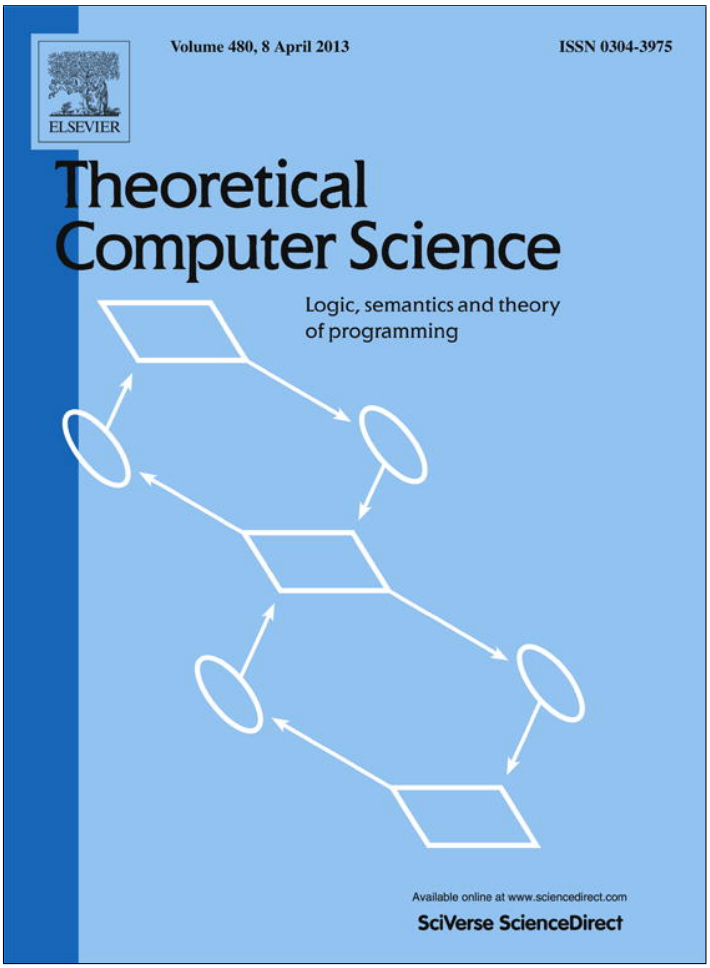


Theoretical Computer Science 480 (2013) 43–68

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Semantics and proof-theory of depth bounded Boolean logics☆

Marcello D'Agostino [a,*], Marcelo Finger [b], Dov Gabbay [c,d,e]

[a] Department of Economics and Management, University of Ferrara, Italy
[b] Department of Computer Science, University of Sao Paulo, Brazil
[c] Bar Ilan University, Israel
[d] King's College London, United Kingdom
[e] University of Luxembourg, Luxembourg

ARTICLE INFO

Article history:
Received 22 July 2012
Received in revised form 13 January 2013
Accepted 1 February 2013
Communicated by A. Avron

Keywords:
Boolean logic
Tractability
Natural deduction
Automated deduction

ABSTRACT

We present a unifying semantical and proof-theoretical framework for investigating depth-bounded approximations to Boolean Logic, namely approximations in which the number of nested applications of a single structural rule, representing the classical Principle of Bivalence, is bounded above by a fixed natural number. These approximations provide a hierarchy of tractable logical systems that indefinitely converge to classical propositional logic. The framework we present here brings to light a general approach to logical inference that is quite different from the standard Gentzen-style approaches, while preserving some of their nice proof-theoretical properties, and is common to several proof systems and algorithms, such as KE, KI and Stålmarck's method.

1. Introduction

Logic is informationally trivial and, yet, computationally hard. This is one of the most baffling paradoxes arising from the traditional account of logical consequence. Triviality stems from the widely accepted characterization of deductive inference as "tautological", in the sense of being *uninformative* or *non-ampliative*: the information carried by the conclusion is (in some sense) contained in the information carried by the premisses. Computational hardness stems from the well-known results showing that most interesting logics are either undecidable or likely to be intractable. In classical quantification theory, the tension between the alleged triviality of logical consequence and the established fact that it admits of no mechanical decision procedure was described by Jaakko Hintikka as a true "scandal of deduction" [38].[1] In Hintikka's view, the scandal was confined to first-order logic: by virtue of its decidability, Boolean Logic was prima facie consistent with the traditional view that logical consequence is uninformative. However, Cook's theorem [14] – according to which Boolean logic is tractable if and only if $\mathscr{P} = \mathscr{NP}$ – strongly suggests that even in the innocuous-looking domain of Boolean languages, logical consequence is probably far from being uninformative.[2] This unsettling situation is also related to the so-called *problem of logical omniscience*: if logic were informationally trivial, then a rational agent should *always* be aware that a certain sentence

☆ We wish to thank Hykel Hosni, Anthony Hunter and Sanjay Modgil for discussing previous versions of this paper and David Makinson for some inspiring suggestions. Thanks are also due to two anonymous referees for helpful comments and corrections. Marcelo Finger would like to acknowledge the partial support of CNPq grant PQ 302553/2010-0, Fapesp grants 2011/19860-4, 2008/03995-5 and 2010/51038-0.

* Corresponding author.
E-mail addresses: dgm@unife.it (M. D'Agostino), mfinger@ime.usp.br (M. Finger), dov.gabbay@kcl.ac.uk (D. Gabbay).

1 For a recent criticism of Hintikka's proposed solution, see [47].

2 On this point see [23], where some of the ideas underlying the present work were first put forward, and [21].

0304-3975/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2013.02.014

is a logical consequence of the data. However, this is clearly not the case when the logic in question lacks a feasible decision procedure.

A typical response to this paradox is that logical systems are *idealizations* and, as such, not intended to faithfully describe the actual deductive behavior of rational agents. As Gabbay and Woods put it:

> A logic is an idealization of certain sorts of real-life phenomena. By their very nature, idealizations misdescribe the behavior of actual agents. This is to be tolerated when two conditions are met. One is that the actual behavior of actual agents can defensibly be made out to approximate to the behavior of the ideal agents of the logician's idealization. The other is the idealization's facilitation of the logician's discovery and demonstration of deep laws. [34, p. 158]

This raises what can be called the *approximation problem* that, in the context of logical systems, can be concisely stated as follows:

Approximation Problem. *Can we define, in a natural way, a hierarchy of logical systems that indefinitely approximate a given idealized Logic (e.g., Boolean Logic) in such a way that, in all practical contexts, suitable approximations can be taken to model the inferential power of actual, resource-bounded agents?*

Stable solutions to this problem are likely to have a significant practical impact in all research areas – from knowledge engineering to economics – where there is an urgent need for more realistic models of deduction, and involve an imaginative re-examination of logical systems as they are usually presented in the literature.[3]

The idea of approximating Boolean logic via tractable subsystems of increasing inferential power has received some attention in Computer Science and Artificial Intelligence [12,25,26,15,48,29,30,32,33,31], but comparatively little attention has been devoted to embedding such efforts in a systematic proof-theoretical and semantic framework. In this paper we aim to fill this gap and propose a unifying approach. Because of the nature of this task, some of the ideas and results presented here are new, while (variants of) others have repeatedly and independently come up in the logical literature – sometimes within different communities that hardly communicate with each other – in such a way that is quite difficult to trace their origin or give proper credit for each of them (see Section 4 for an attempt in this direction). We hope that the present contribution may also shed light on these ideas, clarify their connections, and help building bridges between different research areas.

The unifying approach presented in this paper is rooted in a long-standing research programme started in [41,42,44] and further developed in [16,24,45,18,29–31,23,20]. The heuristic hard-core of this research programme is concisely described by the following statements:

- The classical meaning of the Boolean operators is *overdetermined* by the standard truth-functional semantics and this fact is probably responsible for the lack of a suitable inferential semantics[4] for them.
- Boolean inferences are better construed as arising from the interplay between a weaker basic semantics for the logical operators and the purely structural principles of *Bivalence* and *Non-Contradiction*.
- A suitable inferential semantics for the standard classical operators is given by the *introduction* rules of the system KI [41,16,45] combined with the *elimination* rules of the system KE [42,16,18,19]; these rules are not complete for Boolean Logic, but completeness is achieved by adding suitable structural rules corresponding to the two principles of Bivalence and Non-Contradiction.
- The separation between the inferential role played by the meaning of the logical operators and the inferential role played by the structural principles naturally prompts for the definition of *depth-bounded* approximations in which nested applications of the structural rule expressing the Principle of Bivalence are limited above by a fixed natural number.

A similar heuristic was implicit in the work of Gunnar Stålmarck [49] and has been, independently, pursued [48,8–10] with more practical motivations, leading to efficient and widely used techniques for software verification (see Section 4 for further details).

The present paper is a systematic development of ideas put forward in [23]. We first show (Section 2) that the set of rules obtained by generalizing the introduction rules of KI and the elimination rules of KE to a language with *arbitrary* Boolean operators defines a tractable "natural deduction" system for the chosen operators that (i) is a logic in Tarski's sense, (ii) enjoys the subformula property (with no increase in proof-length) and (iii) allows for a semantic characterization in terms of non-compositional partial valuations or, equivalently, of a certain kind of non-deterministic matrices. Next (Section 3) we investigate the hierarchy of tractable extensions of this basic system that are defined by bounding the application of the Principle of Bivalence in a variety of ways.

[3] This kind of approximation problem arises from the computational idealizations typically made by logical models. But clearly these models involve other kinds of idealizations (see, for instance [28] on this point) that define other kinds of approximation problems.

[4] By "inferential semantics" we mean the approach that identifies the meaning of the logical operators with the role they play in basic inferences. This is usually related to the meaning-as-use approach advocated by the later Wittgenstein and to Gentzen's suggestion in [35] that the rules of his system of Natural Deduction could be taken as definitions of the logical operators. (Indeed, he proposed that the introduction rules would be sufficient for this purpose and that the elimination rules could be ultimately "justified" in terms of the introduction rules.)

More specifically, in Section 2.2 we present a modular semantics for arbitrary n-ary operators based on partial valuations.[5] The resulting consequence relation, although being a proper subsystem of Boolean Logic, is still a logic in Tarski's sense. The modular semantics is subsequently used, in Sections 2.3 and 2.4, to characterize a set of introduction and elimination rules (or *intelim rules*, for short) for arbitrary operators that satisfy a form of the inversion principle[6] and define a natural deduction system with no discharge rules. Section 2.5 shows that every intelim proof can be turned into one that enjoys the subformula property with *no increase in the size of the proof*. Section 2.6 shows that the pure intelim logic is tractable. Next, in Sections 2.8 and 2.9 we first show, in the spirit of [36], that the pure intelim logic is not characterized by any finite valuation system. We also show that the modular semantics of Section 2.2 is equivalent to a certain kind of non-deterministic valuation system that was first proposed by Crawford and Etherington [15] to provide semantics for unit resolution. This result strongly suggests that the intelim system can be seen as a generalization of unit resolution to an unrestricted language with arbitrary operators. In Section 3.1 we re-interpret the Principle of Bivalence as a principle allowing for the manipulation of "virtual information", i.e., information that may not belong to the current information state of a given reasoning agent, but whose consideration is essential to carry out some classical inferences. Finally, in Sections 3.2–3.6 we investigate the tractable logical systems that arise from bounding the search space and the manipulation of virtual information in a variety of ways.

2. 0-depth Boolean logics

2.1. Preliminaries

Let $\mathbb{L}$ be an arbitrary propositional language with a finite number of logical operators of fixed finite arity. Let $F(\mathbb{L})$ be the set of well-formed formulas of $\mathbb{L}$. We use p, q, r, possibly with subscripts, as metalinguistic variables for atomic $\mathbb{L}$-formulas, A, B, C, etc., possibly with subscripts, for arbitrary $\mathbb{L}$-formulas, and Γ, Δ, Λ, possibly with subscripts, for sets of $\mathbb{L}$-formulas. In the context of this paper we shall abuse standard terminology and call *consequence relation* on a language $\mathbb{L}$ any relation $\models$ between sets of formulas of $\mathbb{L}$ and formulas of $\mathbb{L}$ satisfying the following conditions:

$$A \models A \qquad \text{(Reflexivity)}$$

$$\Gamma \models A \Longrightarrow \Gamma \cup \Delta \models A \qquad \text{(Monotonicity)}$$

A *Tarski consequence relation* on $\mathbb{L}$ is a consequence relation on $\mathbb{L}$ in the standard sense, i.e., one satisfying the following additional condition:

$$\Gamma \models A \text{ and } \Gamma \cup \{A\} \models B \Longrightarrow \Gamma \models B. \qquad \text{(Cut)}$$

A *substitution* is a mapping $\sigma : F(\mathbb{L}) \to F(\mathbb{L})$ such that for every n-ary operator $\star$,

$$\sigma(\star(P_1, \ldots, P_n)) = \begin{cases} \star(\sigma(P_1), \ldots, \sigma(P_n)) & \text{if } n > 0 \\ \star & \text{if } n = 0. \end{cases}$$

To economize on parentheses we shall simply write σA instead of $\sigma(A)$. We shall also write $\sigma\Delta$, with Δ a set of formulas, for $\{\sigma A | A \in \Delta\}$. A *Tarskian propositional logic* is a pair $L = \langle \mathbb{L}, \models_L \rangle$, where $\mathbb{L}$ is a propositional language and $\models_L$ is a Tarski consequence relation on $\mathbb{L}$ satisfying the following additional condition:

$$\Gamma \models_L A \Longrightarrow \sigma\Gamma \models_L \sigma A \text{ for every substitution } \sigma. \qquad \text{(Substitution Invariance)}$$

If $\mathbb{L}$ is a propositional language with logical operators $\star_1, \ldots, \star_n$, a *valuation system*[7] $\mathscr{V}$ for $\mathbb{L}$ is a structure $\langle V, D, f_1, \ldots, f_n \rangle$, where:

1. V is a set with at least two elements;
2. D is a proper non-empty subset of V;
3. for each i, $1 \leq i \leq n$, f_i is a mapping of V^{a_i} to V, where a_i is the arity of $\star_i$.

A valuation system is *finite*, if V contains a finite number of elements. A *Boolean valuation system* is one in which $V = \{0, 1\}$ and $D = \{1\}$. An *assignment* relative to a valuation system $\mathscr{V}$ for $\mathbb{L}$ is a mapping of the atomic formulas of $\mathbb{L}$ to V. Each assignment ϕ induces the (total) *valuation* v_ϕ over V defined as follows:

---

[5] Some of the ideas in this section have been inspired by David Makinson (private communication). This modular semantics is related, but not equivalent, to the semantics proposed in [23] for the language with the four standard Boolean operators. In particular, the modular semantics does not validate the "mingle rules" of [23] and so provides a sharp characterization of the pure intelim system.

[6] The Inversion Principle essentially says that no information can be obtained from applying an elimination rule to a sentence A that would not have already been available if A had been obtained by means of an introduction rule. It plays a crucial role in the admissibility of a set of natural deduction rules as definitions of the logical operators. For the historical background and a discussion see [46].

[7] Here we follow the terminology of [27]. In many contexts a valuation system is also called "matrix".

- $v_\phi(p) = \phi(p)$ for all atomic p,
- $v_\phi(\star_i(A_1, \ldots, A_{a_i})) = f_i(v_\phi(A_1), \ldots, v_\phi(A_{a_i}))$.

For every set Γ of $\mathbb{L}$-formulas and every $\mathbb{L}$-formula A, $\Gamma \models_{\mathscr{V}} A$ (Γ *entails* A in $\mathscr{V}$) if and only if, for every assignment ϕ, $v_\phi(A) \in D$ whenever $v_\phi(B) \in D$ for all $B \in \Gamma$. Given a propositional logic $L = \langle \mathbb{L}, \models_L \rangle$ and a valuation system $\mathscr{V}$ for $\mathbb{L}$, we say that

- $\mathscr{V}$ is *faithful* to L if, and only if, for every Γ and A, $\Gamma \models_L A$ implies $\Gamma \models_{\mathscr{V}} A$;
- $\mathscr{V}$ is *characteristic* for L if, and only if, $\mathscr{V}$ is faithful to L and, for all Γ and A, $\Gamma \models_{\mathscr{V}} A$ implies $\Gamma \models_L A$.

An *approximation system* for L is a triple $\mathscr{A} = \langle P, \preceq, \{R_\alpha\}_{\alpha \in P} \rangle$, where $(P, \preceq)$ is a directed set, called the *parameter set*, and $\{R_\alpha\}_{\alpha \in P}$ is a family of consequence relations on $\mathbb{L}$ such that:

- $\alpha \prec \beta$ implies $R_\alpha \subset R_\beta$,
- for each $\alpha \in P$, R_α is decidable in polynomial time,
- $\bigcup_{\alpha \in P} R_\alpha = \models_L$.

We shall call L the *limiting logic* of the approximation system $\mathscr{A}$. Each relation R_α is an *approximation* of L. Clearly, approximation systems are of practical and theoretical interest whenever the limiting logic is known or conjectured to be intractable. In the context of this paper we are concerned with approximation systems where the limiting logic is *classical propositional logic* over some arbitrary propositional language $\mathbb{L}$.

Observe that there is no *a priori* reason for a set of logical operators that is functionally complete for Boolean logic to be functionally complete also for its approximations. So, we shall define approximation systems for any logic $L = \langle \mathbb{L}, \models_L \rangle$, where $\mathbb{L}$ is an arbitrary language and $\models_L$ is a *Boolean consequence relation*, i.e., a consequence relation on $\mathbb{L}$ characterized by the Boolean valuation system for $\mathbb{L}$. To simplify our treatment we shall assume that $\mathbb{L}$ includes the four standard Boolean operators associated with the usual Boolean functions, but the results presented in this paper are independent of this assumption.

