EPiC Series in Computing

Volume 57, 2018, Pages 196–213

LPAR-22. 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning

Reasoning About Prescription and Description Using Prioritized Default Rules

Valentin Cassano[12], Carlos Areces[12], and Pablo Castro[13]

[1] Consejo Nacional de Investigaciones Científicas y Técnicas, Argentina
[2] Sección de Computación, FaMAF, UNC, Argentina
[3] Departamento de Computación, FCEFQyN, UNRC, Argentina
{vcassano,areces}@famaf.unc.edu.ar pcastro@dc.exa.unrc.edu.ar

Abstract

In this paper we introduce a prioritized default logic. We build this logic modularly from Standard Deontic Logic by the addition of default rules and priorities among them. Our main aim is to provide a logical framework to reason about scenarios where prescriptive and descriptive statements coexist and may be incomplete and contradictory. We motivate and illustrate the technical elements of our work with the use of examples (classical, and coming from software engineering). In addition, we present sound, complete, and terminating (with loop check) tableau-based proof calculi for credulous and sceptical reasoning in our logic.

1 Introduction

Prescriptive and descriptive statements are easily found in many fields: Artificial Intelligence, Software Engineering, Legal Argumentation, etc. On the one hand, prescriptive statements refer to ideal behaviors or situations: they indicate how "things ought to be". On the other hand, descriptive statements refer to particular states of affairs: they indicate how "things are". The logical treatment of prescriptive and descriptive statements goes back to the pioneer work of von Wright [49]. Nowadays, it mostly constitute part of Modal Logic [10], with *Standard Deontic Logic* (SDL) being its most famous member (see [15] for a brief introduction to SDL; and [7, 30] for in-depth introductions and historical remarks). SDL is the basic modal logic interpreted in serial models. SDL copes well with some elements of representing, and reasoning from, prescriptive and descriptive statements, e.g., violations. At the same time, it is also acknowledged that it exhibits some shortcomings in situations involving *contrary-to-duties*, prescriptions arising in the context of violations; or, more generally, in scenarios involving stratified prescriptions. These shortcomings are usually exposed by so-called *paradoxes*, prescriptive and descriptive statements that are intuitively consistent but whose logical formalization gives rise to a logical inconsistency.

The nature and logical structure of paradoxes has led some researchers to incorporate tools and techniques coming from non-monotonic logics as a way of handling scenarios involving stratified and overriding prescriptions. We contribute to this line of work. More precisely, we introduce a prioritized default logic built modularly from SDL by the addition of prioritized default rules. Default logics are among the best-known non-monotonic logics. Work on default logics started with Reiter's seminal paper 'A Logic for Default Reasoning' [43] and continued with many variants and *addenda* to Reiter's original ideas [3, 4, 6, 12]. Default logics are characterized by default rules (or defaults). Defaults provide a way of reasoning with incomplete information by augmenting what is known, with what is plausible but cannot be obtained by classical reasoning. They also provide a way of reasoning in the presence of inconsistencies circumventing the classical idea that from a contradiction everything follows. Furthermore, they provide a way of reasoning with defeasible conclusions, i.e., conclusions which can be withdrawn in the light of new information. These features of defaults make them particularly useful to model conditional obligations and what follows from them. The incorporation of priorities among defaults offers the possibility of a faithful representation of scenarios involving stratified prescriptions and overriding. Our main aim is to provide a framework to reason about scenarios where prescriptive and descriptive statements coexist and may be incomplete and contradictory. We illustrate the technical elements of our work via examples.

Initial ideas combining non-monotonic and deontic logics go back to the work of Horty [32], Alchourron [1], and Nute [40]. In brief, Horty adapts Reiter's non-monotonic framework [43] to reason about norms; Alchourron presents a dyadic logic for capturing defeasible reasoning using a revision operator; and Nute described a framework for defeasible reasoning over deontic formulas. Following these seminal work, many other authors developed different kinds of non-monotonic deontic systems. Worth mentioning are the approaches of McCarty [37], proposing to encode normative default reasoning in ProLog; the framework given by van der Torre et al. [45], formalizing different types of obligation overriding; and the more recent non-monotonic systems described in [48] and [39]. The former uses dynamic semantics for epistemic modalities to reason about deontic formulas, and the latter adapts Horty's ideas to reason about rights and contracts between agents. Also worth mentioning are the approaches presented in [25] and [29]. In [25], Goble presents a formalism to reason about *prima facie* obligations introducing an operator to capture conditional obligations, and using a notion similar to the one of extensions we introduce below to provide semantics. One of the main differences between this and our approach is that we use default rules containing prerequisites, justifications, and consequents, thus allowing for a finer look at the notion of extension. In [29], Hansen uses maximally consistent sets to define the semantics of conditional obligation (deontic alternatives). This is related to the use of extensions in our approach, as they may be seen as possible consistent situations where norms are fulfilled. Our approach gains in interest since it is built on SDL, which directly enables the use of tableau methods for constructing extensions. The work reported in [46] is also closely related to ours. Therein a ternary deontic operator O is presented. A formula $O(\alpha \mid \beta \setminus \gamma)$ reads: 'α should be the case if β is the case, unless γ is the case'. As explained by the authors, the 'unless' clause is analogous to the justification in a default rule. This said, the semantics of O is given by means of a modal logic, in contrast to the usual notion of extension for default rules. The approach, however, provides interesting intuitions and examples that illustrate the use of these kinds of normative operators.

We also like to draw attention to the fact that the systems presented in most of the works mentioned above either present no proof calculus, or present Hilbert style proof calculi. Here, we propose a non-monotonic proof calculus for our logic based on tableaux for SDL. The proof calculus is sound, complete, and terminating (with loop check). We believe that having such a proof calculus is important. On the one hand, it bridges the gap between Default Logic and Theorem Proving in a way that avoids the use of meta-logical arguments involving extensions. On the other hand, our proof calculus offers a succinct representation of a consequence relation for our logic, shedding insight into its properties. Tableau systems are popular proof systems in Modal Logic with well-known benefits (see [27] for a good introduction to modal tableau systems). They allow for systematic proof constructions, and the exploration and production of counterexamples. Our proof calculus retains these properties. To the best of our knowledge no tableau-based proof calculus has been proposed in the literature combining Default Logic and SDL. Tableau systems for Default Logic have been developed in, e.g., [2, 44]. Ours is based on ideas found in [13]. Our proof calculus is constructed over tableaux for SDL, but the techniques can be used to obtain non-monotonic versions of tableaux for other deontic logics.

One additional characteristic of our approach is the inclusion of the universal modality [26] into the framework of SDL. This modality becomes useful as it permits a clean presentation of statements that have a global status, in addition to those that have local impact. Its inclusion simplifies the definition of consequence, and of the proof system.

Structure. In Section 2 we recall the basics of SDL. In Section 3 we introduce our extension of Prioritized Default Logic, called $\mathscr{D}$SDL, built on SDL. In Section 4 we motivate and illustrate some of the technical elements of our work with the use of examples. In Section 5 we cover the basic definitions of a tableau-based proof calculus for SDL and introduce a tableau-based proof calculus for $\mathscr{D}$SDL. Finally, in Section 6 we discuss our work and present some final remarks. Proofs of results can be found in Appendix A.

2 Deontic Preliminaries

By *Standard Deontic Logic* (SDL), we mean basic modal logic over the class of models with a serial accessibility relation (see, e.g., [15, 7, 30]). We extend SDL with the global modality A (see [26]), and call the resulting system SDLA. We define the syntax and semantics of SDLA below. We assume basic knowledge of Modal Logic [9, 42].

Syntax: Let $\mathscr{P}$ be a denumerable set of *proposition symbols.* p, q, r are variables for proposition symbols. Lowercase Roman letters (possibly with indices) indicate members of $\mathscr{P}$. The set $\mathscr{F}$ of wffs of SDLA is determined by the grammar:

$$\mathscr{F} ::= p \mid \neg\mathscr{F}_1 \mid \mathscr{F}_1 \wedge \mathscr{F}_2 \mid \mathsf{O}\mathscr{F}_1 \mid \mathsf{A}\mathscr{F}_1$$

The symbols $\neg$ and $\wedge$ are the *boolean logical connectives* of *negation* and *conjunction.* O is the *(deontic) modality* of *obligation.* A is the *universal modality.* The symbols $\top$, $\bot$, $\vee$, and $\supset$ (*truth*, *falsity*, *disjunction*, and *material implication*) are defined from $\neg$ and $\wedge$ in the usual way. The deontic modality of *permission* P is $\neg\mathsf{O}\neg$. The existential modality E is $\neg\mathsf{A}\neg$. The members of $\mathscr{F}$ are *formulas.* Lowercase and uppercase Greek letters indicate formulas and sets of formulas, respectively.

Semantics: The formulas in $\mathscr{F}$ are interpreted over *Kripke models* that are *serial.* A Kripke model, notation $\mathfrak{M}$, is a tuple $\langle W, R, v\rangle$ where: W is a non-empty set; $R \subseteq W \times W$; $v : \mathscr{P} \to 2^W$. The elements of W are called *worlds*, R is the *accessibility relation*, and v is the *valuation function.* $\mathfrak{M}$ is *serial* if for all $w \in W$, there is $w' \in W$ s.t. wRw'. Define the ternary relation

$\mathfrak{M}$ *satisfies* φ at w, notation $\mathfrak{M}, w \models \varphi$, according to the following rules:

$$\begin{array}{lll} \mathfrak{M}, w \models p & \text{iff} & w \in v(p) \\ \mathfrak{M}, w \models \neg\varphi & \text{iff} & \mathfrak{M}, w \not\models \varphi \\ \mathfrak{M}, w \models \varphi \wedge \psi & \text{iff} & \mathfrak{M}, w \models \varphi \text{ and } \mathfrak{M}, w \models \psi \\ \mathfrak{M}, w \models \mathsf{O}\varphi & \text{iff} & \text{for all } w' \in W, \text{ if } wRw', \text{ then } \mathfrak{M}, w' \models \varphi \\ \mathfrak{M}, w \models \mathsf{A}\varphi & \text{iff} & \text{for all } w' \in W, \mathfrak{M}, w' \models \varphi \end{array}$$

For $\Phi \subseteq \mathscr{F}$, $\mathfrak{M}$ *satisfies* Φ *at* w, notation $\mathfrak{M}, w \models \Phi$, if, for all $\varphi \in \Phi$, $\mathfrak{M}, w \models \varphi$. $\mathfrak{M}$ *validates* Φ, notation $\mathfrak{M} \models \Phi$, if, for all w, $\mathfrak{M}, w \models \Phi$.

