Classic-Like Analytic Tableaux for Finite-Valued Logics

Carlos Caleiro[1] and João Marcos[2]

[1] Instituto de Telecomunicações and Dept. Mathematics, IST, TU-Lisbon, Portugal
[2] Department of Informatics and Applied Mathematics, UFRN, Brazil

Abstract. The paper provides a recipe for adequately representing a very inclusive class of finite-valued logics by way of tableaux. The only requisite for applying the method is that the object logic received as input should be sufficiently expressive, in having the appropriate linguistic resources that allow for a bivalent representation. For each logic, the tableau system obtained as output has some attractive features: exactly two signs are used as labels in the rules, as in the case of classical logic, providing thus a uniform framework in which different logics can be represented and compared; the application of the rules is analytic, in that it always reduces complexity, providing thus an immediate proof-theoretical decision procedure together with a counter-model builder for the given logic.

Keywords: many-valued logics, proof theory.

1 Background

The fact that any abstract consequence relation may be represented by way of an adequate many-valued semantics (cf. [16]) makes many-valued logics ubiquitous in the realm of inference systems. Moreover, the compositional feature that characterizes truth-functional semantics makes the latter extremely attractive for computational or linguistic purposes. This much from a purely semantical perspective. From a proof-theoretical perspective, on the other hand, the existence of appropriate and efficient deductive formalisms and theorem-proving frameworks for truth-functional logics provides many-valued logics with useful tools for automating their variegated approaches to entailment and for developing deep computational insights into their underlying reasoning mechanisms.

General procedures for providing arbitrary finite-valued logics with adequate tableau systems are known since long in the literature (cf. [3,8]). Reasonably up-to-date implementation-oriented accounts of such axiom-extraction strategies can be found in [10,4]. The price to pay for the full generality of such approaches is that of a certain semantic intromission in the proof-theoretical formulation of the corresponding object logics: the tableau rules, in each case, contain formulas labeled by as many different signs as there are truth-values, or collections of truth-values. The issue here goes beyond a mere sacrifice in elegance: the staggering and heavily semantic-dependent wealth of signs for formulas culminates

H. Ono, M. Kanazawa, and R. de Queiroz (Eds.): WoLLIC 2009, LNAI 5514, pp. 268–280, 2009.

in irksome difficulties for the task of comparing different logics on what regards their deductive strengths, given the inexistence of a uniform object-language framework for dealing with all of them at once.

The intrinsic bivalence that underlies the usual definition of entailment for many-valued logics has suggested that the many 'algebraic' truth-values of the latter might be represented by the use of just two 'logical' values (cf. [15]). A constructive procedure for producing an equivalent bivalent semantics for any sufficiently expressive finite-valued logic has been proposed in [6]. Suitable machineries for extracting classic-like sequent systems for generous classes of such bivalent semantics were set up in [5,1], and a sketch of how any such bivalent semantics may give origin to classic-like 2-signed tableaux was offered in [6] and implemented in [13]. The advantage of uniformity of framework provided by the mentioned constructive extraction of adequate 2-signed tableau-theoretic formalizations for finite-valued logics was partially canceled, however, by the fact that among such tableau rules a non-analytic dual branching version of the cut rule was to be found. In contrast, the present paper is to show in detail how adequate classic-like tableau systems may be constructively extracted directly from the corresponding finite-valued semantics, this time with the additional advantage of analyticity, a feature that allows for the immediate design of fully automated decision tacticals for the logics characterized by such semantics.

2 Truth-Functionality *vs.* Bivalence

Consider an alphabet consisting of a denumerable set $\mathcal{A} = \{p_0, p_1, p_2, \ldots\}$ of *atoms/variables* and a finite set of *connectives* $\Sigma = \{\odot_0, \odot_1, \ldots, \odot_k\}$. The arity of a given connective $\odot \in \Sigma$ will be denoted by $\mathsf{ar}\odot$. The set $\mathbb{S}$ of *formulas*, as usual, is the algebra freely generated over $\mathcal{A}$ with respect to Σ. Let $\mathcal{V}_n = \{v_0, v_1, \ldots, v_{n-1}\}$ be a set of *truth-values*, partitioned into a set $\mathcal{D} \subseteq \mathcal{V}_n$ of *designated* values and a set $\mathcal{U} = \mathcal{V}_n \setminus \mathcal{D}$ of *undesignated* values. In what follows, it will be handy in many cases to assume $F = v_0$ and $T = v_{n-1}$. In general, an n-(*valued*) *assignment* of truth-values to the atoms is any mapping $\rho : \mathcal{A} \to \mathcal{V}_n$, and an n-(*valued*) *valuation* is any extension $w : \mathbb{S} \to \mathcal{V}_n$ of such an assignment to the set of all formulas. An *n-valent semantics* for $\mathbb{S}$ based on $\mathcal{V}_n$, then, is simply an arbitrary collection of n-valued valuations. In particular, we will call *bivalent* any (classic-like) semantics where $\mathcal{V}_2 = \{F, T\}$ and $\mathcal{D}_2 = \{T\}$; the corresponding valuations are called *bivaluations*. Canonical notions of *entailment* $\models_x \subseteq \mathsf{Pow}(\mathbb{S}) \times \mathbb{S}$ characterizing a *logic* $\mathcal{L}$ may be associated to any valuation w and any n-valent semantics Sem, if one simply sets $\Gamma \models_w \alpha$ iff $(w(\alpha) \in \mathcal{D}$ whenever $w(\Gamma) \subseteq \mathcal{D})$, and $\Gamma \models_{\mathsf{Sem}} \alpha$ iff $(\Gamma \models_w \alpha$ for every $w \in \mathsf{Sem})$, where $\Gamma \cup \{\alpha\} \subseteq \mathbb{S}$. Any pair $\langle \Gamma, \alpha \rangle \in \mathsf{Pow}(\mathbb{S}) \times \mathbb{S}$ such that $\Gamma \models_{\mathsf{Sem}} \alpha$ is called a *valid inference* of Sem.