2.2. Modular semantics

The main aim of this section is to define a basic notion of "information state" and a corresponding notion of 0-depth consequence.[8] This will be done in Definitions 2.10 and 2.13 at the end of a longish sequence of preliminary steps.

In what follows we shall restrict our attention to *non-degenerate* Boolean operators, i.e., to n-ary operators $\star$ such that $f_\star(x_1, \ldots, x_n) = 1$ and $f_\star(y_1, \ldots, y_n) = 0$ for some n-tuples $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ of values in $\{0, 1\}$.

Definition 2.1. For every formula A:

- a *subformula* of A is defined inductively as follows:
  1. A is a subformula of A;
  2. if $\star(B_1, \ldots, B_n)$ is a subformula of A, then so are $B_1, \ldots, B_n$;
- a *proper subformula* of A is any subformula of A that is different from A;
- an *immediate subformula* of A is any proper subformula of A that is not a proper subformula of any proper subformula of A.

Definition 2.2. A *partial valuation* for $\mathbb{L}$ is a partial mapping v from $F(\mathbb{L})$ to $\{0, 1\}$.

We write $v(A) = \bot$ whenever v is undefined for A. Thus, a partial valuation can be seen as a total mapping from $F(\mathbb{L})$ to $\{0, 1, \bot\}$ where $\bot$ is treated as a third value. For the sake of easier exposition, in the sequel we shall use the three-valued representation.

Partial valuations can be interpreted in a variety of ways. According to one interpretation $v(A) = 1$ means intuitively that A is true, $v(A) = 0$ that A is false and $v(A) = \bot$ that A is neither-true–nor-false. According to another, which is the one we shall adopt in this paper, a partial valuation v represents the information held by a given agent a about the truth or falsity of sentences; $v(A) = 1$ means "a holds the information that A is true" , $v(A) = 0$ means "a holds the information that A is false" and $v(A) = \bot$ means "a holds no information about the truth or falsity of A".[9]

We take the three values as partially ordered by the relation $\preceq$ such that $x \preceq y$ ("x is less defined than, or equal to, y") if, and only if, $x = \bot$ or $x = y$ for $x, y \in \{0, 1, \bot\}$. Partial valuations are, in turn, partially ordered by the usual approximation relation $\sqsubseteq$ defined as follows: $v \sqsubseteq w$ (read: "v approximates w" or "w refines v") if and only if $v(A) \preceq w(A)$ for all A. The set of all partial valuations partially ordered by $\sqsubseteq$ forms a meet-semilattice with a bottom element consisting of the partial valuation that takes the undefined value for all formulas. We denote by $v \sqcap w$ the meet of the partial valuations v and w.

8 These notions are *not* equivalent to the similar notions defined in [23]; see footnote 5 above.

9 For other interpretations of partial valuations, see [11].

Our aim, now, is to define an "information state" as a special kind of partial valuation that (i) is "locally" compatible with the classical interpretation of the logical operators and (ii) is closed under a most basic kind of implicit information.

First, we need to pick out, from the set of all partial valuations, those that agree with the classical truth-tables. In the presence of the "undefined" value $\bot$, this can be done in a variety of ways. Here we investigate a *modular* approach, in which agreement is checked "locally" with respect to the main logical operator of a formula, ignoring the other operators that may occur in it, that is, treating the operands as if they were (distinct) atomic formulas.

Definition 2.3. An $\mathbb{L}$-*module* is a set consisting of a non-atomic $\mathbb{L}$-formula, called the *top formula* of the module, and of its *immediate* subformulas, called the *secondary formulas* of the module.

We shall denote by $\mathsf{Mod}(A)$ the unique $\mathbb{L}$-module whose top formula is A.

Definition 2.4. A *valuation module* is any mapping $\alpha : \mathsf{Mod}(A) \to \{0, 1, \bot\}$, where A is some non-atomic formula. Given a partial valuation v, we say that α *is a module of* v if $\alpha = v \mid \mathrm{dom}(\alpha)$.

By extension, we shall call *top formula of* α the top formula of $\mathrm{dom}(\alpha)$, and *main operator of* α the main operator of the top formula of α.

For each non-atomic formula $B = \star(A_1, \ldots, A_n)$ and each $i = 1, \ldots, n+1$, let

$$\pi_i(B) =_{\mathrm{def}} \begin{cases} A_i & \text{if } 1 \le i \le n \\ B & \text{if } i = n+1. \end{cases}$$

In the sequel we shall use the boldface letters $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$, etc. to denote finite vectors with values in $\{0, 1, \bot\}$.

Definition 2.5. For each valuation module $\alpha : \mathsf{Mod}(A) \to \{0, 1, \bot\}$, the *value vector* of α, denoted by $\vec{\alpha}$, is the vector $\mathbf{x} \in \{0, 1, \bot\}^{a+1}$, where a is the arity of the main logical operator of A, such that $x_i = \alpha(\pi_i(A))$ for all $i = 1, \ldots, a+1$.

We say that a vector $\mathbf{x} \in \{0, 1, \bot\}^n$ *approximates* a vector $\mathbf{y} \in \{0, 1, \bot\}^n$, and write $\mathbf{x} \preceq \mathbf{y}$ if, and only if, $x_i \preceq y_i$ for all $i = 1, \ldots, n$. Notice that, for every fixed n, the set of all vectors in $\{0, 1, \bot\}^n$ partially ordered by $\preceq$ is a meet semilattice. We denote by $\mathbf{x} \wedge \mathbf{y}$ the meet of $\mathbf{x}$ and $\mathbf{y}$. For each n-ary operator $\star$, let us denote by $\mathscr{A}_\star$ the set of all vectors in $\{0, 1, \bot\}^{n+1}$ that approximate some vector in the graph of $f_\star$ (i.e., the Boolean function associated with $\star$ by the valuation system that characterizes the limiting logic L), namely the set

$$\{\mathbf{x} \mid \exists \mathbf{y}(\mathbf{x} \preceq \mathbf{y} \text{ and } \mathbf{y} \in f_\star)\},$$

where by "$\mathbf{y} \in f_\star$" we mean that $\mathbf{y}$ belongs to the graph of $f_\star$. Let $\mathbf{M}_\star$ denote the set of all valuation modules α such that the top formula of α is a $\star$-formula.

Definition 2.6. A *module* $\alpha \in \mathbf{M}_\star$ is *admissible* iff its value vector $\vec{\alpha}$ is in $\mathscr{A}_\star$. A *partial valuation* v is *admissible* iff all its modules are admissible.

Thus an admissible partial valuation is one that conveys *partial* information about the sentences of the given language in a way that agrees with the Boolean truth-tables for the logical operators in the modular sense specified above: the value assigned to each formula A is compatible with those assigned to its immediate subformulas according to the truth-table for the *main* logical operator of A.

Now, we want to specify what it means that a certain piece of information, specifying that a given sentence is true or false, is "implicitly contained" in the information that is explicitly conveyed by a partial valuation v.

Let us denote by $\mathbf{x}[i := z]$, for $z \in \{0, 1, \bot\}$ the vector $\mathbf{y}$ such that $y_i = z$ and $y_j = x_j$ for all $j \neq i$.

Definition 2.7. For every n-ary operator $\star$, a vector $\mathbf{x} \in \mathscr{A}_\star$ is *stable in* $\mathscr{A}_\star$ if, and only if, for every i, $1 \le i \le n+1$,

$$x_i = \bot \Longrightarrow \mathbf{x}[i := 0] \in \mathscr{A}_\star \quad \text{and} \quad \mathbf{x}[i := 1] \in \mathscr{A}_\star.$$

In other words, a stable approximation of a vector in the graph of $f_\star$ is a vector such that each of its "undefined" components may indifferently take, in some better approximation, either of the two defined values 0 and 1. On the other hand, a vector in $\mathscr{A}_\star$ is unstable whenever, for some of its undefined components, one of the two possible defined values is ruled out by the truth-table for $\star$, and so the other one is uniquely determined as the only possible defined value that this component can take.

Definition 2.8. A *valuation module* $\alpha \in \mathbf{M}_\star$ is *informationally closed* iff $\vec{\alpha}$ is a stable element of $\mathscr{A}_\star$. A *valuation* v is *informationally closed* if all its modules are informationally closed.

If α is not informationally closed, that is, $\vec{\alpha}$ is *un*stable, then there is an $A \in \mathrm{dom}(\alpha)$ such that $\alpha(A) = \bot$, but α carries implicit information about A, because there is a defined value for A that is *deterministically dictated* by the truth-table for the main operator of α. An informationally closed module is one in which all such implicit information has been made explicit. It is not difficult to verify that:

Proposition 2.9. *For every n-ary operator $\star$, if $\mathbf{x}$ and $\mathbf{y}$ are both stable elements of $\mathscr{A}_\star$, their meet $\mathbf{x} \wedge \mathbf{y}$ is also a stable element of $\mathscr{A}_\star$.*

Proof. By Definition 2.7, if $\mathbf{z} = \mathbf{x} \wedge \mathbf{y}$ and $\mathbf{z}$ is unstable, then for some i, $1 \leq i \leq n+1$, $z_i = \bot$ and either $\mathbf{z}[i := 1] \notin \mathscr{A}_\star$ or $\mathbf{z}[i := 0] \notin \mathscr{A}_\star$. Since $\mathbf{x}$ and $\mathbf{y}$ are both stable, $\mathbf{z} \preceq \mathbf{x}$ and $\mathbf{z} \preceq \mathbf{y}$, it follows that either x_i and y_i are both equal to 0 or they are both equal to 1. Hence $\mathbf{z}_i = 0$ or $\mathbf{z}_i = 1$, against the assumption. □

This is sufficient to ensure that, for every α, the set of all informationally closed β such that $\alpha \sqsubseteq \beta$ has a minimum element. Therefore, for every valuation v, the set of informationally closed valuations w such that $v \sqsubseteq w$ also has a minimum element.

Definition 2.10. A partial valuation is an *information state* if it is informationally closed.

Examples 2.11. According to the above definition, the module α described by the following table:

$$\begin{array}{ccc} p \vee q & p & (p \vee q) \wedge p \\ \hline 1 & \bot & 0 \end{array} \tag{1}$$

is admissible, because its value vector $\vec{\alpha} = (1, \bot, 0)$ approximates the vector $(1, 0, 0)$ which is in the graph of $f_\wedge$ (since $f_\wedge(1, 0) = 0$). On the other hand, the following modules:

$$\begin{array}{ccc} p \vee r & q \to s & (p \vee r) \wedge (q \to s) \\ \hline 0 & \bot & 1 \end{array} \qquad \begin{array}{ccc} p \wedge r & q & (p \wedge r) \vee q \\ \hline 1 & \bot & 0 \end{array}, \tag{2}$$

are *not* admissible, because $(0, \bot, 1)$ does not approximate any vector in the graph of $f_\wedge$ and $(1, \bot, 0)$ does not approximate any vector in the graph of $f_\vee$.

The following modules

$$\begin{array}{ccc} p \vee r & q \to s & (p \vee r) \wedge (q \to s) \\ \hline \bot & \bot & 0 \end{array} \qquad \begin{array}{ccc} p \wedge r & q & (p \wedge r) \vee q \\ \hline \bot & \bot & 1 \end{array}, \tag{3}$$

are informationally closed. The first module, is informationally closed because its value vector $(\bot, \bot, 0)$ is a stable element of $\mathscr{A}_\wedge$, that is, the vectors $(0, \bot, 0)$, $(1, \bot, 0)$, $(\bot, 0, 0)$ and $(\bot, 1, 0)$ are all in $\mathscr{A}_\wedge$. The second one is informationally closed because its value vector $(\bot, \bot, 1)$ is a stable element of $\mathscr{A}_\vee$, that is, $(0, \bot, 1)$, $(1, \bot, 1)$, $(\bot, 0, 1)$ and $(\bot, 1, 1)$ are all in $\mathscr{A}_\vee$. On the other hand, the module (1) is not *informationally* closed, because $(1, 1, 0)$ is not in $\mathscr{A}_\wedge$.

Remark 2.12. Observe that:

- Every Boolean valuation is an information state;
- for every information state v, if $v(A) \neq \bot$ for all $A \in F(\mathbb{L})$, then v is a Boolean valuation.

Let us also say that an information state v *verifies* a formula A if $v(A) = 1$, and that v verifies a set Γ of formulas if v verifies A for all $A \in \Gamma$. We are now in a position to define our basic consequence relation $\models_0$.

Definition 2.13. For every set Γ of formulas and every formula A, $\Gamma \models_0 A$ (read "A is a 0-depth consequence of Γ") if and only if A is verified by every information state that verifies Γ.

Definition 2.14. A set Γ of formulas is 0-*depth inconsistent* if and only if there is no information state that satisfies all the formulas in Γ.

The reader can check that $\models_0$ is a consequence relation in Tarski's sense, i.e. that it satisfies Reflexivity, Monotonicity and Cut (see Section 2.1). Moreover, $\models_0$ also satisfies Substitution Invariance and, therefore, is a Tarskian propositional logic. In Section 2.6 it will be shown that $\models_0$ is a tractable logic.

Finally, an important property of $\models_0$ is that, like Kleene's 3-valued logic [40] and Belnap's 4-valued logic [6,7], it has no "tautologies", i.e., for no formula A, it holds true that $\emptyset \models_0 A$. To see this, it is sufficient to observe that the partial valuation v such that $v(A) = \bot$ for all A is, according to our definition, an information state — we could call it the *null* information state. Hence, there is no formula that is verified by *all* information states; in particular no formula is verified by the null information state.

2.3. Boolean natural deduction: intelim sequences

Signed formulas of $\mathbb{L}$ are expressions of the form $T\,A$ or $F\,A$, where A is a formula of $\mathbb{L}$. The intuitive interpretation of such signed formulas is, respectively, "A is true" and "A is false". By a *conjugate* of a signed formula, we mean the result of replacing T with F or F with T, so that the conjugate of $T\,A$ is $F\,A$ and the conjugate of $F\,A$ is $T\,A$. We shall use "$\mathbf{\Gamma}$, $\mathbf{\Delta}$, $\mathbf{\Lambda}$", etc. as metalinguistic variables for sets of signed formulas. We shall also use "S, S_0, S_1, ..." as variables standing for either of the two signs T and F.

We extend Definition 2.1 to signed formulas in the obvious way:

Definition 2.15. $S_1\,A$ is a *(proper, immediate) signed subformula* of $S_2\,B$ if, and only if, A is a (proper, immediate) subformula of B.

By a *(signed) formula scheme* for $\mathbb{L}$ we mean the expression that results from a (signed) formula of $\mathbb{L}$ by uniformly replacing the atomic formulas with metalinguistic variables for arbitrary formulas, such as A, B, C, etc.

Given a sign $S \in \{T, F\}$, let

$$\mathtt{val}(S) = \begin{cases} 1 & \text{if } S = T \\ 0 & \text{if } S = F. \end{cases}$$

Definition 2.16. For every n-ary operator $\star$, with $n > 0$, $\langle\langle S_1\,\psi_1, \ldots, S_k\,\psi_k\rangle, S_0\,\varphi\rangle$ is a *Boolean introduction rule for* $\star$ if, and only if,

1. φ has the form $\star(\theta_1, \ldots, \theta_n)$, where $\theta_1, \ldots, \theta_n$ are *distinct* atomic formula schemes;
2. $\psi_1, \ldots, \psi_k$, $k \leq n$, are distinct elements of $\{\theta_1, \ldots, \theta_n\}$;
3. the vector $\mathbf{x} \in \{0, 1, \bot\}^{n+1}$ such that, for all $i = 1, \ldots, n+1$,

$$x_i = \begin{cases} 1 & \text{if } T\,\pi_i(\varphi) \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ 0 & \text{if } F\,\pi_i(\varphi) \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ \bot & \text{otherwise} \end{cases}$$

is a minimal[10] vector in $\mathscr{A}_\star$ satisfying the following condition for all $\mathbf{y} \in f_\star$:

$$\mathbf{x} \preceq \mathbf{y} \Longrightarrow y_{n+1} = \mathtt{val}(S_0).$$

Observe that, according to the above definition of the vector $\mathbf{x}$, x_{n+1} is always equal to $\bot$. Thus, $\mathbf{x}$ is unstable and $y_{n+1} = z_{n+1}$ for every stable $\mathbf{y}, \mathbf{z} \in \mathscr{A}_\star$ such that $\mathbf{x} \preceq \mathbf{y}$ and $\mathbf{x} \preceq \mathbf{z}$.

Definition 2.17. For every n-ary operator $\star$, with $n > 0$, $\langle\langle S_1\psi_1, \ldots, S_k\,\psi_k\rangle, S_0\,\varphi\rangle$ is a *Boolean elimination rule for* $\star$ if, and only if,

1. ψ_1 has the form $\star(\theta_1, \ldots, \theta_n)$, where $\theta_1, \ldots, \theta_n$ are *distinct* atomic formula schemes;
2. $\varphi, \psi_2, \ldots, \psi_k$, $k \leq n$, are distinct elements of $\{\theta_1, \ldots, \theta_n\}$;
3. the vector $\mathbf{x} \in \{0, 1, \bot\}^{n+1}$ such that, for all $i = 1, \ldots, n+1$,

$$x_i = \begin{cases} 1 & \text{if } T\,\pi_i\psi_1 \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ 0 & \text{if } F\,\pi_i\psi_1 \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ \bot & \text{otherwise} \end{cases}$$

is a minimal vector in $\mathscr{A}_\star$ satisfying the following condition for all $\mathbf{y} \in f_\star$:

$$\mathbf{x} \preceq \mathbf{y} \Longrightarrow y_j = \mathtt{val}(S_0)$$

where j is the unique element of $\{1, \ldots, n\}$ such that $\varphi = \pi_j(\psi_1) = \theta_j$.

The signed formula scheme $S_1\,\psi_1$ is called *the major premiss scheme* of the rule, while the signed formula schemes $S_i\,\psi_i$, for $i = 2, \ldots, k$ are called *minor premiss schemes*.