Logical Consequence: There are two reasonable notions of *logical consequence* between sets of formulas (assumptions), and formulas (their consequences), that can be built on $\models$: *global logical consequence*, notation $\models_g$; and *local logical consequence*, notation $\models_l$. Define $\Phi \models_g \varphi$ iff for every Kripke model $\mathfrak{M}$, if $\mathfrak{M} \models \Phi$, then, $\mathfrak{M} \models \varphi$. Define $\Phi \models_l \varphi$ iff for every Kripke model $\mathfrak{M}$ and world w, if $\mathfrak{M}, w \models \Phi$, then, $\mathfrak{M}, w \models \varphi$. It follows that $\models_l \subseteq \models_g$, but not *vice versa*. The global modality A enables us to uniformly handle global and local logical consequence: $\Phi \models_g \varphi$ iff $\mathsf{A}(\Phi) \models_l \varphi$, where $\mathsf{A}(\Phi) = \{\mathsf{A}\varphi \mid \varphi \in \Phi\}$ [26]. For this reason, we define logical consequence simply as local logical consequence and drop the superscript l. We write $\models \varphi$ if $\emptyset \models \varphi$. If Φ is finite, $\Phi \models \varphi$ iff $\models (\bigwedge\Phi) \supset \varphi$.

3 Prioritized Deontic Default Logic

Default Logic is a class of Non-monotonic Logics characterized by *defaults* and *extensions*. Defaults and extensions are arrangements of formulas of an underlying logic. Extensions and consequence in the underlying logic are used as a way of justifying a notion of *default consequence*. A Default Logic is called *prioritized* when defaults are structured in a predefined order, imposing additional constraints on extensions [12, 19, 20]. Prioritized Default Logics are useful in many application areas as they naturally impose an ordering among defaults. For instance, if we take defaults as formalizing defeasible conditional norms, we can easily represent the fact that some norms precede others. We introduce a particular Prioritized Default Logic, called $\mathscr{D}$SDLA, built on SDLA. Our presentation brings default consequence to the fore.

Syntax: Formulas are members of $\mathscr{F}$, i.e., formulas of SDLA. Defaults are members of $\mathscr{D}$. The set $\mathscr{D}$ contains all triples (π, ρ, χ), notation $\pi \overset{\rho}{\rightsquigarrow} \chi$, where $\{\pi, \rho, \chi\} \subseteq \mathscr{F}$. The formulas π, ρ, and χ, are called the *prerequisite*, the *justification*, and the *consequent*, of the default, respectively. We single out the uppercase Greek letter Δ for sets of defaults and the lowercase Greek letter δ for defaults. We define $\Pi_\Delta = \{\pi \mid \pi \overset{\rho}{\rightsquigarrow} \chi \in \Delta\}$, $P_\Delta = \{\rho \mid \pi \overset{\rho}{\rightsquigarrow} \chi \in \Delta\}$, and $X_\Delta = \{\chi \mid \pi \overset{\rho}{\rightsquigarrow} \chi \in \Delta\}$.

Semantics: The set $\mathscr{A}$ contains all tuples $\langle \Phi, \Delta, \prec \rangle$ with Φ and Δ finite subsets of $\mathscr{F}$ and $\mathscr{D}$, respectively, and $\prec$ a partial order on Δ. The members of $\mathscr{A}$ are *(default) assumption sets*. The semantics of $\langle \Phi, \Delta, \prec \rangle$ is given by extensions and logical consequence in SDLA. In this setting, an extension may be seen as an interpretation structure of a syntactical kind (taking the usual role of a model). Extensions are defined in Def. 3.4.

Definition 3.1. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$; then, $\pi \overset{\rho}{\rightsquigarrow} \chi$ is *triggered* by Δ' in $\mathbf{A}$ if $\Phi \cup X_{\Delta'} \models \pi$. It is *blocked* by Δ' in $\mathbf{A}$ if there is $\varrho \in (P_{\Delta'} \cup \rho)$ s.t. $\Phi \cup X_{\Delta'} \cup \chi \models \neg\varrho$. It is *detached* by Δ' in $\mathbf{A}$ if it is triggered and not blocked by Δ' in $\mathbf{A}$.

Example 1. Let $\delta_1 = p \overset{q}{\rightsquigarrow} \neg r$ and $\delta_2 = \top \overset{r}{\rightsquigarrow} \neg q$, and $\Delta = \{\delta_1, \delta_2\}$; both δ_1 and δ_2 are detached by $\emptyset$ in $\langle \{p\}, \Delta, \emptyset \rangle$. δ_1 is blocked by $\{\delta_2\}$ in $\langle \{p\}, \Delta, \emptyset \rangle$. δ_2 is blocked by $\{\delta_1\}$ in $\langle \{p\}, \Delta, \emptyset \rangle$.

Definition 3.2. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, and $\Delta' \subseteq \Delta$; then, $\delta \in (\Delta - \Delta')$ is *enabled* in $\mathbf{A}$ w.r.t. Δ' if δ is detached by Δ' in $\mathbf{A}$; and there is no other $\delta'' \in (\Delta - \Delta')$ detached by Δ' in $\mathbf{A}$ s.t. $\delta'' \prec \delta$.

Example 2. Let $\mathbf{A} = \langle \Phi, \{\delta_1, \delta_2\}, \{\delta_1 \prec \delta_2\} \rangle$. If δ_1 and δ_2 are detached by $\emptyset$ in $\mathbf{A}$, then δ_1 is enabled in $\mathbf{A}$ w.r.t. $\emptyset$, and δ_2 is not. If δ_2 is detached by $\{\delta_1\}$ in $\mathbf{A}$, then δ_2 is enabled in $\mathbf{A}$ w.r.t. $\{\delta_1\}$. If δ_1 is not detached by $\emptyset$ in $\mathbf{A}$ but δ_2 is, then δ_2 is enabled in $\mathbf{A}$ w.r.t. $\emptyset$.

Definition 3.3. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$; then, $\Delta' = \bigcup_{0 \leq i} \Delta'_i$ is a *series of enabled detachments* if $\Delta'_0 = \emptyset$ and

$$\Delta'_{i+1} = \begin{cases} \Delta'_i \cup \{\delta'\} & \text{if } \delta' \in (\Delta - \Delta'_i) \text{ is enabled in } \mathbf{A} \text{ w.r.t. } \Delta'_i \\ \Delta'_i & \text{otherwise.} \end{cases}$$

Definition 3.4 (Extension)**.** The set of extensions of $\langle \Phi, \Delta, \prec \rangle$, notation $\mathscr{E}(\Phi, \Delta, \prec)$, has as its elements all $\Phi \cup X_{\Delta'}$ s.t. Δ' is a series of enabled detachments (and only those elements).

Example 3. $\mathscr{E}(\{p\}, \{p \overset{\neg q}{\rightsquigarrow} r, \top \overset{\neg r}{\rightsquigarrow} q\}, \{p \overset{\neg q}{\rightsquigarrow} r \prec \top \overset{\neg r}{\rightsquigarrow} q\}) = \{\{p, r\}\}$.

Default Consequence: We define default consequence in $\mathscr{D}$SDLA as a relation $\approx\!\!\subseteq \mathscr{A} \times \mathscr{F}$. If $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\!\approx} \varphi$, we say that φ is a *default consequence* of $\langle \Phi, \Delta, \prec \rangle$. If $\prec = \emptyset$, we simply write $\langle \Phi, \Delta \rangle \mathrel{|\!\!\approx} \varphi$. Two commonly accepted views of default consequence are *sceptical* and *credulous*. Formally, φ is a sceptical default consequence of $\langle \Phi, \Delta, \prec \rangle$, notation $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\!\approx}^s \varphi$, iff *for every* extension E of $\langle \Phi, \Delta, \prec \rangle$, $E \models \varphi$. In turn, φ is a credulous default consequence of $\langle \Phi, \Delta, \prec \rangle$, notation $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\!\approx}^c \varphi$, iff *for some* extension E of $\langle \Phi, \Delta, \prec \rangle$, $E \models \varphi$. We use $\mathrel{|\!\!\approx}$ when there is no need to distinguish between sceptical and credulous default consequence. Monotonicity in $\mathscr{D}$SDLA reads: if $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\!\approx} \varphi$, then, $\langle \Phi \cup \Phi', \Delta \cup \Delta', \prec' \rangle \mathrel{|\!\!\approx} \varphi$, where $\prec \subseteq \prec'$. $\mathscr{D}$SDLA is non-monotonic.

