Ground Nonmonotonic Modal Logic S5: New Results

MAURICIO OSORIO GALINDO, JUAN ANTONIO NAVARRO PÉREZ,
Universidad de las Américas - Puebla, Computer Science Department,
Email: {josorio,m10890}@mail.udlap.mx

JOSÉ R. ARRAZOLA RAMÍREZ and VERÓNICA BORJA MACÍAS ,
Benemérita Universidad Autónoma de Puebla, Mathematics Department,
Email: {arrazola, vero0304}@fcfm.buap.mx

Abstract

We study logic programs under Gelfond's translation in the context of modal logic S5. We show that for arbitrary logic programs (propositional theories where logic negation is associated with default negation) ground nonmonotonic modal logics between T and S5 are equivalent. Furthermore, we also show that these logics are equivalent to a nonmonotonic logic that we construct using the well known $\mathcal{FOUR}$ bilattice. We will call this semantic GNM-S5 as a reminder of its origin in the logic S5. Finally we show that, for normal programs, our approach is closely related to the Well-Founded-by-Cases Semantics introduced by Schlipf and the WFS$^+$ proposed by Dix. We prove that GNM-S5 has the properties of classicality and extended cut. While WFS$^+$ also supports classicality it fails to satisfy the extended cut principle, an important property available in other semantics such as stable models. Hence, we claim that GNM-S5 is a good candidate for defining a nonmonotonic semantics closer to the direction of classical logic.

Keywords: Modal logic S5, nonmonotonic reasoning, $\mathcal{FOUR}$ bilattice, logic programming, semantics of programming.

1 Introduction

The idea of using modal logic to formalise nonmonotonic reasoning (NMR) can be traced back to McDermott and Doyle [24]. Subsequently McDermott [23] attempted to define nonmonotonic logics based on the standard T, S4 and S5 logics. But he observed that, unfortunately, nonmonotonic S5 collapses to ordinary logic S5. Moore suggested the use of autoepistemic logic (AEL) as an alternative formalization of NMR to avoid the problems encountered with standard modal logics [25]. Moore explains that the real problem with NMR S5 is not the S5 schema, but the adoption of $\Box A \rightarrow A$. He argues that 'the S5 schema merely makes explicit the consequences of adopting $\Box A \rightarrow A$ as a premise schema that are implicit in the logic's natural semantics' [25]. Gelfond also showed in [15] that the perfect models of stratified logic programs can be characterized in terms of expansions of the corresponding autoepistemic theory. His characterization is based on the interpretation of $not\ a$ as $\neg\Box a$. In fact, Baral explains in [3] that the definition of stable models in [16] was inspired by this transformation. Having in mind McDermott and Doyle's work this idea can be interpreted as bounding introspection to objective formulae (non modal sentences). Schwartz [37] proved later the equivalence of AEL with Logic KD45, and more recently Lifschitz was able to characterize the stable semantics for disjunctive programs in terms of AEL via Gelfond's translation [3]. AEL gained a lot of interest (well deserved) and while the approach based on modal logics was almost abandoned. We suggest the reader check [22], where it is possible to find a detailed discussion of the development of the field.

Vol. 15 No. ,
doi:10.1093/logcom/exi042

The NM Logic is called Grounded and it refers to the idea of enabling the agent to make only as assumptions *grounded* in the world's knowledge. According to [12] the notion of groundedness was actually introduced by Konolige in [19]. It is worth mentioning that groundedness has a rather intuitive motivation: 'it corresponds to discarding the reasoning based on epistemic assumptions, which, for example, would enable to conclude that something is true in the world, by assuming to know it' [12]. Donini, Nardi and Rosati [12] renewed interest in nonmonotonic S5 (and other normal modal logics) by studying their grounded versions. They show in particular that grounded nonmonotonic S5 does not collapse with S5. Our paper continues this line of research but we restrict our study to what we call K-basic formulae (sentences with modalities applied only to literals).

Our approach is based on Gelfond's original interpretation and the experience on stable models semantics that shows how it suffices to apply modalities to literals, instead of arbitrary complex formulae, in order to express interesting problems. With our restricted syntax, we show that all ground nonmonotonic modal logics between T and S5 are equivalent. Furthermore, we also show that these logics are equivalent to a nonmonotonic logic that we construct using the well-known $\mathcal{FOUR}$ bilattice. We will call this semantic GNM-S5 as a reminder of its origin in the logic S5. Finally we show that, for normal programs, our approach is closely related to the Well-Founded-by-Cases Semantics introduced by Schlipf [35] and the WFS$^+$ proposed by Dix [9, 10]. We prove that GNM-S5 has the properties of classicality and extended cut. While WFS$^+$ also supports classicality it fails to satisfy the extended cut principle, an important property available in other semantics such as stable models. The reason is just because WFS$^+$ was defined for normal programs that do not include literals as facts. Hence, we claim that GNM-S5 is a good candidate for defining a nonmonotonic semantics closer to the direction of classical logic. We present, in fact, several more results around the topics already mentioned with the aim of understanding nonmonotonic reasoning in further detail.

Our paper is structured as follows. Section 2 describes the general syntax of logic formulae, in particular we present modal formulae (containing modal connectives) and introduce rules for logic programs as a special case of logic formulae. Some of the traditional classes of logic programs, as well as some classes useful for the purposes of this paper, are also defined in this section.

Section 3 introduces the notions of frames and models that are used to provide semantics for modal formulae and theories. In particular, we recall how the satisfiability of some particular formulae can be related to properties of a given frame. One of our first basic results is to present a formula that is able to characterize the number of points (or worlds) in universal frames. We also show how, with respect to an interesting fragment of modal formulae, several classes of frames turn out to be equivalent.

The use of truth values to assign semantics for modal formulae is analysed in Section 4 as well as the $\mathcal{FOUR}$ bilattice. This is later used to define a particular semantic that we call Modal $\mathcal{FOUR}$. We also show that $\mathcal{FOUR}$ is useful to simulate other logics, such as the Gödel multivalued logic G_4, and how the $\mathcal{FOUR}$ valuation is closely related to the universal frame with two points.

Then, in Section 5, we enter into the Proof Theory of modal logics. We review some well known techniques to prove completeness results and present an axiomatization for the logics obtained by universal frames with a fixed (finite) number of frames. The results of previous sections find their conclusion here allowing the study of modal logics which become useful later in the context of logic programming. These sections dealing with modal logics briefly present some of the basic material of [18]. The reader is referred to the cited book for a more detailed introduction to modal logics.

In a slightly different direction, Section 6 develops a natural deduction system intended to model the inference abilities of normal logic programs. Inspired in resolution methods, this section ends with the proposal of two inference rules for natural deduction restricted to the class of normal formulae. The purpose of this section is only to provide partial results of interest in the following

section.

In Section 7 we enter into the fields of Logic Programming. We give first a general definition of semantics for logic programs and introduce the semantics of WFS [39] and answer sets that represent two of the most developed lines of research in logic programming and non-monotonic reasoning. We recall how results in other papers have related answer sets with intermediate and intuitionistic logics, it follows as a simple consequence of results presented here that $\mathcal{FOUR}$ can also be used to model answer sets for the class of disjunctive programs. We also introduce and study GNM-S5. We prove (thanks to results from the previous section) that GNM-S5 satisfies Classicality, namely that if an atom a is a logical consequence in classical logic from a normal program P then a is derived by P under the GNM-S5 semantic.

Section 8 finishes with the conclusions of this paper. This includes comments on applications and consequences of the results obtained, as well as lines of research left open for future work.

At the end of the paper we include four appendices. Some necessary simple results for Section 3 and Section 4 are placed respectively in Appendix A and Appendix B. The inference rule transitions defined in Section 6 are in Appendix C. Finally in Appendix D we find some preliminary lemmas needed in the proof of the principal result of Section 6.

2 Syntax of logic formulae

We introduce in this section the formal syntax of propositional modal formulae. Logic programs consist of *clauses* that are just special cases of formulae restricted to a particular format. We also define several classes of logic programs found in the literature and classes relevant to the purposes of this paper.

2.1 Syntax of modal logic

Formally we consider a language consisting of: An enumerable set $\mathcal{L}$ of elements called *atoms* or *atomic formulae*; the binary connectives $\wedge$, $\vee$ and $\rightarrow$ (for conjunction, disjunction and implication respectively); the unary connectives $\neg$, $\Box$ and $\Diamond$ (for negation, modal necessitation and possibility respectively); the 0-place connectives $\bot$ and $\top$ (for falsity and truth); as well as the auxiliary symbols: $(,)$. Formulae are constructed as usual from this set of connectives.

Depending on the approach, only a few of these connectives are considered to be *primitive*, while the others are introduced as abbreviations. In the context of modal logics, it is standard to accept, for instance, only $\{\rightarrow, \Box, \bot\}$ as primitive connectives and introduce the rest of them according to the following definitions:

$$\begin{array}{cc} \neg A := A \rightarrow \bot & \top := \neg\bot \\ A \vee B := \neg A \rightarrow B & A \wedge B := \neg(\neg A \vee \neg B) \\ \Diamond A := \neg\Box\neg A & \end{array}$$

Another standard defined connective, which is rarely taken as primitive, is the biconditional $A \leftrightarrow B := (A \rightarrow B) \wedge (B \rightarrow A)$. Moreover we may also use, following the tradition in logic programing, $A \leftarrow B$ as an alternative way of writing the formula $B \rightarrow A$.

The connectives $\Box$ and $\Diamond$ are usually known as *modal connectives* and they may have different interpretations. For instance $\Box A$ could be read as 'It is necessary true that A', 'It will always be true that A', 'It ought to be that A', or 'It is known that A'. The possible readings of the formula $\Diamond A$ depend on the one selected for $\Box A$.

A *theory* T is just a set of formulae and its *signature*, denoted $\mathcal{L}_T$, is the set of atoms that occur in it. Given a theory T we also define the negated theory $\neg T = \{\neg A \mid A \in T\}$. A *literal* is either an atom a (a positive literal) or a negated atom $\neg a$ (a negative literal). Given a literal x we also define its complement as $\bar{x}$, i.e. $\overline{(a)} = \neg a$ and $\overline{(\neg a)} = a$.

We find particularly useful the class of formulae where the scope of the primitive connective $\Box$ is restricted to literals. One of the contributions of this paper is to study the properties of this fragment of modal formulae that we refer as K-basic.

DEFINITION 2.1
The class of *K-basic formulae* is the minimal set X satisfying:

- $\bot \in X$.
- If x is a literal then $x, \Box x \in X$.
- If $A, B \in X$ then $(A \rightarrow B) \in X$.

It is easy to observe that, introducing other standard connectives as their usual abbreviations, $\neg A$, $A \vee B$ and $A \wedge B$ are also K-basic formulae provided that both A and B are K-basic. For atomic p, $\Diamond p$ is also a K-basic formula.

2.2 Syntax of logic programs

A logic program is, in general, a propositional theory without modal connectives. Moreover, we restrict our attention to finite programs that have, as a consequence, finite signatures. Formally we consider the set $\{\wedge, \vee, \rightarrow, \bot\}$ as primitive connectives while $\neg$ and $\top$ are introduced as the usual abbreviations.

While we present a very general definition of logic program there are several classes of programs that have been of particular interest in the development of the theory and practice of logic programming. A *disjunctive program*, for instance, is a set of formulae of the form

$$h_1 \vee \cdots \vee h_k \leftarrow b_1 \wedge \cdots \wedge b_n \wedge \neg b_{n+1} \wedge \cdots \wedge \neg b_{n+m} \ ,$$

where each h_i and b_j is an atom in $\mathcal{L}$. A rule of this form with $k = 1$ is known as a *normal rule*. A *normal program* is one containing only normal rules. Moreover, if $n = m = 0$ then the right-hand side of the formula is interpreted as a single $\top$ and the rule is known as a *fact*; while $k = 0$ means to have $\bot$ in the left-hand side of the rule that is known as a *constraint*.

Another class of programs interesting in our context is the class of *basic formulae* that are defined as the class of formulae where the scope of negation is restricted to single atoms. Formally it corresponds to the minimal set X satisfying:

- If $p \in \mathcal{L}$ then $p, \neg p \in$ X.
- If $F, G \in$ X then $(F \wedge G), (F \vee G), (F \rightarrow G) \in$ X.