Observe that, according to the above definition of the vector $\mathbf{x}$, $x_{n+1} = \mathtt{val}(S_1)$. Moreover, if $\varphi = \theta_j$, $x_j = \bot$, otherwise $\mathbf{x}$ would not satisfy the condition in Clause 3. Thus, $\mathbf{x}$ is unstable and $y_j = z_j$ for every stable $\mathbf{y}, \mathbf{z} \in \mathscr{A}_\star$ such that $\mathbf{x} \preceq \mathbf{y}$ and $\mathbf{x} \preceq \mathbf{z}$.

By *Boolean intelim rules* for $\mathbb{L}$ we mean the collection of all Boolean introduction and elimination rules for the logical operators of $\mathbb{L}$. An intelim rule $\langle\langle S_1\psi_1, \ldots, S_k\,\psi_k\rangle, S_0\,\varphi\rangle$ is usually represented as:

$$\frac{\begin{array}{c} S_1\,\psi_1 \\ \vdots \\ S_k\,\psi_k \end{array}}{S_0\,\varphi.}$$

Table 1 lists the intelim rules for the four standard Boolean operators and Table 2 lists the intelim rules for *exclusive or* ($\oplus$), *Peirce's arrow* ($\downarrow$), *Sheffer's stroke* ($|$).

For every intelim rule $\mathscr{R} = \langle\langle S_1\,\psi_1, \ldots, S_k\,\psi_k\rangle, S_0\,\varphi\rangle$, we say that a signed formula $S_0\,A$ *follows from* a set $\mathbf{\Delta}$ of signed formulas *by an application of* $\mathscr{R}$ if there is a substitution ρ of schematic letters with $\mathbb{L}$-formulas such that (i) $\mathbf{\Delta} = \{S_1\,\rho(\psi_1), \ldots, S_k\,\rho(\psi_k)\}$, and (ii) $\rho(\varphi) = A$. The signed formulas in $\mathbf{\Delta}$ and $S_0\,A$ are, respectively, the *premisses* and the *conclusion* of the rule application. If $\mathscr{R}$ is an elimination rule, $S_1\,\rho(\psi_1)$ is the *major premiss* of the rule application.

10 "Minimal" with respect to the $\preceq$ ordering, i.e. minimally defined.

Table 1
Intelim rules for the four standard Boolean operators.

$$\frac{F\,A}{T\,\neg A}\,F\neg\text{-}\mathscr{I} \qquad \frac{T\,A}{F\,\neg A}\,T\neg\text{-}\mathscr{I} \qquad \frac{T\,\neg A}{F\,A}\,T\neg\text{-}\mathscr{E} \qquad \frac{F\,\neg A}{T\,A}\,F\neg\text{-}\mathscr{E}$$

$$\frac{T\,A}{T\,A\vee B}\,T\vee\text{-}\mathscr{I}1 \qquad \frac{T\,B}{T\,A\vee B}\,T\vee\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}F\,A\\F\,B\end{array}}{F\,A\vee B}\,F\vee\text{-}\mathscr{I}$$

$$\frac{\begin{array}{c}T\,A\vee B\\F\,A\end{array}}{T\,B}\,T\vee\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}T\,A\vee B\\F\,B\end{array}}{T\,A}\,T\vee\text{-}\mathscr{E}2 \qquad \frac{F\,A\vee B}{F\,A}\,F\vee\text{-}\mathscr{E}1 \qquad \frac{F\,A\vee B}{F\,B}\,F\vee\text{-}\mathscr{E}2$$

$$\frac{F\,A}{F\,A\wedge B}\,F\wedge\text{-}\mathscr{I}1 \qquad \frac{F\,B}{F\,A\wedge B}\,F\wedge\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}T\,A\\T\,B\end{array}}{T\,A\wedge B}\,T\wedge\text{-}\mathscr{I}$$

$$\frac{\begin{array}{c}F\,A\wedge B\\T\,A\end{array}}{F\,B}\,F\wedge\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}F\,A\wedge B\\T\,B\end{array}}{F\,A}\,F\wedge\text{-}\mathscr{E}2 \qquad \frac{T\,A\wedge B}{T\,A}\,T\wedge\text{-}\mathscr{E}1 \qquad \frac{T\,A\wedge B}{T\,B}\,T\wedge\text{-}\mathscr{E}2$$

$$\frac{F\,A}{T\,A\to B}\,T\to\text{-}\mathscr{I}1 \qquad \frac{T\,B}{T\,A\to B}\,T\to\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}T\,A\\F\,B\end{array}}{F\,A\to B}\,F\to\text{-}\mathscr{I}$$

$$\frac{\begin{array}{c}T\,A\to B\\T\,A\end{array}}{T\,B}\,T\to\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}T\,A\to B\\F\,B\end{array}}{F\,A}\,T\to\text{-}\mathscr{E}2 \qquad \frac{F\,A\to B}{T\,A}\,F\to\text{-}\mathscr{E}1 \qquad \frac{F\,A\to B}{F\,B}\,F\to\text{-}\mathscr{E}2$$

Table 2
Intelim rules for *exclusive or* ($\oplus$), *Peirce's arrow* ($\downarrow$), *Sheffer's stroke* ($|$).

$$\frac{\begin{array}{c}T\,A\\F\,B\end{array}}{T\,A\oplus B}\,T\oplus\text{-}\mathscr{I}1 \qquad \frac{\begin{array}{c}F\,A\\T\,B\end{array}}{T\,A\oplus B}\,T\oplus\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}T\,A\\T\,B\end{array}}{F\,A\oplus B}\,F\oplus\text{-}\mathscr{I}1 \qquad \frac{\begin{array}{c}F\,A\\F\,B\end{array}}{F\,A\oplus B}\,F\oplus\text{-}\mathscr{I}2$$

$$\frac{\begin{array}{c}T\,A\oplus B\\T\,A\end{array}}{F\,B}\,T\oplus\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}T\,A\oplus B\\T\,B\end{array}}{F\,A}\,T\oplus\text{-}\mathscr{E}2 \qquad \frac{\begin{array}{c}T\,A\oplus B\\F\,A\end{array}}{T\,B}\,T\oplus\text{-}\mathscr{E}3 \qquad \frac{\begin{array}{c}T\,A\oplus B\\F\,B\end{array}}{T\,A}\,T\oplus\text{-}\mathscr{E}4$$

$$\frac{\begin{array}{c}F\,A\oplus B\\T\,A\end{array}}{T\,B}\,F\oplus\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}F\,A\oplus B\\T\,B\end{array}}{T\,A}\,F\oplus\text{-}\mathscr{E}2 \qquad \frac{\begin{array}{c}F\,A\oplus B\\F\,A\end{array}}{F\,B}\,F\oplus\text{-}\mathscr{E}3 \qquad \frac{\begin{array}{c}F\,A\oplus B\\F\,B\end{array}}{F\,A}\,F\oplus\text{-}\mathscr{E}4$$

$$\frac{T\,A}{F\,A\downarrow B}\,T\downarrow\text{-}\mathscr{I}1 \qquad \frac{T\,B}{F\,A\downarrow B}\,T\downarrow\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}F\,A\\F\,B\end{array}}{T\,A\downarrow B}\,T\downarrow\text{-}\mathscr{I}$$

$$\frac{\begin{array}{c}F\,A\downarrow B\\F\,A\end{array}}{T\,B}\,F\downarrow\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}F\,A\downarrow B\\F\,B\end{array}}{T\,A}\,F\downarrow\text{-}\mathscr{E}2 \qquad \frac{T\,A\downarrow B}{F\,A}\,T\downarrow\text{-}\mathscr{E}1 \qquad \frac{T\,A\downarrow B}{F\,B}\,T\downarrow\text{-}\mathscr{E}2$$

$$\frac{F\,A}{T\,A|B}\,T|\text{-}\mathscr{I}1 \qquad \frac{F\,B}{T\,A|B}\,T|\text{-}\mathscr{I}2 \qquad \frac{\begin{array}{c}T\,A\\T\,B\end{array}}{F\,A|B}\,F|\text{-}\mathscr{I}$$

$$\frac{\begin{array}{c}T\,A|B\\T\,A\end{array}}{F\,B}\,T|\text{-}\mathscr{E}1 \qquad \frac{\begin{array}{c}T\,A|B\\T\,B\end{array}}{F\,A}\,T|\text{-}\mathscr{E}2 \qquad \frac{F\,A|B}{T\,A}\,F|\text{-}\mathscr{E}1 \qquad \frac{F\,A|B}{T\,B}\,F|\text{-}\mathscr{E}2$$

In the sequel, we shall use X, Y, Z, possibly with subscripts, as metalinguistic variables ranging over signed formulas of $\mathbb{L}$. Moreover, $\overline{X}$ will denote the conjugate of X.

Definition 2.18. Given any set $\mathbf{\Gamma}$ of signed formulas, an *intelim sequence for* $\mathbf{\Gamma}$ is a finite sequence $X_1, \ldots, X_n$ of signed formulas such that for every X_i, $1 \leq i \leq n$, either (i) $X_i \in \mathbf{\Gamma}$, or (ii) X_i follows from some subset of $\{X_1, \ldots, X_{i-1}\}$ by an application of an intelim rule.

Definition 2.19. An *intelim proof* of X from $\mathbf{\Gamma}$ is any intelim sequence for $\mathbf{\Gamma}$ that contains X.

Definition 2.20. An intelim sequence is *closed* if it contains both a signed formula X and its conjugate $\overline{X}$. Otherwise, we say that it is *open*.

Definition 2.21. An *intelim refutation* of $\mathbf{\Gamma}$ is a closed intelim sequence for $\mathbf{\Gamma}$.

Definition 2.22. A *signed* formula X is *intelim-deducible* from a set $\mathbf{\Gamma}$ of *signed* formulas if there is an intelim proof of X from $\mathbf{\Gamma}$. An *unsigned* formula A is *intelim-deducible* from a set Γ of *unsigned* formulas if there is an intelim proof of $T\,A$ from $\{T\,B \mid B \in \Gamma\}$.

For the sake of easier notation, we shall use the same symbol "$\vdash_{IE}$" for the intelim-deducibility relation, no matter whether it refers to signed or unsigned formulas; so, "$\Gamma \vdash_{IE} A$" means that the unsigned formula A is intelim deducible from the set Γ of unsigned formulas, and "$\mathbf{\Gamma} \vdash_{IE} X$" means that the signed formula X is intelim deducible from the set $\mathbf{\Gamma}$ of signed formulas.

Definition 2.23. A set $\mathbf{\Gamma}$ of *signed* formulas is *intelim-inconsistent* if there is an intelim refutation of $\mathbf{\Gamma}$. A set Γ of *unsigned* formulas is *intelim-inconsistent* if there is an intelim refutation of $\{T\,B \mid B \in \Gamma\}$. A set of signed or unsigned formulas is *intelim-consistent* if it is not intelim-inconsistent.

Proposition 2.24. *The binary relation $\vdash_{IE}$ is a Tarskian propositional logic, i.e. it satisfies Reflexivity, Monotonicity, Cut and Substitution Invariance.*

Proof. Left to the reader. □

Proposition 2.25. *If Γ is intelim-inconsistent, then $\Gamma \vdash_{IE} A$ for every formula A.*

Proof. Let A be an arbitrary formula and suppose Γ is intelim-inconsistent. Then there is a closed intelim sequence for $\{T\,B \mid B \in \Gamma\}$, i.e.,

$$\{T\,B \mid B \in \Gamma\} \vdash_{IE} T\,C \text{ and } \{T\,B \mid B \in \Gamma\} \vdash_{IE} F\,C$$

for some formula C. By $T\vee$-$\mathscr{I}1$ (see Table 1) $T\,C \vdash_{IE} T\,C \vee A$, and by $T\vee$-$\mathscr{E}1$, $T\,C \vee A, F\,C \vdash_{IE} T\,A$. Since $\vdash_{IE}$ is closed under Cut, it follows that $\{T\,B \mid B \in \Gamma\} \vdash_{IE} T\,A$ and, therefore, $\Gamma \vdash_{IE} A$. □

2.4. Soundness and completeness of intelim sequences

Given the close correspondence between the proof-theoretical notions of the previous section and the semantical notions of Section 2.2, soundness and completeness of $\vdash_{IE}$ with respect to $\models_0$ are particularly straightforward.

Definition 2.26. A set $\mathbf{\Gamma}$ of signed formulas is *intelim saturated* if it satisfies the following conditions:

1. for no formula A, $T\,A$ and $F\,A$ are both in $\mathbf{\Gamma}$;
2. for every signed formula X, if X follows from some subset of $\mathbf{\Gamma}$ by an application of an intelim rule, then $X \in \mathbf{\Gamma}$.

Proposition 2.27. *A partial valuation v is an information state if and only if the set*

$$\mathbf{\Gamma}_v = \{T\,A | v(A) = 1\} \cup \{F\,A | v(A) = 0\}$$

is intelim saturated.

Proof. Suppose v is an information state and that there is an intelim rule

$$\mathscr{R} = \langle\langle S_1\,\psi_1, \ldots, S_k\,\psi_k\rangle, S_0\,\varphi\rangle$$

such that $S_0\,A$ follows from $\mathbf{\Delta} \subseteq \mathbf{\Gamma}_v$ by an application of $\mathscr{R}$. First, consider the case in which $\mathscr{R}$ is an introduction rule. Let $\alpha = v \mid \mathsf{Mod}(A)$.[11] Since α is a module of v and v is informationally closed, by Definition 2.8 α is informationally closed

[11] Recall that $\mathsf{Mod}(A)$ is the (unique) $\mathbb{L}$-module whose top formula is A.

and $\vec{\alpha}$ is a stable element of $\mathscr{A}_\star$, where $\star$ is the main operator of α. Let $n \geq k$ be the arity of $\star$ and consider the vector $\mathbf{x} \in \{0, 1, \bot\}^{n+1}$ such that, for all $i = 1, \ldots, n+1$,

$$x_i = \begin{cases} 1 & \text{if } T\,\pi_i(\varphi) \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ 0 & \text{if } F\,\pi_i(\varphi) \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\} \\ \bot & \text{otherwise.} \end{cases}$$

Since $S_0\,A$ follows from $\mathbf{\Delta}$ by an application of $\mathscr{R}$, then there is a substitution ρ of schematic letters with $\mathbb{L}$-formulas such that $\mathbf{\Delta} = \{S_1\,\rho(\psi_1), \ldots, S_k\,\rho(\psi_k)\}$ and $\rho(\varphi) = A$. Moreover, $\mathbf{\Delta} \subseteq \mathrm{dom}(\alpha)$. Now, if $x_i = 1$, then $T\,\pi_i(\varphi) \in \{S_1\,\psi_1, \ldots, S_k\,\psi_k\}$, and so:

$$T\,\pi_i(\rho(\varphi)) \in \{S_1\,\rho(\psi_1), \ldots, S_k\,\rho(\psi_k)\} = \mathbf{\Delta}.$$

Since $\rho(\varphi) = A$, then $T\,\pi_i(A) \in \mathbf{\Delta} \subseteq \mathbf{\Gamma}_v$. So, $v(\pi_i(A)) = \alpha(\pi_i(A)) = \vec{\alpha}_i = 1$. Similarly, if $x_i = 0$, then $v(\pi_i(A)) = \alpha(\pi_i(A)) = \vec{\alpha}_i = 0$. Hence, $x_i = \vec{\alpha}_i$ for all i such that $x_i \neq \bot$, and so $\mathbf{x} \preceq \vec{\alpha}$. By Definition 2.16, for every stable $\mathbf{y} \in \mathscr{A}_\star$ such that $\mathbf{x} \preceq \mathbf{y}$,

$$y_{n+1} = \mathtt{val}(S_0). \tag{4}$$

Now, $\vec{\alpha}$ is a stable element of $\mathscr{A}_\star$ and so $\vec{\alpha}_{n+1} = \mathtt{val}(S_0)$. Since, $\alpha(A) = \vec{\alpha}_{n+1}$, then $v(A) = \vec{\alpha}_{n+1}$ and $S_0\,A \in \mathbf{\Gamma}_v$. So $\mathbf{\Gamma}_v$ is closed under $\mathscr{R}$. The argument is similar for the case in which $\mathscr{R}$ is an elimination rule.

We have so shown that $\mathbf{\Gamma}_v$ is closed under all the intelim rules. Moreover, since v is a function, $\mathbf{\Gamma}_v$ satisfies also the condition that for no A, $T\,A$ and $F\,A$ are both in $\mathbf{\Gamma}$. Therefore, $\mathbf{\Gamma}_v$ is intelim saturated.

This concludes the proof of the "only-if" direction, i.e., that $\mathbf{\Gamma}$ is intelim saturated whenever v is an information state. The proof of the converse is left to the reader. □

Proposition 2.28. *A set $\mathbf{\Delta}$ of signed formulas is closed under the intelim rules if and only if $\mathbf{\Delta}$ is closed under intelim-deducibility, i.e., $X \in \mathbf{\Delta}$ whenever $\mathbf{\Gamma} \vdash_{IE} X$ for some $\mathbf{\Gamma} \subseteq \mathbf{\Delta}$.*

Proof. The "if" direction is obvious. For the "only-if" direction, consider that $\vdash_{IE}$ satisfies Reflexivity, Monotonicity and Cut. □

Lemma 2.29. *$\Gamma \models_0 A$ if and only if, for every intelim saturated set $\mathbf{\Delta}$ of signed formulas:*

$$\{T\,B \mid B \in \Gamma\} \subseteq \mathbf{\Delta} \Longrightarrow T\,A \in \mathbf{\Delta}.$$

Proof. Suppose $\Gamma \models_0 A$ and $\mathbf{\Delta}$ is an intelim saturated set that includes $\{T\,B \mid B \in \Gamma\}$. Since $\mathbf{\Delta}$ contains no pair of conjugate signed formulas, the partial valuation v such that

$$v(C) = \begin{cases} 1 & \text{if } T\,C \in \mathbf{\Delta} \\ 0 & \text{if } F\,C \in \mathbf{\Delta} \\ \bot & \text{otherwise} \end{cases}$$

exists and, by Proposition 2.27, is an information state. Moreover, v verifies all the formulas in Γ. Therefore, by definition of $\models_0$, $v(A) = 1$ and so $T\,A \in \mathbf{\Delta}$.