Some Remarks and Easily Established Properties of Extensions: Def. 3.4 corresponds to extensions as defined by Łukaszewicz in [35] (modulo priorities). This definition of extension is better behaved than Reiter's [43], in particular, by this definition any $\langle \Phi, \Delta, \prec \rangle$ always has at least one extension. In addition, this definition of extensions guarantees *semi-monotonicity*, i.e., for every extension E of $\langle \Phi, \Delta, \prec \rangle$, there is an extension E' of $\langle \Phi, \Delta \cup \Delta', \prec' \rangle$ s.t. $E \subseteq E'$, where $\prec \subseteq \prec'$ and for all $\delta_1, \delta_2 \in \Delta$ and $\delta' \in \Delta'$, if $\delta_1 \prec' \delta'$ and $\delta' \prec' \delta_2$, then $\delta_1 \prec \delta'$ and $\delta \prec \delta_2$. This formulation of semi-monotonicity allows us to define a tableau based proof calculus that can take advantage of a partial use of default assumption sets, see Section 5. Clearly, $\prec$ defines a prioritized default logic [12, 19, 20]. If $\delta_1 \prec \delta_2$, we say that δ_1 precedes, or has higher priority than, δ_2. Our view of $\prec$ differs from some standard approaches, e.g., [12], in that defaults are not included in an extension if this depends on defaults with lesser priority. This is closely related to the "prescriptive" approach in [19, 20] (single-pass breadth-first iteration only).

4 Some Features of the Formalism

The examples in this section are meant to clarify some concepts and definitions. They also motivate and illustrate some of the technical elements of our work. We start analysing a classical example from the literature in our formalism. We then move to our intended field of application on the modelling of software requirements. The following terminology is useful: $\pi \overset{\rho}{\rightsquigarrow} \chi$ is *normal*, notation $\pi \rightsquigarrow \chi$, if $\rho = \chi$.

Forrester's Paradox. First presented in [23], this paradox is a puzzle involving contrary-to-duty reasoning. We recall this paradox as presented in [38]. Consider the English statements:

(1) 'It is obligatory that John does not kill his mother'.
(2) 'If John does kill his mother, it is obligatory that he does kill her gently'.
(3) 'John does kill his mother'.
(4) 'If John does kill his mother gently, he does kill his mother'.

In SDL, (1) to (4) are typically formalized as:[1]

(1') $\mathsf{O}(\neg k)$ (2') $k \supset \mathsf{O}g$ (3') k (4') $g \supset k$

Let $\mathbb{E}$ be the set of plain English sentences (1) to (4), and Φ the set of formulas (1') to (4'); Forrester's Paradox is a mismatch between our intuitive understanding of $\mathbb{E}$, their formalization as Φ, and what follows from what in each case. More precisely, let $\models^i$ be our intuitive understanding of reasoning from $\mathbb{E}$, and $\models^f$ be formal reasoning from Φ. The paradox is a mismatch between $\models^i$ and $\models^f$ (an issue of adequacy of formalization).

Let us elaborate on the mismatch between $\models^i$ and $\models^f$ by revisiting the discussion presented in [38]. Therein, it is proven that: $\Phi \models^f \mathsf{O}(\neg k) \wedge k$, i.e., $\mathsf{O}(\neg k) \wedge k$ is a formal consequence of Φ. Let $\mathsf{O}(\neg k) \wedge k$ be the formalization of (†) 'John is obliged not to kill his mother but he does kill her', it seems reasonable that: $\mathbb{E} \models^i$ (†); i.e., (†) is an intuitive consequence of $\mathbb{E}$; alternatively, $\mathbb{E}$ involve a violation (to the obligation not to kill). In [38] it is also proven that: $\Phi \models^f \mathsf{O}g$. Let $\mathsf{O}g$ be the formalization of (‡) 'John is obliged to kill his mother gently'; it also seems reasonable that: $\mathbb{E} \models^i$ (‡); this is a case of so-called *contary-to-duty* reasoning.

In both cases above $\models^i$ and $\models^f$ coincide. But [38] points out that: $g \supset k \models^f \mathsf{O}(g \supset k)$; from which it can be proven that: $\Phi \models^f \mathsf{O}k$. This is rather untenable. If we take $\mathsf{O}k$ as the formalization of (††) 'John is obliged to kill his mother', it is not at all clear that: $\mathbb{E} \models^i$ (††); in fact, it seems reasonable that: $\mathbb{E} \not\models^i$ (††). Making things worse, it can be proven that: $\Phi \models^f \mathsf{O}(\neg k)$ and $\Phi \models^f \mathsf{O}k$; which results in Φ being formally inconsistent. On the other hand, $\mathbb{E}$ is intuitively consistent. The last two cases reveal that $\models^f$ over-generates w.r.t. $\models^i$. Forrester's Paradox is precisely about this over-generation.

The Global Modality: The previous discussion of Forrester's Paradox is not particular to [38] but standard in the literature on deontic reasoning. To be noted, however, in this discussion $\models^f$ intertwines global and local logical consequence, $\models^f_g$ and $\models^f_l$, without drawing a proper distinction between them. Let us be more precise:

(G) Observe that $g \supset k \models^f_g \mathsf{O}(g \supset k)$, whereas $g \supset k \not\models^f_l \mathsf{O}(g \supset k)$. So when it is said that $g \supset k \models^f \mathsf{O}(g \supset k)$, $\models^f$ refers to $\models^f_g$, and so not to $\models^f_l$. Moreover, if $g \supset k \not\models^f \mathsf{O}(g \supset k)$, then, Φ is a consistent set of sentences, and $\Phi \not\models^f \mathsf{O}k$.

(L) But clearly $\models^f$ is not $\models^f_g$ in all cases. If it were, then $k \models^f \mathsf{O}k$, and this would result in $\{\mathsf{O}(\neg k), k\}$ being immediately inconsistent. This is certainly undesirable; as it undermines formal reasoning about violations in the system, $\mathsf{O}(\neg k) \wedge k$, by reducing it to a triviality. Thus, in this case at least, $\models^f$ refers to $\models^f_l$.

The case analysis in **(G)** and **(L)** shows that there is a need to distinguish between formulas like $g \supset k$, which have a global status, and those like k, which have a local status. The extension from SDL to SDLA serves this purpose. SDLA brings the global modality A into the picture.

[1](1') and (3') are fairly natural; as to whether (2) should be formalized as $k \supset \mathsf{O}g$ or $\mathsf{O}(k \supset g)$ see [38].

This modality is well-known in the literature on Modal Logic, but it is not used explicitly in the literature on Deontic Logic. A enables us to draw a clear distinction between the global and the local. In SDLA 'John does kill his mother' is adequately formalized as k, while 'If John does kill his mother gently, he does kill his mother' is adequately formalized as $\mathsf{A}(g \supset k)$. This captures explicitly, and in the object language, the global status of the meaning of 'killing gently' which is in all cases a form of 'killing'. The inclusion of A in the object language also makes it clear why 'John does kill his mother' should be formalized as k and not as $\mathsf{A}k$; as this is true of a particular world, not in all worlds. A also raises the question of whether obligations should be preceded by it. Though the inclusion of A does not solve the paradoxical status of $\mathbb{E}$, insofar as their formalization is concerned, it does improve the clarity of the formalism. In particular, the inclusion of A reduces $\models^f$ to $\models^f_l$; enabling us to work with global and local formulas in a uniform setting while retaining the capability of drawing the proper distinctions.

Deontic Reasoning with Defaults: The mismatch between $\models^i$ and $\models^f$ in Forrester's Paradox is due to the consideration that $\mathbb{E}$ is intuitively consistent, but its formalization in SDL, and SDLA, "makes it" inconsistent. This understanding of Forrester's Paradox can be seen as a particular instance of *reasoning consistently in the presence of inconsistent information*, in this case contrary-to-duty statements. The extension of SDLA to $\mathscr{D}\mathsf{SDLA}$ accommodates for this logical point of view. In $\mathscr{D}\mathsf{SDLA}$ we formalize prescriptive statements as defaults and descriptive statements as formulas. Intuitively, we can understand defaults as conditional defeasible norms, and formulas as particular situations or scenarios.[2]

To illustrate this point of view, let (1) and (2), and (3) and (4) in Forrrester's Paradox above be formalized as the defaults $(\delta_1)\ \top \rightsquigarrow \mathsf{O}(\neg k)$ and $(\delta_2)\ k \rightsquigarrow \mathsf{O}g$, and as the formulas k and $\mathsf{A}(g \supset k)$, respectively. Define $\mathbf{F} = \langle \{k, \mathsf{A}(g \supset k)\}, \{\delta_1, \delta_2\}\rangle$; it follows that:

(i) $\mathbf{F} \mathrel{|\!\!\approx^c} \mathsf{O}(\neg k) \wedge k$ (ii) $\mathbf{F} \mathrel{|\!\!\approx^c} \mathsf{O}g$ (iii) $\mathbf{F} \mathrel{|\!\!\not\approx^c} \bot$

Cases (i) to (iii) offer some insight into Forrester's Paradox: formal and intuitive reasoning coincide. In this sense, $\mathrel{|\!\!\approx^c}$ brings formal reasoning closer to $\models^i$. Yet $\mathrel{|\!\!\approx^c}$ does not solve Forrester's Paradox: $\mathbf{F} \mathrel{|\!\!\approx^c} \mathsf{O}(g \supset k)$ and $\mathbf{F} \mathrel{|\!\!\approx^c} \mathsf{O}(k)$. So there is still a mismatch w.r.t. $\models^i$. Incidentally, this mismatch does not originate from normative reasoning. Instead, it stems from the interaction between A and the K formula $\mathsf{O}(g \supset k) \supset (\mathsf{O}g \supset \mathsf{O}k)$ which is valid in SDL. From $\mathsf{A}(g \supset k)$ we can conclude $\mathsf{O}(g \supset k)$, and from this fact and K we can conclude $\mathsf{O}g \supset \mathsf{O}k$. We obtain $\mathsf{O}k$ as soon as we detach $k \rightsquigarrow \mathsf{O}g$. There is an obvious technical solution for this problem in $\mathscr{D}\mathsf{SDLA}$, replace the sentence $\mathsf{A}(g \supset k)$ by the default $(\delta_3)\top \overset{\neg\mathsf{O}g}{\rightsquigarrow} \mathsf{A}(g \supset k)$ and add extra constraints to block the detachment of this new default if (δ_2) has been detached, e.g., $\delta_2 \prec \delta_3$. In this scenario we cannot credulously conclude $\mathsf{O}k$. This carries with it the benefit of matching our intuition, $\mathrel{|\!\!\approx^c}$ and $\models^i$ coincide, but also the added cost of losing the naturalness of the first formalization. Whether or not this is a cost that we are willing to assume is certainly a matter of debate.

