Well-Founded Semantics, Generalized

Melvin C. Fitting
Department of Mathematics and Computer Science
Lehman College (CUNY)
Bedford Park Boulevard West
Bronx, NY 10468, USA
mlflc@cunyvm.cuny.edu

Abstract

Classical fixpoint semantics for logic programs is based on the $T_{\mathcal{P}}$ immediate consequence operator. The Kripke/Kleene, three-valued, semantics uses $\Phi_{\mathcal{P}}$, which extends $T_{\mathcal{P}}$ to Kleene's strong three-valued logic. Both these approaches generalize to cover logic programming systems based on a wide class of logics, provided only that the underlying structure be that of a bilattice. This was presented in earlier papers. Recently well-founded semantics has become influential for classical logic programs. We show how the well-founded approach also extends naturally to the same family of bilattice-based programming languages that the earlier fixpoint approaches extended to. Doing so provides a natural semantics for logic programming systems that have already been proposed, as well as for a large number that are of only theoretical interest. And finally, doing so simplifies the proofs of basic results about the well-founded semantics, by stripping away inessential details.

1 Introduction

There have been many attempts at providing a semantics for logic programs with negation, but they all tend to agree on one point: if restrictions are not to be placed on programs, partial models must be allowed. For instance, if every formula is required to have either *true* or *false* as its value, a simple program like $p \leftarrow \neg p$ can have no models. Among the approaches allowing partial models are [5, 14, 15, 22, 23, 16]. These are not all the same. For instance, the so-called Kripke/Kleene approach of [5] produces, for the program $p \leftarrow p$, a model in which p lacks a classical truth value, while the well-founded approach of [22] produces a model in which p has the value *false*.

Once partial models are allowed, it is a natural step to shift the paradigm from that of an incompletely specified two-valued model to that of a completely specified three-valued one, in which the additional truth value, $\perp$, can be read as "lacks a classical truth value." It is then also a natural step

to ask, what is special about the use of three-valued logic; are there other logics that would allow the application of similar techniques. Such a question is of more than academic interest, since logic programming languages involving other logics have been proposed: a confidence factor-valued language was described in [19]; one allowing inconsistent information, with four truth values, was presented in [7], and there have been others. In an attempt to extend the so-called Kripke/Kleene semantics of [5] we found the notion of a *bilattice*, [12], to be exactly the right construct. This approach covers the specific logic programming generalizations just mentioned, and a large family of others as well. A detailed presentation will appear in [10], and further work has already appeared in [8].

The bilattice work mentioned above was part of a project to extend, and better understand, the Kripke/Kleene semantics. But the alternative well-founded semantics for conventional logic programs has considerable attractions. Many different approaches to the semantics of conventional logic programming have turned out to be equivalent to the well-founded one, at least under rather broad assumptions about programs. In this paper the well-founded approach will be extended to a richer range of logics than just classical. Again, bilattices provide the right tools. Rather than extending the original characterization of the well-founded model, we will build on the equivalent one of [21] involving "alternating fixpoints." We believe that by moving to a more general setting, not only do we gain a wider range of applicability, but proofs become more transparent and unnecessary details fall away. We note that some of the results appearing here have been obtained, in a less general setting, in [17].

There are curious parallels between much of the work on logic programming semantics and work on the philosophical theory of truth. Stratified semantics, [2], is closely related to Tarski's well-known heirarchy of languages. The Kripke/Kleene approach of [5] was motivated by Kripke's work, [13]. Its bilattice generalization, [10], has its philosophical analog, [6]. So too with the alternating fixpoint approach of [21], which was anticipated in [24]. This convergence of two disciplines makes me want to believe we are doing something right.

2 Syntax

We assume we have a fixed collection of constant, function and relation symbols. Terms and atoms are built up as usual (we allow *false* as an atom). A literal is an atom A or a negated atom $\neg A$. A clause is an expression of the form $A \leftarrow L_1, \ldots, L_n$ where A is an atom other than *false* and $L_1, \ldots, L_n$ is a possibly empty set of literals. A is the head and $L_1, \ldots, L_n$ is the body of the clause. A program is a finite set of clauses. If $\mathcal{P}$ is a program, by $\mathcal{P}^*$ we mean the infinite set consisting of all ground instances of clauses of $\mathcal{P}$, together with all clauses of the form $A \leftarrow$ *false*, where A is a ground atom

which is not otherwise the head of a clause. In the usual way we will think of a body as the conjunction of its literals, and multiple clauses in $\mathcal{P}^*$ with the same head as acting like the disjunction of their bodies. This will be explicitly built into the semantics below.

3 Bilattices

A *bilattice* is a structure with two partial orderings, each of them a complete lattice, with certain relationships postulated between the orderings, [12]. Informally the points of the bilattice are truth values and one of the orderings reflects 'degree of truth' while the other reflects 'degree of knowledge.' Some of the truth values of a bilattice may represent various states of incomplete information, while others may represent inconsistent states.