A particularly interesting case of n-valent semantics corresponds to the ones we call *truth-functional*, namely, semantics provided to the set of formulas $\mathbb{S}$ by defining an appropriate Σ-algebra $\mathbb{V}$ with carrier $\mathcal{V}_n$, associating to each $\odot \in \Sigma$ an $\mathsf{ar}\odot$-ary operator $\widehat{\odot} \in \mathbb{V}$, and collecting in Sem the set of all homomorphisms $\S : \mathbb{S} \to \mathbb{V}$. Any such homomorphism, as is usual in the field of

universal algebra, can be understood as the unique extension of an assignment $\rho : \mathcal{A} \to \mathcal{V}_n$ into a valuation $\S_\rho : \mathbb{S} \to \mathbb{V}$ where one imposes $\S(\odot(\varphi_1, \ldots, \varphi_{\mathsf{ar}\odot})) = \widehat{\odot}(\S(\varphi_1), \ldots, \S(\varphi_{\mathsf{ar}\odot}))$. This way, one might say that such a semantics is *compositional*, in that the meaning it attributes to a complex expression clearly depends (functionally) on the meaning of its directly subordinated subexpressions. Any logic characterized by truth-functional means, for a given $\mathcal{V}_n$, is called *n-valued*; we will say that an n-valued logic $\mathcal{L}$ with an entailment relation $\models_{\mathsf{Sem}}$ is *genuinely n-valued* in case there is no $m < n$ such that $\models_{\mathsf{Sem}}$ can be canonically obtained by way of an m-valued truth-functional semantics.

Example 1. Our running example for this paper will involve a well-known class of truth-functional n-valued logics, namely Łukasiewicz's logics $\mathrm{Ł}_n$, for $n > 2$. Each $\mathrm{Ł}_n$ may be characterized by considering the unary connective $\neg$ and the binary connective $\supset$, together with a set of truth-values $\mathcal{V}_n$ where the designated ones form the singleton $\mathcal{D} = \{v_{n-1}\}$, while interpreting $\widehat{\neg} v_i$ as $v_{(n-1)-i}$ and interpreting $v_i \widehat{\supset} v_j$ as $v_{(n-1)-(i-j)}$ in case $i > j$, and as $v_{(n-1)}$ otherwise, where $0 \leq v_i, v_j \leq n-1$.

Taking advantage of the residual shadow of bivalence that lurks in the distinction between designated and undesignated truth-values, it is easy to see that any entailment relation characterizing an n-valued logic can also be characterized by way of a bivalent semantics. Indeed, consider the total mapping $t : \mathcal{V}_n \to \mathcal{V}_2$ such that $t(v) = T$ iff $v \in \mathcal{D}$ and define, for any valuation $\S : \mathbb{S} \to \mathbb{V}$ of an n-valued semantics Sem, the bivaluation $b_\S = t \circ \S$. Collect all such bivaluations into a semantics Sem_2 and notice that $\Gamma \models_x \alpha$ iff $\Gamma \models_y \alpha$, where $\langle x, y\rangle \in \{\langle \S, b_\S\rangle, \langle \mathsf{Sem}, \mathsf{Sem}_2\rangle\}$.

Now, given a genuinely n-valued logic, for $n > 2$, describing this same logic by way of a bivalent (non-truth-functional) semantics would seem to throw away the fundamental feature of compositionality, making the resulting semantic characterization less appealing both from a meta-theoretical and from a practical point of view. As we will see in what follows, however, this is not necessarily the case, as bivalent semantics can be quite profitable and informative, even in the non-truth-functional case, where in fact an extended notion of compositionality may be entertained.

