Logic Programs under Three-Valued Łukasiewicz Semantics

Steffen Hölldobler and Carroline Dewi Puspa Kencana Ramli

International Center for Computational Logic,
TU Dresden, 01062 Dresden, Germany
sh@iccl.tu-dresden.de
http://www.computational-logic.org/~sh/

Abstract. If logic programs are interpreted over a three-valued logic, then often Kleene's strong three-valued logic with complete equivalence and Fitting's associated immediate consequence operator is used. However, in such a logic the least fixed point of the Fitting operator is not necessarily a model for the program under consideration. Moreover, the model intersection property does not hold. In this paper, we consider the three-valued Łukasiewicz semantics and show that fixed points of the Fitting operator are also models for the program under consideration and that the model intersection property holds. Moreover, we review a slightly different immediate consequence operator first introduced by Stenning and van Lambalgen and relate it to the Fitting operator under Łukasiewicz semantics. Some examples are discussed to support the claim that Łukasiewicz semantics and the Stenning and van Lambalgen operator is better suited to model commonsense and human reasoning.

Keywords: Three Valued Logic Programs, Łukasiewicz Semantics.

1 Introduction

When interpreting logic programs (with negation) under a three-valued semantics, then it appears that with some exceptions (see e.g. [10]) mainly the semantics defined by Fitting in [7] is considered (see e.g. [1]) in the logic programming literature up to now. This semantics combines Kleene's strong three-valued logic for negation, conjunction, disjunction and implication with complete equivalence, which was also introduced by Kleene (see [13]). Complete equivalence was used by Fitting to ensure that a formula of the form $F \leftrightarrow F$ is mapped to true under an interpretation, which maps F to neither true nor false (see [7], p.300). Under the Fitting semantics, the law of equivalence ($F \leftrightarrow G$ is semantically equivalent to $(F \leftarrow G) \wedge (G \leftarrow F)$) does not hold anymore. This is somewhat surprising as Fitting suggests a completion-based approach ([5]), where the if-halves of the definitions in a logic program are completed by adding their corresponding only-if-halves. Under the Fitting semantics, a completed definition $p \leftrightarrow q$ may be mapped to true under an interpretation, which maps neither $p \leftarrow q$ nor $q \leftarrow p$ to true.

P.M. Hill and D.S. Warren (Eds.): ICLP 2009, LNCS 5649, pp. 464–478, 2009.

The Fitting semantics was also considered in a recent book by Stenning and van Lambalgen [18], where they argue in favor of a completion-based logic-programming approach to model human reasoning. Stenning and van Lambalgen introduce an immediate consequence operator, which is slightly different from the one defined by Fitting in [7], and claim that for a given propositional logic program the least fixed point of this operator is the minimal model of the program (Lemma 4(1.) in [18]). Looking into this result we found that the least fixed point may not even be a model for the program (see [12]) and that this stems from the fact that the Fitting semantics does not admit the law of equivalence.

From these observations two questions arose: Why did Fitting combine Kleene's strong three-valued logic with complete equivalence? Is there an alternative semantics under which the results proven in [7] hold and which admits also the law of equivalence?

We can answer the former question only partially: questions of computability[1] and, in particular, termination[2] may have been the driving force. As for the latter, we believe that the Łukasiewicz semantics [15] may be a good candidate.

After reviewing three-valued logics in Section 2 and stating some preliminaries in Section 3 we investigate Fitting's immediate consequence operator in Section 4. In particular, we show that under the Łukasiewicz semantics, a fixed point of the Fitting operator is not only a model for the completion of a given program, but for the program itself. Moreover, we show that the model intersection property holds for logic programs (with negation) under the Łukasiewicz semantics.

In Section 5 we review Stenning and van Lambalgen's immediate consequence operator under Łukasiewicz semantics. The main difference between the Fitting and the Stenning and van Lambalgen operator is the observation that whereas Fitting assumes all undefined predicates to be false within the completion process, Stenning and van Lambalgen allow the user to control which otherwise undefined predicates shall be mapped to false. In order to do so, they introduce so-called negative facts and modify the notion of completion accordingly. In Section 6 we present two examples from commonsense and human reasoning to support the claim that the Stenning and van Lambalgen operator may be better suited for these reasoning tasks than the Fitting operator. In the final Section 7 we summarize our findings and point to some future and related work.

2 Three-Valued Logics

In 1920, the Polish philosopher Łukasiewicz introduced the first three-valued logic [15]. The truth values are not only true or false, but there exists a third, intermediate value. A formula is allowed to be neither true nor false. We can interpret the intermediate truth value as possibility: the truth value is not decided yet but possibly decided at some later time. In this paper, we symbolize truth- and falsehood by $\top$ and $\bot$, respectively. We call the third truth value *undecided* and use the symbol u to denote it.

Łukasiewicz used the following principles and definitions to assign values to formulas, where $\equiv$ denotes semantic equivalence:

[1] Personal communication with Melvin Fitting.

[2] Personal communication with Pascal Hitzler.

Table 1. A truth table for three-valued logics. The indices K and Ł refer to Kleene's and Łukasiewicz's logic, respectively. $\leftrightarrow_C$ denotes the complete equivalence used by Fitting.

F	G	$\neg F$	$F \wedge G$	$F \vee G$	$F \leftarrow_K G$	$F \leftrightarrow_K G$	$F \leftrightarrow_C G$	$F \leftarrow_Ł G$	$F \leftrightarrow_Ł G$
$\top$	$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\bot$	$\bot$	$\top$	$\bot$
$\top$	u	$\bot$	u	$\top$	$\top$	u	$\bot$	$\top$	u
$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
$\bot$	$\bot$	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\top$	$\top$	$\top$
$\bot$	u	$\top$	$\bot$	u	u	u	$\bot$	u	u
u	$\top$	u	u	$\top$	u	u	$\bot$	u	u
u	$\bot$	u	$\bot$	u	$\top$	u	$\bot$	$\top$	u
u	u	u	u	u	u	u	$\top$	$\top$	$\top$

Table 2. Some common laws under Łukasiewicz, Kleene and Fitting semantics

Laws		Łukasiewicz	Kleene	Fitting
Equivalence	$F \leftrightarrow G \equiv (F \rightarrow G) \wedge (G \rightarrow F)$	Yes	Yes	No
Implication	$F \rightarrow G \equiv \neg F \vee G$	No	Yes	Yes
Syllogism	$(F \rightarrow G) \wedge (G \rightarrow H) \equiv F \rightarrow H$	No	Yes	Yes
Excluded Middle	$F \vee \neg F \equiv \top$	No	No	No
Contradiction	$F \wedge \neg F \equiv \bot$	No	No	No

1. The principles of identity and non-identity: $(\bot \leftrightarrow \bot) \equiv (\top \leftrightarrow \top) \equiv \top$, $(\top \leftrightarrow \bot) \equiv (\bot \leftrightarrow \top) \equiv \bot$,
   $(\bot \leftrightarrow u) \equiv (u \leftrightarrow \bot) \equiv (\top \leftrightarrow u) \equiv (u \leftrightarrow \top) \equiv u$, $(u \leftrightarrow u) \equiv \top$.
2. The principles of implication:
   $(\bot \leftarrow \bot) \equiv (\top \leftarrow \bot) \equiv (\top \leftarrow \top) \equiv \top$, $(\bot \leftarrow \top) \equiv \bot$,
   $(u \leftarrow \bot) \equiv (\top \leftarrow u) \equiv (u \leftarrow u) \equiv \top$, $(\bot \leftarrow u) \equiv (u \leftarrow \top) \equiv u$.
3. The definitions of negation, disjunction and conjunction:
   $\neg A \equiv (\bot \leftarrow A)$, $A \vee B \equiv (B \leftarrow (B \leftarrow A))$, $A \wedge B \equiv \neg(\neg A \vee \neg B)$.

