, CA, 1984.

[Moo85] Robert Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25(1), 1985.

[MS02] Jack Minker and Dietmar Seipel. Disjunctive logic programming: A survey and assessment. In Antonis C. Kakas and Fariba Sadri, editors, *Computational Logic. Logic Programming and Beyond*, volume 2407 of *Lecture Notes in Computer Science*, pages 472–511. Springer, 2002.

[Rei80] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[Rob65] J. Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[Sed95] Anthony K. Seda. Topology and the semantics of logic programs. *Fundamenta Informaticae*, 24(4):359–386, 1995.

[SHLG94] Viggo Stoltenberg-Hansen, Ingrid Lindström, and Edward R. Griffor. *Mathematical Theory of Domains*. Cambridge University Press, 1994.

[SNS02] Patrik Simons, Ilkka Niemelä, and Timo Soininen. Extending and implementing the stable model semantics. *Artificial Intelligence*, 138(1–2):181–234, 2002.

[vEK76] Marten H. van Emden and Robert A. Kowalski. The semantics of predicate logic as a programming language. *Journal of ACM*, 23(4):733–742, 1976.

[vG89] Allen van Gelder. The alternating fixpoint of logic programs with negation. In *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Philadelphia, Pennsylvania*, pages 1–10. ACM Press, 1989.

[vGRS91] Allen van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

[Wen02a] Matthias Wendt. Unfolding the well-founded semantics. *Journal of Electrical Engineering, Slovak Academy of Sciences*, 53(12/s):56–59, 2002. (Proceedings of the 4th Slovakian Student Conference in Applied Mathematics, Bratislava, April 2002)[7].

[Wen02b] Matthias Wendt. Unfolding the well-founded semantics. Technical Report WV–02–08, Knowledge Representation and Reasoning Group, Department of Computer Science, Dresden University of Technology, 2002. http://www.wv.inf.tu-dresden.de/Publications/2002/.

[7] Available online as [Wen02b].