Definition 3.1 A *pre-bilattice* is a structure $\langle \mathcal{B}, \leq_t, \leq_k \rangle$ where $\mathcal{B}$ is a non-empty set (of truth values) and $\leq_t$ and $\leq_k$ are partial orderings giving $\mathcal{B}$ the structure of a complete lattice.

We use the following notation. For meet and join under $\leq_t$ we use $\wedge$ and $\vee$ in the finite case, and $\bigwedge$ and $\bigvee$ in the arbitrary case. We use $\otimes$ and $\oplus$ for finite, and $\prod$ and $\sum$ for arbitrary meet and join under $\leq_k$. We use *false* and *true* for least and greatest members under $\leq_t$, and $\bot$ and $\top$ for least and greatest members under $\leq_k$.

The operations $\wedge$ and $\vee$ are meant to generalize their classical counterparts. If we think of $\leq_k$ as representing an ordering by knowledge, then $\otimes$ is a *consensus* operator: $p \otimes q$ is the most that p and q agree on. It will play an important role here. In a similar way, $\oplus$ is an 'accept anything' operator. Since a pre-bilattice has four finitary operations, there are twelve possible finitary distributive laws, and an equal number of infinitary ones.

Definition 3.2 A *distributive bilattice* is a pre-bilattice in which all distributive laws hold.

It should be noted that the definition above requires both finitary and infinitary distributive laws to hold. A typical finitary distributive law is $x \otimes (y \wedge z) = (x \otimes y) \wedge (x \otimes z)$. A typical infinitary distributive law is $x \otimes \bigwedge\{y_i \mid i \in S\} = \bigwedge\{x \otimes y_i \mid i \in S\}$.

Belnap's four-valued logic, from [3], is a distributive bilattice, using the orderings displayed in Figure 1. In the Belnap logic there are two obvious symmetries; we introduce the following terminology for them.

Definition 3.3 Let $\langle \mathcal{B}, \leq_t, \leq_k \rangle$ be a pre-bilattice. It has a *negation* if there is a mapping $\neg$ such that:

1. $x \leq_t y \Longrightarrow \neg y \leq_t \neg x$;
2. $x \leq_k y \Longrightarrow \neg x \leq_k \neg y$;

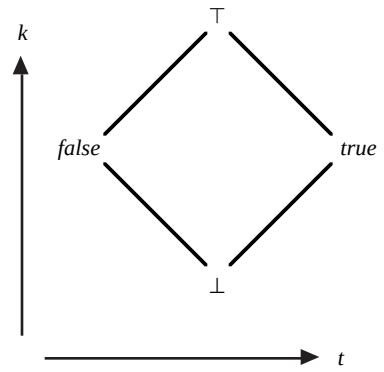

Figure 1: Belnap's 4-valued Logic

3. $\neg\neg x = x$.

The pre-bilattice has a *conflation* if there is a mapping $-$ such that:

1. $x \leq_k y \Longrightarrow -y \leq_k -x$;
2. $x \leq_t y \Longrightarrow -x \leq_t -y$;
3. $--x = x$.

The Belnap logic is an example of a bilattice with both a negation and a conflation. Another natural example is suggested by [19]. Take for truth values pairs, $\langle x, y\rangle$ where x and y are real numbers in the interval $[0, 1]$. To say a sentence p has $\langle x, y\rangle$ as its truth value intuitively means, we believe p with confidence factor x, and doubt it with confidence factor y. Belief and doubt need not add to exactly 1, and so gaps and inconsistencies can be represented. The orderings are defined as follows. $\langle x_1, y_1\rangle \leq_t \langle x_2, y_2\rangle$ if $x_1 \leq x_2$ and $y_1 \geq y_2$ (belief increased but doubt diminished); $\langle x_1, y_1\rangle \leq_k \langle x_2, y_2\rangle$ if $x_1 \leq x_2$ and $y_1 \leq y_2$ (belief and doubt both increased). We set $\neg\langle x, y\rangle = \langle y, x\rangle$; thus negation switches the roles of belief and doubt. Likewise we set $-\langle x, y\rangle = \langle 1 - y, 1 - x\rangle$; thus conflation replaces what was our degree of belief by the degree to which we did not doubt.

There are many other examples of distributive bilattices of considerable interest. Indeed, there is a general method of construction, from [12], and also discussed and extended to take negation and conflation into account, in [6], [8], [9] and [10]. We do not have space here to present it again.

If $\mathcal{B}$ is a distributive bilattice with negation, a *$\mathcal{B}$-valuation* is a mapping from ground atoms to members of the bilattice, mapping the atom *false* to the truth value *false*. Valuations are given pointwise orderings. That is, we set $v_1 \leq_t v_2$ if $v_1(A) \leq_t v_2(A)$ for every ground atom A, and similarly for $\leq_k$. It is easy to check that the space of valuations is again a distributive bilattice. We continue to use $\wedge$ and $\vee$ for finite meet and join of $\mathcal{B}$-valuations

under $\leq_t$, and so on. In fact, $(v_1 \odot v_2)(A) = v_1(A) \odot v_2(A)$ for $\odot$ any of $\wedge$, $\vee$, $\otimes$ or $\oplus$. If we also set $(\neg v)(A) = \neg(v(A))$, the space of valuations is a bilattice with negation. Similarly for conflation, if present. Valuations are extended to arbitrary formulas by setting $v(X \wedge Y) = v(X) \wedge v(Y)$, and so on. It is straightforward to check that if $v_1 \leq_k v_2$ then $v_1(X) \leq_k v_2(X)$ for every formula X, but if $v_1 \leq_t v_2$ then $v_1(X) \leq_t v_2(X)$ provided X has no negations.