Suppose now that $T\,A$ belongs to every intelim saturated set that includes $\{T\,B \mid B \in \Gamma\}$. By Proposition 2.27, for every information state v that verifies all the formulas in Γ, the set $\mathbf{\Delta} = \{T\,C | v(C) = 1\} \cup \{F\,C | v(C) = 0\}$ is intelim saturated and includes $\{T\,B \mid B \in \Gamma\}$. Therefore, $T\,A \in \mathbf{\Delta}$ and $v(A) = 1$. □

Proposition 2.30 (*Soundness and Completeness*)**.** *$\Gamma \vdash_{IE} A$ iff $\Gamma \models_0 A$.*

Proof. Let $\mathbf{\Gamma} = \{T\,B \mid B \in \Gamma\}$. Suppose $\Gamma \vdash_{IE} A$. By Proposition 2.28, the set $\{S\,C \mid \mathbf{\Gamma} \vdash_{IE} S\,C\}$ is included in every intelim saturated set that includes $\mathbf{\Gamma}$. We distinguish two cases.

Case 1: Γ is intelim-inconsistent. Then, there is no intelim saturated set that includes $\mathbf{\Gamma}$. Hence, by Lemma 2.29, $\Gamma \models_0 A$.

Case 2: Γ is intelim-consistent. Then, $\{S\,C \mid \mathbf{\Gamma} \vdash_{IE} S\,C\}$ must contain $T\,A$ and therefore every intelim saturated set $\mathbf{\Delta}$ that includes $\mathbf{\Gamma}$ must contain $T\,A$. Hence, by Lemma 2.29 again, $\Gamma \models_0 A$.

Suppose now that $\Gamma \nvdash_{IE} A$, that is, $\mathbf{\Gamma} \nvdash_{IE} T\,A$. Then, by Proposition 2.25, Γ must be intelim-consistent and so the set $\mathbf{\Delta} = \{S\,C \mid \mathbf{\Gamma} \vdash_{IE} S\,C\}$ is intelim saturated. By the reflexivity of $\vdash_{IE}$, $\mathbf{\Gamma} \subseteq \mathbf{\Delta}$ and, since $\mathbf{\Gamma} \nvdash_{IE} T\,A$, $T\,A \notin \mathbf{\Delta}$. Therefore, by Lemma 2.29 it follows that $\Gamma \not\models_0 A$. □

2.5. Subformula property

While elimination rules preserve the subformula property – the conclusion of their application is always a signed subformula of the major premiss – introduction rules do not. So, it is important, both for theoretical and practical reasons, to show that the search for intelim proofs or refutations can be restricted, without any loss, to sequences involving only signed subformulas of the assumptions or, in the case of proofs, of the signed formula to be proved. In this section we show that the intelim logic satisfies the subformula property in the following form:

1. If $\mathbf{\Gamma}$ is intelim inconsistent, there is an intelim refutation that involves only signed subformulas of the signed formulas in $\mathbf{\Gamma}$;
2. If $\mathbf{\Gamma}$ is intelim consistent and X is intelim provable from $\mathbf{\Gamma}$, there is an intelim proof that involves only signed subformulas of the signed formulas in $\mathbf{\Gamma} \cup \{X\}$.

This version of the subformula property is sufficient for all practical purposes. For, if we are trying to refute $\mathbf{\Gamma}$ we know that the search space can be restricted to intelim sequences that contain only signed subformulas of the signed formulas in $\mathbf{\Gamma}$: $\mathbf{\Gamma}$ is intelim inconsistent if and only if one of these sequences is closed. On the other hand, if we are trying to prove X from $\mathbf{\Gamma}$ the search space can be restricted to intelim sequences that contain only signed subformulas of the signed formulas in $\mathbf{\Gamma} \cup \{X\}$: $\mathbf{\Gamma} \vdash_{IE} X$ if and only if one of these sequences is closed or contains X.

Definition 2.31. Given two sets $\mathbf{\Gamma}$ and $\mathbf{\Delta}$ of signed formulas, we say that $\mathbf{\Gamma}$ and $\mathbf{\Delta}$ *clash* if there is a formula A such that $S\,A \in \mathbf{\Gamma}$ and $\overline{S\,A} \in \mathbf{\Delta}$.

Lemma 2.32 (*Inversion Principle*)**.** *If a conclusion X follows from $\mathbf{\Gamma} \cup \{Y\}$ by an application of an elimination rule, with Y as major premiss, then X belongs to every $\mathbf{\Delta}$ such that (i) Y follows from $\mathbf{\Delta}$ by an application of an introduction rule, and (ii) $\mathbf{\Delta}$ does not clash with $\mathbf{\Gamma}$.*

Proof. Let $\mathbf{\Gamma}$, $\mathbf{\Delta}$ be sets of signed formulas that do not clash. Let us assume that X follows from $\mathbf{\Gamma} \cup \{Y\}$ by an application of an elimination rule, with Y as major premiss, and that Y follows from $\mathbf{\Delta}$ by an application of an introduction rule. We want to show that $X \in \mathbf{\Delta}$. Let $Y = S_1 \star(B_1, \ldots, B_n)$ for some $S_1 \in \{T, F\}$. Then $X = S_2\,B_j$ for some $j \in \{1, \ldots, n\}$ and $S_2 \in \{T, F\}$.

Consider the vectors $\mathbf{x}, \mathbf{y} \in \{0, 1, \bot\}^{n+1}$ such $x_{n+1} = y_{n+1} = \bot$ and for $i = 1, \ldots, n$:

$$x_i = \begin{cases} 1 & \text{if } T\,B_i \in \mathbf{\Gamma} \\ 0 & \text{if } F\,B_i \in \mathbf{\Gamma} \\ \bot & \text{otherwise.} \end{cases} \qquad y_i = \begin{cases} 1 & \text{if } T\,B_i \in \mathbf{\Delta} \\ 0 & \text{if } F\,B_i \in \mathbf{\Delta} \\ \bot & \text{otherwise.} \end{cases}$$

Observe that the join $\mathbf{x} \vee \mathbf{y}$ (with respect to the $\preceq$ ordering) exists whenever $\mathbf{\Gamma}$ and $\mathbf{\Delta}$ do not clash. Since $Y = S_1 \star(B_1, \ldots, B_n)$ results from $\mathbf{\Delta}$ by an application of an introduction rule, it follows from clause (3) of Definition 2.16 that for all $\mathbf{z} \in f_\star$,

$$\mathbf{y} \preceq \mathbf{z} \Longrightarrow z_{n+1} = \mathtt{val}(S_1) \tag{5}$$

and so

$$\mathbf{x} \vee \mathbf{y} \preceq \mathbf{z} \Longrightarrow z_{n+1} = \mathtt{val}(S_1). \tag{6}$$

On the other hand, since $X = S_2\,B_j$ results from $\mathbf{\Gamma} \cup \{Y\}$ by an application of an elimination rule, it follows from (6) and clause (3) of Definition 2.17 that, for all $\mathbf{z} \in f_\star$,

$$\mathbf{x} \vee \mathbf{y} \preceq \mathbf{z} \Longrightarrow z_j = \mathtt{val}(S_2). \tag{7}$$

Now, for every n-ary operator $\star$, every vector whose $n+1$ component is undefined belongs to $\mathscr{A}_\star$. So, given that $x_{n+1} = y_{n+1} = \bot$, if $x_j = y_j = \bot$, $f_\star$ must contain refinements $\mathbf{u}, \mathbf{w}$ of $\mathbf{x} \vee \mathbf{y}$ such that $u_j = 1$ and $w_j = 0$. Since z_j takes a fixed value in $\{0, 1\}$ for all refinements $\mathbf{z}$ of $\mathbf{x} \vee \mathbf{y}$ in $f_\star$, it follows that either x_j or y_j must already be equal to this fixed value. Notice that, by Definition 2.17, neither $S_2\,B_j$ nor its conjugate are in $\mathbf{\Gamma}$ and, therefore, $x_j = \bot$. So y_j is already defined and equal to z_j for every $\mathbf{z} \in f_\star$, that is, $y_j = \mathtt{val}(S_2)$. Hence, $S_2\,B_j \in \mathbf{\Delta}$. □

Lemma 2.33. *If Y follows from $\mathbf{\Gamma}$ by an application of an introduction rule and $\overline{Y}$ follows from $\mathbf{\Delta}$ also by an application of an introduction rule, then $\mathbf{\Gamma}$ and $\mathbf{\Delta}$ clash.*

Proof. Left to the reader. □

Let us use the symbols $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, etc. to denote finite sequences of signed formulas. Given a sequence $\mathbf{a} = X_1, \ldots, X_n$ of signed formulas, a *subsequence* of $\mathbf{a}$ is any sequence $\mathbf{b}$ such that $\mathbf{b} = X_{a(1)}, \ldots, X_{a(k)}$, for some $k \leq n$ and strictly increasing $a : \{1, \ldots, k\} \to \{1, \ldots, n\}$. If $k < n$ we say that $\mathbf{b}$ is a *proper subsequence* of $\mathbf{a}$.

Definition 2.34. We say that a sequence $\mathbf{a}$ of signed formulas is a *non-redundant intelim proof* of X from $\mathbf{\Gamma}$ (*non-redundant intelim refutation* of $\mathbf{\Gamma}$) if $\mathbf{a}$ is an intelim proof of X from $\mathbf{\Gamma}$ (an intelim refutation of $\mathbf{\Gamma}$) and there is no proper subsequence of $\mathbf{a}$ that is an intelim proof of X from $\mathbf{\Gamma}$ (an intelim refutation of $\mathbf{\Gamma}$).

Definition 2.35. Given an intelim sequence $\mathbf{a}$ for $\mathbf{\Gamma}$, a signed formula X occurring in $\mathbf{a}$ is a *detour* in $\mathbf{a}$ if X occurs in $\mathbf{a}$ both as conclusion of an introduction and as major premiss of an elimination.

Definition 2.36. We say that an intelim proof $\mathbf{a}$ of X from $\mathbf{\Gamma}$ is *regular* if for no A, $T\,A$ and $F\,A$ both occur in $\mathbf{a}$ as premisses of a rule application.

Lemma 2.37. *For all X, all $\mathbf{\Gamma}$ and all intelim sequences $\boldsymbol{a}$ for $\mathbf{\Gamma}$,*

1. *if $\boldsymbol{a}$ is a non-redundant refutation of $\mathbf{\Gamma}$, then $\boldsymbol{a}$ contains no detours;*
2. *if $\boldsymbol{a}$ is a non-redundant regular proof of X from $\mathbf{\Gamma}$, then $\boldsymbol{a}$ contains no detours.*

Proof. To show that (1) is the case we can reason as follows. Let **a** be a non-redundant intelim refutation of **Γ**. Suppose Y is a detour in **a**. Then there are signed formulas $U_1, \ldots, U_k, V_1, \ldots, V_m, Z$ occurring in **a** such that:

1. Y occurs in **a** as the conclusion of an introduction with premisses $U_1, \ldots, U_k$
2. Z occurs in **a** as the conclusion of an elimination with Y as major premiss and $V_1, \ldots, V_m$ as minor premisses.

By Lemma 2.32, either (i) the conclusion Z of the elimination is already contained in $\{U_1, \ldots, U_k\}$, or (ii) $\{V_1, \ldots, V_m\}$ clashes with $\{U_1, \ldots, U_k\}$, that is, some minor premiss of the elimination is the conjugate of some of the premisses of the introduction and so, for some A, both $T\,A$ and $F\,A$ occur in **a** before Z. It follows that **a** is a redundant refutation of **Γ** against the assumption that **a** was non-redundant.

As for (2), let **a** be a non-redundant regular proof of X from **Γ**. By the same argument used above, it can be easily checked that if **a** contains a detour, **a** is either redundant or irregular, since it contains a pair of conjugate formulas that are both used as premisses of a rule application. This concludes the proof of the lemma. □

Definition 2.38. An *intelim proof* of X from **Γ** is *analytic* if every signed formula occurring in it is a signed subformula of X or of some signed formula in **Γ**. A *refutation* of **Γ** is *analytic* if every signed formula occurring in it is a signed subformula of some signed formula in **Γ**.

Proposition 2.39. *For all X, all **Γ** and all intelim sequences **a** for **Γ**,*

1. *if **a** is a non-redundant intelim refutation of **Γ**, then **a** is an analytic intelim refutation of **Γ**;*
2. *if **a** is a non-redundant and regular intelim proof of X from **Γ**, then **a** is an analytic intelim proof of X from of **Γ**;*

Proof. Let us define the *logical degree* of a signed formula $S\,A$ as the number of occurrences of logical operators in A. Let **a** be a non-redundant intelim refutation of **Γ**. Let **Δ** be the set of all signed formulas in **a** that are not signed subformulas of any of the signed formulas in **Γ**. Suppose **a** is non-analytic, that is, $\mathbf{\Delta} \neq \emptyset$ and let Y be an element of **Δ** of maximal logical degree. Now, Y cannot be in **Γ** and cannot be obtained from previous signed formulas by an application of an elimination rule, otherwise the major premiss of this application would also be in **Δ** and would have a strictly greater logical degree than that of Y. So, Y must result from an application of an introduction rule.

Since **a** is non-redundant, Y must be used in it, either to close the sequence or as premiss of an intelim rule. In the first case, $\overline{Y}$ (the conjugate of Y) must also occur in **a** and, by the same argument just used for Y, it can only be the conclusion of an introduction. It is not difficult to see, from the definition of introduction rule, that in such a case one of the premisses of the introduction of Y must be the conjugate of one of the premisses of the introduction of $\overline{Y}$. So, there is a closed proper subsequence of **a** that is still an intelim refutation of **Γ**, against the assumption that **a** is non-redundant. In the second case, there must be a signed formula V in **a** such that V is the conclusion of some rule application with Y as one of the premisses. From the assumption that Y has maximal logical degree among the signed formulas in **Δ**, it follows that V must be the conclusion of an elimination with Y as major premiss. For, if V were the conclusion of an introduction, V would be a signed formula in **Δ** with logical degree strictly greater than that of Y. Similarly, if V were the conclusion of an elimination with Y as minor premiss, the major premiss of this elimination would be a signed formula in **Δ** with logical degree strictly greater than that of Y. Hence, Y is at the same time the conclusion of an introduction and the major premiss of an elimination, that is, Y is a detour. Then, by Lemma 2.37(1), **a** is redundant, against the assumption. This concludes the proof of claim 1. The argument for claim 2 is essentially the same, except that: (i) **a** is a non-redundant regular intelim proof of X from **Γ**, (ii) **Δ** is the set of signed formulas in **a** that are not signed subformulas of signed formulas in $\mathbf{\Gamma} \cup \{X\}$, (iii) under the assumption that the intelim proof is regular Y can only be a detour in **a** and therefore, by Lemma 2.37(2), **a** must be redundant against the assumption that it is not. □

Let us define the *size* of a sequence **a** of signed formulas, denoted by $|\mathbf{a}|$, as the total number of occurrences of symbols in **a**.

Corollary 2.40 (*Subformula Property*)**.** *For all **Γ** and A,*

1. *If there is an intelim refutation **a** of **Γ**, then there is an analytic intelim refutation $\mathbf{a}'$ of **Γ** such that $|\mathbf{a}'| \leq |\mathbf{a}|$;*
2. *if there is a regular intelim proof **a** of X from **Γ**, then there is an analytic intelim proof $\mathbf{a}'$ for of X from **Γ** such that $|\mathbf{a}'| \leq |\mathbf{a}|$.*

Proof. The corollary follows immediately from Proposition 2.39 and from the fact that if there is an intelim refutation **a** of **Γ** (regular intelim proof of X from **Γ**), then there always exists a subsequence $\mathbf{a}'$ of **a** such that $\mathbf{a}'$ is a non-redundant intelim refutation of **Γ** (non-redundant and regular intelim proof of X from **Γ**). Clearly the size of $\mathbf{a}'$ is less than or equal to the size of **a**. □

As the above proposition suggests, there are **Γ** and X such that X is intelim deducible from **Γ** by means of an irregular intelim proof, but there is no analytic intelim proof of X from **Γ**. A simple example is the well-known proof of $T\,B$ from $T\,A$ and $T\,\neg A$:

1	$T\,A$	
2	$T\,\neg A$	
3	$F\,A$	from 2 by $T\,\neg$-$\mathscr{E}$
4	$T\,A \vee B$	from 1 by $T\,\vee$-$\mathscr{I}$1
5	$T\,B$	from 4,3 by $T\,\vee$-$\mathscr{E}$1.

However, the existence of an irregular intelim proof of X from Γ means that Γ is intelim inconsistent and, by Claim 1 of Corollary 2.40, in this case there is always an analytic intelim refutation of $\mathbf{\Gamma}$.

It is interesting to observe that, by Corollary 2.40, analytic intelim proofs or refutations are *uniformly shorter* than non-analytic ones. This property can be contrasted with the well-known fact that analytic proofs or refutations in full classical propositional logic may be exponentially longer than non-analytic ones.

Finally, we observe that Corollary 2.40 and Proposition 2.30, taken together, imply that intelim deducibility and refutability can be characterized in terms of informationally closed valuations whose domain is not the whole of $F(\mathbb{L})$, but is restricted to the relevant subformulas.

Given a set Γ of formulas, let $\mathtt{sub}(\Gamma)$ denote the set of all subformulas of the formulas in Γ.