Later, in 1952, Kleene proposed an alternative three-valued logic with the truth values true, false, and undefined. He distinguishes between weak and strong three-valued logics. For our paper only the latter is of interest. It is similar to the Łukasiewicz logic, but differs in the semantics of implication and equivalence, viz., $u \leftrightarrow u \equiv u$ and $u \leftarrow u \equiv u$. Kleene also introduced a *complete equivalence* where $(F \leftrightarrow G) \equiv \top$ if and only if both F and G have the same logical value, else $(F \leftrightarrow G) \equiv \bot$.

The semantics of the connectives is summarized in Table 1. In the Łukasiewicz logic [15] the set of connectives is $\{\neg, \wedge, \vee, \leftarrow_Ł, \leftrightarrow_Ł\}$, in Kleene's strong three-valued logic [13] the set of connectives is $\{\neg, \wedge, \vee, \leftarrow_K, \leftrightarrow_K\}$, and in the Fitting logic [7] the set of connectives is $\{\neg, \wedge, \vee, \leftarrow_K, \leftrightarrow_C\}$. Table 2 gives an overview over some common laws which do not always hold with respect to the Łukasiewicz, Kleene and Fitting logics considered in this paper. Other laws like impotency, commutativity, associativity, absorption, distributivity, double negation, de Morgan and contraposition hold under Kleene, Łukasiewicz and Fitting logics.

3 Preliminaries

In this section we recall some notations and terminologies based on [14].

3.1 First-Order Language

We consider an *alphabet* consisting of (finite or countably infinite) disjoint sets of variables, constants, function symbols, predicate symbols, connectives $\{\neg,\ \vee,\ \wedge,\ \leftarrow,\ \leftrightarrow\}$, quantifiers $\{\forall,\ \exists\}$, and punctuation symbols {"(", ",", ")"}. In this paper we will use upper case letters to denote variables and lower case letters to denote constants, function- and predicate symbols. Terms, atoms, literals and formulas are defined as usual. To avoid having formulas cluttered with brackets, we adopt the following precedence hierarchy to order the connectives: $\neg\ >\ \{\vee,\ \wedge\}\ > \leftarrow\ > \leftrightarrow$. The *language* given by an alphabet consists of the set of all formulas constructed from the symbols occurring in the alphabet. A *sentence* is a formula without free variables. Finally, we extend our language by the symbols $\top$ and $\bot$ denoting a valid and an unsatisfiable formula, respectively.

3.2 Logic Programs

A *(program) clause* is an expression of the form $A \leftarrow B_1 \wedge \cdots \wedge B_n$, where $n \geq 1$, A is an atom, and each $B_i,\ 1 \leq i \leq n$, is either a literal (i.e., atom or negated atom) or $\top$. A is called *head* and $B_1 \wedge \cdots \wedge\ B_n$ *body* of the clause. One should note that the body of a clause must not be empty. A clause of the form $A \leftarrow \top$ is called a *positive fact*.

A *(logic) program* is a finite set of clauses. *ground*$(\mathcal{P})$ denotes the set of all ground instances of the program $\mathcal{P}$. In many cases, *ground*$(\mathcal{P})$ is infinite, but for propositional or datalog programs *ground*$(\mathcal{P})$ is finite. In the sequel we will consider *ground*$(\mathcal{P})$ as a substitute for $\mathcal{P}$, thus ignoring unification issues.

We assume that each non-propositional program contains at least one constant symbol. Moreover, the language $\mathcal{L}$ underlying a program $\mathcal{P}$ shall contain precisely the relation, function and constant symbols occurring in $\mathcal{P}$, and no others.

3.3 Interpretations and Models

The declarative semantics of a logic program is given by a model-theoretic semantics of formulas in the underlying language. We represent interpretations by pairs $\langle I^\top, I^\bot \rangle$, where the set $I^\top$ contains all atoms which are mapped to $\top$, the set $I^\bot$ contains all atoms which are mapped to $\bot$, and $I^\top \cap I^\bot = \emptyset$. All atoms which occur neither in $I^\top$ nor $I^\bot$ are mapped to u. The logical value of formulas can be derived from Table 1 as usual. We use $I_Ł,\ I_K$ and I_F to denote that an interpretation I uses the Łukasiewicz, Kleene or Fitting semantics, respectively. let $\mathcal{I}$ denote the set of all interpretations. One should observe that $(\mathcal{I}, \subseteq)$ is a complete semi-lattice (see [7]).

Let I be an interpretation of a language $\mathcal{L}$ and let F be a sentence of $\mathcal{L}$. I is a *model* for F if F is true with respect to I (i.e., $I(F) = \top$). Let $\mathcal{S}$ be a set of sentences of a language $\mathcal{L}$ and let I be an interpretation of $\mathcal{L}$. We say I is a *model* for $\mathcal{S}$ if I is a model for each sentence of $\mathcal{S}$. Two sentences F and G are said to be *semantically equivalent* if and only if both have same truth value under all interpretations.

3.4 Program Completion

Let $ground(\mathcal{P})$ be a logic program. Consider the following transformation:

1. All clauses with the same head (ground atom) $A \leftarrow Body_1,\ A \leftarrow Body_2, \ldots$ are replaced by the single expression $A \leftarrow Body_1 \vee Body_2 \vee \ldots$.
2. If a ground atom A is not the head of any clause in $ground(\mathcal{P})$ then add $A \leftarrow \bot$, where $\bot$ denotes an unsatisfiable formula.
3. All occurrences of $\leftarrow$ are replaced by $\leftrightarrow$.

The resulting set of formulas is called *completion of* $ground(\mathcal{P})$ and is denoted by $comp(ground(\mathcal{P}))$. One should observe that in step 1 there may be infinitely many clauses with the same head resulting in a countable disjunction. However, its semantic behavior is unproblematic.

4 The Fitting Operator

In this section we will discuss Fitting's immediate consequence operator [7] under the Łukasiewicz semantics. We will show that replacing the Fitting semantics with the Łukasiewicz semantics does not change the behaviors of the Fitting operator. But in addition each model of the completion of a program coincides with a model of the program itself.

Let I be an interpretation and $\mathcal{P}$ a program. *Fitting's immediate consequence operator* is defined as follows: $\Phi_{F,\mathcal{P}}(I) = \langle J^\top, J^\bot \rangle$, where

$$J^\top = \{A \mid \text{there exists } A \leftarrow Body \in ground(\mathcal{P}) \text{ with } I(Body) = \top\} \text{ and}$$
$$J^\bot = \{A \mid \text{for all } A \leftarrow Body \in ground(\mathcal{P}) \text{ we find } I(Body) = \bot\}.$$

Please recall that the body of the program is a conjunction of literals and, hence, $I_Ł(Body) = I_K(Body) = I_F(Body)$ according to Table 1.