The role of negation is an obvious one, but that of conflation is less so. Suppose $\mathcal{B}$ is a bilattice with a negation and a conflation, and these commute with each other. Call a member x of $\mathcal{B}$ *consistent* if $x \leq_k -x$, and *exact* if $x = -x$. In the Belnap logic, the consistent members are *false*, *true*, and $\bot$, members of a three-valued logic; the $\leq_t$ operations, restricted to the consistent members are the operations of the Kleene strong three-valued logic. The exact members are just the classical truth values. Likewise in the probabilistic system above, a member $\langle x, y\rangle$ is consistent if $x + y \leq 1$, and it is exact if $x + y = 1$. The space of truth values used in [19] can be identified with the exact subspace of this probabilistic bilattice.

In general it can be shown that the consistent members of any bilattice with negation and conflation (assuming these commute with each other) will be closed under the $\leq_t$ operations, and similarly for the exact members. Somewhat weaker closure conditions obtain for the $\leq_k$ operations (see [10]). This plays a significant role below.

Finally, we conclude with a few basic results that we will need later, and whose proofs illustrate how one reasons in bilattices. The first is behind a representation theorem in [12].

Proposition 3.1 *In a distributive bilattice,* $p = (p \wedge \bot) \oplus (p \vee \bot)$.

Proof

$$\begin{aligned}
(p \wedge \bot) \oplus (p \vee \bot) &= [p \oplus (p \vee \bot)] \wedge [\bot \oplus (p \vee \bot)] \\
&= [(p \oplus p) \vee (p \oplus \bot)] \wedge [(\bot \oplus p) \vee (\bot \oplus \bot)] \\
&= [p \vee p] \wedge [p \vee \bot] \\
&= p \wedge (p \vee \bot) \\
&= p
\end{aligned}$$

□

The next result shows the *interlacing* conditions hold in distributive bilattices. These played a fundamental role in [10].

Proposition 3.2 *In a distributive bilattice:*

1. $x \leq_k y$ *implies* $x \wedge z \leq_k y \wedge z$ *and* $x \vee z \leq_k y \vee z$;
2. $x \leq_t y$ *implies* $x \otimes z \leq_t y \otimes z$ *and* $x \oplus z \leq_k y \oplus z$.

Proof We show half of part 1. If $x \leq_k y$ then $x \otimes y = x$. Then $(x \wedge z) \otimes (y \wedge z) = (x \otimes y) \wedge z = x \wedge z$ and so $x \wedge z \leq_k y \wedge z$. □

Finally we have a result giving another connection between the two orderings, that will be of direct use later on. We begin with a preliminary lemma.

Lemma 3.3 *In a distributive bilattice, if* $a \leq_t b \leq_t c$ *then:*

1. $(a \wedge \bot) \otimes (c \vee \bot) \leq_k \bot$;
2. $(a \vee \bot) \otimes c \leq_k b$;
3. $(a \wedge \bot) \otimes (c \wedge \bot) \leq_k b$.

Proof First, $\bot \leq_k$ *false*, so by Proposition 3.2 $a \wedge \bot \leq_k a \wedge$ *false* = *false*. Similarly $c \vee \bot \leq_k$ *true*. Then, again by Proposition 3.2, $(a \wedge \bot) \otimes (c \vee \bot) \leq_k$ *false*$\otimes$*true*, and in any distributive bilattice *false*$\otimes$*true* $= \bot$. This establishes part 1. (the hypothesis of the Lemma was not used in this part).

Next, using the hypothesis and Proposition 3.2, $(a \vee \bot) \otimes c \leq_k (a \vee b) \otimes c = b \otimes c \leq_k b$.

Finally, $(a \wedge \bot) \otimes (c \wedge \bot) \leq_k (a \wedge \bot) \otimes (c \wedge b) = (a \wedge \bot) \otimes b \leq_k b$. □

Proposition 3.4 *In a distributive bilattice, if* $a \leq_t b \leq_t c$ *then* $a \otimes c \leq_k b$.