Mine Pump Example: Beyond philosophical considerations, deontic reasoning has interesting applications in Computer Science and Software Engineering [47]. Such is the case in Goal-Oriented Requirements Engineering (GORE) [34]. GORE aims at formally specifying the informal requirements of a system via a set of *goals* applicable to particular *domains.* Goals are prescriptive statements specifying what must be achieved by the system. Domains are declarative statements about the environment in which the system operates. Formalization of informal

[2]In this sense, unlike conditional deontic formulas, defaults are neither true nor false, they are rules of procedure which can be acted in accordance with, voided, or contravened, when applied to particular situations.

requirements brings with it: removal of the ambiguities of informal languages, tool-support, and the possibility of automatic analyses. This is particularly interesting for the analysis of *conflicts*; situations in which the goals in a requirements specification diverge (they are not satisfiable relative to some domain). Conflicts are not obvious, and it is extremely useful to detect them, specially in early stages of analysis. Let us note that conflicts do not necessarily render useless the entire set of goals. In some cases, consistency is restored in some maximal way (e.g., by dropping a minimal set of goals giving rise to the conflict). When uniqueness of the solution is not guaranteed, we may wish to properly inspect all alternatives. Deontic Logic adds to this the possibility of reasoning about "normal" or "ideal" situations, and to distinguish them from those that exhibit "faults," modeled as violations (see [14, 33, 47] for an in-depth introduction to the use of Deontic Logic in fault-tolerance). But Deontic Logic alone does not allow us to adequately deal with the conflicting nature of requirements. It is precisely at this point where Default Logics prove their worth [5]. Let us illustrate what the combination of Deontic Logic and Default Logic can do by means of an example, adapted from [18].

Deontic Reasoning with Defaults (cont.): Suppose that in a gold mine there is a controller which has two sensors indicating the level of water and of methane in the mine. From the information provided by the sensors, the controller turns on/off a pump that regulates the level of water in the mine. Consider the following scenario:

(1) 'If the water level is high, the water must be drained'.
(2) 'If the methane level is high, the pump must be off'.
(3) 'Whenever the pump is off, the water is not drained'.
(4) 'The pump is on'.
(5) 'The level of water is high'.
(6) 'The level of methane is high'.

In this scenario: (1) and (2) are goals, and (3) to (6) are domain properties. Following [5], we formalize goals as defaults, and domain properties as formulas. A possible formalization is:

(1') $w \rightsquigarrow \mathsf{O}d$ (2') $m \rightsquigarrow \mathsf{O}(\neg p)$ (3') $\mathsf{A}(\neg p \supset \neg d)$ (4') p (5') w (6') m

If $\Phi = \{\mathsf{A}(\neg p \supset \neg d)\}$, $\Delta = \{(1'), (2')\}$, $\mathbf{M}_1 = \langle \Phi \cup \{w, m\}, \Delta\rangle$, and $\mathbf{M}_2 = \langle \Phi \cup \{\neg p, w\}, \Delta\rangle$,

(i) $\mathbf{M}_1 \mathrel{|\!\!\approx^s} \mathsf{O}d \vee \mathsf{O}(\neg p)$ (ii) $\mathbf{M}_1 \mathrel{|\!\!\not\approx^s} \mathsf{O}d$ (iii) $\mathbf{M}_1 \mathrel{|\!\!\not\approx^s} \mathsf{O}(\neg p)$ (iv) $\mathbf{M}_2 \mathrel{|\!\!\approx^s} \mathsf{O}d \wedge \neg d$

Intuitively, (i) to (iii) tell us that the goals can be reached individually, but not jointly. This indicates a conflict w.r.t. the domain properties under consideration. A finer analysis reveals that $\mathbf{M}_1$ has two extensions: $\Phi \cup \{p, w, m, \mathsf{O}d\}$ and $\Phi \cup \{p, w, m, \mathsf{O}(\neg p)\}$. The sets of defaults generating these extensions are maximally satisfiable sets of goals. In other words extensions tell us which goals are jointly satisfiable, and which ones are in conflict and need to be revised to restore consistency. (iv) tells us that the domain properties in $\mathbf{M}_2$ result in a violation of (1'). This use of deontic concepts in the analysis of goals enables us to distinguish between normal and faulty behavior in goal specifications.

Prioritization of Defaults: In some application areas we may wish to model a predefined ordering on defaults; or state explicit conditions under which a default must be overridden. For example, suppose that mining safety regulations deem high levels of methane more dangerous than high levels of water (in addition to the risk of explosion, methane gas intoxication may result in loss of consciousness or asphyxia). Under such safety regulations, we may wish to indicate that (2) takes precedence over (1), formally (2') $\prec$ (1'); intuitively, (1') is overridden by (2'). Alternatively, we may wish not to impose any additional structure on defaults, but to formalize (1) as (1") $w \overset{\neg m}{\rightsquigarrow} \mathsf{O}d$. Intuitively, this tells us that (1") is voided if the level of methane is too high. Formally, if $\mathbf{M}'_1 = \langle \Phi \cup \{w, m\}, \Delta, \prec\rangle$, and $\mathbf{M}_3 = \langle \Phi \cup \{w, m\}, \{(1"), (2')\}\rangle$,

(v) $\mathbf{M}'_1 \approx^s \mathsf{O}(\neg p)$ (vi) $\mathbf{M}'_1 \not\approx^s \mathsf{O}d$ (vii) $\mathbf{M}_3 \approx^s \mathsf{O}(\neg p)$ (viii) $\mathbf{M}_3 \not\approx^s \mathsf{O}d$

The multiple extensions problem of the scenario without safety regulations, resulting from a conflict, is resolved in the scenario including safety regulations via prioritization, or by listing the explicit conditions under which a goal is voided. In this particular scenario we obtain the same set of default consequences, albeit for different reasons.

Final Remarks: The discussion of Forrester's Paradox in terms of credulous default consequence in $\mathscr{D}$SDLA yields some interesting observations; in particular w.r.t. the inclusion of the global modality A in deontic reasoning, and the use of defaults in the formalization of conditional norms. It is worth mentioning that credulous default consequence in $\mathscr{D}$SDLA enables us to reason about the violation of an obligation while keeping the violated obligation in force (as opposed to cancelling it). This is a point of departure between $\mathscr{D}$SDLA and some standard works on non-monotonic deontic reasoning (where in cases of violations the violated obligation is cancelled). Evidently, Forrester's Paradox is not solved in $\mathscr{D}$SDLA. We do not claim that this is the case either. Instead, we employ this classical and well-known scenario as a means to illustrate some of the features of our formalism. $\mathscr{D}$SDLA, in line with many other works on nonmonotonic deontic reasoning, sometimes satisfies our intuition, sometimes fails it; in any case, the benefit is to have at hand a set of logical tools for rigorous discussion.

We have also shown a novel use of sceptical default consequence in $\mathscr{D}$SDLA in the context of formal goal oriented requirements. This is an area of research not as well explored from the perspective on (defeasible) deontic reasoning. The formalization of the Mine Pump scenario is similar to Forrester's Paradox. However, in the analysis of goal specifications, practical concerns set aside philosophical considerations. Correct goal specifications are essential for the production of dependable and reliable software based systems. In this respect, methods accommodating different kinds of analyses are highly desirable; even more if they come equipped with tool-support. The presented formalism is a combination of two orthogonal logical frameworks, Default Logic on the one side, and Deontic Logic on the other. An important characteristic of this framework is that it is amenable to tableaux reasoning, as it is shown in the next section.

5 Tableaux-based Proof Calculus

The benefits of equipping a logical system with a proof calculus are well-known. We develop a non-monotonic proof calculus for $\mathscr{D}$SDLA based on tableaux for SDLA. The proof calculus is sound, complete, and terminating (with loop checks). We comment on the technical challenges of the proof system as well as some of its gains.

Basic Tableau Rules: We begin by recalling the basic definitions of a tableau system for SDLA [10, 11, 36, 42]. These will be used in the formulation of the tableau system for $\mathscr{D}$SDLA.

A *tableau* is a tree whose nodes are either a pair of a formula φ and a natural number i, notation $@_i\varphi$ (borrowed from Hybrid Logic [8]), or a pair of natural numbers i and j, notation (i, j). Intuitively, $@_i\varphi$ means that φ is true at world i; whereas (i, j) means that j is accessible from i. By applying rules a tableau is expanded with the addition of new nodes at the level of leaves. A tableau for φ is a tableau having $@_0\varphi$ as its root. A tableau for φ is *well-formed* if is constructed according to the rules in Fig. 1. In this figure, $(\neg\neg)$, $(\wedge)$, and $(\neg\wedge)$, are rules for the boolean logical connectives. (O) and $(\neg\mathsf{O})$ are rules for deontic obligation. (A) and $(\neg\mathsf{A})$ are rules for the global modality. (D) is the rule for seriality. The rules for all other connectives in the language are derived from the rules just introduced. Rules are applied as usual: premisses must belong to the branch; side conditions must be met (if any); the branch is extended according to the consequents.