A *basic program* is a program that contains only basic formulae. Note that a disjunctive program without constraints is, in particular, a basic program. Basic programs are closely related to K-basic theories defined in the previous section, due to a translation given by Gelfond [15].

3 Semantics using frames

This section presents some of the basic notions of frames that are commonly used to give semantics for modal logics. We review some of the basic results in [18] and introduce a modal formula that

Schemata		Frame Property	
T	$\Box A \rightarrow A$	Reflexive	$\forall s(sRs)$
B	$A \rightarrow \Box\Diamond A$	Symmetric	$\forall s\forall t(sRt \rightarrow tRs)$
D	$\Box A \rightarrow \Diamond A$	Serial	$\forall s\exists t(sRt)$
4	$\Box A \rightarrow \Box\Box A$	Transitive	$\forall s\forall t\forall u(sRt \wedge tRu \rightarrow sRu)$
5	$\Diamond A \rightarrow \Box\Diamond A$	Euclidean	$\forall s\forall t\forall u(sRt \wedge sRu \rightarrow tRu)$

TABLE 1. Modal schemata and their corresponding frame properties.

can be used to characterize the number of points in a given frame. This section also shows that the K-basic fragment of modal formulae is invariant under many classes of reflexive frames.

A *frame* is a pair $\mathcal{F} = (S, R)$, where S is a non-empty set, and R is a binary relation on S. A *model* is a triple $\mathcal{M} = (S, R, V)$ where $V : \mathcal{L} \rightarrow 2^S$ is a function that assigns to each atomic formula $p \in \mathcal{L}$ a subset $V(p)$ of S. Informally speaking $V(p)$ can be thought as the set of points where the atomic formula p is 'true'.

For a modal formula A, the relation 'A is true in the point s of the model $\mathcal{M}$', denoted $\mathcal{M} \models_s A$, is defined recursively as follows:

$\mathcal{M} \not\models_s \bot$.	
$\mathcal{M} \models_s p$	if $s \in V(p)$
$\mathcal{M} \models_s A \rightarrow B$	if $\mathcal{M} \models_s A$ implies $\mathcal{M} \models_s B$.
$\mathcal{M} \models_s \Box A$	if sRt implies $\mathcal{M} \models_t A$, $\forall t \in S$.

It is easy to verify that $\mathcal{M} \models_s \Diamond A$ holds if $\exists t \in S$ such that sRt and $\mathcal{M} \models_t A$.

We say that a formula A is *true in model* $\mathcal{M}$, denoted $\mathcal{M} \models A$, if it is true in all points of $s \in S$. And a formula A is *true in the frame* $\mathcal{F}$, denoted $\mathcal{F} \models A$, if it is true for all possible models $\mathcal{M}$ based on the frame $\mathcal{F}$. Moreover, if $\mathcal{C}$ is a class of frames we say *A is true in* $\mathcal{C}$, denoted $\mathcal{C} \models A$, if it is true for every frame $\mathcal{F} \in \mathcal{C}$.

In the notation we have just presented, we may often use, instead of the single formula A, a collection of formulae. The notation is used to specify that all formulae in the collection are true in the corresponding point, model, frame or class of frames.

3.1 Frame properties and schemata

A *schema* is a collection of formulae that share a common syntactic form, i.e. the schema $\Box A \rightarrow A$ represents the collection $\{\Box B \rightarrow B \mid B \text{ is a formula}\}$. It is a common procedure in modal logics to characterize some properties on frames with respect to the validity of some particular schema in the frame. The following well-known theorem formally states the relation between some common schemata and their corresponding frame properties.

THEOREM 3.1
Let $\mathcal{F} = (S, R)$ be a frame. Each one of the schemata in Table 1 is true in the frame $\mathcal{F}$ if and only if $\mathcal{F}$ satisfies the corresponding property (Theorems 1.12 and 1.13 in [18], pp. 12 and 13).

3.2 The counting schemata

A frame is universal if its corresponding relation is universal, i.e. $R = S \times S$. We will use $\mathcal{U}_n$ to denote the universal frame with n different points, i.e. with the set $S = \{0, 1, \ldots, n-1\}$. Universal frames are important since they relate to provability in the modal logic S5 and, as we will see in this paper, they can be useful for logic programming applications. We present now some schemata that can be used to characterize the cardinality of the set S in universal frames.

DEFINITION 3.2
The *counting schema* $\mathbf{F_n}$ is defined, for every integer n, as:

$$\mathbf{F_n} := \bigwedge_{i=1}^{n} \Diamond\Big(A_i \wedge \bigwedge_{j=1}^{i-1} \neg A_j\Big) \rightarrow \Box \bigvee_{i=1}^{n} A_i \; .$$

Some particular instances are the schemata:

$$\begin{aligned}
&\mathbf{F_1} = \Diamond A \rightarrow \Box A \; , \\
&\mathbf{F_2} = \Diamond A \wedge \Diamond(B \wedge \neg A) \rightarrow \Box(A \vee B) \; , \\
&\mathbf{F_3} = \Diamond A \wedge \Diamond(B \wedge \neg A) \wedge \Diamond(C \wedge \neg A \wedge \neg B) \rightarrow \Box(A \vee B \vee C) \; .
\end{aligned}$$

PROPOSITION 3.3
Let $\mathcal{F} = (S, R)$ be a universal frame. $\mathcal{F} \models \mathbf{F_n}$ if and only if the frame satisfies $|S| \leq n$.

PROOF. We will show first that having a frame with $|S| \leq n$ implies validity of the formula $\mathbf{F_n}$. Let $\mathcal{M}$ be an arbitrary model based on $\mathcal{F}$ and suppose that the formula $\bigwedge_{i=1}^{n} \Diamond(A_i \wedge \bigwedge_{j=1}^{i-1} \neg A_j)$ is valid in $\mathcal{M}$. From this it follows that there are n different points $s_i \in S$ —for $i = 1, \ldots, n$— with $\mathcal{M} \models_{s_i} A_i$ and $\mathcal{M} \not\models_{s_i} A_j$ for $j < i$. Since $|S| \leq n$, we must have in fact $S = \{s_i\}_{i=1}^{n}$. Since A_i is true in the point s_i we may conclude that $\Box \bigvee_{i=1}^{n} A_i$ is true in the model $\mathcal{M}$.

For the other implication we follow the contrareciprocal. If $|S| > n$, there are at least $n+1$ distinct points $s_i \in S$ for $i = 1, \ldots, n+1$. Take n different atomic formulae a_i with $i = 1, \ldots, n$ and let $\mathcal{M} = (S, R, V)$ be a model based on $\mathcal{F}$ with the valuation $V(a_i) = \{s_i\}$ for every a_i in the set. It is then clear that the formula $\bigwedge_{i=1}^{n} \Diamond(a_i \wedge \bigwedge_{j=1}^{i-1} \neg a_j)$ is true in $\mathcal{M}$ while $\Box \bigvee_{i=1}^{n} a_i$ is not, since $\bigvee_{i=1}^{n} a_i$ is not true in s_{n+1}. ∎

PROPOSITION 3.4
Given two positive numbers $i < j$ then

$$\{A \mid \mathcal{U}_j \models A\} \subset \{A \mid \mathcal{U}_i \models A\} \, .$$

PROOF. We will first show that $\mathcal{U}_j \models A$ implies $\mathcal{U}_i \models A$. Let $\mathcal{M}_i$ be a model based on $\mathcal{U}_i$ with some valuation function V_i. Then define the model $\mathcal{M}_j$ based on $\mathcal{U}_j$ with a valuation $V_j \colon \mathcal{L} \rightarrow 2^{\{0,1,\ldots,j-1\}}$ as follows:

$$V_j(p) = \begin{cases} V_i(p) \cup \{i, \ldots, j-1\} & \text{if } i-1 \in V_i(p), \\ V_i(p) & \text{otherwise.} \end{cases}$$

It is very simple to show, since the relation is universal, that for $0 \leq k < i$ and every formula B:

$$\mathcal{M}_i \models_k B \quad \text{iff} \quad \mathcal{M}_j \models_k B \; .$$

Since $\mathcal{U}_j \models A$ we know that, in particular, A is true at all points of the model $\mathcal{M}_j$ that we constructed. The previous note makes it clear that A is also true at all points in the model $\mathcal{M}_i$.

To prove that the subset relation is proper just note that, by Proposition 3.3, the schema $\mathbf{F_i}$ is true in $\mathcal{U}_i$ but some instance of it is not true in $\mathcal{U}_j$. ∎

COROLLARY 3.5
A formula is true in the class of universal frames with $|S| \leq n$ if and only if it is true in the frame $\mathcal{U}_n$.

PROOF. If a formula is true in all universal frames with $|S| \leq n$ it is true, in particular, in the frame $\mathcal{U}_n$. Conversely if a formula is true in the frame $\mathcal{U}_n$, by Proposition 3.4, it is true in all universal frames with $|S| \leq n$. ∎

3.3 Frames K-equivalent

In this section we have one of the main building blocks for our principal results. It states that, with respect to K-basic formulae, many classes of frames (that share the property of being reflexive) are equivalent.

If $\mathcal{M} = (S, R, V)$ is a model based on a reflexive frame and s is a fixed point in S we can construct a model $\mathcal{M}'$ based on the universal frame with two points $\mathcal{U}_2$ such that a K-basic formula is true in the original model if and only if it is true in the prime model (see Lemma A.2 in Appendix A).

We say that two classes of frames $\mathcal{C}_1$ and $\mathcal{C}_2$ are K-equivalent if, for every K-basic formula A, it holds that $\mathcal{C}_1 \models A$ iff $\mathcal{C}_2 \models A$. Then we have that all the frames in the class of reflexive frames are K-equivalent to the frame $\mathcal{U}_2$ (see Proposition A.3 in Appendix A).

The previous facts are quite useful to determine K-equivalence as we see in the following theorem.

THEOREM 3.6
The following classes of frames are K-equivalent:

- The class of all reflexive frames.
- The class of all reflexive and transitive frames.
- The class of all symmetric, reflexive and transitive frames.
- The class of all universal frames.
- The class of all universal frames with $|S| \leq 2$.
- The class of all universal frames with $|S| \leq n$.

PROOF. All the equivalences follow immediately from Lemma A.2 and Proposition A.3 since $\mathcal{U}_2$ is a frame in all of these classes. ∎

The previous result is of great significance in order to show that the logics S4, S5, and some others we will construct agree in the class of K-basic formulae. We remark that the class of K-basic formulae is useful in computer science. Michael Gelfond, in fact, uses a subclass of K-basic formulae to characterize stable models using AEL [15].

4 Semantics using truth values

In the context of semantics based on truth values our work takes advantage of the expressive power of the $\mathcal{FOUR}$ system to interpret modal connectives. Belnap originally introduced the four-valued

logic $\mathcal{FOUR}$ [5] trying to deal in a useful way with inconsistent and incomplete information. This structure was further investigated by Ginsberg [17] who proposed bilattices that generalize $\mathcal{FOUR}$, and Fitting showed how this system is useful to provide semantics for logic programs [13].

In this section we introduce the $\mathcal{FOUR}$ structure and then we use it to define a particular truth valuation for modal formulae. We also see how $\mathcal{FOUR}$ is expressive enough to emulate other logics such as the Gödel's G_4. Finally we discover some relations between the universal frame $\mathcal{U}_2$ and the $\mathcal{FOUR}$ system.

4.1 The $\mathcal{FOUR}$ bilattice

Belnap introduced a logic for dealing in a useful way with inconsistent and incomplete information. This logic is based on a structure called $\mathcal{FOUR}$ with four truth values $\{0, 1, 2, 3\}$. These values are usually identified with the symbols $\{\bot, f, t, \top\}$ that provide an intuition of the meaning of the four truth values: the classical t and f, $\bot$ that intuitively denotes lack of information (no knowledge), and $\top$ that indicates inconsistency ('over'-knowledge). We will use, however, numbers instead of these symbols in order to keep the notation simple. These values have two different natural orderings shown in Figure 1.

- Measuring the truth:

  The minimal element is 1, the maximal element is 2 and the values 0, 3 are incomparable. Here we have the inverse involution $\neg_{tr}$, the meet and join operators denoted, respectively, as $\wedge_{tr}$ and $\vee_{tr}$.