Let's first try and find a way of distinguishing each pair of values of a genuinely n-valued logic $\mathcal{L}$. Given $v_i, v_j \in \mathcal{V}$, we write $v_i \sharp v_j$ and say that v_i and v_j are *separated* in case v_i and v_j belong to different classes of truth-values, that is, in case $t(v_i) \neq t(v_j)$. Given any two variables p_i and p_j and any valuation $\S$ such that $v_i = \S(p_i) \neq \S(p_j) = v_j$ yet $b_\S(p_i) = b_\S(p_j)$, we say that a one-variable formula $\theta^{ij}(p)$ of $\mathcal{L}$ *separates* v_i and v_j if $\S(\theta^{ij}(p_i)) \sharp \S(\theta^{ij}(p_j))$ (or, equivalently, $b_\S(\theta^{ij}(p_i)) \neq b_\S(\theta^{ij}(p_j))$). In that case we will also say that the values v_i and v_j of $\mathcal{L}$ are *effectively distinguishable*, as they may be separated using just the original linguistic resources of $\mathcal{L}$. Finally, we will say that the logic $\mathcal{L}$ is *effectively separable* in case its truth-values are pairwise effectively distinguishable, that is, for any pair of distinct values $\langle v_i, v_j\rangle \in \mathcal{D}^2 \cup \mathcal{U}^2$ a one-variable formula $\theta^{ij}(p)$ can be found in $\mathcal{L}$ that separates v_i and v_j. Collect, without repetition, all such one-variable formulas into a finite sequence $\theta_1(p), \ldots, \theta_s(p)$, and assume $\theta_0(p) = p$;

obviously, $\theta_0(p)$ by itself suffices to separate any pair of values $\langle v_i, v_j\rangle \in (\mathcal{D} \times \mathcal{U}) \cup (\mathcal{U} \times \mathcal{D})$. Then, the *binary print* of a value $v \in \mathcal{V}$ will be the sequence $\overline{v} = [b_{\S}(\theta_r(p))]_{r=0}^{s}$, where $\S(p) = v$. Notice that for every pair of distinct values $\langle v_i, v_j\rangle \in \mathcal{V}^2$ it is now obviously the case that $\overline{v_i} \neq \overline{v_j}$.

Example 2. Back to the example of the Ł$_n$, one has to devise a way of pairwise separating each of the $n-1$ undesignated values, in each case. A well-known general method, in the case of the Ł$_n$, is to use the Rosser-Turquette functions (cf. [14]). To give an independent illustration of how the separation can be done in the particular case of Ł$_3$, one might either define $\theta^{01}(p) = \theta_1(p)$ as $\neg p$, using a primitive connective of the language of Ł$_3$, or alternatively define this same $\theta_1(p)$ using a more complex formula such as $\neg p \supset p$. To simplify notation, in this particular case of Ł$_3$, where a single separating formula θ_1 suffices, we shall drop its subscript. For different reasons, it is obvious in each case that $\overline{v_0} \neq \overline{v_1}$. Indeed, using the first definition, the binary prints corresponding respectively to v_0, v_1 and v_2 are $\langle F, T\rangle$, $\langle F, F\rangle$ and $\langle T, F\rangle$; the second definition originates, respectively, the binary prints $\langle F, F\rangle$, $\langle F, T\rangle$ and $\langle T, T\rangle$.

The next sections will show how such effective separation of truth-values, whenever it can be effected —and that is a decidable property of a finite-valued logic—, may be used to automatically produce adequate classic-like analytic tableau systems for the corresponding finite-valued logics.

3 A Uniform Analytic Deductive Formalism

The result, mentioned in the last section, that allowed for the characterization of a finite-valued logic by way of bivalent semantics, coupled with the technique that allows for the separation of the algebraic truth-values of the object logic by way of the binary print defined with the help of the linguistic resources of that very logic, gives a hint on how the corresponding adequate bivalent semantics may be constructively described, in each case. A further step will now be to show in detail how this bivalence may be explored in order to devise an adequate classic-like formalism to investigate a finite-valued logic from a proof-theoretic perspective. Before outlining the general method we intend to propose, having as output an appropriate labeled (in fact, 2-signed) tableau system for some given sufficiently expressive finite-valued logic, let's illustrate it in the present section with a fully worked example. We shall use & to represent conjunction in the classical metalanguage, || to represent disjunction, $\Longrightarrow$ to represent implication, and $\divideontimes$ to represent an absurd.

Now, consider again the case of Ł$_3$, where $\Sigma = \{\neg, \supset\}$, and recall the particular separation of truth-values produced by setting $\theta(p) = \neg p \supset p$. It follows that $\theta(\neg\varphi_1) = \neg\neg\varphi_1 \supset \neg\varphi_1$ and $\theta(\varphi_1 \supset \varphi_2) = \neg(\varphi_1 \supset \varphi_2) \supset (\varphi_1 \supset \varphi_2)$. Using the 3-valued semantics of Ł$_3$ one will then notice that:

$\S(\theta(\neg\varphi_1)) = v_0$ only if $\S(\varphi_1) = v_2$ and $\S(\theta(\neg\varphi_1)) = v_2$ only if $\S(\varphi_1) \in \{v_0, v_1\}$