Proof We use the Lemma, Proposition 3.1 and Proposition 3.2:

$$\begin{array}{rcl} a \otimes c & = & [(a \wedge \bot) \oplus (a \vee \bot)] \otimes c \\ & = & [(a \wedge \bot) \otimes c] \oplus [(a \vee \bot) \otimes c] \\ & \leq_k & [(a \wedge \bot) \otimes c] \oplus b \\ & = & [(a \wedge \bot) \otimes ((c \wedge \bot) \oplus (c \vee \bot))] \oplus b \\ & = & [(a \wedge \bot) \otimes (c \wedge \bot)] \oplus [(a \wedge \bot) \otimes (c \vee \bot)] \oplus b \\ & \leq_k & b \oplus \bot \oplus b \\ & \leq_k & b \end{array}$$

□

4 Immediate Consequence Operators

In [20] and [1] an 'immediate consequence' operator $T_{\mathcal{P}}$ is associated with each program $\mathcal{P}$, mapping classical valuations to classical valuations. Loosely the idea is that one application of $T_{\mathcal{P}}$ represents carrying out a single step of deduction, based on the program $\mathcal{P}$. If $\mathcal{P}$ does not involve negation $T_{\mathcal{P}}$ will be monotonic (essentially in a $\leq_t$ ordering) and so will have a smallest fixed point. This is taken to be the semantic meaning of the program $\mathcal{P}$. Unfortunately, if negations are present $T_{\mathcal{P}}$ will not be monotonic, and may have no fixed points at all. In [5] an operator generally denoted $\Phi_{\mathcal{P}}$ was associated with program $\mathcal{P}$, acting on the space of *partial* valuations, or equivalently, on the Kleene strong three-valued logic, which can be identified with the consistent part of Belnap's logic. Like $T_{\mathcal{P}}$, it may not be monotonic in the $\leq_t$ ordering if negations are present, but it will always be monotonic

in the $\leq_k$ ordering, and this is enough to ensure it has a fixed point. The smallest fixed point of $\Phi_{\mathcal{P}}$ in the $\leq_k$ ordering is taken to be the semantic meaning of $\mathcal{P}$ in the so-called Kripke/Kleene approach. Since $T_{\mathcal{P}}$ is the same as $\Phi_{\mathcal{P}}$ restricted to classical valuations, it is enough here to concentrate on $\Phi_{\mathcal{P}}$. It, in turn, was extended to bilattices satisfying the interlacing conditions in [10], and it was shown that results exactly like those of the three-valued version continued to hold. It is this extension that concerns us now.

Definition 4.1 Let $\mathcal{P}$ be a program, $\mathcal{B}$ be a distributive bilattice with negation, and v be a $\mathcal{B}$-valuation. $\Phi_{\mathcal{P}}(v)$ is the $\mathcal{B}$-valuation such that, for each ground atom A,

$$\Phi_{\mathcal{P}}(v)(A) = \bigvee\{v(\bigwedge B) \mid A \leftarrow B \text{ is in } \mathcal{P}^*\}.$$

We leave it to you to check that in the distributive bilattice of valuations, $\Phi_{\mathcal{P}}$ will be monotonic in $\leq_k$, and will be monotonic in $\leq_t$ if no negations are present. Also, if $\mathcal{B}$ has a conflation operation, notions of *consistent* and of *exact* can be defined, both for $\mathcal{B}$ and for the space of $\mathcal{B}$-valuations. It is shown in [10] that both the sets of consistent, and of exact $\mathcal{B}$-valuations are closed under $\Phi_{\mathcal{P}}$.

5 The Well-Founded Extension

In this section we present an overview of our proposal to extend the well-founded model semantics. All proofs are postponed to the next two sections — indeed, all are simple and are entirely lattice theoretic in nature.

Part of the idea behind both the well-founded and the stable model semantics is to separate the roles of positive and negative information, thus avoiding the kind of symmetry that is present in the Kripke/Kleene approach. Thus we generalize the immediate consequence operator to accept two input $\mathcal{B}$-valuations, one to assign meanings to positive atoms, the other to negative atoms. This means we will sometimes want to think of a negative literal $\neg A$ as simply a funny-looking atom, with no connection to A. We use the term *$\mathcal{B}$-pseudo-valuation* for a mapping from ground *literals* to truth values in $\mathcal{B}$, a distributive bilattice with negation. Then the value a $\mathcal{B}$-pseudo-valuation assigns to $\neg A$ can be quite independent of the value it assigns to A. Pseudo-valuations are extended to conjunctions and disjunctions exactly as valuations are.

Definition 5.1 Suppose v_1 and v_2 are $\mathcal{B}$-valuations. We define a $\mathcal{B}$-pseudo-valuation $v_1 \triangle v_2$ as follows.

$$\begin{array}{lcl} (v_1 \triangle v_2)(A) & = & v_1(A) \\ (v_1 \triangle v_2)(\neg A) & = & \neg v_2(A) \end{array}$$

Thus in $v_1 \triangle v_2$, v_1 supplies the positive, and v_2 the negative information.

Definition 5.2 Let $\mathcal{P}$ be a program, and v_1 and v_2 be $\mathcal{B}$-valuations. We take $\Psi_{\mathcal{P}}(v_1, v_2)$ to be the $\mathcal{B}$-valuation such that, for each ground atom A,

$$\Psi_{\mathcal{P}}(v_1, v_2)(A) = \bigvee\{(v_1 \triangle v_2)(\bigwedge B) \mid A \leftarrow B \text{ is in } \mathcal{P}^*\}.$$

Thus in $\Psi_{\mathcal{P}}$ positive and negative input has been split apart. It is easy to see that $\Phi_{\mathcal{P}}(v) = \Psi_{\mathcal{P}}(v, v)$. Also, for those distributive bilattices with negation and conflation, if both v_1 and v_2 are exact, so is $\Psi_{\mathcal{P}}(v_1, v_2)$, and similarly for consistent.