- Reflecting differences in the amount of knowledge or information:

  The minimal element is 0, the maximal element is 3 and the values 1, 2 are incomparable. Here we have inverse involution $\neg_{kn}$, the meet and join operators denoted, respectively, as $\wedge_{kn}$ and $\vee_{kn}$.

Ginsberg proposed algebraic structures called *bilattices* that generalize Belnap's $\mathcal{FOUR}$; his motivation for introducing bilattices was to provide a uniform approach for a diversity of applications in computer science. The logical role that $\mathcal{FOUR}$ has among bilattices is very similar to the one of two-valued algebra among Boolean algebras [2].

4.2 Modal $\mathcal{FOUR}$

We define a semantic for modal propositional theories using the following definitions for the primitive modal connectives:

$$\begin{aligned} \bot &:= \neg_{tr}\neg_{kn} p \wedge_{kn} p && \text{for some atom } p \\ \Box A &:= A \wedge_{kn} \neg_{tr} A \\ A \rightarrow B &:= \neg_{tr}\neg_{kn} A \vee_{kn} B. \end{aligned}$$

It is easy to verify that, under these previous definitions, $\bot = 0$ and the other two connectives have the following truth tables. Recall that there is no 'typical' $\rightarrow$ connective in $\mathcal{FOUR}$, ours is an abbreviation form of the standard connectives. The $\neg$ connective is defined as usual in terms of $\rightarrow$ and $\bot$.

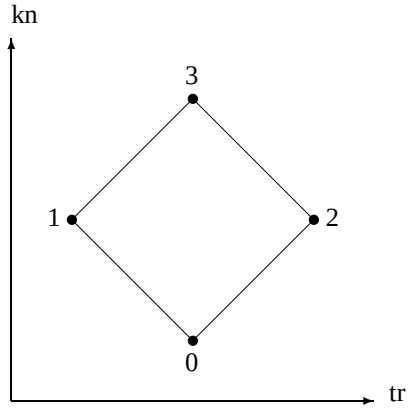

FIGURE 1. $\mathcal{FOUR}$ identified with 0, 1, 2 and 3.

A	$\Box A$
0	0
1	0
2	0
3	3

$\rightarrow$	0	1	2	3
0	3	3	3	3
1	2	3	2	3
2	1	1	3	3
3	0	1	2	3

The next step in using $\mathcal{FOUR}$ for reasoning is to choose its set of *designated* elements. As the Tetravalent Modal Algebras (TMAs) does [14], we let the largest element 3 be our designated value. Then tautologies are defined as usual, i.e. a formula A is a tautology in $\mathcal{FOUR}$ if, for every assignment of the atoms in A of values in $\{0, 1, 2, 3\}$, the formula A values the designated value at 3. We will call this logic *Modal* $\mathcal{FOUR}$. We use $\models_{\mathcal{FOUR}} A$ to denote the fact that the formula A is a tautology in $\mathcal{FOUR}$. Some examples of tautologies in Modal $\mathcal{FOUR}$ are:

$$\Box A \rightarrow A, \quad A \rightarrow \Diamond A, \quad \Box A \equiv \neg\Diamond\neg A \quad \text{and} \quad \Diamond A \equiv \neg\Box\neg A \ ,$$

and some examples of formulae that are not tautologies:

$$A \rightarrow \Box A \quad \text{and} \quad \Diamond A \rightarrow A \ .$$

4.3 Logic G_4 in terms of $\mathcal{FOUR}$

Using the expressive power of the $\mathcal{FOUR}$ bilattice we show in this section how the connectives of the multivalued logic G_4 can also be constructed. First we define a congruence operator † that, in the case of logic, can be interpreted as an identity or equivalence connective. The idea of this new connective is taken from [14] and is defined as:[1]

$$A \dagger B := \big(\neg\Box(A \vee B) \vee \Box(A \wedge B)\big) \wedge \big(\Box\neg(A \vee B) \vee \neg\Box\neg(A \wedge B)\big) \ .$$

[1] In this and the following definition the symbols $\vee$ and $\wedge$ denote the knowledge ordering, i.e. $\vee_{kn}$ and $\wedge_{kn}$ respectively

Now we are able to express the implication connective of G_4 in terms of $\mathcal{FOUR}$ using the following definition.

$$A \rightarrow_{\mathrm{G}_4} B := \Box(A \rightarrow B) \vee (\Diamond B \wedge (A \rightarrow B)) \\ \vee \big(\big(A \dagger (\neg A \wedge_{tr} A)\big) \wedge \big(B \dagger \neg_{tr}(\neg B \wedge_{tr} B)\big)\big).$$

And it is easy to verify that $\rightarrow_{\mathrm{G}_4}$ has the following truth table:

$\rightarrow_{\mathrm{G}_4}$	0	1	2	3
0	3	3	3	3
1	0	3	3	3
2	0	1	3	3
3	0	1	2	3

Moreover the connective $\neg_{\mathrm{G}_4} A$ can be defined, as usual, as an abbreviation of the formula $A \rightarrow_{\mathrm{G}_4} \bot$. Recall, however, that in this Gödel logic the usual definitions for the conjunction and disjunction do not hold. In fact the disjunction connective $\vee_{\mathrm{G}_4}$, that behaves just like the $\max$ function on numbers, can be alternatively defined as:

$$A \vee_{\mathrm{G}_4} B := (A \vee_{tr} B) \vee_{kn} (A \wedge_{kn} \neg_{\mathrm{G}_4} B) \vee_{kn} (\neg_{\mathrm{G}_4} A \wedge_{kn} B) \vee_{kn} \Box A \vee_{kn} \Box B \ .$$

Finally the conjunction connective $\wedge_{\mathrm{G}_4}$ can be obtained using the original $\mathcal{FOUR}$ negation: $A \wedge_{\mathrm{G}_4} B := \neg(\neg A \vee_{\mathrm{G}_4} \neg B)$. It is easy to verify that the defined connectives have the following truth tables:

$\vee_{\mathrm{G}_4}$	0	1	2	3
0	0	1	2	3
1	1	1	2	3
2	2	2	2	3
3	3	3	3	3

$\wedge_{\mathrm{G}_4}$	0	1	2	3
0	0	0	0	0
1	0	1	1	1
2	0	1	2	2
3	0	1	2	3

4.3.1 The frame $\mathcal{U}_2$ and $\mathcal{FOUR}$

This brief section presents one of our first basic results that relates the semantics based on the frame $\mathcal{U}_2$ and the truth values of $\mathcal{FOUR}$.

PROPOSITION 4.1
For any formula A,

$$\mathcal{U}_2 \models A \qquad \text{iff} \qquad \models_{\mathcal{FOUR}} A \ .$$

PROOF. From Lemma B.2 (see Appendix B) it follows immediately that $\mathcal{M} \models A$ iff $I(A) = 3$. For one of the implications, given $I\colon \mathcal{L} \rightarrow \{0,1,2,3\}$ we can define $\mathcal{M}^I$ using the valuation $V^I(p) = g(I(p))$ for every atom p. Completely analogously for the other implication, given a model $\mathcal{M}$ based on $\mathcal{U}_2$ with a valuation V, we can define the $\mathcal{FOUR}$ valuation $I^{\mathcal{M}}(p) = g^{-1}(V(p))$ for every atom p. ∎

5 Proof theory

In this section we will review the proof theory of modal logics. In particular we define a new logic that is based on S5 adding $\mathbf{F_n}$ as a new axiom scheme. Moreover we show that this logic is sound and complete with respect to the semantic in universal frames with n points. All the results in

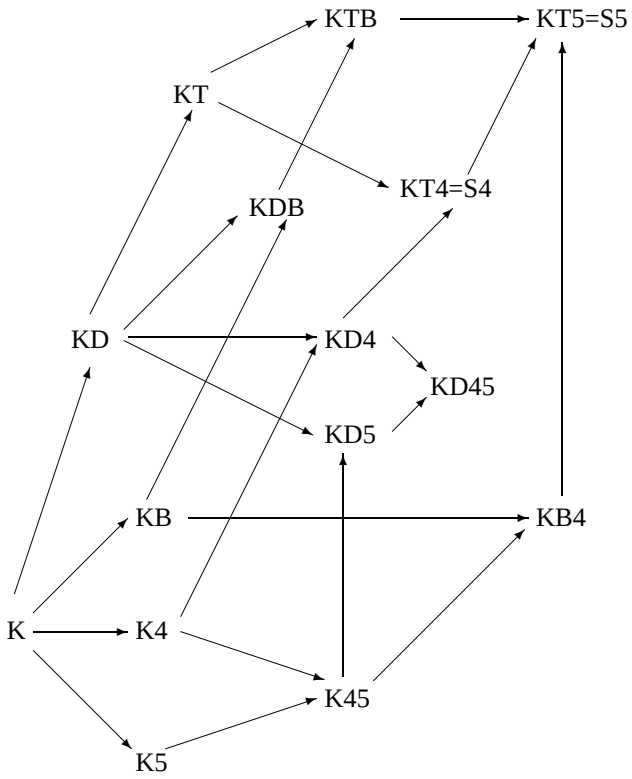

FIGURE 2. Modal logics based on axioms **T**, **B**, **D**, **4** and **5**.

previous sections are helpful here to produce interesting results about the relation of several logics in the K-basic fragment, as well as applications of $\mathcal{FOUR}$.

A *normal logic* can be defined in terms of Hilbert type proof systems as a logic that contains the following axiom schemata:

A1	$A \rightarrow (B \rightarrow A)$
A2	$(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
A3	$(\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$
K	$\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$

And is closed under the following inference rules rules:

- *Modus Ponens*: If the pair of formulae A and $A \rightarrow B$ are provable then B is provable.
- *Necessitation*: If the formula A is provable then $\Box A$ is provable.

Other axiom schemata, like those presented in Table 1, can be added to produce different normal logics. It has become customary to use the notation $\mathrm{K}\Sigma_1 \cdots \Sigma_n$ to refer the smallest normal logic that contains, in addition to axioms A1–A3, the axiom schemata **K**, Σ_1, ..., Σ_n. Figure 2 shows several normal logics, in particular S4 and S5 created by Lewis, according to this notation.

We use the notation $\vdash_\Lambda A$ to denote, for a formula A and a logic Λ, that A is provable in the axiomatic logic Λ. Formulae satisfying $\vdash_\Lambda A$ are also known as theorems of Λ.

Given a set of formulae Γ we also say that a formula A is Λ-deducible from the set Γ, denoted $\Gamma \vdash_\Lambda A$, if there exist $B_0, \ldots, B_{n-1} \in \Gamma$ such that

$$\vdash_\Lambda B_0 \to (B_1 \to (\cdots \to (B_{n-1} \to A) \ldots)) \; .$$

We write $\Gamma \nvdash_\Lambda A$ when A is not Λ-deducible from Γ. We also say that Γ is Λ-consistent if $\Gamma \nvdash_\Lambda \bot$; and Γ is Λ-maximal if Γ is Λ-consistent and, for every formula A, either $A \in \Gamma$ or $\neg A \in \Gamma$.

Given a set of formulae Δ we also use $\Gamma \vdash_\Lambda \Delta$ to denote that every formula in Δ is Λ-deducible from Γ. And we use the symbol $\Gamma \Vdash_\Lambda \Delta$ to denote that Γ is Λ-consistent and $\Gamma \vdash_\Lambda \Delta$.

5.1 Canonical models

In this section we will introduce a pair of generated models, together with some of their properties, that will be helpful to prove our completeness results. It is common to find this kind of model in completeness proofs for modal logics as in [18].

DEFINITION 5.1
Let $\mathcal{M} = (S, R, V)$ be a model and $t \in S$. The *submodel of $\mathcal{M}$ generated by t* is

$$\mathcal{M}^t = (S^t, R^t, V^t) \; ,$$

where

$$\begin{aligned} S^t &= \{u \in S \mid tR^*u\} \; , \\ R^t &= R \cap (S^t \times S^t) \; , \\ V^t(p) &= V(p) \cap S^t \; , \end{aligned}$$

and R^* is the reflexive and transitive closure of R.