Recalling the binary prints of v_0 as $\langle F, F\rangle$, of v_1 as $\langle F, T\rangle$, and of v_2 as $\langle T, T\rangle$, one might rewrite now the above by way of the following first-order schematic sentences, whose consequents are written in a kind of 'disjunctive normal form':

(Ł_3.1) $F{:}\theta(\neg\varphi_1) \Longrightarrow (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1))$
(Ł_3.2) $T{:}\theta(\neg\varphi_1) \Longrightarrow (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1)) \;||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1))$

Similarly, one might also notice that:

$\S(\theta(\varphi_1 \supset \varphi_2)) = v_0$ only if $\S(\varphi_1) = v_2$ and $\S(\varphi_2) = v_0$,
$\S(\theta(\varphi_1 \supset \varphi_2)) \neq v_1$ for every valuation $\S$, and
$\S(\theta(\varphi_1 \supset \varphi_2)) = v_2$, otherwise.

These immediately translate into:

(Ł_3.3) $F{:}\theta(\varphi_1 \supset \varphi_2) \Longrightarrow (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2))$
(Ł_3.4) $T{:}\theta(\varphi_1 \supset \varphi_2) \Longrightarrow (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$

We will call the expressions (Ł_3.1)–(Ł_3.4) *θ-rules*. The full description of Ł_3 will also include the following *non-θ-rules*, obtained by using the binary prints now to describe the original truth-tables of the primitive connectives of Ł_3:

(Ł_3.5) $F{:}\neg\varphi_1 \Longrightarrow (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1)) \;||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1))$
(Ł_3.6) $T{:}\neg\varphi_1 \Longrightarrow (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1))$
(Ł_3.7) $F{:}(\varphi_1 \supset \varphi_2) \Longrightarrow (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2))$
$||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2)) \;||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2))$
(Ł_3.8) $T{:}(\varphi_1 \supset \varphi_2) \Longrightarrow$
$(F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; F{:}\theta(\varphi_2)) \;||\; (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; F{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2)) \;||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; F{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$
$||\; (F{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2)) \;||\; (T{:}\varphi_1 \;\&\; T{:}\theta(\varphi_1) \;\&\; T{:}\varphi_2 \;\&\; T{:}\theta(\varphi_2))$

The above first-order expressions are intended to represent tableau rules: the antecedent of each rule is the head, and the disjunction in the consequent describes the branches that may be created once the head is matched on a previously given branch. In addition to the traditional closure rule for 2-signed tableaux, which says that a branch is closed once it contains two signed formulas of the form $F{:}\varphi$ and $T{:}\varphi$, additional closure rules will be needed in order to exclude each binary print not allowed for the given logic with a given choice of separating formulas. In the present case, as only the pair $\langle T, F\rangle$ fails to be among the pairs allowed as binary prints of the initial collection of truth-values, an additional closure rule will say that branches containing both a signed formula of the form $T{:}\varphi$ and a signed formula of the form $F{:}\theta(\varphi)$ may be closed. One might represent such closure rules by writing:

(Ł_3.C0) $F{:}\varphi \;\&\; T{:}\varphi \Longrightarrow \divideontimes$ (Ł_3.C1) $T{:}\varphi \;\&\; F{:}\theta(\varphi) \Longrightarrow \divideontimes$

There is, of course, a lot of redundancy to be found among the above mechanically extracted θ-rules and non-θ-rules. A simpler tableau system for Ł$_3$ might be defined, though, by any set of expressions first-order-equivalent to (Ł$_3$.1)–(Ł$_3$.8), together with the already mentioned appropriate closure rules. An example of such a simpler axiomatization is provided by (Ł$_3$.C0) and (Ł$_3$.C1) together with:

θ-rules

$(Ł_3.1)^*\ F{:}\theta(\neg\varphi_1) \Longrightarrow T{:}\varphi_1$ $\qquad$ $(Ł_3.2)^*\ T{:}\theta(\neg\varphi_1) \Longrightarrow F{:}\varphi_1$

$(Ł_3.3)^*\ F{:}\theta(\varphi_1 \supset \varphi_2) \Longrightarrow (T{:}\varphi_1\ \&\ F{:}\theta(\varphi_2))$

$(Ł_3.4)^*\ T{:}\theta(\varphi_1 \supset \varphi_2) \Longrightarrow F{:}\varphi_1\ \mid\mid\ T{:}\theta(\varphi_2)$

non-θ-rules

$(Ł_3.5)^*\ F{:}\neg\varphi_1 \Longrightarrow T{:}\theta(\varphi_1)$ $\qquad$ $(Ł_3.6)^*\ T{:}\neg\varphi_1 \Longrightarrow F{:}\theta(\varphi_1)$

$(Ł_3.7)^*\ F{:}(\varphi_1 \supset \varphi_2) \Longrightarrow (T{:}\varphi_1\ \&\ F{:}\varphi_2)\ \mid\mid\ (T{:}\theta(\varphi_1)\ \&\ F{:}\theta(\varphi_2))$