Definition 5.3 A mapping f on a partially ordered space is *monotonic* if $x \leq y$ implies $f(x) \leq f(y)$. A mapping f is *anti-monotonic* if $x \leq y$ implies $f(y) \leq f(x)$.

Under the ordering $\leq_k$, $\Psi_{\mathcal{P}}(v_1, v_2)$ is monotonic in both inputs, v_1 and v_2. But under the $\leq_t$ ordering, $\Psi_{\mathcal{P}}(v_1, v_2)$ is monotonic in v_1 and anti-monotonic in v_2. Then, if we hold v_2 fixed, we have a monotonic operator in v_1, which will have a smallest fixed point. Thus we can define a *derived* operator, called a *stability* operator in [21].

Definition 5.4 The *derived operator* of $\Psi_{\mathcal{P}}$ is the single input mapping $\Psi'_{\mathcal{P}}$ given by: $\Psi'_{\mathcal{P}}(v)$ is the smallest fixed point, in the $\leq_t$ ordering, of the mapping $(\lambda x)\Psi_{\mathcal{P}}(x, v)$.

We will show the derived operator $\Psi'_{\mathcal{P}}$ is anti-monotonic. As such it may have no fixed points, or one, or many. If $\mathcal{B}$ is the Belnap four-valued logic, the exact (= classical) fixed points of $\Psi'_{\mathcal{P}}$, if any, are exactly the stable, or felicitous, models of [11] and [4]. If there is a unique classical fixed point, $\mathcal{P}$ is said to have a stable model semantics, but such a unique classical fixed point may not exist. We will show in Section 7 that for any program $\mathcal{P}$ and any distributive bilattice $\mathcal{B}$ there are always two $\mathcal{B}$-valuations, $\mu_{\mathcal{P}}$ and $\nu_{\mathcal{P}}$, with $\mu_{\mathcal{P}} \leq_t \nu_{\mathcal{P}}$, between which the derived operator oscillates. If the bilattice $\mathcal{B}$ has a conflation operation, and hence a notion of exactness, both $\mu_{\mathcal{P}}$ and $\nu_{\mathcal{P}}$ will be exact. Furthermore, these two $\mathcal{B}$-valuations are extreme, in the sense that if there are any other pairs between which the derived operator oscillates, they must be between $\mu_{\mathcal{P}}$ and $\nu_{\mathcal{P}}$. One can approximate to $\mu_{\mathcal{P}}$ and $\nu_{\mathcal{P}}$ in a manner similar to that by which one approximates to the least fixed point of a monotone mapping. In effect, these two extreme values constitute under and over estimates for a $\mathcal{B}$-valuation. We propose taking the consensus of these extreme values, as a natural model for the program.

Definition 5.5 Let $\mathcal{P}$ be a program, and let $\mu_{\mathcal{P}}$ and $\nu_{\mathcal{P}}$ be the two extreme $\mathcal{B}$-valuations between which $\Psi'_{\mathcal{P}}$ oscillates. By the *extended well-founded model* for $\mathcal{P}$ we mean $\mu_{\mathcal{P}} \otimes \nu_{\mathcal{P}}$.

If $\mathcal{B}$ is the Belnap four-valued logic, the extended well-founded model coincides with the alternating fixpoint model of [21], and hence with the well-founded model. We will show in Section 7 that for any choice of $\mathcal{B}$, $\mu_{\mathcal{P}} \otimes \nu_{\mathcal{P}}$ is a model for program $\mathcal{P}$ in an appropriate sense. We will also show that $\mu_{\mathcal{P}} \otimes \nu_{\mathcal{P}}$ is below any fixed point of $\Psi'_{\mathcal{P}}$, in the $\leq_k$ ordering, and hence contains no more knowledge than any stable model of $\mathcal{P}$.

6 Monotonic and Anti-monotonic Operators

In this section we begin the presentation of proofs of the assertions made in the previous section. By the Knaster-Tarski theorem, [18], a monotonic mapping on a complete lattice has a smallest and a biggest fixed point. This is a standard result, but similar results concerning anti-monotonic operators are less well known. In order to prepare for them we briefly sketch the proof of the Knaster-Tarski theorem as background.

Suppose $\langle \mathbf{L}, \leq \rangle$ is a complete lattice (infinite as well as finite meets and joins exist). And suppose $M : \mathbf{L} \to \mathbf{L}$ is *monotonic*. Set $\mathbf{A} = \{x \in \mathbf{L} \mid M(x) \leq x\}$ and set $\mathbf{B} = \{x \in \mathbf{L} \mid x \leq M(x)\}$. Let $\mu = \bigwedge \mathbf{A}$ and $\nu = \bigvee \mathbf{B}$. Then μ is the smallest, and ν the biggest fixed point of M, by the following argument.