LEMMA 5.2 (Submodel lemma)
For any formula A and $u \in S^t$,

$$\mathcal{M}^t \models_u A \quad \text{iff} \quad \mathcal{M} \models_u A \; .$$

DEFINITION 5.3 (Canonical model)
The *canonical model* of a consistent normal logic Λ is the structure

$$\mathcal{M}^\Lambda = (S^\Lambda, R^\Lambda, V^\Lambda) \; ,$$

where

$$\begin{aligned} S^\Lambda &= \{s \subseteq \mathrm{Fma}(\mathcal{L}) \mid s \text{ is } \Lambda\text{-maximal}\} \; , \\ sR^\Lambda t \quad &\text{iff} \quad \{A \in \mathrm{Fma}(\mathcal{L}) \mid \Box A \in s\} \subseteq t \; , \\ V^\Lambda(p) &= \left\{s \in S^\Lambda \mid p \in s\right\} \; . \end{aligned}$$

The symbol $\mathrm{Fma}(\mathcal{L})$ denotes the set of formulae built from a fixed set of atomic propositions $\mathcal{L}$.

LEMMA 5.4 (Truth lemma)
For any formula A,

$$\mathcal{M}^\Lambda \models A \qquad \text{iff} \qquad \vdash_\Lambda A.$$

THEOREM 5.5
If a normal logic Λ contains any one of the schemata in Table 1, then R^Λ satisfies the corresponding property (Theorem in [18]).

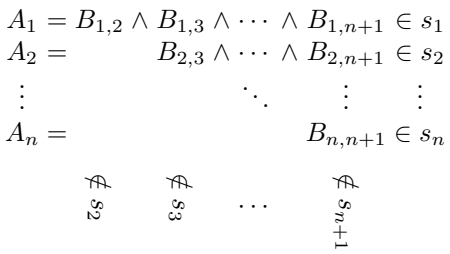

FIGURE 3. Construction of formulae for Theorem 5.6

5.2 The logic S5$_n$

Schiller in 1951 [38] proved that S5 could be characterized by an infinite multivalued logic and moreover that every proper normal extension of S5 can be characterized by a finite multivalued logic. We continue Schiller's work by studying these extensions in further detail.

The logic S5$_n$ is obtained adding the axiom schema $\mathbf{F_n}$ to the axioms of S5, this corresponds to the logic KT4BF$_n$. It is a well-known result that the logic S5 is determined by the class of universal frames (Theorem 3.10 in [18], p. 29). We will now show a similar result for the S5$_n$ logics.

THEOREM 5.6
S5$_n$ is determined by the class of universal frames with $|S| \leq n$.

PROOF. Soundness is easy. All universal frames satisfy, by Theorem 3.1, the schemata $\mathbf{T}$, $\mathbf{4}$ and $\mathbf{B}$. Moreover, since $|S| \leq n$, Proposition 3.3 shows that $\mathbf{F_n}$ is valid in the frame.

For completeness, suppose $\nvdash_{\mathrm{S5}_n} A$. By Lemma 5.4, A is false at some point t in the canonical model $\mathcal{M}^{\mathrm{S5}_2}$. Let $\mathcal{M}^t$ be the submodel of $\mathcal{M}^{\mathrm{S5}_n}$ generated by t. By the submodel lemma, A is also false at t in $\mathcal{M}^t$. Since S5$_n$ contains the schemata $\mathbf{T}$, $\mathbf{4}$ and $\mathbf{B}$ the canonical relation R^{S5_n} is, by Theorem 5.5, an equivalence relation. This shows that $(R^{\mathrm{S5}_n})^* = R^{\mathrm{S5}_n}$ and, in particular, the generated relation R^t is universal. It would suffice to show that $|S^t| \leq n$.

Suppose, by contradiction, that $|S^t| > n$ so that there are $n+1$ different points $s_i \in S$ for $i = 1, \ldots, n+1$. For every $1 \leq i < k \leq n+1$, let $B_{i,k}$ be a formula such that $B_{i,k} \in s_i$ and $B_{i,k} \notin s_k$ (it is easy to see that such formulae must exist). Let $A_i = \bigwedge_{k=i+1}^{n+1} B_{i,k}$ for $1 \leq i \leq n$. It is clear by construction, see Figure 3 for an illustration, that $A_i \in s_i$ and $j < i$ implies $\neg A_j \in s_i$. In particular $\Diamond(A_i \wedge \bigwedge_{j=1}^{i-1} \neg A_j)$ is true in $\mathcal{M}^t$. We also know that $\mathbf{F_2}$ is true in the model $\mathcal{M}^t$, follows from truth and submodel lemmas. So that, by logical consequence, $\Box \bigvee_{i=1}^{n} A_i$ should be true in $\mathcal{M}^t$. But, by construction, $A_i \notin s_{n+1}$ for $1 \leq i \leq n$ and $\Box \bigvee_{i=1}^{n} A_i$ is not true in the submodel $\mathcal{M}^t$. ∎

If we use X $\subset$ Y to denote the proper inclusion of the set of theorems for two given logics X and Y then we have the following result.

THEOREM 5.7
S5 $\subset \cdots \subset$ S5$_{i+1} \subset$ S5$_i \subset \cdots \subset$ S5$_3 \subset$ S5$_2 \subset$ S5$_1$.

PROOF. The fact that S5 is a lower bound to all the logics S5$_i$ comes from proof theory, since logics S5$_i$ have one additional axiom with respect to the axioms of the logic S5.

The fact that S5$_{i+1} \subset$ S5$_i$ follows by Theorem 5.6 and Proposition 3.4. Moreover this shows that S5 $\neq$ S5$_i$, for all natural numbers i, since there is another logic, namely S5$_{i+1}$, that also contains S5 but is properly contained in S5$_i$. ∎

The particular case of the logic $S5_2$ is also quite interesting and serves to connect our previous results in $\mathcal{FOUR}$ and the frame $\mathcal{U}_2$ with proof theory.

COROLLARY 5.8
$S5_2$ is determined by the class of universal frames with $|S| \leq 2$.

PROOF. Follows by Theorem 5.6. ∎

THEOREM 5.9
For any formula A,

$$\vdash_{S5_2} A \qquad \text{iff} \qquad \models_{\mathcal{FOUR}} A.$$

PROOF. We know $\vdash_{S5_2} A$ iff, by Corollary 5.8, A is true in the class of universal frames with $|S| \leq 2$ iff, by Corollary 3.5, A is true in the frame $\mathcal{F}_2$ iff, by Proposition 4.1, $\models_{\mathcal{FOUR}} A$. ∎

Also, with respect to K-basic formulae, we have the following result. As with classes of frames, we say that two theories X and Y are K-equivalent if, for every K-basic formula F, $\vdash_X F$ iff $\vdash_Y F$.

THEOREM 5.10
The following logics are K-equivalent:

$$\text{KT}, \quad \text{S4}, \quad \text{S5}, \quad S5_2, \quad S5_n.$$

PROOF. Follows by Theorem 3.6, Corollary 5.8, and some other well-known results of soundness and completeness, i.e. logic S4 is sound and complete with respect to reflexive and transitive frames. ∎

5.3 Alternative presentation of S5$_n$

It is important to mention that there are other possible representations for the axiomatic logics $S5_n$. The axiom $\mathbf{F_n}$ can be alternatively replaced by the following axiom schemata

$$\mathbf{F'_n} := \bigwedge_{i=1}^{n} \Diamond A_i \wedge \bigwedge_{1 \leq i < j \leq n} \neg\Diamond(A_i \wedge A_j) \rightarrow \bigvee_{i=1}^{n} A_i \ ,$$

as already sketched in [27, 28].

It is easy to prove that any of the two axiom schemata produce the same logic $S5_n$ since, in particular letting $A_i = B_i \wedge \bigwedge_{j=1}^{i-1} \neg B_j$ in the axiom scheme $\mathbf{F'_n}$ produces, after simplifications, an instance of the axiom scheme $\mathbf{F_n}$. On the other hand $\vdash_{S5} F_n \rightarrow F'_n$ where F_n and F'_n are two instances of the corresponding axiom schemata constructed with the same subformulae A_i.

6 Natural deduction systems

A *natural deduction system* is another mathematical structure useful to study and define provability relations. Here we will propose a set of inference rules that will be useful to provide a natural deduction system for normal logic programs. The main result obtained is Proposition 6.4 and it will be applied in the next section.

6.1 Motivation in resolution

It is a well-known result that resolution is sound and complete with respect to classical logic. For propositional formulae in conjunctive normal form (CNF) resolution is based on just one inference rule namely $\frac{\alpha \vee c \quad \beta \vee \neg c}{\alpha \vee \beta}$ Res and a formula F is derived in classical logic by a theory T if and only if the CNF version of $T \cup \{(\neg F)\}$ derives the empty formula (i.e. contradiction). The following inference rules appear when we try to study resolution techniques applied to normal programs.

DEFINITION 6.1
We define the following inference rules for normal logic programs,

$$\frac{x \leftarrow \alpha \wedge d \qquad d \leftarrow \beta}{x \leftarrow \alpha \wedge \beta}\ \mathsf{G} \qquad \frac{x \leftarrow \alpha \wedge c \qquad x \leftarrow \beta \wedge \neg c}{x \leftarrow \alpha \wedge \beta}\ \mathsf{Ca}$$

$$\frac{x \leftarrow \alpha \wedge c \qquad y \leftarrow \beta \wedge \neg c}{x \leftarrow \neg y \wedge \alpha \wedge \beta}\ \mathsf{R}$$

where x, y, c and d are atoms, and the formulae α, β represent an arbitrary conjunction of literals.

The notation $\frac{\alpha \quad \beta}{\gamma}$ X in previous definition denotes that the formula γ is a logic consequence of the pair of formulae α, β. Moreover, the application of an inference rule is annotated with a letter on the right (G, Ca or R as just defined) that indicates the name of the inference rule used. We usually use X to denote the application of an arbitrary inference rule.

DEFINITION 6.2
Given a set $\mathcal{S}$ of inference rules, a *proof* $P \vdash_{\mathcal{S}} \alpha$ is defined recursively as follows:

- If $\alpha \in P$ then $P \vdash_{\mathcal{S}} \alpha$.
- If $P \vdash_{\mathcal{S}} \beta$, $P \vdash_{\mathcal{S}} \gamma$ and $\frac{\beta \quad \gamma}{\alpha}$ X with some rule $\mathsf{X} \in \mathcal{S}$ then $P \vdash_{\mathcal{S}} \alpha$.

DEFINITION 6.3
Given two sets of inference rules $\mathcal{S}$ and $\mathcal{T}$, a *proof* $P \vdash_{\mathcal{S}\text{-}\mathcal{T}} \alpha$ is defined recursively as follows:

- If $P \vdash_{\mathcal{S}} \alpha$ then $P \vdash_{\mathcal{S}\text{-}\mathcal{T}} \alpha$.
- If $P \vdash_{\mathcal{S}\text{-}\mathcal{T}} \beta$, $P \vdash_{\mathcal{S}\text{-}\mathcal{T}} \gamma$ and $\frac{\beta \quad \gamma}{\alpha}$ X with some rule $\mathsf{X} \in \mathcal{T}$ then $P \vdash_{\mathcal{S}\text{-}\mathcal{T}} \alpha$.

Since these proofs have a tree-like structure, we will sometimes call them proof trees. However, we are really thinking in Hilbert-style proofs with no axioms but only hypothesis and inferences rules.

6.2 A system for normal programs

The main goal of this section is to prove that the third rule R is not required to infer a single atom from a normal logic program, the formal statement of this result given in Proposition 6.4. Thus the system GCa is enough to obtain, using resolution techniques, all the atomic inferences of a given normal logic program. We do not know if this result is already known or, more likely, can follow as a direct consequence of some particular theorem available from the theory of resolution. Moreover the proof presented here is not very 'neat' but valid and formal anyway. Nevertheless, the result is of importance for the next section.

The sketch of the proof is as follows: first we prove that the rule G can always be moved to the beginning of the proof tree so that it does not have a strong interaction with the other pair of rules

Ca and R. Then we show how, in a proof using the system CaR, it is possible to remove applications of the rule R one by one. There are some minor details that we have to handle with care but they are addressed appropriately in the development of the proof.

In order to make this section easier to read we avoid here some preliminary results needed in the following proof, but the reader can find these in Appendix D.

PROPOSITION 6.4
Let P be a normal program and let a be an atom. If $P \vdash_{\mathsf{GCaR}} a$ then $P \vdash_{\mathsf{GCa}} a$.