$(Ł_3.8)^*\ T{:}(\varphi_1 \supset \varphi_2) \Longrightarrow F{:}\theta(\varphi_1)\ \mid\mid\ T{:}\varphi_2\ \mid\mid\ (F{:}\varphi_1\ \&\ T{:}\theta(\varphi_2))$

Figure 1 (see Appendix) shows an example of a tableau for Ł$_3$, using the above simpler set of rules. There, the branches (2.1) and (2.2) originate from the application of rule (Ł$_3$.7)* to the signed formula (1), and the same rule applies to (2.1.2) to originate branches (3.1) and (3.2). The branch that goes through (4) originates from the application of rule (Ł$_3$.3)* to the signed formula (2.2.2). The usual closure rule for tableaux, (Ł$_3$.C0), closes the leftmost branch, in view of the nodes (2.1.1) and (3.1.2). Similarly for the rightmost branch, in view of (2.2.1) and (4.2), even though the involved formula in this case is non-atomic. As for the innermost branch, notice that (3.2.2) has the form $F{:}\theta(p_0)$, so the additional closure rule of Ł$_3$, (Ł$_3$.C1), finishes off the branch, in view of (2.1.1).

Special care must be exercised, in the present environment, as θ-rules might interfere with non-θ-rules, or with other θ-rules, and as the blind application of tableau rules might make the same signed formula appear over and over again, in the same branch. Even worse, it would appear that, applied in the wrong order, some signed formulas might give rise to increasingly more complex signed formulas. For instance, suppose that in the tableau from Fig. 1 one never applied the combination of rules (Ł$_3$.3)* and (Ł$_3$C0) after (2.2.2), but applied instead rule (Ł$_3$.8)* to the signed formula $T{:}\theta(p_0)$ that appears in (2.2.1). One of the originating branches would then extend this branch exactly by adding the signed formula $F{:}\theta(\neg p_0)$. If, further on, to this new signed formula one applied rule (Ł$_3$.8)* instead of rule (Ł$_3$.1)*, a new branch would originate in which the signed formula $T{:}\theta(\neg\neg p_0)$ were added. Such unfortunate sequential choice of rules could of course go on forever, producing more and more complex signed formulas, originating thus a tableau branch that would never be closed.

A way of avoiding the above phenomenon would consist in only allowing rule application in building tableaux when the signed formulas originating from a given node are strictly 'less complex' than the signed formula present in that node. One might realize such intent by choosing an appropriate *complexity measure* $\ell : \mathbb{S} \to \mathbb{N}$ for formulas that is guaranteed to decrease with rule application. In the particular case of Ł$_3$ one might define $\ell(p) = 0$, $\ell(\theta(\varphi)) = \ell(\varphi)$, $\ell(\neg\varphi_1) = \ell(\varphi_1) + 1$ and $\ell(\varphi_2 \supset \varphi_3) = \ell(\varphi_2) + \ell(\varphi_3) + 1$, where $p \in \mathcal{A}$, $\varphi, \varphi_1, \varphi_2, \varphi_3 \in \mathbb{S}$, and where φ_2 is not of the form $\neg\varphi_3$, that is, $\varphi_2 \supset \varphi_3$ does not

appear at the head of a θ-rule. One might now use such complexity measure as a guide while constructing tableaux for Ł$_3$, observing that: (i) no rule applies to an atom p, and similarly no rule should be applied to a formula of the form $\theta(p)$, as both p and $\theta(p)$ have complexity zero; (ii) θ-rules contribute more to the reduction of complexity than non-θ-rules, as a formula of the form $\theta(\varphi)$ has the same complexity as φ, and the consequent of an application of a θ-rule involves only formulas of lower complexity, of the forms φ_r or $\theta(\varphi_r)$, where φ_r is a proper subformula of φ. Furthermore, although it is not the case for Ł$_3$, it may happen in general that more than one θ-rule is applicable to the same signed formula. In such a case, as we will impose below, one ought to choose the θ-rule whose head is more 'concrete'. As it turns out, it is always possible to order the rules in a way that solves all ambiguities while guaranteeing also that all tableau constructions terminate. We shall dub such tableau-building heuristics a *requirement of analyticity*, and, as we shall see, it will guarantee that our tableau proofs are normalized and terminate. In the case of the previous example, the requirement of analyticity would guarantee that rule (Ł$_3$.8)* would never be applied to node (2.2.1), as the formula with sign T that appears in the that node already has complexity zero. Indeed, as we have seen, modifying the above example in allowing a non-θ-rule to be applied before a θ-rule in a situation involving non-atomic formulas turned out to allow for the production of increasingly more complex formulas — as it is obviously the case, for instance, that $\ell(\theta(\varphi)) < \ell(\theta(\neg\varphi))$.

The next section will show how the above illustrated procedure may be generalized so as to provide adequate and well-behaved proof-theoretical formalisms, together with a classic-like decision procedure, for a very inclusive class of truth-functional logics.