Fitting shows in [7] that $\Phi_{F,\mathcal{P}}$ is monotone on $(\mathcal{I}, \subseteq)$. Moreover, from [19] and [16] follows that for finite $ground(\mathcal{P})$ the operator $\Phi_{F,\mathcal{P}}$ is also continuous. We call a program $\mathcal{P}$ *F-acceptable* if $\Phi_{F,\mathcal{P}}$ is continuous.

Given a program $\mathcal{P}$. An interpretation I is said to be *fixed point of* $\Phi_{F,\mathcal{P}}$ iff $I = \Phi_{F,\mathcal{P}}(I)$. If $\Phi_{F,\mathcal{P}}$ is continuous, then it admits a least fixed point denoted by $lfp(\Phi_{F,\mathcal{P}})$. It can be computed by iterating $\Phi_{F,\mathcal{P}}$ starting with the empty interpretation as follows, where ω is an arbitrary limit ordinal:

$$\begin{aligned} \Phi_{F,\mathcal{P}} \uparrow_0 &= \langle \emptyset, \emptyset \rangle, \\ \Phi_{F,\mathcal{P}} \uparrow_{(\alpha+1)} &= \Phi_{F,\mathcal{P}}(\Phi_{F,\mathcal{P}} \uparrow_\alpha), \\ \Phi_{F,\mathcal{P}} \uparrow_\omega &= \bigcup \{\Phi_{F,\mathcal{P}} \uparrow_\alpha \mid \alpha < \omega\}. \end{aligned}$$

As examples consider the programs $\mathcal{P}_1 = ground(\mathcal{P}_1) = \{p \leftarrow q\}$ and $\mathcal{P}_2 = ground(\mathcal{P}_2) = \{p \leftarrow q,\ q \leftarrow p\}$. Their completions are $comp(ground(\mathcal{P}_1)) = \{p \leftrightarrow q,\ q \leftrightarrow \bot\}$ and $comp(ground(\mathcal{P}_2)) = \{p \leftrightarrow q,\ q \leftrightarrow p\}$. In both cases, the Fitting operator is continuous and we obtain the least fixed points $lfp(\Phi_{F,\mathcal{P}_1}) = \langle \emptyset, \{p, q\} \rangle$ and $lfp(\Phi_{F,\mathcal{P}_2}) = \langle \emptyset, \emptyset \rangle$. It is easy to verify that the least fixed points are models of the completions under the Fitting semantics, which is no coincidence as formally proven in [7]. This property holds also under the Łukasiewicz semantics.

Proposition 1. *Let $\mathcal{P}$ be a program.*

1. *$I_Ł$ is a fixed point of $\Phi_{F,\mathcal{P}}$ iff $I_Ł$ is a model of* $\text{comp}(\text{ground}(\mathcal{P}))$.
2. *If $I_Ł = lfp(\Phi_{F,\mathcal{P}})$ then $I_Ł$ is the least model of* $\text{comp}(\text{ground}(\mathcal{P}))$.

Proof. 1. To show the if-part, suppose I is a fixed point of $\Phi_{F,\mathcal{P}}$. As shown in [7], in this case I is a model of $comp(ground(\mathcal{P}))$ under the Fitting semantics. Comparing the columns labeled $F \leftrightarrow_C G$ and $F \leftrightarrow_Ł G$ in Table 1 we observe that if $I(F \leftrightarrow_C G) = \top$ then $I(F \leftrightarrow_Ł G) = \top$. Consequently, I is also model for $comp(ground(\mathcal{P}))$ under the Łukasiewicz semantics.
To show the only-if-part, suppose $I_Ł(comp(ground(\mathcal{P}))) = \top$. In this case we have to show that $I_Ł = \langle I^\top, I^\perp \rangle$ is a fixed point of $\Phi_{F,\mathcal{P}}$, i.e., $\Phi_{F,\mathcal{P}}(I_Ł) = I_Ł$. Let $\Phi_{F,\mathcal{P}}(I_Ł) = J = \langle J^\top, J^\perp \rangle$. $J = I$ if and only if $J^\top = I^\top$ and $J^\perp = I^\perp$. We distinguish four cases:
   (a) Suppose $A \in I^\top$, i.e., $I_Ł(A) = \top$. Because $I_Ł(comp(ground(\mathcal{P}))) = \top$ we find $A \leftrightarrow Body_1 \vee Body_2 \vee \ldots \in comp(ground(\mathcal{P}))$ such that $I_Ł(Body_1 \vee Body_2 \vee \ldots) = \top$. Hence, there exists $A \leftarrow Body_i \in ground(\mathcal{P})$, $i \geq 1$, such that $I_Ł(Body_i) = \top$. Therefore, $A \in J^\top$.
   (b) Suppose $A \in J^\top$. By the definition of $\Phi_{F,\mathcal{P}}$, we find $A \leftarrow Body_i \in ground(\mathcal{P})$, $i \geq 1$, such that $I_Ł(Body_i) = \top$. Hence, we find $A \leftrightarrow Body_1 \vee Body_2 \vee \ldots \in comp(ground(\mathcal{P}))$ and $I_Ł(Body_1 \vee Body_2 \vee \ldots) = \top$. Because $I_Ł(comp(ground(\mathcal{P}))) = \top$, we find $I_Ł(A) = \top$. Hence, $A \in I^\top$.
   (c) Suppose $A \in I^\perp$, i.e., $I_Ł(A) = \perp$. Because $I_Ł(comp(ground(\mathcal{P}))) = \top$ we find $A \leftrightarrow F \in comp(ground(\mathcal{P}))$ such that $I_Ł(F) = \perp$. In this case either $F = \perp$ or $F = Body_1 \vee Body_2 \vee \ldots$ and for all $i \geq 1$ we find $I_Ł(Body_i) = \perp$. By definition of $\Phi_{F,\mathcal{P}}$ we find $A \in J^\perp$ in either case.
   (d) Suppose $A \in J^\perp$. By the definition of $\Phi_{F,\mathcal{P}}$ we find for all $A \leftarrow Body_i \in ground(\mathcal{P})$, $i \geq 1$, that $I_Ł(Body_i) = \perp$. Hence, with $F = \perp \vee Body_1 \vee Body_2 \vee \ldots$ we find $I_Ł(F) = \perp$. Because $I_Ł(comp(ground(\mathcal{P}))) = \top$ and $A \leftrightarrow F \in comp(ground(\mathcal{P}))$ we conclude $I_Ł(A) = \perp$. Consequently, $A \in I^\perp$.

2. Suppose $I_Ł = lfp(\Phi_{F,\mathcal{P}})$ and $I_Ł$ is not the least model of $comp(ground(\mathcal{P}))$. Then we find an interpretation $J_Ł$ such that $J_Ł(comp(ground(\mathcal{P}))) = \top$ and $J_Ł \subset I_Ł$. By 1., $J_Ł$ will be a fixed point of $\Phi_{F,\mathcal{P}}$, which contradicts the assumption that $I_Ł$ is the least fixed point of $\Phi_{F,\mathcal{P}}$. □

A fixed point of the Fitting operator under the Fitting semantics is a model of the completion of the program, but it is not necessarily a model of the program itself. Reconsider $\mathcal{P}_2 = \{p \leftarrow q,\ q \leftarrow p\}$. $lfp(\Phi_{F,\mathcal{P}_2}) = \langle \emptyset, \emptyset \rangle$ is not a model for $\mathcal{P}_2$. This is because under Fitting semantics, if p and q are mapped to u, then both implications are mapped to u as well. However, under the Łukasiewicz semantics, if p and q are mapped to u, then both implications are mapped to $\top$. Hence, $lfp(\Phi_{F,\mathcal{P}_2}) = \langle \emptyset, \emptyset \rangle$ is a model for $\mathcal{P}_2$ under the Łukasiewicz semantics.