Boolean Rules:

$$\frac{@_i\neg\neg\varphi}{@_i\varphi}\ (\neg\neg) \qquad \frac{@_i(\varphi\wedge\psi)}{\begin{array}{c}@_i\psi\\ @_i\varphi\end{array}}\ (\wedge) \qquad \frac{@_i\neg(\varphi\wedge\psi)}{@_i\neg\varphi \quad @_i\neg\psi}\ (\neg\wedge)$$

Deontic Rules:

$$\frac{\begin{array}{c}@_i\mathsf{O}\varphi\\ (i,j)\end{array}}{@_j\varphi}\ (\mathsf{O}) \qquad \frac{@_i\neg\mathsf{O}\varphi}{\begin{array}{c}(i,j)\\ @_j\neg\varphi\end{array}}\ (\neg\mathsf{O})^\dagger \qquad \frac{}{(i,j)}\ (\mathsf{D})^*$$

Global Rules:

$$\frac{@_i\mathsf{A}\varphi}{@_j\varphi}\ (\mathsf{A})^\ddagger \qquad \frac{@_i\neg\mathsf{A}\varphi}{@_j\neg\varphi}\ (\neg\mathsf{A})^\dagger$$

† for j new (i.e., not used before in the branch).
‡ for j in the branch.
* for i in the branch, j new, and provided there is no other (i,k) already in the branch.

Figure 1: Tableau Rules for SDLA

The discussion that follows is standard but useful to establish some basic terminology. Tableaux formulate a *proof calculus* for *provability*, i.e., proofs without assumptions. In particular, any tableau for $\neg\varphi$ is an *attempt at proving* φ; and any closed tableau (Def. 5.1) for $\neg\varphi$ is a *successful proof attempt*, or a *proof of* φ, notation $\vdash\varphi$. This proof calculus is *sound* and *complete*, i.e., $\vdash\varphi$ iff $\models\varphi$ [42]. Though in its present form the calculus is non-terminating, termination can be achieved with the use of loop checks [11].

Definition 5.1. A branch is *closed*, noted $(\blacktriangle)$, if $@_i\varphi$ and $@_i(\neg\varphi)$ occur in the branch; otherwise it is *open*, noted $(\blacktriangledown)$. A tableau is *closed* if all of its branches are closed; otherwise it is *open*.

We extend the proof calculus for provability to one for *deducibility* (proofs from a set Φ of assumptions) with the addition of the rule $(\mathbf{A})$ in Fig. 2.

Assumption Rules: $$\frac{}{@_0\varphi}\ (\mathbf{A}) \text{ for } \varphi\in\Phi$$

Figure 2: Tableau Rules for SDLA (cont.)

Intuitively, $(\mathbf{A})$ can be understood as stating that assumptions are always true in the "current" world. $(\mathbf{A})$ is not strictly necessary: $\Phi\models\varphi$ iff $\models\bigwedge\Phi\supset\varphi$ (for Φ finite). Nonetheless, dealing with assumptions in this way simplifies the definitions and understanding of the tableau-based proof calculus for $\mathscr{D}\mathsf{SDLA}$. When $(\mathbf{A})$ is involved, we talk about a tableau for φ from Φ. Such a tableau is *well-formed* if: its root is $@_0\varphi$; the rules in Fig. 1 are applied as usual; and $(\mathbf{A})$ is applied to the formulas in Φ. The proof calculus for deducibility is obtained by defining any well-formed tableau for $@_0(\neg\varphi)$ from Φ as an *attempt at proving* that φ is a syntactical consequence of Φ; and any closed tableau as a *successful proof attempt*, i.e., a *proof* of φ from

Φ, notation $\Phi \vdash \varphi$. This proof calculus for deducibility is also *sound* and *complete*: $\Phi \vdash \varphi$ iff $\Phi \models \varphi$ [36]. Again, termination can be ensured with the use of loop-checks.

Tableaux enjoy an interesting property: if a tableau for $@_0(\neg\varphi)$ from Φ has a branch that is open and saturated (Def. 5.2), there is no proof of φ from Φ, notation $\Phi \nvdash \varphi$ [36]. This result enables us to use tableaux also as a system to establish the *non-existence* of proofs.

Definition 5.2. A branch of a tableau is *saturated*, notation ($\blacklozenge$), if the application of any rule results in the addition of a new node that is already present in the branch, or the application of the rule is blocked by loop checking.

Tableau Rules for Defaults: Let $\mathbf{T}$ be the tableaux system for SDLA just described, we could formulate a tableau system $\mathbf{T}_{\mathscr{D}}$ for $\mathscr{D}\mathsf{SDLA}$ by constructing suitable tableaux in $\mathbf{T}$ for $E \supset \varphi$, for E an extension of $\langle \Phi, \Delta, \prec \rangle$. It is also not difficult to see how we could use $\mathbf{T}$ for generating an arbitrary extension E by traversing $\prec$ in a breadth-first manner (see, e.g., [2, 44]). Generating all extensions requires some ingenuity, e.g., we must keep track of already explored combinations of default rules, but it is possible. This naive formulation of $\mathbf{T}_{\mathscr{D}}$, however, suffers from some drawbacks. First, the exploration of extensions would be defined outside of the framework of $\mathbf{T}$. Second, it necessarily requires the complete construction of extensions; something that we wish to avoid whenever possible given its computational cost [28]. Third, it offers no insight into what it means to reason with defaults; a clear departure from the traditional understanding of a proof calculus. The technical challenge in formulating a reasonable notion of tableaux for $\mathscr{D}\mathsf{SDLA}$ is *to internalize reasoning with defaults in the framework of tableaux allowing for partial use of extensions.* We show that by treating defaults similarly to how we treat formulas in the rule (**A**) in Fig. 2 we can achieve this *desideratum.* The resulting tableau system formulates a proof calculus for $\mathscr{D}\mathsf{SDLA}$ which retains some the desirables properties of this kind of calculi.

A *default tableau* is a tree whose nodes are either a pair of a formula φ and a natural number i, notation $@_i\varphi$; a pair of natural numbers i and j, notation (i, j); or a default $\pi \overset{\rho}{\rightsquigarrow} \chi$. We talk about a *credulous* or *sceptical* default tableau for φ from $\langle \Phi, \Delta, \prec \rangle$. A credulous default tableau is *well-formed* if its root is $@_0\varphi$; the rules in Fig. 1 are applied as usual; the rule (**A**) in Fig. 2 is applied to the formulas in Φ; and the rule (**C**) in Fig. 3 is applied to the defaults in Δ. A sceptical default tableau is *well-formed* if its root is $@_0\varphi$; the rules in Fig. 1 are applied as usual; the rule (**A**) in Fig. 2 is applied to the formulas in Φ; and the rule (**S**) in Fig. 3 is applied to the defaults in Δ. Observe that the rules (**C**) and (**S**) in Fig. 3 are applicable only if their side conditions are met.[3] These side conditions need (auxiliary) tableaux in SDLA but are decidable. Intuitively, (**C**) and (**S**) correspond to the use of defaults as *defeasible conditional assumptions.* Their side conditions guarantee the correctness of reasoning with defaults in *some* (*credulous*) or *all* (*sceptical*) extensions, respectively.

Definition 5.3. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$; then $\pi \overset{\rho}{\rightsquigarrow} \chi$ is $\vdash$-triggered by Δ' in $\mathbf{A}$ if there is a closed tableau for $\neg\pi$ from $\Phi \cup X_{\Delta'}$, i.e., $\Phi \cup X_{\Delta'} \vdash \pi$. It is $\vdash$-blocked by Δ' in $\mathbf{A}$ if for some $\varrho \in (P_{\Delta'} \cup \rho)$, there is a closed tableau for ϱ from $\Phi \cup X_{\Delta'}$, i.e., $\Phi \cup X_{\Delta'} \cup \chi \vdash \neg\varrho$. It is $\vdash$-detached Δ' in $\mathbf{A}$ if it is $\vdash$-triggered and not $\vdash$-blocked by Δ' in $\mathbf{A}$.

Definition 5.4. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, and $\Delta' \subseteq \Delta$; then $\delta \in (\Delta - \Delta')$ is $\vdash$-enabled in $\mathbf{A}$ w.r.t. Δ' if δ is $\vdash$-detached by Δ' in $\mathbf{A}$, and no other $\delta'' \in (\Delta - \Delta')$ $\vdash$-detached by Δ' in $\mathbf{A}$ is s.t. $\delta'' \prec \delta$.

A proof calculus for *credulous (sceptical) default deducibility* is obtained by defining any well-formed credulous (sceptical) default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$ as an *attempt at proving*

[3]Compare Defs. 3.1 and 3.2 with Defs. 5.3 and 5.4.

Default Rules:

$$\frac{}{\begin{array}{c}\pi \overset{\rho}{\rightsquigarrow} \chi\\ @_0\chi\end{array}}\ (\mathbf{C})^\dagger$$

for $\pi \overset{\rho}{\rightsquigarrow} \chi$ a $\vdash$-enabled (Def. 5.4) default in $\langle \Phi, \Delta, \prec \rangle$ w.r.t. the defaults in the branch.

$$\frac{}{\begin{array}{c}\pi_1 \overset{\rho_1}{\rightsquigarrow} \chi_1\\ @_0\chi_1\end{array} \quad \dots \quad \begin{array}{c}\pi_n \overset{\rho_n}{\rightsquigarrow} \chi_n\\ @_0\chi_n\end{array}}\ (\mathbf{S})$$

for $\{\pi_i \overset{\rho_i}{\rightsquigarrow} \chi_i\}_{i \in [1,n]}$ all $\vdash$-enabled defaults in $\langle \Phi, \Delta, \prec \rangle$ w.r.t. the defaults in the branch.

† $(\mathbf{C})$ extends all branches of the default tableaux.