4 The Extraction of Adequate Classic-Like Tableau Systems for Finite-Valued Logics

Let $\mathcal{L}$ be an effectively separable n-valued logic with a set of formulas $\mathbb{S}$ generated over the denumerable set of atoms $\mathcal{A} = \{p_0, p_1, p_2, \ldots\}$ with respect to the set of connectives $\Sigma = \{\odot_0, \odot_1, \ldots, \odot_k\}$ and having $\mathcal{D} \subseteq \mathcal{V}_n$ as its set of designated values, and assume that its binary prints are produced by a convenient sequence of one-variable separating formulas $\theta_1(p), \ldots, \theta_s(p)$. Recall that we set $\theta_0(p) = p$. So, for each truth-value $v \in \mathcal{V}_n$, we might take an atom p and an n-valued assignment $\S$ such that $\S(p) = v$, and consider the distinctive characterizing bivalent sequence $\overline{v} = [b_\S(\theta_r(p))]_{r=0}^s$. As a matter of convention, we shall say that an n-valued valuation $\S$ *satisfies* a labeled formula of the form $X{:}\delta$ if $b_\S(\delta) = X$.

As for the associated tableau rules, consider first the usual classic-like closure rule (C0) of the form: $F{:}\varphi$ & $T{:}\varphi \Longrightarrow *$. Furthermore, let $\mathsf{BP} = \{[c_r]_{r=0}^s : c_r \in \{F, T\}\}$ be the set of all possible $(s+1)$-long binary prints, and let $\mathsf{CL} = \mathsf{BP} \setminus \{\overline{v} : v \in \mathcal{V}_n\}$ be the set of all such bivalent sequences that are *not* produced as binary prints of truth-values of $\mathcal{L}$. Intuitively, any *closuring sequence* $\widetilde{c} \in \mathsf{CL}$ brings about information that is unobtainable by way of the initial truth-values of $\mathcal{L}$, allowing one thus to close a tableau branch that contains such a sequence.

Information, even if partial, leading unambiguously to a binary print in CL should always give rise to a closed tableau. Let a *partial binary print* be any sequence $\widetilde{c}_R = [c_r]_{r\in R}$ such that $R \subseteq \{0,\ldots,s\}$ and each $c_r \in \{F,T\}$ (this definition includes, of course, the total binary prints in BP, as strict partiality occurs exactly when R is a proper subset of $\{0,\ldots,s\}$). Given two partial binary prints $\widetilde{c}_{R_1}$ and $\widetilde{d}_{R_2}$, we say that $\widetilde{c}_{R_1}$ extends $\widetilde{d}_{R_2}$ if $R_2 \subsetneq R_1$ and $d_r = c_r$ for every $r \in R_2$. We can now conclude that closuring information is carried by any partial binary print $\widetilde{c}_R$ such that all of its $2^{s+1-\mathsf{Card}(R)}$ possible total extensions are in CL. Hence, it would be reasonable to add a different closure rule for each such partial closuring information. However, it suffices to take into account just the minimal closuring situations, that is, closuring partial binary prints $\widetilde{c}_R$ that cannot be obtained as extensions of any other closuring partial binary print.

In general, where $\widetilde{c}_R = [c_r]_{r\in R}$ is some partial binary print, and δ stands for an arbitrary schematic formula, we write $\widetilde{c}^{\mathbb{S}}_R(\delta) = [c_r{:}\theta_r(\delta)]_{r\in R}$ for the linguistic 2-signed version of such partial binary print. Accordingly, for each minimal closuring partial binary print $\widetilde{c}_R$, consider an additional closure rule (C#) of the form: $\&(\widetilde{c}^{\mathbb{S}}_R(\varphi)) \Longrightarrow *$.

Recall, moreover, that for each connective $\odot : \mathbb{S}^j \to \mathbb{S}$ of Σ with arity $j = \mathsf{ar}\odot$ there is an associated operator $\widehat{\odot} : \mathcal{V}_n^j \to \mathcal{V}_n$ in the algebra of truth-values. This interpretation mapping can immediately be extended homomorphically to any formula δ of any given arity, and we shall denote this by $\widehat{\delta}$. Finally, consider again the flattening total mapping $t : \mathcal{V}_n \to \{F,T\}$ such that $t(v) = T$ iff $v \in \mathcal{D}$. Given $X \in \{F,T\}$, a j-ary connective $\odot$ and a separating formula θ, let $B^{\theta\odot}_X([\varphi_i]_{i=1}^j) = \{\&[\overline{v_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^j : t(\widehat{\theta}(\widehat{\odot}([v_i]_{i=1}^j))) = X\}$. For each $B^{\theta\odot}_X([\varphi_i]_{i=1}^j)$ consider then a rule of the form: $X{:}\theta(\odot([\varphi_i]_{i=1}^j)) \Longrightarrow \mid\mid B^{\theta\odot}_X([\varphi_i]_{i=1}^j)$. Such rules are called *θ-rules* when $\theta = \theta_i$, for some $0 < i \leq s$; otherwise, when $\theta = \theta_0(p)$, they are called *non-θ-rules*. Notice that those rules generate as many branches as there are members (conjunctions) of $B^{\theta\odot}_X([\varphi_i]_{i=1}^j)$. The number of different non-θ-rules is $2\times\mathsf{Card}(\Sigma)$, and there are $2\times s\times\mathsf{Card}(\Sigma)$ different θ-rules. For each fixed j-ary connective $\odot$ and each fixed separating formula θ, the summed number of branches of the rules $B^{\theta\odot}_F([\varphi_i]_{i=1}^j)$ and $B^{\theta\odot}_T([\varphi_i]_{i=1}^j)$ generated by the above procedure always amounts to n^j; additionally, each branch will have exactly $s+1$ labeled formulas. In the best case, one might use $\mathsf{Ceiling}(\log_2 \mathsf{Max}(d, n-d))$ separating formulas, besides identity, where $d = \mathsf{Card}(\mathcal{D})$, to pairwise distinguish the n truth-values of $\mathcal{L}$; in the worst case, $n-1$ such connectives will be needed. The case in which, say, $B^{\theta\odot}_F([\varphi_i]_{i=1}^j)$ turns out to be empty originates in fact a new closure rule, and in that case the rule $B^{\theta\odot}_T([\varphi_i]_{i=1}^j)$ can thus be omitted; the case in which $B^{\theta\odot}_T([\varphi_i]_{i=1}^j)$ turns out to be empty is entirely symmetric.