Proposition 2. *Let $\mathcal{P}$ be a program.*
If $I_Ł$ $(\text{comp}(\text{ground}(\mathcal{P}))) = \top$*, then $I_Ł$*$(\text{ground}(\mathcal{P})) = \top$.

Proof. If $I_Ł(comp(ground(\mathcal{P}))) = \top$, then for all $A \leftrightarrow F \in comp(ground(\mathcal{P}))$ we find $I_Ł(A \leftrightarrow F) = \top$. By the law of equivalence we conclude $I_Ł((A \leftarrow F) \wedge (F \leftarrow$

$A)) = \top$ and, consequently, $I_Ł(A \leftarrow F) = \top$. If $F = \bot$ then *ground*($\mathcal{P}$) does not contain a clause with head A. Otherwise, $F = Body_1 \vee Body_2 \vee \ldots$ and we distinguish three cases:

1. If $I_Ł(A) = \top$, then we find $I_Ł(A \leftarrow Body_i) = \top$ for all $A \leftarrow Body_i \in ground(P)$.
2. If $I_Ł(A) = \bot$, then for all $i \geq 1$ we find $I_Ł(Body_i) = \bot$ and, consequently, $I_Ł(A \leftarrow Body_i) = \top$ for all $A \leftarrow Body_i \in ground(P)$.
3. If $I_Ł(A) = u$ then either $I_Ł(F) = \bot$ or $I_Ł(F) = u$. The former possibility being similar to case 2. we concentrate on the latter. If $I_Ł(F) = u$ then for at least one i we find $I_Ł(Body_i) = u$ and for all $i \geq 1$ either $I_Ł(Body_i) = u$ or $I_Ł(Body_i) = \bot$. In any case, we find $I_Ł(A \leftarrow Body_i) = \top$ for all $A \leftarrow Body_i \in ground(\mathcal{P})$. □

Corollary 1. *Let $\mathcal{P}$ be a program.*
If $I_Ł$ is a fixed point of $\Phi_{F,\mathcal{P}}$ then $I_Ł(\text{ground}(\mathcal{P})) = \top$.

Proof. The corollary follows immediately from Propositions 1 and 2. □

Although a fixed point of the Fitting operator is not always a model of the given program under the Fitting semantics, the program itself may have models. Returning to the example $\mathcal{P}_2 = \{p \leftarrow q,\ q \leftarrow p\}$, its minimal models under the Fitting semantics are $\langle\emptyset, \{p,q\}\rangle$ and $\langle\{p,q\}, \emptyset\rangle$. Their intersection $\langle\emptyset,\emptyset\rangle$ is no model of $\mathcal{P}_2$ under the Fitting semantics. In other words, the model intersection property does not hold under the Fitting semantics. Under the Łukasiewicz semantics, however, $\langle\emptyset,\emptyset\rangle$ is a model for $\mathcal{P}_2$ and, as we will show in the following, the model intersection property does hold under the Łukasiewicz semantics.

Proposition 3. *Let $\mathcal{P}$ be a program. If $I_Ł = \langle I^\top, I^\bot\rangle$ is a model of* ground($\mathcal{P}$)*, then $I'_Ł = \langle I^\top, \emptyset\rangle$ is also a model of* ground($\mathcal{P}$).

Proof. Let $\mathcal{P}$ be a program. Suppose $I_Ł = \langle I^\top, I^\bot\rangle$ is a model of *ground*($\mathcal{P}$). Let $A \leftarrow Body$ be a clause in *ground*($\mathcal{P}$). In order to show $I'_Ł(A \leftarrow Body) = \top$ we distinguish three cases:

1. If $A \in I^\top$, then $I'_Ł(A \leftarrow Body) = \top$.
2. If $A \in I^\bot$, then $I_Ł(A) = \bot$ and $I'_Ł(A) = u$. Because $I_Ł(A \leftarrow Body) = \top$ we conclude that $I_Ł(Body) = \bot$. Hence, we find a literal C in *Body* with $I_Ł(C) = \bot$. For each literal B occurring in *Body* we find:
   (a) if B is an atom and $B \in I^\top$, then $I_Ł(B) = \top$ and $I'_Ł(B) = \top$,
   (b) if B is an atom and $B \in I^\bot$, then $I_Ł(B) = \bot$ and $I'_Ł(B) = u$,
   (c) if B is an atom and $B \notin I^\top \cup I^\bot$, then $I'_Ł(B) = I_Ł(B) = u$,
   (d) if B is of the form $\neg B'$ and $B' \in I^\top$, then $I_Ł(B) = \bot$ and $I'_Ł(B) = \bot$,
   (e) if B is of the form $\neg B'$ and $B' \in I^\bot$, then $I_Ł(B) = \top$ and $I'_Ł(B) = u$,
   (f) if B is of the from $\neg B'$ and $B' \notin I^\top \cup I^\bot$, then $I'_Ł(B) = I_Ł(B) = u$,
   Because C must belong to either case (b) or (d) and, hence, $I'_Ł(C)$ is either u or $\bot$, we conclude that $I'_Ł(Body)$ is either $\bot$ or u as well. Because $I'_Ł(A) = u$ we conclude that $I'_Ł(A \leftarrow Body) = \top$.
3. If $A \notin I^\top \cup I^\bot$, then $I_Ł(A) = I'_Ł(A) = u$. Because $I_Ł(A \leftarrow Body) = \top$ we distinguish two cases:

(a) If $I_Ł(\textit{Body}) = \bot$, then we conclude as in case 2. that $I'_Ł(\textit{Body})$ is either $\bot$ or u and, consequently, $I'_Ł(A \leftarrow \textit{Body}) = \top$.
(b) If $I_Ł(\textit{Body}) = u$, then *Body* must contain a literal B with $I_Ł(B) = u$. In this case, $I'_Ł(B) = u$ as well and, consequently, $I'_Ł(Body)$ is either $\bot$ or u. As in the previous sub-case we conclude that $I'_Ł(A \leftarrow \textit{Body}) = \top$. □

As an example consider the program $\mathcal{P}_3 = \{p \leftarrow q \wedge \neg r\}$. In the remainder of this paragraph all models are considered under the Łukasiewicz semantics. $\langle\{p,q\},\{r\}\rangle$ is a model for $\mathcal{P}_3$, and so is $\langle\{p,q\},\emptyset\rangle$. $\langle\{p,r\},\{q\}\rangle$ is a model for $\mathcal{P}_3$, and so is $\langle\{p,r\},\emptyset\rangle$. $\langle\{r\},\{q\}\rangle$ is a model for $\mathcal{P}_3$, and so is $\langle\{r\},\emptyset\rangle$. $\langle\emptyset,\emptyset\rangle$ is the least model of $\mathcal{P}_3$.