Figure 3: Tableau Rules for Defaults

$$\begin{tikzcd}
 & \text{(a) } @_0(\neg(\mathsf{O}(\neg k) \wedge k)) \arrow[dl] \arrow[dr] & \\
\begin{array}{l}\text{(b) } @_0(\neg\mathsf{O}(\neg k))\\ \text{(g) } \top \rightsquigarrow \mathsf{O}(\neg k)\\ \text{(h) } @_0\mathsf{O}(\neg k)\\ (\blacktriangle)\text{ b, h}\end{array} & & \begin{array}{l}\text{(c) } @_0(\neg k)\\ \text{(d) } @_0 k\\ \text{(e) } \top \rightsquigarrow \mathsf{O}(\neg k)\\ \text{(f) } @_0\mathsf{O}(\neg k)\\ (\blacktriangle)\text{ c, d}\end{array}
\end{tikzcd}$$

Figure 4

$$\begin{array}{l}\text{(a) } @_0(\neg\mathsf{O}g)\\ \text{(b) } k \rightsquigarrow \mathsf{O}g\\ \text{(c) } @_0\mathsf{O}g\\ (\blacktriangle)\text{ b, c}\end{array}$$

Figure 5

$$\begin{array}{l}\text{(a) } @_0(\neg\mathsf{O}(g \supset k))\\ \text{(b) } (0,1)\\ \text{(c) } @_1(\neg(g \supset k))\\ \text{(d) } @_0\mathsf{A}(g \supset k)\\ \text{(e) } @_1(g \supset k)\\ (\blacktriangle)\text{ c, e}\end{array}$$

Figure 6

$$\begin{tikzcd}
 & \begin{array}{l}\text{(a) } @_0(\neg\mathsf{O}k)\\ \text{(b) } (0,1)\\ \text{(c) } @_1(\neg k)\\ \text{(d) } @_0\mathsf{A}(g \supset k)\\ \text{(e) } @_1(g \supset k)\end{array} \arrow[dl] \arrow[dr] & \\
\begin{array}{l}\text{(f) } @_1(\neg g)\\ \text{(g) } k \rightsquigarrow \mathsf{O}g\\ \text{(h) } @_0\mathsf{O}g\\ \text{(i) } @_1 g\\ (\blacktriangle)\text{ f, i}\end{array} & & \begin{array}{l}\text{(j) } @_1 k\\ \text{(k) } k \rightsquigarrow \mathsf{O}g\\ \text{(l) } @_0\mathsf{O}g\\ (\blacktriangle)\text{ c, j}\end{array}
\end{tikzcd}$$

Figure 7

that φ is a syntactical credulous (sceptical) default consequence of $\langle \Phi, \Delta, \prec \rangle$; and any closed (Def. 5.1) credulous (sceptical) default tableau as a *successful credulous (sceptical) default proof attempt*, i.e., a *credulous (sceptical) default proof* of φ from $\langle \Phi, \Delta, \prec \rangle$, notation $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\sim^c} \varphi$ ($\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\sim^s} \varphi$). The proof calculus for credulous (sceptical) default deducibility is *sound* and *complete* (w.r.t. *extensions*). Termination is guaranteed by the termination of tableaux for SDLA. This claim is made precise in Thm. 5.1; the proof of which can be found in Appendix A.

Theorem 5.1 (Soundness, Completeness, and Termination)**.** $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\sim^c} \varphi$ iff $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\approx^c} \varphi$. $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\sim^s} \varphi$ iff $\langle \Phi, \Delta, \prec \rangle \mathrel{|\!\approx^s} \varphi$. If $\vdash$ terminates, then, both $\mathrel{|\!\sim^c}$ and $\mathrel{|\!\sim^s}$ terminate.

Example 4. Let $\mathbf{F} = \langle \{k, \mathsf{A}(g \supset k)\}, \{(\top \rightsquigarrow \mathsf{O}(\neg k)), (k \rightsquigarrow \mathsf{O}g)\}\rangle$; Figs. 4 to 7 are credulous default proofs of $\mathsf{O}(\neg k) \wedge k$, $\mathsf{O}g$, $\mathsf{O}(g \supset k)$, and $\mathsf{O}k$, from $\mathbf{F}$, respectively (cf. Forrester's Paradox). Let $\Delta = \{(w \rightsquigarrow \mathsf{O}d), (m \rightsquigarrow \mathsf{O}(\neg p))\}$, and $\mathbf{M}_1 = \langle \{\mathsf{A}(\neg p \supset \neg d), w, m\}, \Delta\rangle$; Fig. 8 is a sceptical default proof of $\mathsf{O}d \vee \mathsf{O}(\neg p)$ from $\mathbf{M}_1$. Fig. 9 shows an open sceptical default for $\neg\mathsf{O}(\neg p)$ from $\mathbf{M}_1$. Let $\mathbf{M}_2 = \langle \{\mathsf{A}(\neg p \supset \neg d), \neg p, w\}, \Delta\rangle$; Fig. 10 is a sceptical default proof of $\mathsf{O}d \wedge \neg d$ from $\mathbf{M}_2$. Let $(w \rightsquigarrow \mathsf{O}d) \prec (m \rightsquigarrow \mathsf{O}(\neg p))$, and $\mathbf{M}_1' = \langle \{\mathsf{A}(\neg p \supset \neg d), w, m\}, \Delta, \prec\rangle$; pruning the right sub-tree from the node (a) in Fig. 9 results in a sceptical default proof of $\mathsf{O}(\neg p)$ from $\mathbf{M}_1$.

Final Remarks Our tableaux based proof calculi for credulous and sceptical default consequence internalize reasoning with defaults: tableau rules for defaults are rules of the calculi.

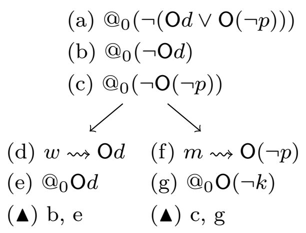
(a) $@_0(\neg(\mathsf{O}d \vee \mathsf{O}(\neg p)))$
(b) $@_0(\neg \mathsf{O}d)$
(c) $@_0(\neg \mathsf{O}(\neg p))$
(d) $w \rightsquigarrow \mathsf{O}d$
(e) $@_0\mathsf{O}d$
(▲) b, e
(f) $m \rightsquigarrow \mathsf{O}(\neg p)$
(g) $@_0\mathsf{O}(\neg k)$
(▲) c, g

Figure 8

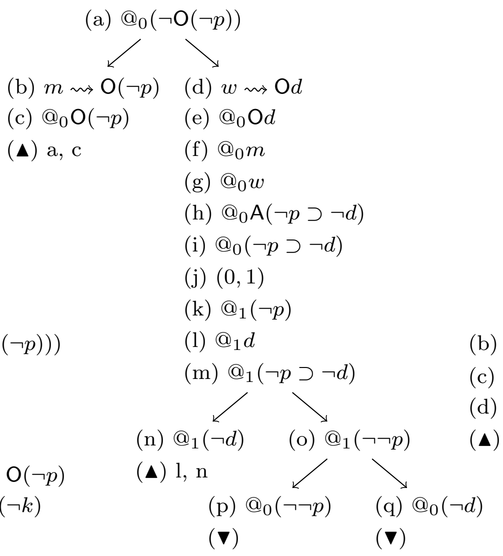
(a) $@_0(\neg \mathsf{O}(\neg p))$
(b) $m \rightsquigarrow \mathsf{O}(\neg p)$
(c) $@_0\mathsf{O}(\neg p)$
(▲) a, c
(d) $w \rightsquigarrow \mathsf{O}d$
(e) $@_0\mathsf{O}d$
(f) $@_0 m$
(g) $@_0 w$
(h) $@_0\mathsf{A}(\neg p \supset \neg d)$
(i) $@_0(\neg p \supset \neg d)$
(j) $(0, 1)$
(k) $@_1(\neg p)$
(l) $@_1 d$
(m) $@_1(\neg p \supset \neg d)$
(n) $@_1(\neg d)$
(▲) l, n
(o) $@_1(\neg\neg p)$
(p) $@_0(\neg\neg p)$
(▼)
(q) $@_0(\neg d)$
(▼)

Figure 9

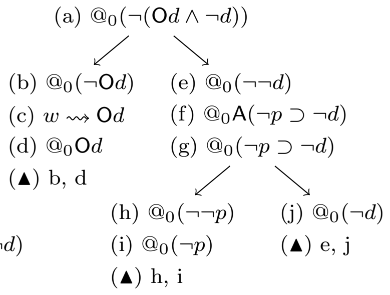
(a) $@_0(\neg(\mathsf{O}d \wedge \neg d))$
(b) $@_0(\neg \mathsf{O}d)$
(c) $w \rightsquigarrow \mathsf{O}d$
(d) $@_0\mathsf{O}d$
(▲) b, d
(e) $@_0(\neg\neg d)$
(f) $@_0\mathsf{A}(\neg p \supset \neg d)$
(g) $@_0(\neg p \supset \neg d)$
(h) $@_0(\neg\neg p)$
(i) $@_0(\neg p)$
(▲) h, i
(j) $@_0(\neg d)$
(▲) e, j

Figure 10

In this way, they avoid meta-logical arguments involving extensions, bridging the gap between Default Logic and Theorem Proving, and between Non-monotonic and Classical reasoning.

Our proof calculi are also remarkably simple. They are so in the following sense. In Fitting's terms [22], a tableau system is a formal proof procedure, existing in many varieties and for several logics, but always having certain characteristics. It is a refutation procedure: to prove a formula, the initial step is to begin with a syntactical expression asserting the contrary. Successive steps syntactically break down this assertion into cases. Finally, there are impossibility conditions for closing cases. The sought after proof is obtained if all cases are closed. Credulous and sceptical default tableaux operate in the way just described. To prove that φ is a default consequence of $\langle \Phi, \Delta, \prec \rangle$, our proof attempt begins with $@_0(\neg\varphi)$. We continue by syntactically breaking down formulas into their components. We are interested in deducibility from default assumption sets, formulas and prioritized defaults. The rule (**A**) in Fig. 2 allow us to incorporate formulas in a proof attempt; the rules (**C**) and (**S**) in Fig. 3 allow us to incorporate prioritized defaults into a proof attempt, depending on whether we are proving credulously or sceptically. Finally, closing cases indicate the impossibility conditions that are needed to make a proof attempt an actual proof. The key observation is that simplicity is obtained by how we treat defaults in a proof attempt.