PROOF. Using Lemma D.2 (see Appendix D) we obtain that $P \vdash_{\mathsf{G\text{-}CaR}} a$. Let $Q = \{\delta \mid P \vdash_{\mathsf{G}} \delta\}$, so that $Q \vdash_{\mathsf{CaR}} a$. Assume that we have a proof for $Q \vdash_{\mathsf{CaR}} a$ that uses the rule R as few times as possible. We claim that, in fact, the rule R is not required. Assume that the proof does require some applications of the rule R. Let R be a set such that $Q \vdash_{\mathsf{CaR}} R$ and there are a pair of formulae $\alpha, \beta \in R$ such that $\frac{\alpha \quad \beta}{\gamma}$ R and $R, \gamma \vdash_{\mathsf{Ca}} a$. Lemma D.6 (see Appendix D) shows that $R \vdash_{\mathsf{GCa}} a$. This shows that there is a proof of $Q \vdash_{\mathsf{GCaR}}$ with less occurrences of the rule R in the proof tree. But again, using Lemma D.2 (see Appendix D), $Q \vdash_{\mathsf{G\text{-}CaR}} a$ with the same number of uses of the rule R in the proof. But, since Q is already closed for the rule G, it is possible to show that $Q \vdash_{\mathsf{CaR}} a$ with less applications of the rule R arising contradiction. ∎

7 Nonmonotonic reasoning and logic programming

In this final section all the results and constructions previously developed find their conclusion. First we will provide a general definition of 'semantics' for logic programs, in particular we briefly describe the popular semantics of answer sets and WFS. Then we present GNM-S5, a proposal for a new semantic whose behaviour is closer to the notion of well-behaved semantics as defined by Dix [9, 10]. We also prove several properties of this new semantics that serve to justify the previous claim. We would like to stress that the notion of 'well behaved semantics' served us only as motivation. The reader does not really need to accept or know about this topic in order to follow the rest of our paper. However, the interested reader is invited to see [9, 10] for more information on this subject.

7.1 Semantics for logic programs

A *model* or *interpretation* of a program P is a set $M \subseteq \mathcal{L}_P$. This kind of interpretations are closely related with two valued truth functions since, intuitively, the set M denotes the atoms in P that are considered as 'true' while the atoms not in M are to be taken as 'false'. Given a class of programs C, a *scenarios semantic operator* is a function S that assigns to each program $P \in C$ a set of possible interpretations for P [11].

There is another kind of semantics that assign one single intended model to each logic program. Formally a *one-model semantic operator* is a function S that assigns to each program P, in some given class C, a subset of literals from P. This models are usually consistent (i.e. there is no atom a such that both a and $\neg a$ appear in the model) or total (i.e. all the literals relevant to P). Moreover, observe that these models have a three-valued nature: an atom can appear either positive or negative in the model (to denote a 'true' or 'false' fact respectively) or not appear at all (denoting a third 'undetermined' state).

Given a scenarios semantic function S it is also common to define the *skeptical semantics* S^{sk} for

each program P as follows:

$$S^{sk}(P) = \bigcap_{M \in S(P)} (M \cup \neg(\mathcal{L}_P \setminus M)) \ .$$

Observe that the skeptical version of a scenarios semantics corresponds to a one-model semantics.

Given the one-model semantic functions S_1 and S_2, we define: $S_1 \leq S_2$ if for every program P it is true that $S_1(P) \subseteq S_2(P)$. The relations $S_1 = S_2$ and $S_1 < S_2$ are defined as usual. We say that one semantic operator is stronger than the other according to this ordering.

Answer sets. A-Prolog, Stable Logic Programming [16] or Answer Set Programming is the realization of much theoretical work on nonmonotonic reasoning and AI applications of logic programming in the last 15 years. The semantic of answer sets is a scenarios semantics that supports one of the logic programming paradigms with greatest acceptance in the community. Efficient software to compute answer sets and a large list of applications to model real-life problems justify this assertion. The two most well-known systems that compute stable models are `dlv` [20] and `smodels` [26].

A characterization of answer sets in terms of intuitionistic logic I has been provided as follows: a literal is entailed by a disjunctive program in the answer set semantics if and only if it belongs to every I-complete and consistent extension of the program formed by adding only negated atoms [33]. We adopted the following formal notation to express this fact[2]:

$$\text{AS}(P) = \{M \subseteq \mathcal{L}_P \mid P \cup \neg(\mathcal{L}_P \setminus M) \Vdash_{\text{I}} M\} \ .$$

This logical approach provides the foundations to define the notion of nonmonotonic inference of any propositional theory (using the standard connectives) in terms of a monotonic logic (namely intuitionistic logic I), see [29, 30, 31, 33]. Notions such as conservative extensions and transformations, equivalence, strong equivalence (see [21, 29]), among others are now better understood thanks to this logical approach. These notions are very important if one wants to push forward a logic based program development approach.

Moreover, in [32] it was proved that any intermediate logic (strictly weaker than classical logic and stronger or equal to intuitionistic) can also be used, instead of intuitionistic logic, to characterise answer sets. In particular we have that

$$\text{AS}(P) = \{M \subseteq \mathcal{L}_P \mid P \cup \neg(\mathcal{L}_P \setminus M) \Vdash_{\text{G}_4} M\} \ .$$

It follows, since the results presented in subsection 4.3 show how the logic G_4 can be emulated using the connectives of $\mathcal{FOUR}$, that the test condition for a model to be considered an answer can also be evaluated using $\mathcal{FOUR}$ and the set of defined connectives $\{\neg_{\text{G}_4}, \rightarrow_{\text{G}_4}, \wedge_{\text{G}_4}, \vee_{\text{G}_4}\}$.

Well-founded semantics. The well-founded semantics (WFS) is another paradigm that developed in parallel to the answer set semantics [39]. The main difference between AS and WFS is that in the definition of the former a *guess* is made and then a particular (2-valued) model is constructed and used to justify the guess or to reject it. However, in the definition of WFS, more and more atoms are declared to be true (or false): once a decision has been drawn, it will never be rejected. WFS is defined as a one-model semantics and is based on a 3-valued intended model.

There are some cases, however, where the well-founded semantics does not give the expected results. As the following example from [1] shows, the program $P = \{p \leftarrow \neg q, q \leftarrow \neg p, r \leftarrow p, r \leftarrow q\}$ has two stable models. The first model satisfies p while the second satisfies q. The authors in [1]

[2] Recall that $\Vdash_{\text{X}}$ was defined earlier in the paper

also mention that the formula $p \vee q$ should be, in a reasonable definition for a semantic operator, considered valid and thus set the value the atom r to true. The WFS operator leaves, contrary to this intuition, the atom r undefined.

Several extensions of WFS have been proposed, trying to overcome this kind of arguments against the semantic, that integrate some additional mechanisms on top of the original definition (i.e. EWFS [8], GWFS [4], WFS$^+$ [6]). Dix noticed, however, that many of these new semantics have more serious shortcomings than the original WFS and hence he defined a set of principles which all semantics should be checked against [10]. Such notions helped Dix to propose the concept of *well behaved semantics*.

7.2 The GNM-S5 semantics

Several attempts have been made to model the notion of nonmonotonic reasoning using modal logics. McDermott and Doyle introduced nonmonotonic versions of modal logics by the use of expansions. Given a modal theory T an *X-expansion* of T, based on the modal X, is a theory E that satisfies the equation

$$E = \mathrm{Cn}_{\mathrm{X}}(T \cup \{\neg\Box F \mid F \notin E\}) \ ,$$

where $\mathrm{Cn}_{\mathrm{X}}(\Gamma)$ denotes the consequence closure of the theory Γ with respect to the particular logic X, i.e. $\mathrm{Cn}_{\mathrm{X}}(\Gamma) = \{F \mid \Gamma \vdash_{\mathrm{X}} F\}$. Depending on the approach some particular X-expansion or the intersection of all X-expansions is considered as the set of nonmonotonic consequences of T.

This is a useful method to produce, for instance, a non-monotonic version of the logic S4. We expected, however, S5 to be a better approach to model the notions of logic programming. Very disappointing was that, as McDermott was able to prove, the nonmonotonic version of S5 is equivalent to the plain monotonic logic S5.

Observe that McDermott's expansions are constructed adding formulae to the original theory. Another approach could be to consider only simple formulae of the form $\neg\Box a$, with a an atom. In fact, Gelfond [15] was able to characterize the answer set semantics of stratified normal programs using AEL with this idea and the following translation: A normal clause

$$a_0 \leftarrow a_1 \wedge \cdots \wedge a_n \wedge \neg a_{n+1} \wedge \cdots \wedge \neg a_{n+m} \ ,$$

is translated to the modal formula

$$a_0 \leftarrow a_1 \wedge \cdots \wedge a_n \wedge \neg\Box a_{n+1} \wedge \cdots \wedge \neg\Box a_{n+m} \ .$$

Generalizing this idea we propose the following translation of basic into K-basic formulae.

DEFINITION 7.1
We define the translation $^\circ$ for basic formulae into K-basic modal formulae as follows:

$$p^\circ = p \qquad (F \wedge G)^\circ = F^\circ \wedge G^\circ$$
$$(\neg p)^\circ = \neg\Box p \qquad (F \vee G)^\circ = F^\circ \vee G^\circ$$
$$(F \rightarrow G)^\circ = F^\circ \rightarrow G^\circ$$

Then we are able to propose a framework to define semantics for basic logic programs in terms of modal logics.

DEFINITION 7.2
Given a modal logic Λ we define the GNM$_\Lambda$ semantics for the class of basic programs as follows: A set of atoms $M \subseteq \mathcal{L}_P$ is an intended model of P, if the following equation is satisfied:

$$M = \mathrm{ACn}_{\mathrm{X}}(P^\circ \cup \{\neg\Box a \mid a \in \mathcal{L}_P \setminus M\}) \ ;$$

where $\mathrm{ACn_X}(\Gamma)$ denotes the atomic consequence closure of the theory Γ with respect to logic X, i.e. $\mathrm{ACn_X}(\Gamma) = \{a \in \mathcal{L}_\Gamma \mid \Gamma \vdash_{\mathrm{X}} a\}$.

By our previous Theorem 5.10 it follows that any of the modal logics KT, S4, S5, S5$_n$ and S5$_2$ produce the same semantic operator when used as in the previous definition. Moreover, by Theorem 5.9, we can use the $\mathcal{FOUR}$ truth values to validate this condition for any of this logics. We will therefore omit the subscript Λ when it is K-equivalent to, say, logic S5. For historical reasons we call it GNM-S5 semantic. Note that other modal logics without the reflexive frame property, logic K for instance, can produce different semantics as the following example shows.

EXAMPLE 7.3
Take the following logic program P:

$$b \leftarrow \neg a.$$
$$a \leftarrow \neg b.$$
$$a \leftarrow \neg p.$$
$$p \leftarrow \neg p.$$

It is easy to verify that GNM-S5$(P) = \{\{a, p\}, \{b, p\}\}$, while $\mathrm{GNM_K}(P) = \emptyset$.

There are some counterexamples of the 'well' behaviour for several WFS extensions (such as GWFS and EWFS) that do not apply for GNM-S5. In particular GNM-S5 is different from GWFS, EWFS, WFS$^+$, and the revised stable semantics. The following examples serve to prove this affirmation.

EXAMPLE 7.4
Consider the EWFS semantics that only has a skeptical semantics. The CUT rule and the following program P, all of them taken from [9]:

$$a \leftarrow \neg a.$$
$$b \leftarrow \neg x \wedge a.$$
$$y \leftarrow \neg b.$$
$$z \leftarrow \neg y.$$

Here EWFS$(P) = \{a, b, \neg x\}$, however EWFS$(P \cup \{b\}) = \{a, b, z, \neg x, \neg y\}$. This example shows that the EWFS semantics does not satisfy the CUT rule. But GNM-S5$(P) =$ GNM-S5$(P \cup \{b\}) = \{a, b, z, \neg x, \neg y\}$. In particular the semantics GNM-S5 is different from EWFS. Here of course GNM-S5 refers to its skeptical version.