Definition 2.41. A set Δ of $\mathbb{L}$-formulas is an $\mathbb{L}$-*domain* if $\Delta = \mathtt{sub}(\Delta)$.

A special case of an $\mathbb{L}$-domain is the set $F(\mathbb{L})$ consisting of all the formulas of the language $\mathbb{L}$.

We can then consider *local partial valuations*, namely mappings $v : \Delta \to \{0, 1, \bot\}$ where Δ is an $\mathbb{L}$-domain that is properly included in $F(\mathbb{L})$ and extend all the definitions in Section 2.2 in the obvious way. In particular:

Definition 2.42. For every $\mathbb{L}$-domain Δ, an *information state over* Δ is a mapping $v : \Delta \to \{0, 1, \bot\}$ such that every module of v is informationally closed.

Corollary 2.43. $\Gamma \models_0 A$ *iff* $v(A) = 1$ *for all information states* v *over* $\mathtt{sub}(\Gamma \cup \{A\})$ *such that* $v(B) = 1$ *for all* $B \in \Gamma$.

2.6. Tractability of 0-depth Boolean logics

Corollary 2.40 guarantees the existence of a decision procedure for the intelim logic by delimiting the space of the signed formulas that have to be taken into consideration as possible conclusions of the introduction rules. In this section we show that the decision problem for the intelim logic is tractable.

Definition 2.44. The *subformula graph* for Γ is the oriented graph $\langle V, E\rangle$ such that $V = \mathtt{sub}(\Gamma)$ and $\langle A, B\rangle \in E$ if and only if A is an immediate subformula of B.

Definition 2.45. A *G-module* is any subgraph M of G whose set of nodes is an $\mathbb{L}$-module, i.e., consists of a formula with all its immediate subformulas. The *top formula* of M is the top formula of the underlying $\mathbb{L}$-module.

Definition 2.46. A *labeled subformula graph* for Γ is a pair $\langle G, \lambda\rangle$, where G is the subformula graph for Γ and λ is a partial function, called the *labeling function*, from the vertices of G into $\{0, 1\}$.

Definition 2.47. An *intelim graph for* Γ *based on* Δ, with $\Delta \subseteq \Gamma$ is a labeled subformula graph $\langle G, \lambda\rangle$ for Γ satisfying the following conditions:

1. for every $A \in \Delta$, $\lambda(A) = 1$;
2. for every A in $\mathtt{sub}(\Gamma)$ such that $A \notin \Delta$, $\lambda(A)$ is defined and equal to 1 (or 0) *only if* there are $B_1, \ldots, B_k$ in $\mathtt{sub}(\Gamma)$ such that $T\,A$ (or $F\,A$) follows from

$$\{T\,B_i \mid 0 \leq i \leq k, \lambda(B_i) = 1\} \cup \{F\,B_i \mid 0 \leq i \leq k, \lambda(B_i) = 0\}$$

by an application of an intelim rule.

The *initial* intelim graph for Γ based on Δ is the intelim graph for Γ based on Δ such that $\lambda(A) = \bot$ for all $A \notin \Delta$.

An intelim graph for Γ is *completed* if it satisfies also the converse of (2).

Simple decision procedures for intelim refutability and deducibility are illustrated in Algorithms 2.1–2.2, and consist in building the initial intelim graph for the set of formulas that are mentioned in the specification of the problem (just the assumptions for refutation problems, the assumptions plus the conclusion for deduction problems) and turning it into a complete intelim graph by saturating it in accordance with the intelim rules. Both algorithms call the subroutine `Expand` described in Algorithm 2.3. The latter, in turn, calls the subroutine `Apply_Intelim` described in Algorithm 2.4.

The symbol "$\top$" stands for the "inconsistent labeling function". This is only a way of speaking to mean that there is no labeling function consistent with the intelim rules.

The correctness of both decision procedures follows from the fact that they return `true` if and only if the set $\{T\,A \mid \lambda(A) = 1\} \cup \{F\,A \mid \lambda(A) = 1\}$ is intelim saturated. We omit a detailed proof and just briefly discuss their complexity.

For each $x \in \{0, 1\}$, let $\mathtt{sign}(x) = T$ iff $x = 1$ and $\mathtt{sign}(x) = F$ iff $x = 0$. Recall also that, for $S \in \{T, F\}$, $\mathtt{val}(S) = 1$ iff $S = T$ and $\mathtt{val}(S) = 0$ iff $S = F$. Let $|A|$ denote the *size* of the formula A, i.e., the total number of occurrences of symbols in A. The size of a finite set Γ of formulas is defined as $\sum_{A\in\Gamma} |A|$ and denoted by $|\Gamma|$.

The input for Algorithm 2.1 is a list of all the formulas in Γ, while the input for Algorithm 2.2 is a pair consisting of a list of all the formulas in Γ and the formula A. Let n be the total size of the input, namely $O(|\Gamma|)$ for Algorithm 2.1 and $O(|\Gamma \cup \{A\}|)$ for Algorithm 2.2.

Algorithm 2.1 Decision Procedure for intelim refutability

Input: A finite set Γ of formulas;
1: build the subformula graph G for Γ;
2: set $\lambda(A) = 1$ for each $A \in \Gamma$;
3: set $\lambda(B) = \bot$ for each B such that $B \notin \Gamma$;
4: `expanded_graph` $=$ `Expand`$(\langle G, \lambda\rangle)$;
5: **if** `expanded_graph` $= \langle G, \top\rangle$, **then**
6: **return true**;
7: **else**
8: **return false**.
9: **end if**

Algorithm 2.2 Decision Procedure for intelim deducibility

Input: A finite set Γ of formulas and a formula A;
1: build the subformula graph G for $\Gamma \cup \{A\}$;
2: set $\lambda(A) = 1$ for each $A \in \Gamma$;
3: set $\lambda(B) = \bot$ for each B such that $B \notin \Gamma$;
4: `expanded_graph` $=$ `Expand`$(\langle G, \lambda\rangle)$;
5: **if** `expanded_graph` $= \langle G, \top\rangle$, **then**
6: **return true**;
7: **else if** `expanded_graph` $= \langle G, \lambda'\rangle$ and $\lambda'(A) = 1$, **then**
8: **return true**;
9: **else**
10: **return false**.
11: **end if**

Algorithm 2.3 `Expand`$(\langle G, \lambda\rangle)$

1: push all formulas A such that $\lambda(A) \neq \bot$ into `formula_stack`;
2: **while** $\lambda \neq \top$ and `formula_stack` is not empty **do**
3: pop a formula A from `formula_stack`
4: `Apply_Intelim`$(A, \langle G, \lambda\rangle)$
5: **end while**

Notice that:

1. the number of nodes in the subformula graph G (line 1 of both decision procedures) is $O(n)$;
2. the cost of building the initial labeled subformula graph $\langle G, \lambda\rangle$ is $O(n^2)$;
3. the **while** loop in the `Expand` subroutine (Algorithm 2.3) is executed at most as many times as there are nodes in G, namely $O(n)$ times; (here the key observation is that in Line 3 the formula A can be safely removed from `formula_stack`, that is each formula in `formula_stack` needs to be visited at most once;)
4. the essential cost of each run of the **while** loop consists in the cost of the `Apply_Intelim` subroutine (Algorithm 2.4);
5. for each formula A in G there are at most $O(n)$ G-modules containing A (line 1 of Algorithm 2.4);
6. the maximum number of nodes in a G-module is $a + 1$, where a is the maximum arity of a logical operator in $\mathbb{L}$;
7. the cost of each run of the **forall** loop in the `Apply_Intelim` subroutine is $O(a)$.

It follows from (1)–(7) that:

Proposition 2.48. *Whether or not A is intelim deducible from Γ (Γ is intelim refutable) can be decided in time $O(n^2)$, where $n = |\Gamma \cup \{A\}|$ ($n = |\Gamma|$).*

2.7. Expressive completeness

Given a logic $L = \langle \mathbb{L}, \models_L\rangle$, we say that an n-ary logical operator $\circ$ of $\mathbb{L}$ is *definable* in terms of other logical operators $\star_1, \ldots, \star_k$ of $\mathbb{L}$ if for every formula A containing $\circ$ there exists an equivalent formula B built up from the atomic formulas occurring in A by the only means of the logical operators $\star_1, \ldots, \star_k$. But, what do we mean here by "equivalent"? Compare the following answers:

1. if the logic admits of a standard valuation system, in the sense specified in Section 2.1, A is equivalent to B if, for every assignment ϕ, $v_\phi(A) = v_\phi(B)$;
2. A and B are equivalent whenever $A \models_L B$ and $B \models_L A$.

Algorithm 2.4 `Apply_Intelim`$(A, \langle G, \lambda\rangle)$

for all the G-modules M containing A **do**
  let B be the top formula of M;
  consider the set $\mathbf{\Delta} = \{T\ C \mid C \in M \text{ and } \lambda(C) = 1\} \cup \{F\ C \mid C \in M \text{ and } \lambda(C) = 0\}$;
  if $\mathbf{\Delta}$ contains all the premisses for an application of an introduction rule with conclusion $S\ B$ **then**
    if $\lambda(B) = |1 - \mathtt{val}(S)|$ **then**
      return $\lambda = \top$
    else if $\lambda(B) = \bot$ **then**
      set $\lambda(B) := \mathtt{val}(S)$ and push B into `formula_stack`;
    else
      do nothing;
    end if
  else
    for all $C \in M$ **do**
      if $\mathbf{\Delta}$ contains all the premisses for an application of an elimination rule with major premiss $\mathtt{sign}(\lambda(B))\ B$ and conclusion $S\ C$, **then**
        if $\lambda(C) = |1 - \mathtt{val}(S)|$ **then**
          return $\lambda = \top$
        else if $\lambda(C) = \bot$ **then**
          set $\lambda(C) := \mathtt{val}(S)$ and push C into `formula_stack`;
        else
          do nothing;
        end if
      end if
    end for
  end if
end for

Clearly, sense (2) is weaker than sense (1), for when there are more than two values it may be the case that $v_\phi(A) \in D$ if and only if $v_\phi(B) \in D$, but $v_\phi(A) \neq v_\phi(B)$ for some ϕ.

Under the consequence relation $\models_0$, no finite set of operators is sufficient to define, even in the weaker sense (2), all the others. For example, $A \vee B$ is not definable, under $\models_0$, as $\neg(\neg A \wedge \neg B)$, for $A \vee B \nvdash_{IE} \neg(\neg A \wedge \neg B)$ and $\neg(\neg A \wedge \neg B) \nvdash_{IE} A \vee B$, despite the fact that $\neg(A \vee B) \vdash_{IE} \neg A \wedge \neg B$ and $\neg A \wedge \neg B \vdash_{IE} \neg(A \vee B)$.

This situation implies that, in general, the 0-depth consequence relation defines different logics for different choices of the logical operators in the language $\mathbb{L}$, since there is no finite set of logical operators that is expressively complete.

2.8. Lack of a finite valuation system for 0-depth Boolean Logics

We now show that the modular semantics described in Section 2.2 is not equivalent to any standard finitely-valued semantics.

Proposition 2.49. *There is no finite valuation system characteristic for* $\langle \mathbb{L}, \models_0 \rangle$.

Proof. Let $\mathscr{V}$ be a valuation system such that V contains n elements. Given $n + 2$ distinct atomic formulas $p_0, \ldots, p_{n+1}$ of $\mathbb{L}$, let $\Gamma = \{p_i \to p_{i+1}\}_{0 \leq i \leq n}$ and let

$$C = \bigvee_{2 \leq j \leq n+1} \bigvee_{0 \leq i \leq j-2} p_i \to p_j.$$

For example, when $n = 3$,

$$\Gamma = \{p_0 \to p_1, p_1 \to p_2, p_2 \to p_3, p_3 \to p_4\}$$

and

$$C = (p_0 \to p_2) \vee (p_0 \to p_3) \vee (p_1 \to p_3) \vee (p_0 \to p_4) \vee (p_1 \to p_4) \vee (p_2 \to p_4).$$

Now, let ϕ be an assignments such that $v_\phi(B) \in D$ for all $B \in \Gamma$. Observe that, since there are only n distinct elements in V, then for some j, $2 \leq j \leq n + 1$, and some i, $0 \leq i \leq j - 2$, it must be the case that $\phi(p_j) = \phi(p_{i+1})$ and, therefore, $v_\phi(p_i \to p_j) = v_\phi(p_i \to p_{i+1}) \in D$. In the above example for $n = 3$, it is impossible that

$$\begin{array}{lll} \phi(p_2) \neq \phi(p_1) & & \\ \phi(p_3) \neq \phi(p_2) & \phi(p_3) \neq \phi(p_1) & \\ \phi(p_4) \neq \phi(p_3) & \phi(p_4) \neq \phi(p_2) & \phi(p_4) \neq \phi(p_1) \end{array}$$

are all simultaneously true inequalities and, therefore, one of the disjuncts in C must receive the same designated value as one of the formulas in Γ.

Hence, since $\mathscr{V}$ is characteristic for $\langle \mathbb{L}, \models_0 \rangle$ and $A \models_0 A \vee B$ for all $A, B \in \mathbb{L}$, it follows that $v_\phi(C) \in D$ and so $\Gamma \models_{\mathscr{V}} C$. However, it is not difficult to check that, by the decision procedure for intelim deducibility, $\Gamma \nvdash_{IE} C$ and therefore, by the completeness of $\vdash_{IE}$, $\Gamma \not\models_0 C$. □

2.9. 3-valued non-deterministic semantics

The negative result of the previous section, the lack of a finite valuation system for the intelim logic, does not prevent the latter, as shown in Section 2.6, from being feasibly decidable. Indeed, it turns out the this tractable logic admits of a non-standard valuation system in which the evaluation of formulas on the basis of a given assignment is *non-deterministic*.

Let V be a set of truth-values. A *non-deterministic valuation* over V is a mapping $\hat{v} : F(\mathbb{L}) \to V$ such that (i) for every atomic $p \in F(\mathbb{L})$, $\hat{v}(p) \in V$ and (ii) for every n-ary operator $\star$ and every formula $A \in F(\mathbb{L})$ with $\star$ as main operator, $\hat{v}(\star(B_1, \ldots, B_n)) \in \hat{f}_\star(\hat{v}(B_1), \ldots, \hat{v}(B_n))$ where $\hat{f}_\star$ is some function $V^n \to 2^V$.

Notice that, unlike standard valuations, non-deterministic valuations are not uniquely determined by an assignment ϕ to the atomic formulas. In general, for every such assignment ϕ we have a set $\mathbf{V}_\phi$ of non-deterministic valuations $\hat{v}$ such that, for all atomic p, $\hat{v}(p) = \phi(p)$.

Non-deterministic valuations have been intensively studied by Arnon Avron (see [1,4,2,5,3] among others) and used as a tool for investigating proof-theoretical properties of Gentzen-style sequent calculi. Crawford and Etherington [15] introduced 3-valued non-deterministic valuations as a tool for investigating tractable inference and claimed (without proof) that they provide an adequate semantics for (an extension of) unit resolution. In this section we generalize Crawford and Etherington's semantics and show that this generalization can be justified in terms of the modular semantics of Section 2.2, so building a bridge between the two approaches.

Let **3** be the set of values $\{0, 1, \bot\}$. For every n-ary operator $\star$, let $\hat{f}_\star$ be the function from $\mathbf{3}^n$ to $2^{\mathbf{3}}$ defined as follows:

$$y \in \hat{f}_\star(x_1, \ldots, x_n) \text{ iff } \langle x_1, \ldots, x_n, y \rangle \text{ is a stable element of } \mathscr{A}_\star. \tag{8}$$

Examples 2.50. The following tables describe the functions $\hat{f}_\wedge, \hat{f}_\vee, \hat{f}_\to$ and $\hat{f}_\neg$ for the usual Boolean operators:

$\wedge$	1	0	$\bot$
1	1	0	$\bot$
0	0	0	0
$\bot$	$\bot$	0	$\bot, 0$

$\vee$	1	0	$\bot$
1	1	1	1
0	1	0	$\bot$
$\bot$	1	$\bot$	$\bot, 1$

$\to$	1	0	$\bot$
1	0	0	$\bot$
0	1	1	1
$\bot$	1	$\bot$	$\bot, 1$

$\neg$	
1	0
0	1
$\bot$	$\bot$.

These are the tables introduced by Crawford and Etherington [15] whose connection with the modular semantics of the previous section is given by (8).

The following tables describe the non-deterministic functions for *exclusive or* ($\oplus$), *Peirce's arrow* ($\downarrow$), *Sheffer's stroke* ($|$) and *equivalence* ($\equiv$), built in accordance with (8):

$\oplus$	1	0	$\bot$
1	0	1	$\bot$
0	1	0	$\bot$
$\bot$	$\bot$	$\bot$	$\bot, 1, 0$

$\downarrow$	1	0	$\bot$
1	0	0	0
0	0	1	$\bot$
$\bot$	0	$\bot$	$\bot, 0$

$\mid$	1	0	$\bot$
1	0	1	$\bot$
0	1	1	1
$\bot$	$\bot$	1	$\bot, 1$

$\equiv$	1	0	$\bot$
1	1	0	$\bot$
0	0	1	$\bot$
$\bot$	$\bot$	$\bot$	$\bot, 1, 0$.