Default tableaux offer some insight into proving with defaults. Moreover, they enjoy some desirable properties. In particular, they do not require the complete construction of extensions for proofs. This can be immediately seen from the fact that a default tableau might close from a subset of the defaults being considered. In some cases, this becomes a practical advantage.

Sceptical default tableaux can be used for exploring and constructing counterexamples (Fig. 9). An analogy with tableaux for SDLA becomes useful to explain why this is the case. Every saturated and open branch of a tableau for $\neg\varphi$ from Φ is a syntactical description of a model of Φ that is also a model of $\neg\varphi$. If there is one such model, then, φ is not a consequence of Φ. In sceptical reasoning extensions play the role of models (see Section 3). In this respect, every saturated and open branch of a sceptical default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$ is a description of an extension $E \in \mathscr{E}(\Phi, \Delta, \prec)$ s.t. $E \not\models \varphi$. This extension is a counterexample for the claim that φ is a sceptical default consequence of $\langle\Phi,\Delta,\prec\rangle$. This is made precise in Prop. 5.1

Proposition 5.1. If a sceptical default tableau for $\neg\varphi$ from $\langle\Phi,\Delta,\prec\rangle$ has an open and saturated branch, then, there is an extension $E \in \mathscr{E}(\Phi,\Delta,\prec)$ s.t. $E \not\models \varphi$, i.e., $\langle\Phi,\Delta,\prec\rangle \not\approx^{s} \varphi$.

Building a suitable counterexample to the claim that φ is a credulous default consequence of $\langle\Phi,\Delta,\prec\rangle$ is not as direct. Observe that a closed credulous default tableau for $\neg\varphi$ from $\langle\Phi,\Delta,\prec\rangle$ indicates that there is an extension $E \in \mathscr{E}(\Phi,\Delta,\prec)$ s.t. $E \models \varphi$, i.e., $\langle\Phi,\Delta,\prec\rangle \approx^{c} \varphi$. However, a saturated and open branch of a credulous default tableau for $\neg\varphi$ from $\langle\Phi,\Delta,\prec\rangle$ is *not* an indication that $\langle\Phi,\Delta,\prec\rangle \not\approx^{c} \varphi$. Instead, it is an indication that the particular extension $E \in \mathscr{E}(\Phi,\Delta,\prec)$ considered in the default tableau is s.t. $E \not\models \varphi$. This extension is not a suitable counterexample since there might be a different extension $E' \in \mathscr{E}(\Phi,\Delta,\prec)$, not considered in the default tableaux s.t. $E' \models \varphi$, resulting in $\langle\Phi,\Delta,\prec\rangle \approx^{c} \varphi$. The issue with credulous default tableaux is that they are not *confluent*: it is not possible, in general, to extend a partially constructed credulous default tableau for $\neg\varphi$ from $\langle\Phi,\Delta,\prec\rangle$ to a closed one, even if one such closed credulous default tableau is guaranteed to exist. This is an intrinsic characteristic of credulous reasoning, as credulous default consequences depend on a particular selection of defaults, not of arbitrary ones. Lack of confluence for credulous default tableaux can be successfully dealt with by applying enabled defaults in a breadth-first manner; exploring the search space of credulous default tableaux until a closed one is found, or no closed one exists. Nonetheless, the problem of producing a suitable counterexample remains: suppose that we exhaust the search space and we have found that there are no closed credulous default tableaux, what do we exhibit as a suitable counterexample? The answer to this question requires further exploration.

6 Conclusions and Final Remarks

We developed a Prioritized Default Logic. This logic, which we call $\mathscr{D}$SDLA, is built modularly from SDLA via the addition of defaults and priorities among them. We have placed special attention to the definition of default consequence bringing to the fore credulous and sceptical default consequence. We illustrated some of the features of $\mathscr{D}$SDLA with the use of examples. In particular, we have discussed credulous default consequence in $\mathscr{D}$SDLA resorting to Forrester's Paradox. Departing from some standard works on nonmonotonic deontic reasoning, credulous default consequence in $\mathscr{D}$SDLA enables us to reason about the violation of obligations without cancelling the violated obligation. We have also shown a novel application of sceptical default consequence in $\mathscr{D}$SDLA in the context of Goal Oriented Requirements Engineering. We believe that this lays open an interesting avenue for application of our logic.

In addition, we developed tableau-based proof calculi for credulous and sceptical default consequence in $\mathscr{D}$SDLA. Our proof calculi are sound, complete and terminating using loop-checks. We have shown that they also have some desirable properties. In particular, they internalize reasoning with defaults, and do not require the complete construction of extensions to prove whether a formula is a default consequence. Sceptical default tableaux retain the capability of producing counter-examples (as extensions). We discussed some reasons why the production of counter-examples is not trivial for credulous default tableaux; but this issue requires further exploration.

As some of the further work that we plan to undertake we list: efficient implementations of our default proof calcului in a tableaux system, e.g., in [31], proof strategies for building shorter proofs, similar to those listed in [16], and tight complexity bounds on the calculi, building on results reported in [10, 21, 28].

References

[1] C. Alchourrón. Detachment and defeasibility in deontic logic. *Studia Logica*, 57(1):5–18, 1996.

[2] G. Amati, L. Aiello, D. Gabbay, and F. Pirri. A proof theoretical approach to default reasoning I: Tableaux for default logic. *Journal of Logic and Computation*, 6(2):205–231, 1996.

[3] G. Antoniou. A comparative survey of default logic variants. In *International Conference on Formal and Applied Practical Reasoning (FAPR'96)*, volume 1085 of *LNCS*, pages 15–28. Springer, 1996.

[4] G. Antoniou. *Nonmonotonic Reasoning*. The MIT Press, Cambridge, 1997.

[5] G. Antoniou. The role of nonmonotonic representations in requirements engineering. *International Journal of Software Engineering and Knowledge Engineering*, 8(3):385–399, 1998.

[6] G. Antoniou and K. Wang. Default logic. In D. Gabbay and J. Woods, editors, *The Many Valued and Nonmonotonic Turn in Logic*, volume 8 of *Handbook of the History of Logic*, pages 517–555. North-Holland, 2007.

[7] L. Åqvist. Deontic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 8, pages 147–264. Springer Netherlands, Dordrecht, 2007.

[8] C. Areces and B. ten Cate. Hybrid logics. In Blackburn et al. [10].

[9] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge U Press, 2001.

[10] P. Blackburn, J. van Benthem, and F. Wolter, editors. *Handbook of Modal Logic*. Elsevier, 2007.

[11] T. Bolander and P. Blackburn. Termination for hybrid tableaus. *Journal of Logic and Computation*, 17(3):517–554, 2007.

[12] G. Brewka and T. Eiter. Prioritizing default logic. In S. Hölldobler, editor, *Intellectics and Computational Logic*, volume 19 of *Applied Logic Series*, pages 27–45. Kluwer, 2000.

[13] V. Cassano, C. López Pombo, and T. Maibaum. A propositional tableaux based proof calculus for reasoning with default rules. In *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2015)*, volume 9323 of *LNAI*, pages 6–21. Springer, 2015.

[14] P. Castro. *Deontic Action Logics for Specification and Analysis of Fault-Tolerance*. PhD thesis, Department of Computer and Software, McMaster University, 2009.

[15] B. Chellas. *Modal Logic: An Introduction*. Cambridge U Press, 1980.

[16] M. D'Agostino. Tableau methods for classical propositional logic. In D'Agostino et al. [17], pages 45–123.

[17] M. D'Agostino, D. M. Gabbay, R. Hahnle, and J. Posegga, editors. *Handbook of Tableau Methods*. Springer, 1999.

[18] R. Degiovanni, N. Ricci, D. Alrajeh, P. Castro, and N. Aguirre. Goal-conflict detection based on temporal satisfiability checking. In *31st IEEE/ACM International Conference on Automated Software Engineering (ASE 2016)*, pages 507–518, 2016.

[19] J. Delgrande and T. Schaub. Expressing preferences in default logic. *Artificial Intelligence*, 123(1-2):41–87, 2000.

[20] J. Delgrande, T. Schaub, H. Tompits, and K. Wang. A classification and survey of preference handling approaches in nonmonotonic reasoning. *Computational Intelligence*, 20(2):308–334, 2004.

[21] U. Egly and H. Tompits. Proof-complexity results for nonmonotonic reasoning. *ACM Transactions on Computational Logic*, 2(3):340–387, 2001.

[22] M. Fitting. Introduction. In D'Agostino et al. [17], pages 1–43.

[23] J. Forrester. Gentle murder, or the adverbial samaritan. *The Journal of Philosophy*, 81:193–197, 1984.

[24] D. Gabbay, J. Horty, X. Parent, R. van der Meyden, and L. van der Torre, editors. *Handbook of Deontic Logic and Normative Systems*, volume 1. College Publications, 2013.

[25] L. Goble. Prima facie norms, normative conflicts, and dilemmas. In Gabbay et al. [24], pages 241–352.

[26] V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.

[27] R. Gore. Tableau methods for modal and temporal logics. In D'Agostino et al. [17], pages 297–396.

[28] G. Gottlob. Complexity results for nonmonotonic logics. *Journal of Logic and Computation*, 2(3):397–425, 1992.

[29] J. Hansen. Prioritized conditional imperatives: problems and a new proposal. *Autonomous Agents and Multi-Agent Systems*, 17(1):11–35, 2008.