EXAMPLE 7.5
Consider the following two program examples taken from [9]:

$$p \leftarrow \neg b.$$
$$b \leftarrow c.$$
$$c \leftarrow p \wedge \neg a.$$
$$a \leftarrow \neg b.$$

and

$$p \leftarrow \neg b.$$
$$b \leftarrow p \wedge \neg a.$$
$$a \leftarrow \neg b.$$

One may expect the same semantics of both programs, at least with respect to the common language. However, GWFS infers p in the first program, but it does not in the second. GNM-S5 gives the same answer in both programs which consists in deriving only $\neg c$, under the skeptical version. Hence, GNM-S5 is different to GWFS.

EXAMPLE 7.6
Consider the program P, taken from [11]:

$a \leftarrow \neg b.$
$b \leftarrow \neg a.$
$x \leftarrow \neg a.$
$x \leftarrow \neg b.$

Note that GNM-S5$(P) = \{\{a, x\}, \{b, x\}\}$. Hence, its skeptical version gives GNM-S5$(P) = \{x\}$. This shows that GNM-S5 is different from WFS$^+$, since WFS$^+$ does not infers x.

EXAMPLE 7.7
Consider this last program P, taken from [34]:

$a \leftarrow \neg b.$
$x \leftarrow \neg y.$
$b \leftarrow \neg a.$
$y \leftarrow \neg x.$
$x \leftarrow a \wedge \neg c.$
$z \leftarrow x \wedge \neg z.$

Note that GNM-S5$(P) = \{\{a, c, x, z\}, \{a, c, y\}, \{b, x, z\}, \{b, y\}\}$. This shows that GNM-S5 is different from the revised stable semantics by Pereira and Pinto. The difference is that they exclude $\{a, c, x, z\}$ from the solution. However, they consider the interesting choice to include it. In order to do it, they also propose in [34] the notion of combination revised stable models (CRSM) that adds again $\{a, c, x, z\}$ to the solution. We do not know at this point if both semantics GNM-S5 and CRSM agree. CRSM is presented in a very different setting than GNM-S5 and is defined just for normal programs.

In the following definition we restrict our attention to skeptical semantics. We say that a semantic S satisfies the extended Cut principle if for every program P and pair of literals a and b it holds that: $a \in S(P)$ and $b \in S(P \cup \{a\})$ implies that $b \in S(P)$, see [9].

We have the informal claim that the particular interpretation that Dix gives to the notion of well behaved semantics has a small defect and in order to correct it we propose to reject to interpret $P \cup M$ as the reduct operation P^M (for a definition on the reduct operator see [10]). Note that the notion P^M is a syntactic transformation, not required when $P \cup M$ has a logical meaning.

Take for instance, the following program $P = \{a \leftarrow b, b \leftarrow \neg a\}$ also presented in [10]. This example is used to show that WFS$^+$ does not satisfy the Extended Cut principle. While WFS$^+(P) = \{a, \neg b\}$, WFS$^+(P \cup \{\neg b\}) = \{\neg a, \neg b\}$. But the set $\{\neg a, \neg b\}$ is neither a 2-valued model, nor a 3-valued model of the logic program $P \cup \{\neg b\}$. This happens because WFS$^+$ does not have a 'logical' definition for programs with constraints (negated formulae). Hence, Dix has to interpret $P \cup \{\neg b\}$ as $P^{\{\neg b\}} = \{a \leftarrow b\}$.

GNM-S5 is defined for any basic propositional theory and, in particular, allows the use of constraints. In this particular example, with the same logic program P, we get GNM-S5$(P) =$ GNM-S5$(P \cup \{\neg b\}) = \{a, \neg b\}$. In fact, thanks to the logical approach of GNM-S5 it is straightforward to prove that:

PROPOSITION 7.8
GNM-S5 satisfies Extended Cut.

We propose to reconsider Dix's work on well-behaved semantics, towards the direction of making it more general and logical based.

Classicality. One feature in nonmonotonic reasoning that has been recognized as important by many authors is that semantics should have a closed behaviour with respect to classical logic [7, 9, 10, 11, 36]. Classicality, a condition that shows part of this expected behaviour, basically states that atomic classical consequences of a program P can be added to the logic program without modifying its semantics. The results in Section 6 serve to prove that GNM-S5 satisfies supraclassicality.

PROPOSITION 7.9
Let P be a normal program and let α be a normal formula. It follows that

$$P \vdash_{\mathsf{C}} \alpha \quad \text{if and only if} \quad P \cup T \vdash_{\mathsf{GCaR}} \alpha \ ,$$

where $T = \{a \leftarrow a \mid a \in \mathcal{L}_P\}$.

PROOF. It is well known that classical logic is sound and complete with respect to resolution. In fact $P \vdash_{\mathsf{C}} F$ if and only if $P \cup \{\neg F\}$ yields $\bot$ (the empty formula) using resolution. For the case of an atomic formula a it is easy to see that this is equivalent to the program P deriving a by the use of resolution. It is a simple exercise to verify that, with respect to the syntax of normal formulae, resolution takes the form of the inference rules G, Ca and R. There is the need, in fact, of a fourth syntactic transformation rule namely:

$$\frac{x \leftarrow \neg x \wedge \alpha}{x \leftarrow \alpha}$$

But, having the set of basic tautologies T (namely, formulae of the form $x \leftarrow x$, for every atom x) this rule follows as a particular case of the application of Ca. ∎

PROPOSITION 7.10
Let P be a normal program and let a be an atom. If $P \vdash_{\mathsf{C}} a$ then $P \models_{\mathcal{FOUR}} a$.

PROOF. Suppose that $P \vdash_{\mathsf{C}} a$, by Propositions 7.9 and 6.4 we may conclude that $P \cup T \vdash_{\mathsf{GCa}} a$. But again, it is easy to observe that the inference rules G and Ca are valid in Modal $\mathcal{FOUR}$; moreover the set T is not needed since it contains basic tautologies already part of Modal $\mathcal{FOUR}$. So, finally $P \models_{\mathcal{FOUR}} a$. ∎

THEOREM 7.11 (Classicality)
Let P be a normal program and let a be an atom. If $P \vdash_{\mathsf{C}} a$ then $a \in$ GNM-S5(P). Moreover Supraclassicality also holds, namely that GNM-S5$(P) =$ GNM-S5$(P \cup \{a\})$.

PROOF. By Proposition 7.10 we know that $P \models_{\mathcal{FOUR}} a$ and, from this, it follows that $P \equiv_{\mathcal{FOUR}} P \cup \{a\}$ making both programs interchangeable in the definition of GNM-S5 with respect to Modal $\mathcal{FOUR}$. ∎

8 Conclusions

In this paper we were able to show how the extensive available background of mathematical logic in general and modal logics in particular is quite useful to study and better understand logic programming and/or NMR. We also note how a very general framework, based on completions with

simple negated formulae and parametrized with a particular logic, can be used to model several non-monotonic systems.

We propose in particular a new semantic operator whose behaviour is closer to the WFS^+ semantics that satisfies several of the principles of well-behaved semantics while keeping a closed relation with respect to classical logic. Moreover, we see how $\mathcal{FOUR}$, a four-valued inference system, is expressive enough to characterize both answer sets and our proposed new semantic. We conjecture that a three-valued approach is also possible by the use of, for instance, the Gödel's (G_3) and Lukasevich's (Ł_3) tree-valued logics.

Interesting lines of research for future work could include the study of other well-behaved conditions on the proposed new semantic and its properties when dealing with broader classes of programs (i.e. disjunctive, nested, ...). It would also by interesting to study other forms of completions based, for instance, on negated literals (not only atoms) or some sort of basic modal formulae. We expect that this kind of results will be helpful to bring logic and logic programming even closer, and serve to better understand important notions such as nonmonotonic reasoning.

References

[1] K. R. Apt and R. Bol. Logic programming and negation: A survey. *Journal of Logic Programming*, **19/20**, 9–71, 1994.

[2] A. Avron. On the expressive power of three-valued and four-valued languages. *Journal of Logic and Computation*, **9**, 977–994, 1999.

[3] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, 2003.

[4] C. Baral, J. Lobo and J. Minker. Generalized Well-founded Semantics for Logic Programs. In *10th International Conference on Automated Deduction*, M. E. Stickel, ed., pp. 102–116. *Lecture Notes in Artificial Intelligence* 449, Springer-Verlag, Berlin, 1990.

[5] N. D. Belnap. A useful four-valued logic. In *Modern Uses of Multiple-Valued Logic*, J. Michael Dunn and G. Epstein, eds, pp. 8–37. D. Reidel, 1977.

[6] G. Brewka, J. Dix, and K. Konolige. *Nonmonotonic Reasoning: An Overview*. CSLI Lecture Notes 73. CSLI Publications, Stanford, CA, 1997.

[7] M. Denecker, N. Pelov and M. Bruynooghe. Ultimate well-founded and stable semantics for logic programs with aggregates. In *Logic Programming. 17th International Conference, ICLP 2001*, pp. 212–226. Vol. 2237 in *Lecture Notes in Computer Science*, Springer, 2001.

[8] J. Dix. Cumulativity and Rationality in Semantics of Normal Logic Programs. In *Proceedings of the first Workshop on Nonmonotonic and Inductive Logic 1990 in Karlsruhe*, J. Dix, K. P. Jantke, and P. H. Schmitt, eds, pp. 13–37. Vol. 543 of *Lecture Notes in Computer Science*, Springer, 1991.

[9] J. Dix. A classification theory of semantics of normal logic programs: I. strong properties. *Fundamental Informaticae*, **22**, 227–255, 1995.

[10] J. Dix. A classification theory of semantics of normal logic programs: II. weak properties. *Fundamental Informaticae*, **22**, 257–288, 1995.

[11] J. Dix, M. Osorio and C. Zepeda. A general theory of confluent rewriting systems for logic programming and its applications. *Annals of Pure and Applied Logic*, **108**, 153–188, 2001.

[12] F. M. Donini, D. Nardi and R. Rosati. Ground nonmonotonic modal logics. *Logic and Computation*, **7**, 1997.

[13] M. C. Fitting. Bilattices and the semantics of logic programming. *Journal of Logic Programming*, **11**, 91–116, 1991.

[14] J. Maria Font and M. Rius. An absract logic approach to tetravalent modal logics. *Journal of Symbolic Logic*, **65**, 481–518, 2000.

[15] M. Gelfond. On stratified auto-epistemic theories. In *Proceedings of AAAI*, pp. 207–211. Morgan Kaufman, 1987.

[16] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *5th Conference on Logic Programming*, R. Kowalski and K. Bowen, eds, pp. 1070–1080. MIT Press, 1988.

[17] M. Ginsberg. Multivalued logics. *Computational Intelligence*, **4**, 1988.

[18] R. Goldblatt. *Logics of Time and Computation*. Number 7 in CSLI Lecture Notes. Stanford, 2nd edn, 1992.

[19] K. Konolige. On the relation between default and autoepistemic logic. In *Readings in Nonmonotonic Reasoning*, M. L. Ginsberg, ed., pp. 195–226. Kaufmann, Los Altos, CA, 1987.

[20] N. Leone, G. Pfeifer, W. Faber, F. Calimeri, T. Dell'Armi, T. Eiter, G. Gottlob, G. Ianni, G. Ielpa, C. Koch, S. Perri and A. Polleres. The DLV system. In *JELIA '02: Proceedings of the European Conference on Logics in Artificial Intelligence*, S. Flesca, S. Greco, G. Ianni and N. Leone, eds, pp. 537–540. Vol. 2424 of *Lecture Notes in Computer Science*, Springer-Verlag, 2002.

[21] V. Lifschitz, D. Pearce and A. Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, **2**, 526–541, 2001.

[22] W. Marek and M. Truszczyński. *Nonmonotonic Logics; Context-Dependent Reasoning*. Springer, Berlin, 1st edn, 1993.

[23] D. McDermott. Nonmonotonic logic II: Nonmonotonic modal theories. *ACM Transactions on Computer Systems*, **29**, 33–57, 1982.

[24] D. McDermott and J. Doyle. Non-monotonic logic I. *Artificial Intelligence*, **13**, 41–72, 1980.

[25] R. C. Moore. Autoepistemic logic. In *Non-Standard Logics for Automated Reasoning*, P. Smets, E. H. Mamdani, D. Dubois, and H. Prade, eds. Academic Press, 1988.

[26] I. Niemelä, P. Simons and T. Syrjänen. Smodels: A system for answer set programming. In *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning*, Breckenridge, Colorado, USA, April 2000.