Definition 2.51. A CE-valuation is a non-deterministic valuation $\hat{v}$ over **3** satisfying the following condition, for every n-ary operator $\star$:

$$\hat{v}(\star(A_1, \ldots, A_n)) \in \hat{f}_\star(\hat{v}(A_1), \ldots, \hat{v}(A_n)).$$

It immediately follows from the above definitions that:

Proposition 2.52. *A partial valuation v for $\mathbb{L}$ is an information state if and only if v is a CE-valuation.*

Therefore:

Corollary 2.53. *For every Γ, A,*

1. *$\Gamma \models_0 A$ if, and only if, A is verified by all CE-valuations that verify all the formulas in Γ;*
2. *Γ is 0-depth inconsistent if and only if there is no CE-valuation that verifies all the formulas in Γ.*

3. Depth bounded Boolean logics

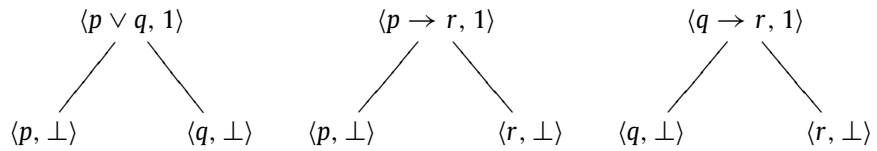
The content of Section 2 can be summarized by saying that $\models_0$ is the smallest Tarski consequence relation that is closed under the intelim rules for the logical operators of $\mathbb{L}$. In this section we use $\models_0$ as a basis for defining several approximation systems for Boolean Logic, i.e., directed sets of consequence relations that indefinitely approximate Boolean logic. Depending on the approximation system, the successors of $\models_0$ may or may not be Tarski consequence relations.

3.1. Virtual information

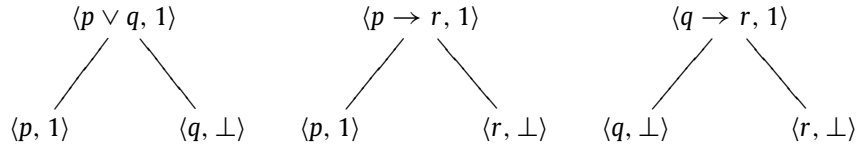
Information states are valuations in which each module is informationally closed in the sense explained in Section 2.2. This means that, each module α is closed under all the information that is uniquely *determined* by the truth-table for the main operator of α. This is the most basic mechanism for extracting implicit information from the data (expressed by a partial valuation). Any "deeper" processing of the data must introduce and use information that is not uniquely determined by it. For example, to establish that r is a Boolean consequence of $p \vee q$, $p \to r$ and $q \to r$, we start with the partial valuation v such that

$$v(A) = \begin{cases} 1 & \text{if } A \in \{p \vee q, p \to r, q \to r\} \\ \bot & \text{otherwise.} \end{cases}$$

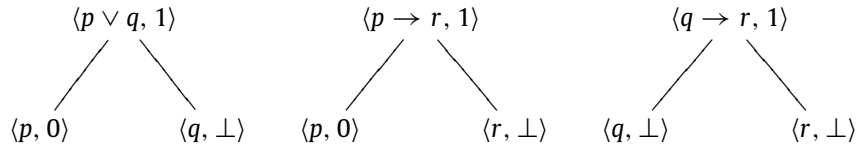
Given the subformula property of intelim deducibility (Corollary 2.40, see also Corollary 2.43), the only valuation modules of v that are significant for the deduction problem under consideration are those extensionally described by the following diagrams (the links represent the subformula relation):



These valuation modules are all informationally closed: in each of them there is no way of determining the value of some undefined formula from the information explicitly stored in the module. To extract the information that r must be true, we necessarily have to consider possible refinements of v containing information that is not implicitly determined by it. For example, we can consider its possible alternative refinements v_1 and v_2 such that $v_1(p) = 1$, $v_2(p) = 0$ and $v_1(A) = v_2(A) = v(A)$ for all $A \neq p$. Both such refinements of v contain information concerning p that is not contained in v. This is what we call *virtual information*. We may assume that either v_1 or v_2 must "eventually" obtain, that is, one of the two valuations will eventually express the information that is explicitly held by the given agent. Or, alternatively, argue that either v_1 or v_2 must agree with all "possible worlds" that agree with v. In either case, the relevant modules of v_1 and v_2 are, respectively,



and



It is easy to check that, in both cases, the least valuation (with respect to the $\sqsubseteq$ relation) in which all the relevant modules are informationally closed must verify r. So, we may wish to conclude that the piece of information that r is true is also implicitly contained in v. However, this essentially requires the consideration of refinements of v that contain virtual information, in the above example the information that p is true in v_1 and the information that p is false in v_2, namely information that is not implicitly contained in the original valuation v.

The unbounded use of virtual information, in the way just explained, turns an information state into a Boolean valuation. A natural way of approximating Boolean valuations, starting from the information states defined in Section 2.2 – which correspond to the basis case in which *no* virtual information is allowed – consists in imposing an upper bound on the depth at which the nested use of virtual information is allowed. However, as we shall see in the next section, this is by no means sufficient to define an approximation system. Because of the failure of the subformula property for the logics with depth-bounded use of virtual information, we also need to introduce further restrictions on the kind of formulas on which virtual information can be assumed (e.g., atomic formulas) and on the kind of formulas that can be taken into consideration as conclusions of the introduction rules (e.g., subformulas of the initial premisses or of the conclusion to be proven).

3.2. Search space and virtual space

A fine-grained control on the manipulation of virtual information involves a good deal of subtleties, both from the proof-theoretical and the semantical viewpoint. In this section we propose a general approach to deal with a variety of approximation strategies that differ from each other on two respects: (i) the delimitation of the *search space*, namely the set of formulas we are "interested in" as potential premisses and conclusions of our inference steps and (ii) the delimitation of the *virtual space*, namely the subset of the search space consisting of the formulas that we are allowed to use as "virtual assumptions" (up to a certain fixed depth). For each deduction problem of the form "$\Gamma \vdash A$?" the search space is determined as a function $f(\Gamma \cup \{A\})$ of the set of the formulas occurring in its specification, and the virtual space as a function $g(f(\Gamma \cup \{A\}))$ of the search space. In general, the choice of specific functions to yield suitable values of these two parameters for each particular deduction problem is the result of decisions that are conveniently made by the system designer, depending on the intended application. Such decisions affect the deductive power of each given k-depth consequence relation, and so the "speed" at which the approximation process converges to full Boolean logic. An important special case is the one in which both f and g are equal to the subformula function $\mathtt{sub}$.[12] In this case, since $\mathtt{sub}(\mathtt{sub}(\Gamma \cup \{A\})) = \mathtt{sub}(\Gamma \cup \{A\})$ both the search space and the virtual space are equal to the set $\mathtt{sub}(\Gamma \cup \{A\})$ consisting of all the subformulas of the formulas that occur in the definition of the given deduction problem. In what follows we propose a general characterization of the kind of functions that can be properly used to delimit the search space and the virtual space.

Let $\mathscr{F}$ be the set of all unary operations f on $2^{F(\mathbb{L})}$ such that, for all sets Δ of $\mathbb{L}$-formulas:

1. $\Gamma \subseteq \Delta \Longrightarrow f(\Gamma) \subseteq f(\Delta)$.
2. $f(f(\Delta)) = f(\Delta)$.

Let also

$$\mathscr{F}^u = \{f \in \mathscr{F} \mid \mathtt{sub}(\Delta) \subseteq f(\Delta)\}, \tag{9}$$

and

$$\mathscr{F}^d = \{g \in \mathscr{F} \mid g(\Delta) \subseteq \mathtt{sub}(\Delta)\}. \tag{10}$$

It immediately follows from the above definitions that $\mathtt{sub}$ is the only operation of $\mathscr{F}$ that is both in $\mathscr{F}^u$ and in $\mathscr{F}^d$. Observe also that, for every $f \in \mathscr{F}^u$, $\mathtt{sub}(f(\Delta)) \subseteq f(f(\Delta)) = f(\Delta)$ and, obviously, $f(\Delta) \subseteq \mathtt{sub}(f(\Delta))$; therefore for every $f \in \mathscr{F}^u$,

$$\mathtt{sub}(f(\Delta)) = f(\Delta) \tag{11}$$

that is, $f(\Delta)$ is an $\mathbb{L}$-domain (see above, Definition 2.41). Moreover, for every $f \in \mathscr{F}^u$ and every $g \in \mathscr{F}^d$,

$$g(f(\Delta)) \subseteq \mathtt{sub}(f(\Delta)) = f(\Delta), \tag{12}$$

that is the virtual space is always a subset of the search space.

Typical examples of operations in $\mathscr{F}^u$ are :

- the operation $\mathtt{sub}$;
- the constant operation $\mathtt{F}$ such that $\mathtt{F}(\Delta) = F(\mathbb{L})$;
- the operation $\mathtt{F}^*$ such that $\mathtt{F}^*(\Delta)$ is the set of all formulas of $\mathbb{L}$ that can be built up from the atomic formulas that occur in the formulas of Δ;
- the operation $\mathtt{c}^u$ such that $\mathtt{c}^u(\Delta)$ is the set of all $A \in \mathtt{F}^*(\Delta)$ whose logical complexity (i.e., the number of occurrences of connectives in A) is less than or equal to the logical complexity of some formula in Δ;
- the operation $\mathtt{d}^u$ such that $\mathtt{d}^u(\Delta)$ is the set of all $A \in \mathtt{F}^*(\Delta)$ whose logical depth (i.e., the number of alternations of connectives in A) is less than or equal to the logical depth of some formula in Δ.

Typical examples of operations in $\mathscr{F}^d$ are:

- the operation $\mathtt{sub}$;
- the constant operation $\varnothing$ such that $\varnothing(\Delta) = \emptyset$;
- the operation $\mathtt{at}$ such that $\mathtt{at}(\Delta)$ is the set of all atomic formulas occurring in the formulas in Δ;
- the operation $\mathtt{c}^d_k$ such that $\mathtt{c}^d_k(\Delta)$ is the subset of $\mathtt{sub}(\Delta)$ consisting of the formulas A whose logical complexity (i.e., the number of occurrences of logical operators in A) is no greater than k for some fixed k (so, $\mathtt{at} = \mathtt{c}^d_0$);
- the operation $\mathtt{d}^d_k$ such that $\mathtt{d}^d_k(\Delta)$ is the subset of $\mathtt{sub}(\Delta)$ consisting of the formulas A whose logical depth (i.e., the number of alternation of logical operators in A) is no greater than k for some fixed k.

Notice that $\mathtt{F}$ is the only constant operation in $\mathscr{F}^u$ and $\varnothing$ is the only constant operation in $\mathscr{F}^d$.

12 Recall that $\mathtt{sub}(\Delta)$ is the set of all subformulas of the formulas in Δ.

3.3. Weak depth-bounded consequence relations

The simplest kind of depth-bounded consequence relations can be straightforwardly defined by means of the simple notion of information state introduced in Definition 2.10.

Definition 3.1. Given $f \in \mathscr{F}^u$ and $g \in \mathscr{F}^d$, the *weak k-depth consequence relation of type* $\langle f, g\rangle$ is the relation $\left|\!\overset{\langle f,g\rangle}{\underset{k}{=\!=}}\right.$ in $2^{F(\mathbb{L})} \times F(\mathbb{L})$ defined as follows:

1. $\Gamma \left|\!\overset{\langle f,g\rangle}{\underset{0}{=\!=}}\right. A$ if and only if A is verified by every information state over $f(\Gamma \cup \{A\})$ that verifies all the formulas in Γ (see Definition 2.42);
2. for $k > 0$, $\Gamma \left|\!\overset{\langle f,g\rangle}{\underset{k}{=\!=}}\right. A$ if and only if there is a $B \in g(f(\Gamma \cup \{A\}))$ such that:

$$\Gamma \cup \{B\} \left|\!\overset{\langle f,g\rangle}{\underset{k-1}{=\!=}}\right. A \quad \text{and} \quad \Gamma \cup \{\neg B\} \left|\!\overset{\langle f,g\rangle}{\underset{k-1}{=\!=}}\right. A.$$

It follows from Corollary 2.43 that, for every $f \in \mathscr{F}^u$ and every $g \in \mathscr{F}^d$:

$$\Gamma \left|\!\overset{\langle f,g\rangle}{\underset{0}{=\!=}}\right. A \Longleftrightarrow \Gamma \left|\!\overset{\langle f,\varnothing\rangle}{\underset{0}{=\!=}}\right. A \Longleftrightarrow \Gamma \left|\!\overset{\langle \mathrm{F},\varnothing\rangle}{\underset{0}{=\!=}}\right. A \Longleftrightarrow \Gamma \left|\!\overset{\langle \mathtt{sub},\varnothing\rangle}{\underset{0}{=\!=}}\right. A \Longleftrightarrow \Gamma \models_0 A. \tag{13}$$

Moreover, it follows from Proposition 2.48 that:

Proposition 3.2. *For every fixed k,* $\left|\!\overset{\langle \mathtt{sub},g\rangle}{\underset{k}{=\!=}}\right.$ *can be decided in time* $O(n^{k+2})$.

Hint. For all $g \in \mathscr{F}^d$, $g(\mathtt{sub}(\Delta)) \subseteq \mathtt{sub}(\Delta)$. Moreover, the number of formulas in $\mathtt{sub}(\Delta)$ is bounded above by $|\Delta|$. □

Let us define

$$f_1 \trianglelefteq f_2 \quad \text{if and only if} \quad \forall \Delta, f_1(\Delta) \subseteq f_2(\Delta).$$

It is not difficult to show that

Proposition 3.3. *If* $\mathtt{at} \trianglelefteq g$, $\left|\!\overset{\langle \mathtt{sub},g\rangle}{\underset{k}{=\!=}}\right.$ *approaches classical propositional logic as* $k \to \infty$.

Proof Sketch. Suppose that Γ classically implies A. Let $\mathtt{at}(\Gamma \cup \{A\}) = \{p_1, \ldots, p_n\}$ and consider all the possible sets $\{\ell_1, \ldots, \ell_n\}$ of literals such that every ℓ_i, $i = 1, \ldots, n$, is equal to p_i or $\neg p_i$. There are 2^n such sets. For each of these sets consider the set $\Gamma \cup \{\ell_1, \ldots, \ell_n\}$. Since $\vdash_{IE}$ can simulate, by means of the introduction rules, each line of a classical truth-table, either $\{\ell_1, \ldots, \ell_n\} \vdash_{IE} \neg B$ for some $B \in \Gamma$ and so $\Gamma \cup \{\ell_1, \ldots, \ell_n\}$ is intelim inconsistent – in which case, by Proposition 2.30, no information state that verifies Γ can verify also $\{\ell_1, \ldots, \ell_n\}$ – or $\{\ell_1, \ldots, \ell_n\} \vdash_{IE} A$ and, therefore, $\{\ell_1, \ldots, \ell_n\} \models_0 A$, that is every information state that verifies $\{\ell_1, \ldots, \ell_n\}$ verifies also A. It follows that $\Gamma \left|\!\overset{\langle \mathtt{sub},\mathtt{at}\rangle}{\underset{n}{=\!=}}\right. A$. Hence, for sufficiently large k, $\left|\!\overset{\langle \mathtt{sub},\mathtt{at}\rangle}{\underset{k}{=\!=}}\right.$ can justify the validity of any classical inference. □

The proposition above shows that the family $\left\{ \left|\!\overset{\langle \mathtt{sub},g\rangle}{\underset{k}{=\!=}}\right. \right\}_{k\in\mathbb{N}}$ is an approximation system for Boolean Logic whenever $\mathtt{at} \trianglelefteq g$. Observe that, for a fixed k, the consequence relation $\left|\!\overset{\langle \mathtt{sub},g\rangle}{\underset{k}{=\!=}}\right.$ does not, in general, satisfy Cut and so it is not a Tarskian consequence relation.

3.4. Strong depth-bounded consequence relations

The characterization of stronger consequence relations that satisfy some form of Cut prompts for a general notion of k-depth information state.

Definition 3.4. For all $\mathbb{L}$-domains Δ and all Λ, such that $\Lambda \subseteq \Delta$, the set $\mathbf{S}\binom{\Delta}{\Lambda}_k$ is recursively defined as follows:

1. $\mathbf{S}\binom{\Delta}{\Lambda}_0 = \mathbf{S}\binom{\Delta}{\emptyset}_0$ is the set of all information states over Δ (see Definition 2.42);
2. $\mathbf{S}\binom{\Delta}{\Lambda}_{k+1}$ is the set of all information states v over Δ such that v is closed under the following condition:

$$\forall A \in \Delta, \text{if } \exists B \in \Lambda \text{ such that}$$
$$\left(\forall w \in \mathbf{S}\binom{\Delta}{\Lambda}_k\right)(v \sqsubseteq w \wedge w(B) \neq \bot \Longrightarrow w(A) = x \in \{0, 1\}),$$

then $v(A) = x$.

Elements of $\mathbf{S}\binom{\Delta}{\Lambda}_k$ are called *k-depth information states of type* $\binom{\Delta}{\Lambda}$.

$$\mathbf{S}\begin{pmatrix} F(\mathbb{L}) \\ \emptyset \end{pmatrix}_0 = \mathbf{S}\begin{pmatrix} F(\mathbb{L}) \\ \Lambda \end{pmatrix}_0 = \mathbf{S}\begin{pmatrix} F(\mathbb{L}) \\ \emptyset \end{pmatrix}_k$$

is the set of all information states discussed in Section 2.2.