Tableaux are built as usual, by applying the above rules, given an initial sequence of 2-signed formulas, and a branch is said to be *closed* if its closure is obtained by the application of any of the (C#) rules, including (C0). Branches that are not closed are said to be *open*. A tableau is said to be *closed* in case all of its branches are closed. By the construction of the above rules, it is easy to check the following soundness result with respect to the initially given truth-functional semantics:

Theorem 1. If a valuation satisfies some initial sequence of 2-signed formulas, then it satisfies all the formulas in some open branch of any tableau that originates from that set of formulas.

To enforce the requirement of analyticity for our tableaux the following strategy will be helpful. Notice that, from the point of view of analyticity, only a formula in the set $\Theta = \{\theta_r(\varphi) : 0 < r \leq s, \varphi \in \mathbb{S} \setminus \mathcal{A}\}$ is possibly 'problematic', as more than one rule may apply to (an appropriate labelling of) it. In the case of such a formula we will always apply a carefully chosen θ-rule, as follows. In general, given $\delta \in \Theta$, let $I_\delta = \{0 < r \leq s :$ there exists $\odot_r([\delta_i]_{i=1}^{j_r}) \in \mathbb{S}$ such that $\delta = \theta_r(\odot_r([\delta_i]_{i=1}^{j_r}))\}$. To ensure the termination of the tableau construction procedure, the choice of rule to be applied in each case will be guided by the following complexity measure $\ell : \mathbb{S} \to \mathbb{N}$, defined for the formulas of $\mathcal{L}$:

($\ell 0$) $\ell(p) = \ell(\theta_r(p)) = 0$, where $p \in \mathcal{A}$;
($\ell 1$) $\ell(\delta) = 1 + \mathsf{Min}_{r \in I_\delta} \left(\sum_{i=1}^{j_r} \ell(\delta_i)\right)$, where $\delta \in \Theta$;
($\ell 2$) $\ell(\odot([\varphi_i]_{i=1}^{j})) = 1 + \sum_{i=1}^{j} \ell(\varphi_i)$, otherwise.

Accordingly, no rule application should be allowed in a tableau for nodes of complexity 0; moreover, applications of θ-rules should always precede applications of non-θ-rules, as the former clearly contribute more than the latter to decreasing the overall complexity of the corresponding nodes. A particularly interesting situation arises in the case where the heads of more than one θ-rule are matched by the same node. In that case, the θ-rule to be applied may be chosen by heeding the minimality requirement in clause ($\ell 1$) of the definition of the complexity measure. This choice is typically very simple. Indeed, consider the situation in which a formula δ can be obtained either as $\theta_i(\varphi_i)$ or as $\theta_j(\varphi_j)$. In that case, $\theta_i(p_i)$ might be thought of as a formula on the variable p_i, of which $\theta_i(\varphi_i)$ is a substitution instance, and similarly for $\theta_j(p_j)$ as a formula on the variable p_j. Now, by Robinson's unification algorithm, $\theta_i(p_i)$ and $\theta_j(p_j)$ will have a most general unifier, that is, either it is the case that $p_i = \sigma(p_j)$ or else that $p_j = \sigma(p_i)$, for an appropriate substitution σ. In the former case, where $p_i = \sigma(p_j)$, the θ-rule to be applied first is the one for θ_j, as $\theta_j(p_j) = \theta_i(\sigma(p_j))$; the latter case is entirely symmetric. As it can easily be seen, this prioritary application of the 'most concrete' rule will provide the greater decrease in complexity for a given node, and will help implementing the requirement of analyticity. The situation is a bit more complex in the remaining case, where the most general unifier of $\theta_i(p_i)$ and $\theta_j(p_j)$ asserts simultaneously that $\sigma(p_i) = \delta_i$ and $\sigma(p_j) = \delta_j$ where δ_i, δ_j are formulas in which the variables p_i, p_j, respectively, do not occur. In this (peculiar) case, and only for the formula $\theta_i(\delta_i) = \theta_j(\delta_j)$, we must directly check which of the two corresponding θ-rules matches the minimality requirement.