1. If $x \in \mathbf{A}$ then $M(x) \leq x$ so by monotonicity $M^2(x) \leq M(x)$, and so $M(x) \in \mathbf{A}$. In a similar way $\mathbf{B}$ is closed under M.

2. Suppose $x \in \mathbf{A}$. Then $\mu \leq x$, so $M(\mu) \leq M(x) \leq x$. Since x was an arbitrary member of $\mathbf{A}$, $M(\mu) \leq \mu$. Then $\mu \in \mathbf{A}$. In a similar way, $\nu \in \mathbf{B}$.

3. Since $\mu \in \mathbf{A}$, $M(\mu) \in \mathbf{A}$, hence $\mu \leq M(\mu)$. Then $M(\mu) = \mu$. Also since every fixed point of M is in $\mathbf{A}$, and μ is the greatest lower bound of $\mathbf{A}$, μ must be the *smallest* fixed point. By a similar argument, ν is the *biggest* fixed point.

Incidentally, the argument shows the stronger fact that not only is μ least among fixed points, but if $M(x) \leq x$, $\mu \leq x$.

A typical application of the Knaster-Tarski result in logic programming is to provide a semantics for programs without negations. Customarily these are produced in the space of *classical* valuations (ordered by $\leq_t$), but it might be wondered whether allowing *partial* valuations could change things. In fact, it will not. This is a result that extends to arbitrary bilattices with negation and conflation: the least fixed point, under the $\leq_t$ ordering, of the immediate consequence operator for a program without negations must be exact. This follows from the facts that the exact members must be closed under $\bigwedge$ and $\bigvee$, *false* and *true* must be exact, and the following argument.

Suppose the complete lattice $\mathbf{L}$ has a complete sublattice $\mathbf{L}_0$ containing the least and greatest members of $\mathbf{L}$, and that meets and joins of subsets

of $\mathbf{L}_0$ are the same, whether evaluated in $\mathbf{L}_0$ or in $\mathbf{L}$. And suppose M is a monotonic mapping on $\mathbf{L}$, such that $\mathbf{L}_0$ is closed under M. Then the least fixed point of M is the same in $\mathbf{L}$ and in $\mathbf{L}_0$. (Similarly for the greatest fixed point.) A proof of this can be based on the well-known way of approximating to the least fixed point of a monotonic mapping by beginning with the least member of the lattice, iterating applications of the mapping, and taking sups at limit stages. By our assumptions, the least member of $\mathbf{L}$ is the same as the least member of $\mathbf{L}_0$. Then, again by our assumptions, the sequence of approximations to the least fixed point of M will be the same whether calculated in $\mathbf{L}$ or in $\mathbf{L}_0$, which is enough to establish the claim.

Now we turn to the anti-monotonic version; suppose $N : \mathbf{L} \to \mathbf{L}$ is *anti-monotonic*. We will show N has two points, one below the other, between which it oscillates. Further, we will show that if there is another pair of points between which N oscillates, they must lie between these.

Since N is anti-monotonic, N^2 is monotonic. Set $\mathbf{A} = \{x \in \mathbf{L} \mid N^2(x) \leq x\}$, and $\mathbf{B} = \{x \in \mathbf{L} \mid x \leq N^2(x)\}$, and set $\mu = \bigwedge \mathbf{A}$ and $\nu = \bigvee \mathbf{B}$. By the argument above, μ and ν are the smallest and biggest fixed points of N^2. Then $\mu \leq \nu$, of course. Also, if N oscillates between some other pair of points, c and d say, both c and d will be fixed points of N^2, and hence will come between μ and ν. Finally, that μ and ν are interchanged by N is shown as follows.

1. If $x \in \mathbf{A}$ then $N^2(x) \leq x$ so by anti-monotonicity $N(x) \leq N^3(x)$, and so $N(x) \in \mathbf{B}$. Similarly, if $x \in \mathbf{B}$ then $N(x) \in \mathbf{A}$.

2. Exactly as in the Knaster-Tarski argument, $\mu \in \mathbf{A}$ and $\nu \in \mathbf{B}$. Then $N(\mu) \in \mathbf{B}$ and so $N(\mu) \leq \nu$. Likewise $N(\nu) \in \mathbf{A}$, so $\mu \leq N(\nu)$.

3. Since $\mu \leq N(\nu)$, by anti-monotonicity $N^2(\nu) \leq N(\mu)$. But ν is a fixed point of N^2, so we have $\nu \leq N(\mu)$, and so $N(\mu) = \nu$. Similarly $N(\nu) = \mu$.

We mentioned above that the smallest fixed point of a monotone operator can be approximated to by beginning with the least element of $\mathbf{L}$ and continually iterating the operator, taking sups at limit stages. In a similar way, such an iteration process can be carried out for an anti-monotonic operator, producing a sequence with two subsequences, one of which approximates from below to the point μ, the other of which approximates from above to the point ν. Inferior and superior limits take the place of simple limits in the monotonic case. This is the approach taken in [24]. Finally, above we showed the least fixed point of a monotonic operator would be the same in a lattice as in a sublattice with the same meets and joins. Since the oscillation points of an anti-monotonic operator are the extreme fixed points of the square of the operator, a similar result applies to them.