Proposition 4. *Let* $I_{Ł1} = \left\langle I_1^\top, \emptyset\right\rangle$ *and* $I_{Ł2} = \left\langle I_2^\top, \emptyset\right\rangle$ *be two models for a program* $\mathcal{P}$. *Then* $I_{Ł3} = \left\langle I_1^\top \cap I_2^\top, \emptyset\right\rangle$ *is a model for* $\mathcal{P}$ *as well.*

Proof. Suppose $I_{Ł3} = \left\langle I_3^\top, I_3^\bot\right\rangle = \left\langle I_1^\top \cap I_2^\top, \emptyset\right\rangle$ is not a model for $\mathcal{P}$. Then we find $A \leftarrow \textit{Body} \in \mathcal{P}$ such that $I_{Ł3}(A \leftarrow \textit{Body}) \neq \top$. According to Table 1 one of the following cases must hold:

1. $I_{Ł3}(A) = \bot$ and $I_{Ł3}(\textit{Body}) = \top$.
2. $I_{Ł3}(A) = \bot$ and $I_{Ł3}(\textit{Body}) = u$.
3. $I_{Ł3}(A) = u$ and $I_{Ł3}(\textit{Body}) = \top$.

Because $I_3^\bot = \emptyset$ we find $I_{Ł3}(A) \neq \bot$ and, consequently, cases 1. and 2. cannot apply. Therefore, we turn our attention to case 3. If $I_{Ł3}(A) = u$ then there must exist $j \in \{1,2\}$ such that $I_{Łj}(A) = u$. Because $I_{Łj}$ is a model for $\mathcal{P}$ we find $I_{Łj}(A \leftarrow \textit{Body}) = \top$ and, thus, $I_{Łj}(\textit{Body})$ is either u or $\bot$. In this case, $\textit{Body} \neq \top$. Let $\textit{Body} = B_1 \wedge \ldots \wedge B_n$ with $n \geq 1$.

Because $I_{Ł3}(\textit{Body}) = \top$ and $I_3^\bot = \emptyset$ we find for all $1 \leq i \leq n$ that B_i is an atom with $I_{Ł3}(B_i) = \top$. Hence, $\{B_1, \ldots, B_n\} \subseteq I_3^\top$ and, consequently, $\{B_1, \ldots, B_n\} \subseteq I_j^\top$, which contradicts the assumption that $I_{Łj}(\textit{Body})$ is either u or $\bot$. □

Proposition 4 does not hold for arbitrary models of $\mathcal{P}$. For instance, suppose $\mathcal{P}_4 = \{p \leftarrow q_1 \wedge r_1,\ p \leftarrow q_2 \wedge r_2\}$, $I_{Ł1} = \langle\emptyset, \{p, q_1, r_2\}\rangle$ and $I_{Ł2} = \langle\emptyset, \{p, q_2, r_1\}\rangle$. We can easily show that $I_{Ł1}$ and $I_{Ł2}$ are models for $\mathcal{P}_4$. Their intersection $\langle\emptyset, \{p\}\rangle$, however, is not a model for $\mathcal{P}_4$.

Proposition 5. *Let* $\mathcal{M}_Ł$ *be the set of all models of a program* $\mathcal{P}$ *under the Łukasiewicz semantics. Then,* $\bigcap \mathcal{M}_Ł$ *is a model for* $\mathcal{P}$ *as well.*

Proof. The result follows immediately from Propositions 3 and 4. □

The least model of $\mathcal{P}_4$ under the Łukasiewicz semantics is $\langle\emptyset,\emptyset\rangle$, whereas the least model of $\mathcal{P}_5 = \{p \leftarrow \top,\ q \leftarrow p,\ r \leftarrow q \wedge \neg s\}$ under the Łukasiewicz semantics is $\langle\{p,q\},\emptyset\rangle$. The last example also exhibits that the least fixed point of the Fitting operator is not necessarily the least model of the underlying program because $\textit{lfp}(\Phi_{F,\mathcal{P}_4}) = \langle\{p,q,r\},\{s\}\rangle$.

5 The Stenning and van Lambalgen Operator

In their quest for models of human reasoning Stenning and van Lambalgen [18] have introduced an immediate consequence operator for propositional programs, which differs slightly from the Fitting operator. Here, we extend the operator to first-order programs. Let I be an interpretation and $\mathcal{P}$ be a program. *Stenning and van Lambalgen's immediate consequence operator* is defined as follows: $\Phi_{SvL,\mathcal{P}}(I) = \langle J^\top, J^\bot \rangle$, where

$$J^\top = \{A \mid \text{there exists } A \leftarrow Body \in ground(\mathcal{P}) \text{ with } I(Body) = \top\} \text{ and}$$
$$J^\bot = \{A \mid \text{there exists } A \leftarrow Body \in ground(\mathcal{P}) \text{ and for all } A \leftarrow Body \in ground(\mathcal{P}) \text{ we find } I(Body) = \bot\}$$

and the difference to the Fitting operator has been highlighted. Stenning and van Lambalgen consider programs under the Fitting semantics. In addition, Stenning and van Lambalgen allow so-called *negative facts* of the form $A \leftarrow \bot$ as program clauses. An *extended (logic) program* is a finite set of clauses and negative facts.

Stenning and van Lambalgen show in [18] that $\Phi_{SvL,\mathcal{P}}$ is monotone on $(\mathcal{I}, \subseteq)$. Moreover, from [19] and [16] follows that for finite $ground(\mathcal{P})$ the operator $\Phi_{SvL,\mathcal{P}}$ is also continuous. We call a program $\mathcal{P}$ *SvL-acceptable* if $\Phi_{SvL,\mathcal{P}}$ is continuous.

If $\Phi_{SvL,\mathcal{P}}$ is continuous then we can compute the least fixed point of $\Phi_{SvL,\mathcal{P}}$ by iterating $\Phi_{SvL,\mathcal{P}}$ starting from empty interpretation. Let I be the least fixed point of $\Phi_{SvL,\mathcal{P}}$ and let

$$I_0 = \langle \emptyset, \emptyset \rangle \tag{1}$$
$$I_\alpha = \Phi_{SvL,\mathcal{P}}(I_{\alpha-1}) \text{ for every non-limit ordinal } \alpha > 0 \tag{2}$$
$$I_\alpha = \bigcup_{\beta<\alpha} I_\beta \text{ for every limit ordinal } \alpha \tag{3}$$

Then for some ordinal ω we find $I = I_\omega$.

Before discussing further properties of the new operator we reconsider $\mathcal{P}_1 = \{p \leftarrow q\}$. Its completion is $comp(ground(\mathcal{P}_1)) = \{p \leftrightarrow q,\ q \leftrightarrow \bot\}$. $\Phi_{SvL,\mathcal{P}}$ admits a least fixed point for $\mathcal{P}_1$ and we obtain $lfp(\Phi_{SvL,\mathcal{P}_1}) = \langle \emptyset, \emptyset \rangle$. One should note that this result differs from $lfp(\Phi_{F,\mathcal{P}_1}) = \langle \emptyset, \{p, q\} \rangle$. Now consider $\mathcal{P}_1' = \{p \leftarrow q,\ q \leftarrow \bot\}$. Its completion is $comp(ground(\mathcal{P}_1')) = \{p \leftrightarrow q,\ q \leftrightarrow \bot\} = comp(ground(\mathcal{P}_1))$ and $lfp(\Phi_{SvL,\mathcal{P}_1'}) = lfp(\Phi_{F,\mathcal{P}_1}) = \langle \emptyset, \{p, q\} \rangle$. Thus, by adding negative facts, Stenning and van Lambalgen's operator can simulate Fitting's operator. But it is more liberal in that if there is no clause with head A in the extended program, then its meaning remains undefined.