[27] M. Osorio, V. Borja and J. Arrazola. Closing the gap between the stable semantics and extensions of WFS. In *Mexican International Conference on Artificial Intelligence*, pp. 202–211, 2004.

[28] M. Osorio and J. A. Navarro. Modal logic $S5_2$ and FOUR (abstract). In *2003 Annual Meeting of the Association for Symbolic Logic*, Chicago, June 2003.

[29] M. Osorio, J. A. Navarro and J. Arrazola. Equivalence in answer set programming. In *Logic Based Program Synthesis and Transformation. 11th International Workshop, LOPSTR 2001*, A. Pettorossi, ed., pp. 57–75. Vol. 2372 of *Lecture Notes in Computer Science*, Springer 2001.

[30] M. Osorio, J. A. Navarro and J. Arrazola. A logical approach for A-Prolog. In *9th Workshop on Logic, Language, Information and Computation (WoLLIC)*, R. de Queiroz, L. C. Pereira and E. H. Haeusler, eds, pp. 265–275. Vol. 67 of *Electronic Notes in Theoretical Computer Science*, Elsevier, 2002.

[31] M. Osorio, J. A. Navarro and J. Arrazola. Applications of intuitionistic logic in answer set programming. *Theory and Practice of Logic Programming*, **4**, 325–354, 2004.

[32] M. Osorio, J. A. Navarro and J. Arrazola. Safe beliefs for propositional theories. Accepted to appear at Annals of Pure and Applied Logic, 2004.

[33] D. Pearce. Stable inference as intuitionistic validity. *Logic Programming*, **38**, 79–91, 1999.

[34] L. M. Pereira and A. M. Pinto. Revised stable models - a new semantics for logic programs. In *Convegno Italiano di Logica Computazionale (CILC'04)*, Parma, Italy, July 2004.

[35] J. S. Schlipf. Formalizing a logic for logic programming. In *Proceedings of the International Symposium on Artificial Intelligence '90*, 1990.

[36] J. S. Schlipf. Formalizing a logic for logic programming. *Annals of Mathematics and Artificial Intelligence*, **5**, 279–302, 1992.

[37] G. Schwarz. Autoepistemic logic of knowledge. In *Proceedings of LPNMR*, pp. 260–274. *Lecture Notes in Artificial Intelligence*, 1991.

[38] S. J. Scroggs. Extensions of the Lewis system S5. *Journal of Symbolic Logic*, **16**, 112–120, 1951.

[39] A. van Gelder, K. A. Ross and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, **38**, 620–650, 1991.

Appendix

A K-basic formulae in reflexive frames

DEFINITION A.1

Let $\mathcal{M} = (S, R, V)$ be a model based on a reflexive frame $\mathcal{F}$, and let s be a fixed point in S. We define $\mathcal{M}'$ as the model based on $\mathcal{U}_2$, the universal frame with two points, with the valuation $V' \colon \mathcal{L} \to 2^{\{0,1\}}$ defined as follows:

$$V'(p) = \begin{cases} \{0,1\} & \text{if } s \in V(p) \text{ and } T \subseteq V(p) \\ \{0\} & \text{if } s \in V(p) \text{ and } T \not\subseteq V(p) \\ \{1\} & \text{if } s \notin V(p) \text{ and } T \cap V(p) \neq \emptyset \\ \emptyset & \text{if } s \notin V(p) \text{ and } T \cap V(p) = \emptyset \end{cases}$$

where $T = \{t \mid sRt \text{ and } s \neq t\}$.

LEMMA A.2
Let $\mathcal{M} = (S, R, V)$ be a model based on a reflexive frame $\mathcal{F}$, and let s be a fixed point in S. If A is a K-basic formula then

$$\mathcal{M} \models_s A \text{ if and only if } \mathcal{M}' \models_0 A \ .$$

PROOF. The proof is by structural induction over the construction of the K-basic formula A:

- $A = \bot$. The result follows trivially since neither $\mathcal{M} \models_s \bot$ nor $\mathcal{M}' \models_0 \bot$.
- $A = p$, for atomic p. It follows immediately by the definition of V' since

$$\mathcal{M} \models_s p \text{ iff } s \in V(p) \text{ iff } 0 \in V'(p) \text{ iff } \mathcal{M}' \models_0 p \ .$$

- $A = \neg p$, for atomic p. Analogously

$$\mathcal{M} \models_s \neg p \text{ iff } s \notin V(p) \text{ iff } 0 \notin V'(p) \text{ iff } \mathcal{M}' \models_0 \neg p \ .$$

- $A = \Box p$, for atomic p.

$$\begin{array}{lll} \mathcal{M} \models_s \Box p \text{ iff } (\forall t, sRt \text{ implies } \mathcal{M} \models_t p) & & \text{by definition} \\ \quad \text{iff } \{t \mid sRt\} \subseteq V(p) & & \\ \quad \text{iff } V'(p) = \{0, 1\} & & \text{by definition of } V' \\ \quad \text{iff } \mathcal{M}' \models_0 p \text{ and } \mathcal{M}' \models_1 p & & \\ \quad \text{iff } \mathcal{M}' \models_0 \Box p \ . & & \end{array}$$

- $A = \Box\neg p$, for atomic p.

$$\begin{array}{lll} \mathcal{M} \models_s \Box\neg p \text{ iff } (\forall t, sRt \text{ implies } t \notin V(p)) & & \text{by definition} \\ \quad \text{iff } \{t \mid sRt\} \cap V(p) = \emptyset & & \\ \quad \text{iff } V'(p) = \emptyset & & \text{by definition of } V' \\ \quad \text{iff } \mathcal{M}' \models_0 \neg p \text{ and } \mathcal{M}' \models_1 \neg p & & \\ \quad \text{iff } \mathcal{M}' \models_0 \Box\neg p \ . & & \end{array}$$

- $A = B \rightarrow C$, for some K-basic formulae B, C.

$$\begin{array}{lll} \mathcal{M} \models_s B \rightarrow C \text{ iff } (\mathcal{M} \models_s B \text{ implies } \mathcal{M} \models_s C) & & \text{by definition} \\ \quad \text{iff } (\mathcal{M}_s \models_0 B \text{ implies } \mathcal{M}_s \models_0 C) & & \text{by induction} \\ \quad \text{iff } \mathcal{M}_s \models_0 B \rightarrow C & & \text{by definition.} \end{array}$$

∎

PROPOSITION A.3
Let $\mathcal{C}$ be a class of frames such that every frame in $\mathcal{C}$ is reflexive and the frame $\mathcal{U}_2 \in \mathcal{C}$. Then, for every K-basic formula A:

$$\mathcal{C} \models A \quad \text{iff} \quad \mathcal{U}_2 \models A \ .$$

PROOF. The 'only if' part is trivial since $\mathcal{U}_2 \in \mathcal{C}$. For the 'if' part we will prove the contrareciprocal, i.e. $\mathcal{C} \not\models A$ implies $\mathcal{U}_2 \not\models A$. Assume $\mathcal{C} \not\models A$, then there is a frame $\mathcal{F} = (S, R) \in \mathcal{C}$, a model $\mathcal{M}$ based on $\mathcal{F}$, and a point $s \in S$ such that $\mathcal{M} \not\models_s A$. Let $\mathcal{M}'$ be constructed as in Definition A.1 so that, by Lemma A.2, $\mathcal{M}' \not\models_0 A$. Since the model $\mathcal{M}'$ is based on $\mathcal{U}_2$, this shows that $\mathcal{U}_2 \not\models A$. ∎

B The frame $\mathcal{U}_2$ and $\mathcal{FOUR}$

DEFINITION B.1
Let $g\colon \{0, 1, 2, 3\} \rightarrow 2^{\{0,1\}}$ be the mapping:

$$\begin{array}{ll} g(0) = \emptyset & g(2) = \{1\} \\ g(1) = \{0\} & g(3) = \{0, 1\} \ . \end{array}$$

LEMMA B.2
Let $I\colon \mathcal{L} \to \{0,1,2,3\}$ be a $\mathcal{FOUR}$ valuation. The domain of the valuation is extended to arbitrary modal formulae as defined in subsection 4.2. Also let $\mathcal{M}$ be an arbitrary model for the frame $\mathcal{U}_2$ with some valuation V. If it turns out that $V(p) = g(I(p))$ for every propositional formula p then, for every formula A,

$$\mathcal{M} \models_s A \qquad \text{iff} \qquad s \in g(I(A))\ .$$

PROOF. The proof is by induction on the length of the formula A, there are three cases:

- $A = p$, for atomic p. The is trivial since $\mathcal{M} \models_s p$ iff $s \in V(p) = g(I(p))$.
- $A = B \to C$. To simplify writing the proof let $W(F) = \{s \mid \mathcal{M} \models_s F\}$ for any formula F. Observe that the lemma's statement is equivalent to show that $W(A) = g(I(A))$. Also note that $g(I(\neg_{tr}\neg_{kn}F)) = S \setminus g(I(F))$ and $g(I(F \vee_{kn} G)) = g(I(F)) \cup g(I(G))$, where $S = \{0,1\}$, and F, G are any pair of formulae. From this it follows:

$$\begin{aligned}
W(A \to B) &= \{s \mid \mathcal{M} \models_s B \to C\} \\
&= \{s \mid \mathcal{M} \models_s B \text{ implies } \mathcal{M} \models_s C\} && \text{by definition} \\
&= \{s \mid s \in W(B) \text{ implies } s \in W(C)\} \\
&= (S \setminus W(B)) \cup W(C) && \text{by simple set theory} \\
&= (S \setminus g(I(B))) \cup g(I(C)) && \text{by induction} \\
&= g(I(\neg_{tr}\neg_{kn}B \vee_{kn} C)) && \text{by previous notes} \\
&= g(I(B \to C)) && \text{by definition.}
\end{aligned}$$

- $A = \Box B$. The proof is as follows:

$$\begin{aligned}
\mathcal{M} \models_s \Box B \quad &\text{iff} \quad \forall t \in \{0,1\}, \mathcal{M} \models_t B && \text{by definition in } \mathcal{F}_2 \\
&\text{iff} \quad \forall t \in \{0,1\}, t \in g(I(B)) && \text{by induction} \\
&\text{iff} \quad g(I(B)) = \{0,1\} \\
&\text{iff} \quad g(I(\Box B)) = \{0,1\} && \text{by definition} \\
&\text{iff} \quad s \in g(I(\Box B))\ .
\end{aligned}$$

For the last step observe that $s \in g(I(\Box B))$ implies that the set $g(I(\Box B))$ is not empty, and the only possible case is $g(I(\Box B)) = \{0,1\}$. The other implication is trivial.

∎

C Inference rule transitions

Left side transitions.

Rule Ca

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \wedge c \qquad x \leftarrow \beta \wedge \neg c}{x \leftarrow \alpha \wedge \beta \wedge d}\,\mathsf{Ca} \qquad d \leftarrow \gamma}{x \leftarrow \alpha \wedge \beta \wedge \gamma}\,\mathsf{G}$$

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \wedge c \qquad d \leftarrow \gamma}{x \leftarrow \alpha \wedge \gamma \wedge c}\,\mathsf{G} \qquad x \leftarrow \beta \wedge \neg c}{x \leftarrow \alpha \wedge \beta \wedge \gamma}\,\mathsf{Ca}$$

Rule R

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \wedge c \qquad y \leftarrow \beta \wedge \neg c}{x \leftarrow \neg y \wedge \alpha \wedge \beta \wedge d}\,\mathsf{R} \qquad d \leftarrow \gamma}{x \leftarrow \neg y \wedge \alpha \wedge \beta \wedge \gamma}\,\mathsf{G}$$

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \wedge c \qquad d \leftarrow \gamma}{x \leftarrow \alpha \wedge \gamma \wedge c}\,\mathsf{G} \qquad y \leftarrow \beta \wedge \neg c}{x \leftarrow \neg y \wedge \alpha \wedge \beta \wedge \gamma}\,\mathsf{R}$$

Right side transitions.