Definition 3.5. For all $f \in \mathscr{F}^u$ and all $g \in \mathscr{F}^d$, the *strong k-depth consequence relation of type* $\langle f, g\rangle$ is the relation $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle}$ in $2^{F(\mathbb{L})} \times F(\mathbb{L})$ such that:

$$\Gamma \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} A \text{ iff } A \text{ is verified by all information states } v \in \mathbf{S}\binom{f(\Gamma\cup\{A\})}{g(f(\Gamma\cup\{A\}))}_k \text{ such that } v \text{ verifies all the formulas in } \Gamma. \tag{14}$$

Not all strong k-depth consequence relations $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle}$ are consequence relations in Tarski's sense, for they may or may not satisfy Cut depending on their type. A sufficient condition obtains when the operation f that delimits the search space is the operation F^* such that $\mathrm{F}^*(\Delta)$ is the set of all formulas that can be built up from the atomic formulas in $\mathtt{at}(\Delta)$:

Proposition 3.6. *For all* $g \in \mathscr{F}^d$, $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle \mathrm{F}^*,g\rangle}$ *is a Tarski consequence relation.*

Proof. Left to the reader. □

In particular, the following instances of $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle}$ are all Tarski consequence relations:

$$\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle \mathrm{F}^*,\mathrm{F}^*\rangle}, \quad \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle \mathrm{F}^*,\mathtt{at}\rangle}, \quad \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle \mathrm{F}^*,f_\Lambda\rangle},$$

where f_Λ is the operation in $\mathscr{F}^d$ such that $f_\Lambda(\Delta) = \mathtt{at}(\Delta) \cap \Lambda$ for some fixed set Λ of atomic formulas of the language. The reader can verify that all these three special cases are consequence relation that satisfy unrestricted cut. However, only the first satisfies Substitution Invariance (is a propositional logic in the sense of Section 2.1). That Proposition 3.6 does not express a necessary condition is shown by Corollary 2.43. The corollary says that $\left|\!\!\left|\!\!\frac{\quad}{0}\right.^{\langle \mathtt{sub},\emptyset\rangle}$ is equal to $\left|\!\!\left|\!\!\frac{\quad}{0}\right.^{\langle \mathrm{F},\emptyset\rangle}$, which is a Tarski consequence relation.

Although they may not satisfy unrestricted Cut, strong k-depth consequence relations always satisfy the following restricted version of it:

$$\text{For all } B \in f(\Gamma \cup \{A\}), \quad \Gamma \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} B \quad \text{and} \quad \Gamma \cup \{B\} \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} A \Longrightarrow \Gamma \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} A. \tag{Bounded Cut}$$

An important special case of Bounded Cut is the one in which $f = g = \mathtt{sub}$, and so

$$f(\Gamma \cup \{A\}) = g(f(\Gamma \cup \{A\})) = \mathtt{sub}(\Gamma \cup \{A\}),$$

in which case we speak of *Analytic Cut*.

It immediately follows from the above definitions that:

Proposition 3.7. *For all* $g, g' \in \mathscr{F}^d$, *all* $f, f' \in \mathscr{F}^u$ *and all* $k, m \in \mathbb{N}$

- *if* $g \trianglelefteq g'$ *and* $f \trianglelefteq f'$, *then* $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} \subseteq \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f',g'\rangle}$;
- *if* $k \leq m$, *then* $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle} \subseteq \left|\!\!\left|\!\!\frac{\quad}{m}\right.^{\langle f,g\rangle}$.

We shall use the notation $\left|\!\!\left|\!\!\frac{\quad}{\infty}\right.^{\langle f,g\rangle}$ to refer to the limit of the sequence of all k-depth consequence relations of type $\langle f, g\rangle$, that is, $\left|\!\!\left|\!\!\frac{\quad}{\infty}\right.^{\langle f,g\rangle} =_{def} \bigcup_{k=0}^{\infty} \left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle}$.

Remarkably, not all Tarski consequence relations $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle f,g\rangle}$ are adequate to generate a hierarchy of depth-bounded approximations to Boolean Logic.

Proposition 3.8. $\left|\!\!\left|\!\!\frac{\quad}{k}\right.^{\langle \mathrm{F},\mathrm{F}\rangle} = \left|\!\!\left|\!\!\frac{\quad}{1}\right.^{\langle \mathrm{F},\mathrm{F}\rangle} = \models_L$ *for all* $k \geq 1$.

Proof Sketch. Every $w \in \mathbf{S}\binom{F(\mathbb{L})}{F(\mathbb{L})}_1$ verifies *all* classical tautologies. It is not difficult to check, using the decision procedure for $\vdash_{IE}$, that for any axiom A of any of the standard Hilbert style axiomatic systems for classical propositional logic, $\neg A$ is intelim inconsistent, and so there is no 0-depth information state that falsifies A. It then follows that $\left|\!\!\left|\!\!\frac{\quad}{1}\right.^{\langle \mathrm{F},\mathrm{F}\rangle} A$ for every such axiom A. Moreover, $\left|\!\!\left|\!\!\frac{\quad}{1}\right.^{\langle \mathrm{F},\mathrm{F}\rangle}$ is closed under Modus Ponens (because $\models_0$ is) and, therefore, every classical tautology is a tautology also for $\left|\!\!\left|\!\!\frac{\quad}{1}\right.^{\langle \mathrm{F},\mathrm{F}\rangle}$. □

3.5. Depth-bounded intelim proofs

Given a signed formula X, let $\mathtt{Uns}(X)$ denote the unsigned formula A such that $X = S\,A, S \in \{T, F\}$. Moreover, given a set $\mathbf{\Gamma}$ of signed formulas, let $\mathtt{Uns}(\mathbf{\Gamma}) = \{\mathtt{Uns}(X) \mid X \in \mathbf{\Gamma}\}$.

The deducibility relation $\vdash^{\langle f,g\rangle}_{k}$ that corresponds to the weak k-depth consequence relation $\models^{\langle f,g\rangle}_{k}$ can be defined in a straightforward way, starting from $\vdash_{IE}$ (see Definition 2.22 and recall that, by Corollary 2.40, $\vdash_{IE}$ enjoys the subformula property):

Definition 3.9. For all $f \in \mathscr{F}^u$ and all $g \in \mathscr{F}^d$,

1. $\mathbf{\Gamma} \vdash^{\langle f,g\rangle}_{0} Y$ iff $\mathbf{\Gamma} \vdash_{IE} Y$;
2. $\mathbf{\Gamma} \vdash^{\langle f,g\rangle}_{k+1} Y$ if and only if there is $B \in g(f(\mathtt{Uns}(\mathbf{\Gamma}) \cup \{\mathtt{Uns}(Y)\})$, such that $\mathbf{\Gamma} \cup \{T\,B\} \vdash^{\langle f,g\rangle}_{k} Y$ and $\mathbf{\Gamma} \cup \{F\,B\} \vdash^{\langle f,g\rangle}_{k} Y$.

This is equivalent to augmenting the intelim rules with the following depth-increasing *proof rule*:

$$\frac{\mathbf{\Gamma} \cup \{T\,B\} \vdash^{\langle f,g\rangle}_{j} X \qquad \mathbf{\Gamma} \cup \{F\,B\} \vdash^{\langle f,g\rangle}_{j} X}{\mathbf{\Gamma} \vdash^{\langle f,g\rangle}_{j+1} X}.$$

Observe that, by the monotonicity of each $\vdash^{\langle f,g\rangle}_{k}$, $\vdash^{\langle f,g\rangle}_{k} \subseteq \vdash^{\langle f,g\rangle}_{k+1}$ for all $k \geq 0$.

Proofs of $\mathbf{\Gamma} \vdash^{\langle f,g\rangle}_{k} X$ can be represented in the format of a tree whose branches are intelim sequences and virtual information is introduced by a single branching rule of the form:

For all $B \in g(f(\mathtt{Uns}(\mathbf{\Gamma}) \cup \{\mathtt{Uns}(X)\}))$,

$$\frac{}{T\,B \mid F\,B}. \tag{PB}$$

When proofs are presented in this format, $\mathbf{\Gamma} \vdash^{\langle f,g\rangle}_{k} X$ if and only if there is a tree $\mathscr{T}$ such that (i) each branch of $\mathscr{T}$ contains at most k virtual assumptions introduced via the branching rule PB (and so $\mathscr{T}$ contains at most 2^k branches), (ii) each branch is an intelim sequence for $\mathbf{\Gamma} \cup \mathbf{\Delta}$, where $\mathbf{\Delta}$ is the set of all the virtual assumptions on that branch; and (iii) X occurs in all *open* branches (namely, those that do not contain $T\,A$ and $F\,A$ for any sentence A).[13]

We now discuss the deducibility relations that correspond to the strong k-depth consequence relations (of each given type).

Definition 3.10. For all $f \in \mathscr{F}^u$ and all $g \in \mathscr{F}^d$, the relation $\Vdash^{\langle f,g\rangle}_{k}$ is recursively defined as follows:

1. $\mathbf{\Gamma} \Vdash^{\langle f,g\rangle}_{0} Y$ iff $\mathbf{\Gamma} \vdash_{IE} Y$;
2. $\mathbf{\Gamma} \Vdash^{\langle f,g\rangle}_{k+1} Y$ if and only if there is a finite sequence $X_1, \ldots, X_n$ of signed formulas such that $X_n = Y$ and, for every X_i, $1 \leq i \leq n$,
   - $\mathtt{Uns}(X_i) \in f(\mathtt{Uns}(\mathbf{\Gamma} \cup \{Y\}))$ and
   - $\exists B \in g(f(\mathtt{Uns}(\mathbf{\Gamma} \cup \{Y\})))$ such that

$$X_1, \ldots, X_{i-1}, T\,B \Vdash^{\langle f,g\rangle}_{k} X_i \quad \text{and} \quad X_1, \ldots, X_{i-1}, F\,B \Vdash^{\langle f,g\rangle}_{k} X_i.$$

The sequence $X_1, \ldots, X_n$ is called a *k-depth proof of Y from $\mathbf{\Gamma}$ of type $\langle f, g\rangle$*.

We shall use the notation $\Vdash^{\langle f,g\rangle}_{\infty}$ to refer to the limit of the sequence of all k-depth deducibility relations of type $\langle f, g\rangle$, that is, $\Vdash^{\langle f,g\rangle}_{\infty} =_{def} \bigcup_{k=0}^{\infty} \Vdash^{\langle f,g\rangle}_{k}$. The proof of soundness and completeness of $\Vdash^{\langle f,g\rangle}_{k}$ with respect to $\models^{\langle f,g\rangle}_{k}$ is routine and can be omitted:

Proposition 3.11. *$\Gamma \Vdash^{\langle f,g\rangle}_{k} A$ iff $\Gamma \models^{\langle f,g\rangle}_{k} A$.*

Proposition 3.12. *For every $k \in \mathbb{N}$, if $\Gamma \Vdash^{\langle \mathtt{F},\mathtt{F}\rangle}_{k} A$, then there exists $m \in \mathbb{N}$ such that $m \geq k$ and $\Gamma \Vdash^{\langle \mathtt{sub},\mathtt{at}\rangle}_{m} A$.*

---

13 For unbounded k, this is a proof system for classical propositional logic that enjoys the subformula property. However, by removing the restriction on B in the rule PB this presentation of classical logic allows also for representing proofs that do not have the subformula property. On the connection between the rule PB and the cut rule of Gentzen's sequent calculus, as well as on the advantages of cut-based formalizations of classical logic, see [16,17,24,18].

Proof Reference. A proof can be adapted from [19]. This paper proves a normalization theorem for a natural deduction system for classical propositional logics that corresponds to $\left|\!\left|\!\frac{\langle \mathtt{F},\mathtt{F}\rangle}{\infty}\right.\right.$, except that unsigned formulas are used instead of signed formulas. The normalization theorem immediately implies that every k-depth proof of type $\langle \mathtt{F}, \mathtt{F}\rangle$ can be transformed, for some $m \geq k$, into an m-depth proof of type $\langle \mathtt{sub}, \mathtt{at}\rangle$ of the same conclusion from the same assumptions. □

It immediately follows from Propositions 3.7 and 3.12, that:

Corollary 3.13. $\left|\!\left|\!\frac{\langle \mathtt{F},\mathtt{F}\rangle}{\infty}\right.\right. = \left|\!\left|\!\frac{\langle \mathtt{F},\mathtt{at}\rangle}{\infty}\right.\right. = \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{at}\rangle}{\infty}\right.\right.$.

3.6. Analytic vs non-analytic consequence relations

Let us consider the depth-bounded consequence relation $\left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right.$. Like any strong depth-bounded consequence relation, it satisfies Bounded Cut, which in this special case corresponds to:

$$\text{for all } A \in \mathtt{sub}(\Gamma \cup \{B\}), \quad \Gamma \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. A \quad \text{and} \quad \Gamma \cup \{A\} \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. B \Longrightarrow \Gamma \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. B. \qquad \text{(Analytic Cut)}$$

Moreover, it is easily seen that $\left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right.$ satisfies also Substitution Invariance. A decision procedure for $\Gamma \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. A$ can

Algorithm 3.1 `Depth-Expand`$(k, \langle G, \lambda\rangle)$

```
1: if k = 0, then
2:   Expand(⟨G, λ⟩);
3: else
4:   push all formulas A such that λ(A) = ⊥ into undefined_formulas;
5:   while λ ≠ ⊤ and undefined_formulas is not empty do
6:     pop a formula A from undefined_formulas;
7:     let λ1(A) = 1 and λ1(B) = λ(B) for all B ≠ A;
8:     let λ2(A) = 0 and λ2(B) = λ(B) for all B ≠ A;
9:     let G1 = Depth-Expand(k − 1, ⟨G, λ1⟩);
10:    let G2 = Depth-Expand(k − 1, ⟨G, λ2⟩);
11:    if λ1 = ⊤, then
12:      set λ = λ2;
13:      remove from undefined_formulas the formulas A such that λ(A) ≠ ⊥;
14:    else
15:      if λ2 = ⊤, then
16:        set λ = λ1,
17:        remove from undefined_formulas the formulas A such that λ(A) ≠ ⊥;
18:      else
19:        set λ(A) = x for all A such that λ1(A) = λ2(A)
20:        remove from undefined_formulas the formulas A such that λ(A) ≠ ⊥;
21:      end if
22:    end if
23:  end while
24: end if
```

be obtained from that for $\Gamma \vdash_{IE} A$ discussed in Section 2.6 by generalizing the `Expand` subroutine (Algorithm 2.3) into the one illustrated in Algorithm 3.1. A simple analysis shows that:

Proposition 3.14. *The complexity of the decision of* $\Gamma \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. A$ *is* $O(n^{2k+2})$, *where* $n = |\Gamma \cup \{A\}|$.

Hence, the time required by the decision of $\Gamma \left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right. A$ is bounded above by a polynomial in the size of $\Gamma \cup \{A\}$. It follows from Proposition 3.7, 3.8 and 3.11 and Corollary 3.13 that the limit $\left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{\infty}\right.\right.$ of the sequence of k-depth analytic consequence relations is equivalent to full Boolean logic. Thus, the hierarchy of the $\left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right.$ provides an approximation system to full Boolean Logic whose semantic counterpart is given by the depth-bounded consequence relation $\left|\!\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right.\right.$.

Algorithm 3.1 can be adapted to show tractability whenever the size of the search space is bounded above by a polynomial in the size of the input problem. Let $\mathscr{PF}^u$ be the set of all operations in $\mathscr{F}^u$ such that, for some fixed polynomial p, $|f(\Delta)| \leq p(|\Delta|)$ for all Δ.

Proposition 3.15. *For every* $f \in \mathscr{PF}^u$ *the relation* $\left|\!\left|\!\frac{\langle f,\mathtt{sub}\rangle}{k}\right.\right.$ *is tractable.*

1	$T\, p \to (p \to A)$	
2	$T\, \neg p \to (\neg p \to B)$	
3	$T\, q \to (B \to C)$	
4	$T\, \neg q \to (A \to C)$	
5	$T\, q \to (q \to \neg A)$	
6	$T\, \neg q \to (\neg q \to \neg B)$	
7	$T\, p$	$F\, p$
8	$T\, p \to A$	$T\, \neg p$
9	$T\, A$	$T\, \neg p \to B$
10	$T\, A \vee B$	TB
		$T\, A \vee B$
12	$T\, A \vee B$	
13	$T\, q$	$F\, q$
14	$T\, B \to C$	$T\, \neg q$
15	$T\, q \to \neg A$	$T\, A \to C$
16	$T\, \neg A$	$T\, \neg q \to \neg B$
17	$F\, A$	$T\, \neg B$
18	$T\, B$	$F\, B$
19	$T\, C$	$T\, A$
		$T\, C$
21	$T\, C$	

Fig. 1. A non-analytic proof that cannot be transformed into an analytic one without raising the depth of the proof.

We can now ask ourselves whether or not $\left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right|$ is a Tarskian consequence relation. This boils down to the question whether the deducibility relation $\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right|$ enjoys the subformula property, i.e., whether or not $\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right| = \left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{k}\right|$.

We have shown that

$$\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{0}\right| = \left|\!\frac{\langle \mathtt{F},\varnothing\rangle}{0}\right| = \vdash_{IE},$$

enjoys the subformula property, i.e., that

$$\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{0}\right| = \left|\!\frac{\langle \mathtt{F},\varnothing\rangle}{0}\right| = \left|\!\frac{\langle \mathtt{sub},\varnothing\rangle}{0}\right|,$$

and it follows from Proposition 3.12 that so does the limit

$$\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{\infty}\right| = \bigcup_{k=0}^{\infty} \left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right|$$

of the sequence of k-depth deducibility relations of type $\langle \mathtt{F}, \mathtt{sub}\rangle$. However, for k finite and greater than 0, the relation $\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right|$ *does not* enjoy the subformula property. A counterexample is illustrated in Fig. 1 showing a non-analytic 1-depth proof of C from assumptions 1–6 that cannot be transformed into an analytic one without raising the depth of the proof. The proof is not analytic, because $A \vee B$ is not a subformula either of the assumptions or of the conclusion. Moreover, it can be verified that there is no analytic 1-depth proof. This counterexample shows that $\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{1}\right| \neq \left|\!\frac{\langle \mathtt{sub},\mathtt{sub}\rangle}{1}\right|$, which raises the problem of the tractability and decidability of Tarskian depth-bounded consequence relations, suggesting that they may not be suitable to define approximation systems, even when the use of virtual information is restricted to subformulas of the assumptions or of the conclusion. Despite the failure of the subformula property, we conjecture that the Tarskian consequence relation $\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right|$ is indeed tractable because the size of the search space can be polynomially bounded without loss of deductive power:

Conjecture 3.16. *There exists an operation $f \in \mathscr{PF}^u$ such that*

$$\left|\!\frac{\langle \mathtt{F},\mathtt{sub}\rangle}{k}\right| = \left|\!\frac{\langle f,\mathtt{sub}\rangle}{k}\right|.$$

If true, the above conjecture would imply, via Proposition 3.15 that $\left|\!\left|\!\frac{\langle \mathrm{F},\mathtt{sub}\rangle}{k}\right.$ is a tractable Tarskian consequence relation.