Obviously, completion as defined in Section 3.4 is unsuitable for extended programs $\mathcal{P}$. If we omit step 2. in the completion transformation, then the resulting set of formulas is called *weak completion of* $ground(\mathcal{P})$ and is denoted by $wcomp(ground(\mathcal{P}))$. Returning to the examples, we find $wcomp(ground(\mathcal{P}_1)) = \{p \leftrightarrow q\}$ and $wcomp(ground(\mathcal{P}_1')) = \{p \leftrightarrow q,\ q \leftrightarrow \bot\}$.

In the following we relate the Stenning and van Lambalgen operator and weak completion under the Łukasiewicz semantics.

Lemma 1. *Let* $I_Ł$ *be the least fixed point of* $\Phi_{SvL,\mathcal{P}}$ *and* $J_Ł$ *be a model of* wcomp(ground($\mathcal{P}$)) *then* $I_Ł \subseteq J_Ł$*.*

Proof. Let $I_Ł = \langle I^\top, I^\bot \rangle$ be the least fixed point of $\Phi_{SvL,\mathcal{P}}$ and $J_Ł = \langle J^\top, J^\bot \rangle$ be a model of *wcomp*(*ground*($\mathcal{P}$)). $I_Ł \subseteq J_Ł$ iff $I^\top \subseteq J^\top$ and $I^\bot \subseteq J^\bot$ iff the following propositions hold: (i) if $I_Ł(A) = \top$, then $J_Ł(A) = \top$ and (ii) if $I_Ł(A) = \bot$, then $J_Ł(A) = \bot$. By transfinite induction it can be shown that for every ordinal α and every atom A we find: (iii) if $I_\alpha(A) = \top$, then $J_Ł(A) = \top$ and (iv) if $I_\alpha(A) = \bot$, then $J_Ł(A) = \bot$. The claim follows immediately by the definition of least fixed point of $\Phi_{SvL,\mathcal{P}}$ because it implies that there is an ordinal ω such that $I_Ł = I_\omega$. □

Proposition 6. *Let* $\mathcal{P}$ *be an extended program. If* $I_Ł$ *is the least fixed point of* $\Phi_{SvL,\mathcal{P}}$*, then* $I_Ł$ *is a minimal model of* wcomp(ground($\mathcal{P}$)).

Proof. First we will show that $I_Ł$ is a model of *wcomp*(*ground*($\mathcal{P}$)). Let's pick an arbitrary formula $(A \leftrightarrow F) \in$ *wcomp*(*ground*($\mathcal{P}$)). In order to show that $I_Ł(A \leftrightarrow F) = \top$ we consider three cases according to the truth value of A in $I_Ł$:

a) If $I_Ł(A) = \top$, then according to the definition of $\Phi_{SvL,\mathcal{P}}$, there exists a clause $(A \leftarrow \mathit{Body}_i) \in \mathit{ground}(\mathcal{P})$ such that $I_Ł(\mathit{Body}_i) = \top$. Because Body_i is one of the disjuncts of F, this implies $I_Ł(F) = \top$ and hence $I_Ł(A \leftrightarrow F) = \top$.
b) If $I_Ł(A) = \bot$, then according to the definition of $\Phi_{SvL,\mathcal{P}}$, there is a clause $(A \leftarrow \mathit{Body}_i) \in \mathit{ground}(\mathcal{P})$ and for every clause $(A \leftarrow \mathit{Body}_i) \in \mathit{ground}(\mathcal{P})$ we have $I_Ł(\mathit{Body}_i) = \bot$ for all i. Consequently, all disjuncts in F are false under $I_Ł$ and, therefore, $I_Ł(F) = \bot$. Hence, $I_Ł(A \leftrightarrow F) = \top$.
c) If $I_Ł(A) = u$, then according to the definition of $\Phi_{SvL,\mathcal{P}}$ there is no clause $(A \leftarrow \mathit{Body}_i) \in \mathit{ground}(\mathcal{P})$ with $I_Ł(\mathit{Body}_i) = \top$ and there are some clauses $(A \leftarrow \mathit{Body}_j) \in \mathit{ground}(\mathcal{P})$ with $I_Ł(\mathit{Body}_j) \neq \bot$. So none of the disjuncts in F is true, but it is also not the case that all of them are false. Therefore $I_Ł(F) = u$ and $I_Ł(A \leftrightarrow F) = \top$.

To prove that $I_Ł$ is a minimal model of *wcomp*(*ground*($\mathcal{P}$)), let $I_Ł = \langle I_Ł^\top, I_Ł^\bot \rangle$. By Lemma 1 we learn that any model $J_Ł = \langle J_Ł^\top, J_Ł^\bot \rangle$ of *wcomp*(*ground*($\mathcal{P}$)) will be such that $I_Ł^\top \subseteq J_Ł^\top$ and $I_Ł^\bot \subseteq J_Ł^\bot$. Hence, no proper subset of $I_Ł$ can be a model of *wcomp*(*ground*($\mathcal{P}$)). Consequently, $I_Ł$ is a minimal model of *wcomp*(*ground*($\mathcal{P}$)). □

Proposition 7. *Let* $\mathcal{P}$ *be an extended program. If* $I_Ł$ *is a minimal model of* wcomp(ground($\mathcal{P}$))*, then* $I_Ł$ *is the least fixed point of* $\Phi_{SvL,\mathcal{P}}$*.*

Proof. Let $I_Ł = \langle I_Ł^\top, I_Ł^\bot \rangle$ be a minimal model of *wcomp*(*ground*($\mathcal{P}$)) and let $J_Ł = \langle J_Ł^\top, J_Ł^\bot \rangle$ be the least fixed point of $\Phi_{SvL,\mathcal{P}}$. By Lemma 1 we know that $J_Ł^\top \subseteq I_Ł^\top$ and $J_Ł^\bot \subseteq I_Ł^\bot$. Further, by Proposition 6 we have that $J_Ł$ is a minimal model of *wcomp*(*ground*($\mathcal{P}$)). But then it must be the case that $I_Ł = J_Ł$ because otherwise we have a conflict with the minimality of $I_Ł$. □

Corollary 2. *Let* $\mathcal{P}$ *be an extended program.* $I_Ł$ *is the least fixed point of* $\Phi_{SvL,\mathcal{P}}$ *iff* $I_Ł$ *is the least model of* wcomp(ground($\mathcal{P}$)).

Proof. Follows from Propositions 6 and 7 and the fact that the least fixed point of $\Phi_{SvL,\mathcal{P}}$ is unique. □

One should observe, that Corollary 2 does not hold if we consider *comp*$(\mathit{ground}(\mathcal{P}))$ and the Fitting semantics instead of the Łukasiewicz semantics. As an example consider again $\mathcal{P}_1 = \{p \leftarrow q\}$ and let $I = \langle\emptyset, \{p, q\}\rangle$. I_F is a model for *comp*$(\mathcal{P}_1)$, but $\Phi_{SvL,\mathcal{P}_1}(I) = \langle\emptyset, \{p\}\rangle \neq I$. This is counter example for Lemma 4(3) in [18].

Proposition 8. *Let $\mathcal{P}$ be an extended program.*
If $I_Ł$(wcomp(ground($\mathcal{P}$))) $= \top$, then $I_Ł$(ground($\mathcal{P}$)) $= \top$.