7 Proofs of Basic Results

In this section we present proofs of the results that were stated in Section 5. Since the arguments are largely lattice-theoretic in nature, we abstract slightly to clear away useless details. Assume, for the rest of this section, that $\mathcal{B}$ is a distributive bilattice with negation, and that $\Psi(v_1, v_2)$ maps $\mathcal{B}$-valuations to $\mathcal{B}$-valuations, and meets the following conditions:

1. under the ordering $\leq_t$, $\Psi(v_1, v_2)$ is monotonic in v_1 and anti-monotonic in v_2;
2. under the ordering $\leq_k$, $\Psi(v_1, v_2)$ is monotonic in both arguments;
3. if there is a notion of exactness, then if v_1 and v_2 are both exact, so is $\Psi(v_1, v_2)$;

We set $\Phi(x) = \Psi(x, x)$. Also we set $\Psi'(v)$ to be the smallest fixed point, in the $\leq_t$ ordering, of the mapping $(\lambda x)\Psi(x, v)$. By item 3. and remarks in the previous Section, if v is exact, the least fixed point of $(\lambda x)\Psi(x, v)$ will also be exact, and will be the same whether evaluated in the space of all valuations, or just in the space of exact valuations. Then $\Psi'(v)$ will be exact if v is exact. Also, by definition,

$$\Psi(\Psi'(v), v) = \Psi'(v)$$

$$\Psi(x, v) = x \Longrightarrow \Psi'(v) \leq_t x$$

and further, from the proof of the Knaster-Tarski result,

$$\Psi(x, v) \leq_t x \Longrightarrow \Psi'(v) \leq_t x.$$

Proposition 7.1 *Ψ' is anti-monotonic.*

Proof Suppose $v_1 \leq_t v_2$. Then for each x, $\Psi(x, v_2) \leq_t \Psi(x, v_1)$. It follows from this, and the facts above, that $\Psi(\Psi'(v_1), v_2) \leq_t \Psi(\Psi'(v_1), v_1) = \Psi'(v_1)$, and so $\Psi'(v_2) \leq_t \Psi'(v_1)$. □

Using Proposition 7.1, Ψ' has two extreme values between which it oscillates. If there is a notion of exactness, these will be exact $\mathcal{B}$-valuations and, by results in the previous Section, will be the same whether evaluated in the space of all $\mathcal{B}$-valuations or only of exact $\mathcal{B}$-valuations. For the rest of this section we use μ and ν for these two values, assuming $\mu \leq_t \nu$. Then, $\Psi'(\mu) = \nu$, and $\Psi'(\nu) = \mu$. Note the following simple equations: $\Psi(\mu, \nu) = \Psi(\Psi'(\nu), \nu) = \Psi'(\nu) = \mu$, and similarly $\Psi(\nu, \mu) = \nu$.

We have not yet defined the notion of a *model* for a logic program; we do so now. The idea is simple: require program heads to be at least as true as bodies.

Definition 7.1 A $\mathcal{B}$-valuation v is a *model* for a program $\mathcal{P}$ provided, for each clause $A \leftarrow B$ in $\mathcal{P}^*$, $v(A) \geq_t v(\bigwedge B)$.

It is straightforward to check that v will be a model for a program $\mathcal{P}$ if and only if $\Phi_{\mathcal{P}}(v) \leq_t v$. We will show $\mu_{\mathcal{P}} \otimes \nu_{\mathcal{P}}$ is a model for $\mathcal{P}$ by showing the stronger fact that $\mu_{\mathcal{P}} \otimes \nu_{\mathcal{P}}$ is a *fixed point* for $\Phi_{\mathcal{P}}$. In proving this, the main role will be played by the ordering $\leq_k$, rather than by $\leq_t$.

Proposition 7.2 $\Phi(\mu \otimes \nu) \leq_k \mu \otimes \nu$.

Proof $\mu \otimes \nu \leq_k \mu$ and $\mu \otimes \nu \leq_k \nu$, so $\Psi(\mu \otimes \nu, \mu \otimes \nu) \leq_k \Psi(\nu, \mu) = \nu$ and $\Psi(\mu \otimes \nu, \mu \otimes \nu) \leq_k \Psi(\mu, \nu) = \mu$. It follows that $\Phi(\mu \otimes \nu) = \Psi(\mu \otimes \nu, \mu \otimes \nu) \leq_k \mu \otimes \nu$. □

Thus we have half of the proof that $\mu \otimes \nu$ is a fixed point. The other half makes use of general results about distributive bilattices.

Proposition 7.3 $\mu \otimes \nu \leq_k \Phi(\mu \otimes \nu)$.