[30] R. Hilpinen and P. McNamara. Deontic logic: A historical survey and introduction. In Gabbay et al. [24], pages 3–136.

[31] G. Hoffmann and C. Areces. Htab: a terminating tableaux system for hybrid logic. *Electronic Notes in Theoretical Computer Science*, 231:3–19, 2009.

[32] J. Horty. Nonmonotonic foundations for deontic logic. In Nute [41], pages 17–44.

[33] S. Khosla and T. Maibaum. The prescription and description of state based systems. In *Temporal Logic in Specification*, volume 398 of *LNCS*, pages 243–294. Springer, 1987.

[34] A. Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley, 2009.

[35] W. Łukaszewicz. Considerations on default logic: An alternative approach. *Computational Intelligence*, 4:1–16, 1988.

[36] F. Massacci. Single step tableaux for modal logics. *Journal of Automated Reasoning*, 24(3):319–364, 2000.

[37] T. McCarty. Defeasible deontic reasoning. *Fundamenta Informaticae*, 21(1/2):125–148, 1994.

[38] P. McNamara. Deontic logic. In D. M. Gabbay and J. Woods, editors, *Logic and the Modalities in the Twentieth Century*, volume 7 of *Handbook of the History of Logic*, pages 197–288. North-Holland, 2006.

[39] Robert Mullins. Rights in default logic. In *13th International Conference Deontic Logic in Computer Science (DEON 2016)*, pages 18–21. College Publications, 2016.

[40] D. Nute. Apparent obligation. In *Defeasible Deontic Logic* [41], pages 287–315.

[41] D. Nute, editor. *Defeasible Deontic Logic*. Kluwer, 1997.

[42] G. Priest. *An Introduction to Non-classical Logic: From If to Is*. Cambridge U Press, 2000.

[43] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1-2):81–132, 1980.

[44] V. Risch. Analytic tableaux for default logics. *Journal of Applied Non-Classical Logics*, 6(1):71–88, 1996.

[45] L. van der Torre and Y. Tan. The many faces of defeasibility in defeasible deontic logic. In Nute [41], pages 79–121.

[46] L. van der Torre and Y. Tan. Contextual deontic logic. In *Formal Models of Agents (ModelAge'97)*, volume 1760 of *LNCS*, pages 240–251. Springer, 1999.

[47] R. Wieringa and J.-J. Meyer. Applications of deontic logic in computer science: A concise overview. In J.-J. Meyer and R. Wieringa, editors, *Deontic Logic in Computer Science*, pages 17–40. John Wiley & Sons, 1994.

[48] M. Willer. Dynamic foundations for deontic logic. In N. Charlow and M. Chrisman, editors, *Deontic Modality*. Oxford U Press, 2016.

[49] G. H. Von Wright. Deontic logic. *Mind*, 60(237):1–15, 1951.

A Proof of Selected Theorems

We refer to a default tableau when there is no need to distinguish between a credulous or sceptical default tableau. We assume all default tableaux are well-formed. We assume that $\mathbf{T}$ is the set of all tableaux for SDLA and $\mathbf{T}_{\mathscr{D}}$ is the set of all default tableaux for $\mathscr{D}$SDLA.

Definition A.1. Let $\tau \in \mathbf{T}_{\mathscr{D}}$; define $\mathsf{p}_{\mathbf{T}}(\tau)$ as the tree obtained by removing all nodes n of τ that are a default rule, re-linking the immediate predecessor of n with the immediate successor of n. Define $\mathsf{p}_{\mathscr{D}}(\tau)$ as the set of defaults in τ. Let β be a branch of τ, define $\mathsf{p}_{\mathscr{D}}(\beta)$ as the set of defaults in β. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, $E = \Phi \cup \Delta' \in \mathscr{E}(\mathbf{A})$, and τ be a default tableau from $\mathbf{A}$; define $\mathsf{p}_{E}(\tau) = \tau'$, where τ' is obtained from τ by pruning all sub-trees of τ starting from a node with a default not in Δ'.

Lemma A.1. If τ is a default tableau from $\langle \Phi, \Delta, \prec \rangle$, $\mathsf{p}_{\mathbf{T}}(\tau)$ is a tableau from $\Phi \cup X(\mathsf{p}_{\mathscr{D}}(\tau))$. *Proof.* Immediate by construction. □

Lemma A.2. Let τ be a default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$; if β is an open and saturated branch of τ; $\Phi \cup X(\mathsf{p}_{\mathscr{D}}(\beta)) \not\models \varphi$. *Proof.* Immediate from Lemma A.1 and from completeness of $\mathbf{T}$. □

Lemma A.3. Let τ be a default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$; if β is an open and saturated branch of τ, $\Phi \cup X_{\mathsf{p}_{\mathscr{D}}(\beta)}$ is an extension of $\langle \Phi, \Delta, \prec \rangle$. *Proof.* It suffices to show that $\mathsf{p}_{\mathscr{D}}(\beta)$ is a series of enabled detachments (Def. 3.1). This fact is immediate by construction. □

Proposition A.1. Let τ be a sceptical default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$; if β is an open and saturated branch of τ, $\langle \Phi, \Delta, \prec \rangle \not\approx^{s} \varphi$. *Proof.* Immediate from Lemmas A.2 and A.3. □

Lemma A.4. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, and τ be a closed sceptical default tableau for $\neg\varphi$ from $\mathbf{A}$; for every $E \in \mathscr{E}(\mathbf{A})$, $\mathsf{p}_{\mathbf{T}}(\mathsf{p}_{E}(\tau))$ extends to a closed tableau. *Proof.* By cases. Let $E = \Phi \cup \Delta'$ be any extension of $\mathbf{A}$; if β is a closed branch of τ s.t. $\mathsf{p}_{\mathscr{D}}(\beta) \subseteq \Delta'$, then, β is a closed branch of $\mathsf{p}_{\mathbf{T}}(\mathsf{p}_{E}(\tau))$. If β is a closed branch of τ s.t. $\mathsf{p}_{\mathscr{D}}(\beta) \not\subseteq \Delta'$, then, there is a prefix β' of β that is a branch of $\mathsf{p}_{E}(\tau)$. This prefix β' is not necessarily closed (we may have pruned the suffix closing the branch). Nonetheless, the side conditions of the default expansion rule (S) in Fig. 3 guarantee that β' is a prefix of a closed branch β'' of $\mathsf{p}_{E}(\tau)$. Thus β' extends to a closed branch of $\mathsf{p}_{\mathbf{T}}(\mathsf{p}_{E}(\tau))$. □

Proposition A.2. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, and τ be a closed sceptical default tableau for $\neg\varphi$ from $\mathbf{A}$; $\mathbf{A} \approx \varphi$. *Proof.* By contradiction. Suppose that there is $E \in \mathscr{E}(\mathbf{A})$ s.t. $E \not\models \varphi$. Then, every tableau for $\neg\varphi$ from E is open (from the soundness of $\mathbf{T}$). But $\mathsf{p}_{\mathbf{T}}(\mathsf{p}_{E}(\tau))$ extends to a closed tableau for $\neg\varphi$ from E (from Lemma A.4). □

Lemma A.5. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$, and τ be a credulous default tableau from $\mathbf{A}$; there is $E \in \mathscr{E}(\mathbf{A})$ s.t. $\Phi \cup \mathsf{p}_{\mathscr{D}}(\beta) \subseteq E$; in other words, $\mathsf{p}_{\mathbf{T}}(\mathsf{p}_{E}(\tau)) = \mathsf{p}_{\mathbf{T}}(\tau)$; alternatively, $\mathsf{p}_{E}(\tau)$ is a tableau from E. *Proof.* Immediate by construction. □

Proposition A.3. Let τ be a closed credulous default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$; $\langle \Phi, \Delta, \prec \rangle \approx^{c} \varphi$. *Proof.* Immediate from Lemma A.5 and the soundness of $\mathbf{T}$. □

Lemma A.6. Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$; for every $E \in \mathscr{E}(\mathbf{A})$, if $E \models \varphi$, there is a closed credulous default tableau for $\neg\varphi$ from $\mathbf{A}$. *Proof.* If $E \models \varphi$, there is a closed tableau τ for $\neg\varphi$ from E. The key observation is to use the construction of τ to construct a closed credulous default tableau τ' for $\neg\varphi$ from $\mathbf{A}$. If $\Phi \models \varphi$, the construction of τ' is direct. Otherwise, let $E = \Phi \cup X_{\Delta'}$; since

$E \models \varphi$ and $\Phi \not\models \varphi$, τ uses some of the consequents of the defaults in Δ' to close the tableau. Each time that τ uses one such consequent $X_{\{\delta\}}$, τ' uses the default expansion rule (C) in Fig. 3 to introduce all defaults in Δ' preceding δ before introducing δ (this is due to the fact that we need to satisfy the side condition of the default expansion rule (C) before incorporating δ to τ'). The result follows from this fact. □

Proposition A.4. If $\langle \Phi, \Delta, \prec \rangle \approx^c \varphi$, then there is a closed credulous default tableau for $\neg\varphi$ from $\langle \Phi, \Delta, \prec \rangle$. *Proof.* Immediate from Lemma A.6. □

Soundness, Completeness, and Termination: Let $\mathbf{A} = \langle \Phi, \Delta, \prec \rangle$; the soundness and completeness of sceptical default tableaux, i.e., $\mathbf{A} \mid\!\sim^s \varphi$ iff $\mathbf{A} \approx^s \varphi$, follows from Props. A.1 and A.2. The soundness and completeness of credulous default tableaux, i.e., $\mathbf{A} \mid\!\sim^c \varphi$ iff $\mathbf{A} \approx^c \varphi$, follows from Props. A.3 and A.4. Termination of default tableau is guaranteed by the use of loop-checks and the fact that the default expansion rules (S) and (C) in Fig. 3 can only be applied a finite number of times.