Proof. If $I_Ł(\mathit{wcomp}(\mathit{ground}(\mathcal{P}))) = \top$, then for all $A \leftrightarrow F \in \mathit{wcomp}(\mathit{ground}(\mathcal{P}))$ we find $I_Ł(A \leftrightarrow F) = \top$. By the law of equivalence we conclude $I_Ł((A \leftarrow F) \wedge (F \leftarrow A)) = \top$ and, consequently, $I_Ł(A \leftarrow F) = \top$. Let $F = \mathit{Body}_1 \vee \mathit{Body}_2 \vee \ldots$. We distinguish three cases:

1. If $I_Ł(A) = \top$, then we find $I_Ł(A \leftarrow \mathit{Body}_i) = \top$ for all $A \leftarrow \mathit{Body}_i \in \mathit{ground}(P)$.
2. If $I_Ł(A) = \bot$, then for all $i \geq 1$ we find $I_Ł(\mathit{Body}_i) = \bot$ and, consequently, $I_Ł(A \leftarrow \mathit{Body}_i) = \top$ for all $A \leftarrow \mathit{Body}_i \in \mathit{ground}(P)$.
3. If $I_Ł(A) = u$ then either $I_Ł(F) = \bot$ or $I_Ł(F) = u$. The former possibility being similar to case 2. we concentrate on the latter. If $I_Ł(F) = u$ then we find an i with $I_Ł(\mathit{Body}_1) = u$ and for all $i \geq 1$ either $I_Ł(\mathit{Body}_i) = u$ or $I_Ł(\mathit{Body}_i) = \bot$. In any case, we find $I_Ł(A \leftarrow \mathit{Body}_i) = \top$ for all $A \leftarrow \mathit{Body}_i \in \mathit{ground}(\mathcal{P})$. □

From Proposition 6 and Proposition 8 we can derive Corollary 3 for the Stenning and Lambalgen operator.

Corollary 3. *Let $\mathcal{P}$ be an extended program.*
If $I_Ł$ is the least fixed point of $\Phi_{SvL,\mathcal{P}}$ then $I_Ł$(ground($\mathcal{P}$)) $= \top$.

Proof. The corollary follows immediately from Propositions 6 and 8. □

One should observe that contrary to Lemma 4(1.) of [18] this corollary does not hold under the Fitting semantics. Reconsider $\mathcal{P}_1 = \{p \leftarrow q\}$, then $\mathit{lfp}(\Phi_{SvL,\mathcal{P}_1}) = \langle\emptyset, \emptyset\rangle$ and, thus, both p and q are mapped to u. Under this interpretation $\mathcal{P}_1$ is mapped to u as well. One should also note that the least fixed point of the Stenning and van Lambalgen operator for a given program $\mathcal{P}$ is not necessarily the least model of $\mathcal{P}$ under the Fitting semantics. Reconsidering $\mathcal{P}_1' = \{p \leftarrow q,\ q \leftarrow \bot\}$ we find $\mathit{lfp}(\Phi_{SvL,\mathcal{P}_1'}) = \langle\emptyset, \{p, q\}\rangle$ whereas the least model of $\mathcal{P}_1'$ under the Łukasiewicz semantics is $\langle\emptyset, \emptyset\rangle$.

6 Two Examples

In this section we present two examples to illustrate the difference between the Fitting and the Stenning and van Lambalgen operator. Suppose we want to model an agent driving a car. One rule would be that he may cross an intersection if the traffic light shows green and there is no unusual situation:

$$cross \leftarrow green, \neg unusual\_situation.$$

An unusual situation occurs if an ambulance wants to cross the intersection from a different direction:

$$unusual\_situation \leftarrow ambulance\_crossing.$$

In addition, suppose that the green light is indeed on:

$$green \leftarrow \top.$$

Let $\mathcal{P}_6$ be the set of these clauses. It is easy to see that

$$lfp(\Phi_{F,\mathcal{P}_6}) = \langle\{green, cross\}, \{unusual\_situation, ambulance\_crossing\}\rangle .$$

Hence, not knowing anything about an ambulance, our agent will assume that no ambulance is present, hit the accelerator, and speed into the intersection. One should observe that not knowing anything about an ambulance may be caused by the fact that the agent's camera is blurred or the agent's microphone is damaged. His assumption that no ambulance is present is made by default. On the other hand,

$$lfp(\Phi_{SvL,\mathcal{P}_6}) = \langle\{green\}, \emptyset\}\rangle .$$

In this case, the agent doesn't know whether he may cross the intersection. Inspecting his rules he may find that in order to satisfy the conditions for the first rule, he must verify that no ambulance is crossing. In doing so, he may extend $\mathcal{P}_6$ to $\mathcal{P}_6' = \mathcal{P}_6 \cup \{ambulance\_crossing \leftarrow \bot\}$ yielding

$$lfp(\Phi_{SvL,\mathcal{P}_{6'}}) = \langle\{green, cross\}, \{unusual\_situation, ambulance\_crossing\}\rangle .$$

Now, the agent can safely cross the intersection.

The second example is taken from [4]. Byrne has confronted individuals with sentences like *If Marian has an essay to write, she will study late in the library. She does not have an essay to write. If she has textbooks to read, she will study late in the library.* The individuals are then asked to draw conclusions. In this example, only 4% of the individuals conclude that Marian will not study late in the library. Although Byrne uses these and similar examples to conclude that (classical) logic is inadequate for human reasoning, Stenning and van Lambalgen have argued in [18] that the use of three-valued logic programs under completion semantics is indeed adequate for human reasoning. They represent the scenario by

$$\mathcal{P}_7 = \{l \leftarrow e \wedge \neg ab_1,\ e \leftarrow \bot,\ ab_1 \leftarrow \bot,\ l \leftarrow t \wedge \neg ab_2, ab_2 \leftarrow \bot\},$$

where l denotes that Marian will study late in the library, e denotes that she has an essay to write, t denotes that she has a textbook to read, and ab denotes abnormality. In this case, we find $lfp(\Phi_{SvL,\mathcal{P}_7}) = \langle\emptyset, \{ab_1, ab_2, e\}\rangle$, from which we conclude that it is unknown whether Marian will study late in the library. On the other hand, $lfp(\Phi_{F,\mathcal{P}_7}) = \langle\emptyset, \{ab_1, ab_2, e, t, l\}\rangle$. Using the Fitting operator one would conclude that Marian will not study late in the library. Thus, this operator leads to a wrong answer with respect to the discussed scenario from human reasoning, whereas the Stenning and van Lambalgen operator does not.

Table 3. A comparison between the Fitting and the Łukasiewicz semantics for logic programs. We have highlighted the results which were obtained by formal proofs or by counter examples in this paper. The result marked by † was formally proven in [7]. The result marked by * was not proven formally in [18] nor in this paper, but we conjecture that it holds.

Property	Fitting	Łukasiewicz
Model Intersection	No	Yes
Fixed points of $\Phi_{F,\mathcal{P}}$ are models of $comp(ground(\mathcal{P}))$	Yes†	Yes
Fixed points of $\Phi_{F,\mathcal{P}}$ are models of $\mathcal{P}$	No	Yes
The least fixed point of $\Phi_{SvL,\mathcal{P}}$ is the least model of $wcomp(ground(\mathcal{P}))$	Yes*	Yes
The least fixed point of $\Phi_{SvL,\mathcal{P}}$ is a model of $\mathcal{P}$	No	Yes

7 Conclusion

Table 3 compares the Fitting and Łukasiewicz semantics for logic programs as discussed in this paper. In [18] many more examples are given to support the claim that human reasoning can be adequately modelled using completion-based propositional logic programs and the Stenning and van Lambalgen operator. Here, we have extended this approach to first-order programs and have given rigorous proofs of some of the properties of the operator under Łukasiewicz semantics.