Proof Since $\mu \leq_t \nu$, by Proposition 3.2, $\mu = \mu \otimes \mu \leq_t \mu \otimes \nu \leq_t \nu \otimes \nu = \nu$. Then $\mu = \Psi(\mu, \nu) \leq_t \Psi(\mu \otimes \nu, \mu \otimes \nu) = \Phi(\mu \otimes \nu)$, and $\nu = \Psi(\nu, \mu) \geq_t \Psi(\mu \otimes \nu, \mu \otimes \nu) = \Phi(\mu \otimes \nu)$.

Since $\mu \leq_t \Phi(\mu \otimes \nu) \leq_t \nu$, by Proposition 3.4, $\mu \otimes \nu \leq_k \Phi(\mu \otimes \nu)$. □

Proposition 7.4 *$\mu \otimes \nu$ is a fixed point of Φ.*

Proof By Proposition 7.2 and Proposition 7.3. □

Finally, since every fixed point of Ψ' must lie between μ and ν in the $\leq_t$ ordering, Proposition 3.4 immediately gives us the following, which says $\mu \otimes \nu$ is a lower bound, in the $\leq_k$ ordering, for stable models.

Proposition 7.5 *If v is a fixed point of Ψ' then $\mu \otimes \nu \leq_k v$.*

Acknowledgements

This research was supported in part by NSF Grant CCR-8901489.

References

[1] Krzystof R. Apt and Maartin van Emden. Contributions to the theory of logic programming. *Journal of the Assoc. for Comp. Mach.*, 29:841–862, 1982.

[2] Krzystoff R. Apt, Howard Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–142. Morgan Kaufmann, 1988.

[3] Nuel D. Belnap, Jr. A useful four-valued logic. In Jon Michael Dunn and George Epstein, editors, *Modern Uses of Multiple-Valued Logic*. D. Reidel, 1977.

[4] Kit Fine. The justification of negation as failure. In Jens Eric Fenstad, editor, *Logic, Methodology and Philosophy of Science VIII*, pages 263–301. Elsevier, 1989.

[5] Melvin C. Fitting. A Kripke/Kleene semantics for logic programs. *Journal of Logic Programming*, pages 295–312, 1985.

[6] Melvin C. Fitting. Bilattices and the theory of truth. *Journal of Philosophical Logic*, 18:225–256, 1989.

[7] Melvin C. Fitting. Negation as refutation. In Rohit Parikh, editor, *Proceedings of the Fourth Annual Symposium on Logic in Computer Science*, pages 63–70. IEEE, 1989.

[8] Melvin C. Fitting. Bilattices in logic programming. In George Epstein, editor, *The Twentieth International Symposium on Multiple-Valued Logic*, pages 238–246. IEEE, 1990.

[9] Melvin C. Fitting. Kleene's three-valued logics and their children. In *Proceedings of the Bulgarian Kleene 90 Conference*, 1990. Forthcoming.

[10] Melvin C. Fitting. Bilattices and the semantics of logic programming. *Journal of Logic Programming*, 1991. Forthcoming.

[11] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert Kowalski and Ken Bowen, editors, *Proc. of the Fifth Logic Programming Symposium*, pages 1070–1080, 1988.

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

[13] Saul Kripke. Outline of a theory of truth. *The Journal of Philosophy*, 72:690–716, 1975. Reprinted in *New Essays on Truth and the Liar Paradox*, R. L. Martin, ed., Oxford (1983).

[14] Kenneth Kunen. Negation in logic programming. *Journal of Logic Programming*, 4:289–308, 1987.

[15] Kenneth Kunen. Some remarks on the completed database. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming, Proc. of the Fifth Intl. Conf. and Symp.*, pages 978–992. The MIT Press, 1988.

[16] Teodor C. Przymusinski. Stationary semantics for disjunctive logic programs and deductive databases. In Saumya Debray and Manuel Hermenegildo, editors, *Logic Programming, Proc. of the 1990 North American Conf.*, pages 40–59. The MIT Press, 1990.

[17] Teodor C. Przymusinski and Halina Przymusinska. Semantic issues in deductive databases. In *Formal Techniques in Artificial Intelligence*. North-Holland, 1990.

[18] Alfred Tarski. A lattice-theoretical theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[19] Maartin van Emden. Quantitative deduction and its fixpoint theory. *Journal of Logic Programming*, 3:37–53, 1986.

[20] Maartin van Emden and Robert Kowalski. The semantics of predicate logic as a programming language. *Journal of the Assoc. for Comp. Mach.*, 23:733–742, 1976.

[21] Allen Van Gelder. The alternating fixpoint of logic programs with negation. In *Proc. 8th ACM Symp. on Principles of Database Systems*, pages 1–10, Philadelphia, 1989. ACM.

[22] Allen Van Gelder, Kenneth A. Ross, and John S. Schlipf. Unfounded sets and well-founded semantics for general logic programs. In *Proc. Seventh Symp. on Principles of Database Systems*, pages 221–230, 1988.

[23] Allen Van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *JACM*, to appear.

[24] Stephen Yablo. Truth and reflection. *Journal of Philosophical Logic*, 14:297–349, 1985.