4. Related work and concluding remarks

In this final section we briefly discuss a cluster of ideas and methods that are most closely related to the semantic and proof-theoretical framework presented here.

4.1. KE and KI

Let us consider a propositional language $\mathbb{L}$ consisting only of the four standard Boolean operators $\wedge$, $\vee$, $\neg$, $\rightarrow$. We may restrict our attention to the intelim sequences that contain only applications of the elimination rules of Table 1, that we may call "E-sequences". An *eliminative refutation* of Γ is a closed elimination sequence for $\{T\ B \mid B \in \Gamma\}$.

On the other hand, we may also restrict our attention to the intelim sequences that contain only applications of the introduction rules of Γ, that we may call "I-sequences". An *introductory proof* of A from Γ is an introductory sequence for $\{T\ B \mid B \in \Gamma\}$ that contains $T\ A$.

Each of these two restricted notions is turned into a complete system for classical propositional logic by adding a single structural rule, corresponding to the classical *Principle of Bivalence*, that allows us to split a sequence into two branches that are developed in parallel:

$$\frac{}{\mathrm{T}A \mid \mathrm{F}A} \qquad \text{(PB)}$$

A *KE-tree* for a set $\mathbf{\Gamma}$ of a signed formulas is a tree whose nodes results from applications of the *elimination rules* of Table 1, starting from the signed formulas in $\mathbf{\Gamma}$. A *KE-refutation* of a set Γ of formulas is a closed KE-tree for $\{T\ B \mid B \in \Gamma\}$, namely one in which all branches are *closed*, i.e., contain a pair of conjugate signed formulas. KE is a complete refutation system for classical propositional logic and remains complete if the applications of the branching rule PB are restricted to subformulas of the formulas in the initial set $\mathbf{\Gamma}$.

A *KI-tree* for a set $\mathbf{\Gamma}$ of a signed formulas is a tree whose nodes results from applications of the *introduction rules* of Table 1, starting from the signed formulas in $\mathbf{\Gamma}$. A *KI-proof* of a A from Γ is a KI-tree for $\{T\ B \mid B \in \Gamma\}$ such that $T\ A$ occurs in all *open* branches. KI is a complete proof system for classical propositional logic and remains complete if the applications of the branching rule PB are restricted to atomic formulas occurring either in the assumptions or in the conclusion. Moreover, the applications of the introduction rules can be restricted, without loss of completeness, to subformulas of the assumptions or of the conclusion.

The system KE and KI were introduced by Marco Mondadori [41–43] and their relative complexity was later investigated in [16,17,24,45,18], showing that both methods dominate Smullyan's analytic tableaux in terms of the p-simulation relation [13].

It has been shown [44,19,22] that combining KE and KI leads to a sort of "natural deduction" system in which the inferential behavior of the logical operators is regulated by introduction and elimination rules that mirror their classical meaning – while Gentzen's original rules are better seen as an explication of their intuitionistic meaning – and satisfy Prawitz's inversion principle. These operational rules require no manipulation of virtual information (no "discharge" of provisional assumptions), the latter being governed only by the structural rule PB. Normalization of proofs and subformula property for this system of natural deduction are shown in [19]. Given the presence of the introduction rules, this natural deduction system can be used, indifferently, as a refutation method or as a proof method for classical propositional logic.

In all these tree methods, the operational rules are not branching and the only branching rule is the structural rule PB. It is therefore quite natural to investigate the subsystems that result from limiting the applications of this single structural rule. A suggestion in this sense for the KE system can be found in [24, Prop. 5.6]; see also [18, Prop. 46]. KE with restricted bivalence has been more thoroughly investigated in [29–31]. A similar idea for the natural deduction system that results from combining KE and KI is discussed in [19]. In general, the depth-bounded consequence relations that can be characterized simply by limiting the nested applications of the PB-rule in the combined KE/KI system are the *weak* k-depth consequence relations discussed in Section 3.2. The relationship between KE, KI and Gentzen-style natural deduction has been recently discussed at length in [39] with special attention to the complexity of proof-search methods.

4.2. Stålmarck's method

Stålmarck's method is a patented algorithm for tautology checking that was first described in [49]. The method has proved to be quite useful for several industrial and civil applications involving the efficient verification of critical properties that are typically expressed by means of (complex) Boolean formulas. The main heuristic principles underlying this algorithm are very similar to those underlying the KE/KI approach: perform Boolean inference by means of non-branching operational rules augmented by a single branching rule (closely related to the PB rule of KE and KI), and delay the application of the

latter as long as possible. The two research programmes developed independently and in parallel, addressing different communities. Perhaps their connection was partially obscured by the fact that, in its original presentation, Stålmarck's algorithm applied to formulas in implication normal form and was based on rewrite rules whose format was apparently very distant from that of typical inference rules in more conventional proof-theoretical frameworks, such as analytic tableaux or natural deduction. The gap has been bridged in [48,8], where the connection with KE and KI, as well as the contrast with Gentzen style calculi, are clearly and explicitly stated. (See also [37, Section 5.4] on the connection between KE/KI and Stålmarck's method).

In essence, the proof-theoretical backbone of Stålmarck's method can be described as a natural deduction system – like the one described in [19,23][14] and generalized in Section 2.3 of this paper – combining the elimination rules of KE, the introduction rules of KI and a variant of PB called "dilemma" (see [48, Section 3.7] [8, pp. 17–20]). Like in KE/KI the latter is the only branching rule and the only one that involves the introduction and subsequent discharge of (complementary) virtual assumptions. Stålmarck's method is a refutation algorithm exploiting the fact that the unbounded system – where the number of nested applications of PB is not limited – enjoys the subformula property (see [19] for a procedure to turn arbitrary proofs into ones that enjoy the subformula property), so that the formulas occurring in the conclusion of the introduction rules and in the PB rule can be restricted to (weak) subformulas of the initial assumptions or of the conclusion. However, the algorithm adds to this proof-theoretical hardcore two kinds of enhancements: *branch merging*, that allows to unify adjacent branches by deleting all the formulas that do not occur in both of them, and *formula relations*, recording the fact that two formulas must have the same (unknown) truth value.

If we leave formula relations aside, Stålmarck's algorithm for 0-depth saturation is a procedure for recognizing the intelim inconsistency, in the sense of Definition 2.23, of formulas in implication normal form. It follows from Corollary 2.40 that allowing for unrestricted applications of the introduction rules does not generate any new valid inference and so, given Proposition 2.30, the algorithm is sound and complete for 0-depth inconsistency, defined in terms of the Tarskian consequence relation $\models_0$, which is characterized by the modular semantics of Section 2.2 or, alternatively, by the 3-valued non-deterministic semantics of Section 2.9. Proposition 2.49 also applies, to the effect that there is no finite standard valuation system characteristic for Stålmarck's 0-depth procedure (the proof can be adapted for the case of formulas in implication normal form). Stålmarck's k-depth algorithm corresponds to k-depth inconsistency defined in terms of the relations $\|\!\!\frac{\langle \mathrm{sub},\mathrm{sub}\rangle}{k}$ and $\|\!\!\frac{\langle \mathrm{sub},\mathrm{sub}\rangle}{k}$ of Section 3.6. As shown by the counterexample in Fig. 1, for $k > 0$, $\|\!\!\frac{\langle \mathrm{sub},\mathrm{sub}\rangle}{k}$ is not equivalent to $\|\!\!\frac{\langle \mathrm{F},\mathrm{sub}\rangle}{k}$ and so, unlike the 0-depth case, restricting the conclusions of the introduction rules to subformulas of the initial formulas involves some loss of inferential power. (Recall also that for any $k > 0$, $\|\!\!\frac{\langle \mathrm{F},\mathrm{F}\rangle}{k}$ is full Boolean Logic, see Proposition 3.8.)

With the addition of formula relations the situation is less transparent. Although these enhancements may significantly improve the efficiency of the k-depth saturation procedure, for each fixed k, they seem not to fit the "harmonic" pattern of introductions and eliminations that is exhibited by the "pure" rules (see Lemma 2.32 in this respect), which is likely to affect the very possibility of a well-behaved theory of meaning for the logical operators. So, the semantic characterizations given in this paper do not appear to be easily adaptable to provide similar characterizations for Stålmarck's k-depth saturation. From a theoretical viewpoint, it seems more convenient to investigate the systems based on the pure rules and treat other enhancements as shortcuts for refutations that are more aptly presented in a higher-depth system. In any case, we believe that the proof-theoretical and semantic framework presented here can be useful also to further investigate the properties of Stålmarck's procedures.

References

[1] A. Avron, Natural 3-valued logics. characterization and proof theory, The Journal of Symbolic Logic 56 (1) (1991) 276–294.
[2] A. Avron, Non-deterministic matrices and modular semantics of rules, in: J-Yve Beziau (Ed.), Logica Universalis, Birkäuser, 2005, pp. 155–174.
[3] A. Avron, A non-deterministic view of non-classical negation, Studia Logica 80 (2/3) (2005).
[4] A. Avron, I. Lev, Canonical propositional Gentzen-type systems, in: R. Gore, A. Leitsch, T. Nipkov (Eds.), Automated Reasoning: First Joint International Conference (IJCAR), in: Lecture Notes in Artificial Intelligence, Springer, 2001, pp. 529–544.
[5] A. Avron, A. Zamansky, Cut elimination and quantification in canonical systems, Studia Logica 82 (2006) 157–176.
[6] N.D. Belnap Jr., How a computer should think, in: G. Ryle (Ed.), Contemporary Aspects of Philosophy, Oriel Press, 1976, pp. 30–55.
[7] N.D. Belnap Jr., A useful four-valued logic, in: J.M. Dunn, G. Epstein (Eds.), Modern uses of multiple-valued logics, Reidel, Dordrecht, 1977, pp. 8–37.
[8] M. Björk, Stålmarkc's method for automated theorem proving in first order logic. Ph.D. Thesis, Göteborg University, 2003.
[9] M. Björk, A first-order extension of Stålmarck's method, in: G. Sutcliffe, A. Voronkov (Eds.), LPAR, in: LNAI, vol. 3835, Springer, 2005, pp. 276–291.
[10] M. Björk, First-order Stålmarck. Universal lemmas through branch merges, Journal of Automated Reasoning 42 (2009) 99–122.
[11] S. Blamey, Partial logic, in: D.M. Gabbay, F. Guenthner (Eds.), Handbook of Philosophical Logic, vol. 3, Kluwer, 1986, pp. 1–70. Republished in the 2nd edition, volume 5, Kluwer, Dordrecht, 2002.
[12] M. Cadoli, M. Schaerf, Approximate reasoning and non-omniscient agents, in: TARK'92: Proceedings of the 4th Conference on Theoretical Aspects of Reasoning about Knowledge, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc, 1992, pp. 169–183.
[13] S.A. Cook, R. Reckhow, On the length of proofs in the propositional calculus, in: Proceedings of the 6th Annual Symposium on the Theory of Computing, 1974, pp. 135–148.
[14] S.A. Cook, The complexity of theorem-proving procedures, in: STOC'71: Proceedings of the Third Annual ACM Symposium on Theory of Computing, ACM Press, New York, NY, USA, 1971, pp. 151–158.

14 In these papers the authors were not aware of the exposition in [48].

[15] J.M. Crawford, D.W. Etherington, A non-deterministic semantics for tractable inference, in: AAAI/IAAI, 1998, pp. 286–291.
[16] M. D'Agostino, Investigations into the complexity of some propositional calculi, PRG Technical Monographs, vol. 88 Oxford University Computing Laboratory 1990.
[17] M. D'Agostino, Are tableaux an improvement on truth tables? Cut-free proofs and bivalence, Journal of Logic, Language and Information 1 (1992) 235–252.
[18] M. D'Agostino, Tableau methods for classical propositional logic, in: M. D'Agostino, D.M. Gabbay, R. Hähnle, J. Posegga (Eds.), Handbook of Tableaux Methods, Kluwer Academic Publishers, 1999, pp. 45–123.
[19] M. D'Agostino, Classical natural deduction, in: S.N. Artëmov, H. Barringer, A.S. d'Avila Garcez, L.C. Lamb, J. Woods (Eds.), We Will Show Them! (1), College Publications, 2005, pp. 429–468.
[20] M. D'Agostino, Tractable depth-bounded logics and the problem of logical omniscience, in: F. Montagna, H. Hosni (Eds.), Probability, Uncertainty and Rationality, in: CRM series, Springer, 2010, pp. 245–275.
[21] M. D'Agostino, Analytic inference and the informational meaning of the logical operators. Logique et Analyse, 2013, (in press).
[22] M. D'Agostino, K. Broda, M. Mondadori, A solution to a problem of Popper, in: M. Alai, G. Tarozzi (Eds.), Popper Philosopher of Science, Rubbettino, 2006, pp. 147–168.
[23] M. D'Agostino, L. Floridi, The enduring scandal of deduction. Is propositionally logic really uninformative? Synthese 167 (2009) 271–315.
[24] M. D'Agostino, M. Mondadori, The taming of the cut, Journal of Logic and Computation 4 (1994) 285–319.
[25] M. Dalal, Anytime families of tractable propositional reasoners, in: Proceedings of the Fourth International Symposium on AI and Mathematics, AI/MATH-96, 1996, pp. 42–45.
[26] M. Dalal, Anytime families of tractable propositional reasoners, Annals of Mathematics and Artificial Intelligence 22 (1998) 297–318.
[27] M. Dummett, Elements of Intuitionism, Clarendon Press, Oxford, 1977.
[28] R. Epstein, Propositional Logics. The semantic Foundations of Logic, 3rd ed., Advanced Reasoning Forum, 2012.
[29] M. Finger, Polynomial approximations of full propositional logic via limited bivalence, in: 9th European Conference on Logics in Artificial Intelligence, JELIA 2004, in: Lecture Notes in Artificial Intelligence, vol. 3229, Springer, 2004, pp. 526–538.
[30] M. Finger, Towards polynomial approximations of full proposi- tional logic, in: A.L.C. Bazzan, S. Labidi (Eds.), XVII Brazilian Symposium on Artificial Intel ligence (SBIA 2004), in: Lecture Notes in Artificial Intel lingence, vol. 3171, Springer, 2004, pp. 11–20.
[31] M. Finger, D.M. Gabbay, Cut and pay, Journal of Logic, Language and Information 15 (3) (2006) 195–218.
[32] M. Finger, R. Wassermann, Approximate and limited reasoning: semantics, proof theory, expressivity and control, Journal of Logic and Computation 14 (2) (2004) 179–204.
[33] M. Finger, R. Wassermann, The universe of propositional approximations, Theoretical Computer Science 355 (2) (2006) 153–166.
[34] D.M. Gabbay, J. Woods, The new logic, Logic Journal of the IGPL 9 (2) (2001) 141–174.
[35] G. Gentzen, Unstersuchungen über das logische Schliessen, Mathematische Zeitschrift 39 (1935) 176–210. English translation in [50].
[36] K. Gödel, Zum intuitionistischen Aussagenkalkül, Anzeiger Akademie der Wissenschaften Wien (Math.-naturwiss. Klasse) 69 (1932) 65–66.
[37] R. Hähnle, Tableaux and related methods, in: A. Robinson, A. Voronkov (Eds.), Handbook of Automated Reasoning, vol. 1, North Holland, Amsterdam, 2001, pp. 101–178. (chapter 3).
[38] J. Hintikka, Logic, Language Games and Information. Kantian Themes in the Philosophy of Logic, Clarendon Press, Oxford, 1973.
[39] A. Indrzejczak, Natural Deduction, Hybrid Systems and Modal Logics, Springer Verlag, 2010.
[40] S.C. Kleene, Introduction to Metamatehmatics, North-Holland, Amsterdam, 1952.
[41] M. Mondadori, Classical analytical deduction. Annali dell'Università di Ferrara; Sez. III, Discussion paper 1, Università di Ferrara, 1988.
[42] M. Mondadori, Classical analytical deduction, part II. Annali dell'Università di Ferrara; Sez. III, Discussion paper 5, Università di Ferrara, 1988.
[43] M. Mondadori, On the notion of a classical proof, in: Temi e prospettive della logica e della filosofia della scienza contemporanee, vol. I, CLUEB, Bologna, 1988, pp. 211–224.
[44] M. Mondadori, An improvement of Jeffrey's deductive trees. Annali dell'Università di Ferrara; Sez. III. Discussion paper 7, Università di Ferrara, 1989.
[45] M. Mondadori, Efficient inverse tableaux, Logic Journal of the IGPL 3 (6) (1995) 939–953.
[46] E. Moriconi, L. Tesconi, On inversion principles, History and Philosophy of Logic 29 (2) (2008) 103–113.
[47] S. Sequoiah-Grayson, The scandal of deduction. Hintikka on the information yield of deductive inferences, The Journal of Philosophical Logic 37 (1) (2008) 67–94.
[48] M. Sheeran, G. Stålmarck, A tutorial on Stalmarck's proof procedure for propositional logic, Formal Methods in System Design 16 (2000) 23–58.
[49] G. Stålmarck, A system for determining propositional logic theorems by applying values and rules to triplets that are generated from a formula. Swedish Patent No. 467 076 (approved 1992), U.S. Patent No. 5 276 907 (1994), European Patent No. 0403 454 (1995), 1992.
[50] M. Szabo (Ed.), The Collected Papers of Gerhard Gentzen, North-Holland, Amsterdam, 1969.