Naish in [17] considers yet another three-valued semantics, which differs from the Fitting and Łukasiewics semantics studied in this paper as far as the truth table for the implication is concerned. Although Naish shows several model intersection results for his logic, these results do not subsume our model intersection result nor is our result an immediate consequence of Naish's results. Likewise, Naish introduces new immediate consequence operators, but they differ from the Stenning and van Lambalgen operator studied in this paper and, again, the results by Naish do not subsume our results nor are our results immediate consequences of Naish's results. There is an underlying reason for the differences: Naish focuses on programming and debugging, whereas the work by Stenning and van Lambalgen, which underlies this paper, focuses on human reasoning.

In recent years, the Fitting semantics for logic programs has not been used much. It has been overtaken in interest by the well-founded semantics [20] and stable model semantics [9]. The latter extends the former in a well-understood manner, and provides a two-valued semantics for logic programs. Both capture transitive closure and other recursive rule behavior and, thus, are useful for programming. However, there are trade-offs between the Fitting semantics and well-founded semantics. The ability of well-founded semantics to capture properties like graph reachability means that it cannot be modelled by a finite first-order theory such as completion. Well-founded semantics also has a higher complexity than the Fitting semantics. The relationship of the Fitting semantics and the well-founded semantics is brought forward in [11] using level mappings. These are mappings from Herbrand bases to ordinals, i.e., they induce orderings on the set of ground atoms while disallowing infinite descending chains. The result shows that well-founded semantics is a stratified version of the Fitting semantics.

It has been argued recently in [18] that a completion-based approach captures many aspects of commonsense reasoning. Unlike most approaches to logically modelling commonsense reasoning which rely on introspection to characterize common sense, Stenning and van Lambalgan base their model on the large corpus of cognitive science. The result is already helping logic programming to be re-examined in fields such as medical decision-making.

In [18] and [12] connectionist implementations of the Stenning and van Lambalgen operator are given. The latter is based on the core method (connectionist model generation using recurrent networks with feed-forward core, see e.g. [2]), which has been applied to propositional, first-order, multi-valued as well as modal logic programs (see e.g. [3,6]).

The role of negative facts in extended logic programs needs to be discussed. The name *negative fact* is considered only with respect to the (weak) completion of a program as, otherwise, a negative fact like $A \leftarrow \bot$ is also mapped to true by interpretations which map A to u or $\top$. If in addition a program contains a clause with head A, then negative facts can be eliminated without changing the semantics of the program. This is hardly the intention of a negative fact in human reasoning, where an individual may gather some support for a fact as well as its negation. An alternative idea would be to add $\bot \leftarrow A$ to a program and treat this as a constraint, but this needs to be investigated in the future.

We would like to find a syntactic characterization of SvL-acceptability and relate it to corresponding characterizations of F-acceptability. Likewise, we would like to find conditions under which the Stenning and van Lambalgen operator is a contraction and relate it to corresponding findings with respect to the Fitting operator (see [8]).

Last but not least it remains to be seen which semantics is better suited for logic programming, common sense as well as human reasoning. It appears that the Łukasiewicz semantics has nicer theoretical properties, but we still have to investigate how this semantics relates to questions concerning computability and termination. It also appears that the Łukasiewicz semantics gives more flexibility than the Fitting semantics concerning common sense reasoning problems. As far as human reasoning is concerned we would like to find out how individuals treat implications where the premise as well as the conclusion are undefined as this is the distinctive feature between the Łukasiewicz and the Fitting semantics.

Acknowledgement. The authors would like to thank Bertram Fronhöfer for many fruitful discussions and the anonymous referees for many useful comments and suggestions.

References

1. Apt, K.R., Pedreschi, D.: Reasoning about termination of pure Prolog programs. In: Information and Computation (1993)
2. Bader, S., Hölldobler, S.: The core method: Connectionist model generation. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) ICANN 2006. LNCS, vol. 4132, pp. 1–13. Springer, Heidelberg (2006)
3. Bader, S., Hitzler, P., Hölldobler, S.: Connectionist model generation: A first-order approach. Neurocomputing 71, 2420–2432 (2008)
4. Byrne, R.M.J.: Suppressing valid inferences with conditionals. Cognition 31, 61–83 (1989)
5. Clark, K.L.: Negation as failure. In: Gallaire, H., Minker, J. (eds.) Logic and Databases, pp. 293–322. Plenum, New York (1978)

6. d'Avila Garcez, A.S., Broda, K., Gabbay, D.M.: Neural-Symbolic Learning Systems: Foundations and Applications. Springer, Heidelberg (2002)
7. Fitting, M.: A Kripke–Kleene semantics for logic programs. Journal of Logic Programming 2(4), 295–312 (1985)
8. Fitting, M.: Metric methods – three examples and a theorem. Journal of Logic Programming 21(3), 113–127 (1994)
9. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proceedings of the International Joint Conference and Symposium on Logic Programming, pp. 1070–1080. MIT Press, Cambridge (1988)
10. Hitzler, P., Seda, A.K.: Characterizations of classes of programs by three-valued operators. In: Gelfond, M., Leone, N., Pfeifer, G. (eds.) LPNMR 1999. LNCS, vol. 1730, pp. 357–371. Springer, Heidelberg (1999)
11. Hitzler, P., Wendt, M.: The well-founded semantics is a stratified fitting semantics. In: Jarke, M., Koehler, J., Lakemeyer, G. (eds.) KI 2002. LNCS, vol. 2479, pp. 205–221. Springer, Heidelberg (2002)
12. Hölldobler, S., Kencana Ramli, C.D.: Logics and networks for human reasoning. Technical report, International Center for Computational Logic, TU Dresden (submitted, 2009)
13. Kleene, S.C.: Introduction to Metamathematics. North-Holland, Amsterdam (1952)
14. Lloyd, J.W.: Foundations of Logic Programming. Springer, Berlin (1993)
15. Łukasiewicz, J.: O logice trójwartościowej. Ruch Filozoficzny 5, 169–171 (1920); On Three-Valued Logic. In: Borkowski, L. (ed.) Jan Łukasiewicz Selected Works, pp. 87–88. North Holland (1990) (English translation)
16. Mycroft, A.: Logic programs and many-valued logic. In: Fontet, M., Mehlhorn, K. (eds.) STACS 1984. LNCS, vol. 166, pp. 274–286. Springer, Heidelberg (1984)
17. Naish, L.: A three-valued semantics for logic programmers. Theory and Practice of Logic Programming 6(5), 509–538 (2006)
18. Stenning, K., van Lambalgen, M.: Human Reasoning and Cognitive Science. MIT Press, Cambridge (2008)
19. Stoy, J.E.: Denotational Semantics. MIT Press, Cambridge (1977)
20. van Gelder, A., Ross, K.A., Schlipf, J.S.: The well-founded semantics for general logic programs. Journal of the ACM 38, 620–650 (1991)