Rule Ca

$$\dfrac{x \leftarrow \alpha \wedge d \qquad \dfrac{d \leftarrow \beta \wedge c \qquad d \leftarrow \gamma \wedge \neg c}{d \leftarrow \beta \wedge \gamma}\,\mathsf{Ca}}{x \leftarrow \alpha \wedge \beta \wedge \gamma}\,\mathsf{G}$$

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \qquad d \leftarrow \beta \wedge c}{x \leftarrow \alpha \wedge \beta \wedge c}\,\mathsf{G} \qquad \dfrac{x \leftarrow \alpha \wedge d \qquad d \leftarrow \gamma \wedge \neg c}{x \leftarrow \alpha \wedge \gamma \wedge \neg c}\,\mathsf{G}}{x \leftarrow \alpha \wedge \beta \wedge \gamma}\,\mathsf{Ca}$$

Rule R

$$\dfrac{x \leftarrow \alpha \wedge d \qquad \dfrac{d \leftarrow \beta \wedge c \qquad y \leftarrow \gamma \wedge \neg c}{d \leftarrow \neg y \wedge \beta \wedge \gamma}\,\mathsf{R}}{x \leftarrow \neg y \wedge \alpha \wedge \beta \wedge \gamma}\,\mathsf{G}$$

$$\dfrac{\dfrac{x \leftarrow \alpha \wedge d \qquad d \leftarrow \beta \wedge c}{x \leftarrow \alpha \wedge \beta \wedge c}\,\mathsf{G} \qquad y \leftarrow \gamma \wedge \neg c}{x \leftarrow \neg y \wedge \alpha \wedge \beta \wedge \gamma}\,\mathsf{R}$$

D Preliminary results Section 6

LEMMA D.1
If $P \vdash_{\mathsf{CaR}} \alpha$, $P \vdash_{\mathsf{CaR}} \beta$ and $\frac{\alpha \quad \beta}{\gamma}\,\mathsf{G}$ then $P \vdash_{\mathsf{G\text{-}CaR}} \gamma$. Moreover it is possible to ensure that the number of times that R is used in $P \vdash_{\mathsf{G\text{-}CaR}} \gamma$ is the sum of the number of times that it is used in $P \vdash_{\mathsf{CaR}} \alpha$ and $P \vdash_{\mathsf{CaR}} \beta$.

PROOF. The proof is by induction on $n = s + t$ where s and t are the sizes of the proofs for α and β respectively. If $n = 0$ then both $\alpha \in P$ and $\beta \in P$, it follows immediately that $P \vdash_{\mathsf{G}} \gamma$ and, therefore, also $P \vdash_{\mathsf{G\text{-}CaR}} \gamma$.

Assume now that $n > 0$, then one of the two proofs if not empty. Let us suppose that the size of the proof $P \vdash_{\mathsf{CaR}} \alpha$ is non-zero, the other case is analogous. Then we have that $P \vdash_{\mathsf{CaR}} \delta$ and $P \vdash_{\mathsf{CaR}} \phi$ for a pair of clauses such that $\frac{\delta \quad \phi}{\alpha}\,\mathsf{Ca\ or\ R}$. The situation is illustrated for clarity.

$$\dfrac{\dfrac{\delta \qquad \phi}{\alpha}\,\mathsf{Ca\ or\ R} \qquad \beta}{\gamma}\,\mathsf{G}$$

Left side transitions, see Appendix C, show how to construct a formula α' such that $\frac{\delta\ (\text{or } \phi) \quad \beta}{\alpha'}\,\mathsf{G}$. Applying the inductive hypothesis we obtain that $P \vdash_{\mathsf{G\text{-}CaR}} \alpha'$. Left side transitions also show how $\frac{\alpha' \quad \phi\ (\text{or } \delta)}{\gamma}\,\mathsf{Ca\ or\ R}$ so that we may conclude $P \vdash_{\mathsf{G\text{-}CaR}} \gamma$.

If the size of the proof $P \vdash_{\mathsf{CaR}} \alpha$ is zero then the size of the proof $P \vdash_{\mathsf{CaR}} \beta$ must be non-zero. In this case we can follow a similar argument but using the right side transitions. Note that the right side transition for rule Ca shows how to provide two formulae, say α' and α'', with shorter proofs in which induction can be applied. Then γ is derived from α' and α'' and the same argument already provided follows. We can also see at Appendix C that in every case the number of times that R is used is preserved. In the application of rule Ca, one may think that the number of times that R is used could increase since the proof of $x \leftarrow \alpha \wedge d$ is required twice. However, is just the conclusion that we need to reuse and hence the number of times that R is used can be really preserved. ∎

LEMMA D.2
If $P \vdash_{\mathsf{GCaR}} \alpha$ then $P \vdash_{\mathsf{G\text{-}CaR}} \alpha$. Moreover it is possible to ensure that R is used the same number of times in $P \vdash_{\mathsf{G\text{-}CaR}} \alpha$ as in $P \vdash_{\mathsf{GCaR}} \alpha$.

PROOF. The proof is by induction on n, the size of the proof for α. If $n = 0$ then the result is immediate since $\alpha \in P$ and $P \vdash_{\mathsf{G\text{-}CaR}} \alpha$.

Assume now that $n > 0$. Then $P \vdash_{\mathsf{GCaR}} \beta$ and $P \vdash_{\mathsf{GCaR}} \gamma$ for a pair of clauses such that $\frac{\beta \quad \gamma}{\alpha}$ G, Ca or R. We can use the inductive hypothesis to obtain $P \vdash_{\mathsf{G\text{-}CaR}} \beta$ and $P \vdash_{\mathsf{G\text{-}CaR}} \gamma$ using R the same number of times as their corresponding cases. If the actual rule used is either Ca or R we can easily conclude since, by definition, $P \vdash_{\mathsf{G\text{-}CaR}} \alpha$.

On the other case, if the rule used is G, let $Q = \{\delta \mid P \vdash_{\mathsf{G}} \delta\}$. Observe that $Q \vdash_{\mathsf{CaR}} \beta$, $Q \vdash_{\mathsf{CaR}} \gamma$ and, using Lemma D.1, we obtain that $Q \vdash_{\mathsf{G\text{-}CaR}} \alpha$. It follows then, by the construction of Q, that $P \vdash_{\mathsf{G\text{-}CaR}} \alpha$. It is easy to see that the property about the use of R is also preserved. ∎

LEMMA D.3
Let P be a normal program and let P' be a program such that $(a \leftarrow A) \in P$ implies $(a \leftarrow A') \in P'$ for some $A' \subseteq A$. If $P \vdash_{\mathsf{Ca}} a \leftarrow A$ then $P' \vdash_{\mathsf{Ca}} a \leftarrow A'$ for some $A' \subseteq A$.

PROOF. The proof is by induction over n, the length of the proof of $P \vdash_{\mathsf{Ca}} a \leftarrow A$. If $n = 0$ then $(a \leftarrow A) \in P$ and, by hypothesis, there is a clause $(a \leftarrow A') \in P'$ with $A' \subseteq A$, it follows that $P' \vdash_{\mathsf{Ca}} a \leftarrow A'$.

If $n > 0$ then we must have $A = B \cup C$ for a pair of sets B, C such that $P \vdash_{\mathsf{Ca}} a \leftarrow B \wedge y$ and $P \vdash_{\mathsf{Ca}} a \leftarrow C \wedge \neg y$. Using the inductive hypothesis we obtain $P' \vdash_{\mathsf{Ca}} a \leftarrow B'$ and $P' \vdash_{\mathsf{Ca}} a \leftarrow C'$ for some sets $B' \subseteq B \cup \{y\}$ and $C' \subseteq C \cup \{\neg y\}$ respectively. If it is the case that $y \notin B'$ (or $\neg y \notin C'$) we are done by taking $A' = B' \subseteq B \subseteq A$ (or $A' = C'$). If it is not the case then let $B'' = B' \setminus \{y\}$ and $C'' = C' \setminus \{\neg y\}$, we can use the Ca rule on previous premises to obtain $P' \vdash_{\mathsf{Ca}} B'' \cup C''$. In this final case letting $A' = B'' \cup C''$ we solve the lemma. ∎

For the following lemma we remind the reader about the notion $\bar{x}$ defined earlier in the paper.

LEMMA D.4
If $P, (a \leftarrow \bar{x}) \vdash_{\mathsf{Ca}} a \leftarrow A$ with $\bar{x} \notin A$ then $P \vdash_{\mathsf{Ca}} a \leftarrow A'$ with $A' \subseteq A \cup \{x\}$.

PROOF. The proof is by induction over n, the length of the proof. If $n = 0$ then, since $\bar{x} \notin A$, we obtain that $(a \leftarrow A) \in P$. It follows immediately that $P \vdash_{\mathsf{Ca}} a \leftarrow A$ and $A \subseteq A \cup \{x\}$.

If $n > 0$ then we must have $A = B \cup C$ for a pair of sets B, C such that $P, (a \leftarrow \bar{x}) \vdash_{\mathsf{Ca}} a \leftarrow B \wedge y$ and $P, (a \leftarrow \bar{x}) \vdash_{\mathsf{Ca}} a \leftarrow C \wedge \neg y$. Recall that $\bar{x} \notin A = B \cup C$ and, therefore, we have both $\bar{x} \notin B$ and $\bar{x} \notin C$. If $x = y$ we can apply the inductive hypothesis to the first subproof obtain $P \vdash_{\mathsf{Ca}} a \leftarrow B'$ for some $B' \subseteq B \cup \{x\} \subseteq A \cup \{x\}$ and, letting $A' = B'$, find a solution to the lemma conditions. A similar argument follows in case $x = \neg y$ using induction on the second subproof.

Otherwise, neither $x = y$ nor $x = \neg y$, we can apply the inductive hypothesis to both subproofs obtaining $P \vdash_{\mathsf{Ca}} a \leftarrow B'$ and $P \vdash_{\mathsf{Ca}} a \leftarrow C'$ for some sets $B' \subseteq B \cup \{x, y\}$ and $C' \subseteq C \cup \{x, \neg y\}$ respectively. If it is the case that $y \notin B'$ (or $\neg y \notin C'$) we are done by taking, again, $A' = B'$ (or $A' = C'$). If it is not the case then let $B'' = B' \setminus \{y\}$ and $C'' = C' \setminus \{\neg y\}$, we can use the Ca rule on previous premises to obtain $P \vdash_{\mathsf{Ca}} B'' \cup C''$. In this final case letting $A' = B'' \cup C''$ we solve the lemma. ∎

COROLLARY D.5
If $P, (a \leftarrow \bar{x} \wedge B) \vdash_{\mathsf{Ca}} a$ then either $P \vdash_{\mathsf{Ca}} a$ or $P \vdash_{\mathsf{Ca}} a \leftarrow x$.

PROOF. Using Lemma D.3 we get that $P, (a \leftarrow \bar{x}) \vdash_{\mathsf{Ca}} a$ and then, from Lemma D.4, $P \vdash_{\mathsf{Ca}} a \leftarrow X$ for some $X \subseteq \{x\}$. So that either $X = \emptyset$ or $X = \{x\}$. ∎

LEMMA D.6
If $\frac{\alpha \quad \beta}{\gamma}$ R and $P, \alpha, \gamma \vdash_{\mathsf{Ca}} a$ then $P, \alpha, \beta \vdash_{\mathsf{GCa}} a$.

PROOF. Observe that, since only Ca rules are allowed in the proof of a then both γ and α must have the atom a in the head. It follows that α must have the form $a \leftarrow A \wedge c$, while $\beta = b \leftarrow B \wedge \neg c$ so that $\gamma = a \leftarrow \neg b \wedge A \wedge B$. Also define the formula γ' as $a \leftarrow A \wedge B$.

Using Corollary D.5 we get that either $P, \alpha \vdash_{\mathsf{Ca}} a$, in which case we are done, or $P, \alpha \vdash_{\mathsf{Ca}} a \leftarrow b$. In the second case observe that

$$\dfrac{(\alpha)\ a \leftarrow A, c \qquad \dfrac{a \leftarrow b \qquad (\beta)\ b \leftarrow B, \neg c}{a \leftarrow B, \neg c}\ \mathsf{G}}{(\gamma')\ a \leftarrow A, B}\ \mathsf{Ca}$$

so we have $P, \alpha, \beta \vdash_{\mathsf{GCa}} \gamma'$. But recall that, using Lemma D.3 on the original lemma's hypothesis, also $P, \alpha, \gamma' \vdash_{\mathsf{Ca}} a$. This way we can finally conclude that $P, \alpha, \beta \vdash_{\mathsf{GCa}} a$. ∎

Received 6 October 2004