Say that a tableau branch is *exhausted* if it is closed, or if the appropriate θ-rules have been applied to every node whose formulas have non-null complexity and non-θ-rules have been applied to every other non-atomic node. Our main normalization result guarantees that:

Lemma 1. Exhausted tableaux always exist.

A nice thing about exhausted tableaux is that all counter-models for non-valid inferences can be built from them. Indeed, using the above lemma one may easily prove now the following completeness result:

Theorem 2. From every open branch of an exhausted tableau for a given initial sequence of 2-signed formulas one may extract a valuation satisfying those formulas.

As an immediate byproduct of the previous results, it follows that:

Corollary 1. For a given logic $\mathcal{L}$ with semantics Sem:

$\gamma_1, \ldots, \gamma_m \models_{\mathsf{Sem}} \alpha$ iff there is a closed tableau for $T{:}\gamma_1, \ldots, T{:}\gamma_m, F{:}\alpha$.

5 Future Work

There are a number of possible directions for further extension of our present results. Previous work on extraction of non-analytic tableau rules for sufficiently expressive finite-valued logics (cf. [6]) has been implemented (cf. [13]) for the extraction of appropriate axioms in the framework of `Isabelle`'s intuitionistic higher-order logic. A similar implementation might now be performed for the presently illustrated procedure, with the advantage that analyticity guarantees the existence of fully automated decision procedures, and these can be constructively assembled as tacticals in `Isabelle`'s meta-language. Moreover, the 2-signed 'normal form' that underlies the statement of our tableau rules in the (intuitionistic) meta-language may also be used as a basis for the study of classic-like automated reasoning mechanisms based on satisfiability checking or signed resolution (cf. [11,4,9]).

On what concerns the issue about 'sufficient expressiveness', a requisite for the application of our present procedure to a given finite-valued logic, it should be noted that the calculation of a convenient collection of separating formulas, whenever it exists, may also be performed automatically. Moreover, as it can be shown, for the logics that turn out *not* to be sufficiently expressive, a conservative extension of the original language may be devised by the addition of a convenient number of 0-ary connectives in order to make such logics amenable to our method. Such is the case, for instance, for Gödel-Dummett n-valued logics, with finite $n > 3$. The complexity of such procedures, not yet described nor implemented in detail, is still to be fully explored.

Finally, it will be interesting to investigate the amount in which the above procedures can be extended so as to apply to logics characterized by semantics that broaden the notion of truth-functionality, such as non-deterministic semantics (cf. [2]) and possible-translations semantics (cf. [12]). For one thing, as we shall show in future studies, the adequacy results that connect bivalent semantics to the corresponding tableau systems can in fact be generalized so as to cover many other logics characterized by what we call 'dyadic semantics' [6,7], including numerous interesting infinite-valued logics. Such generic results, in fact, also take into account logics whose non-truth-functional semantics is formulated using more than 2 'logical values'.

Acknowledgments. The first author was partially supported by FCT and EU FEDER via the KLog project PTDC/MAT/68723/2006 of SQIG-IT. The second author acknowledges partial support by SQIG-IT and CNPq. The authors are deeply indebted to their colleagues W. Carnielli and M. Coniglio for many fruitful discussions on topics related to this work, and also to D. Mendonça and three anonymous referees for their attentive reading and valuable suggestions of improvements on earlier drafts of the paper.

References

1. Aguzzoli, S., Ciabattoni, A., Di Nola, A.: Sequent calculi for finite-valued Łukasiewicz logics via Boolean decompositions. Journal of Logic and Computation 10(2), 213–222 (2000)
2. Avron, A., Lev, I.: Non-deterministic multiple-valued structures. Journal of Logic and Computation 15, 241–261 (2005)
3. Surma, S.J.: An algorithm for axiomatizing every finite logic. In: Rine, D.C. (ed.) Computer Science and Multiple-Valued Logics, Selected Papers from the IV International Symposium on Multiple-Valued Logics, 2nd edn., pp. 143–149. North-Holland, Amsterdam (1974) 2nd edn. (1984)
4. Baaz, M., Fermüller, C.G., Salzer, G.: Automated deduction for many-valued logics. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 1355–1402. Elsevier and MIT Press (2001)
5. Béziau, J.-Y.: Sequents and bivaluations. Logique et Analyse (N.S.) 44(176), 373–394 (2001)
6. Caleiro, C., Carnielli, W., Coniglio, M.E., Marcos, J.: Two's company: The humbug of many logical values. In: Béziau, J.-Y. (ed.) Logica Universalis, pp. 169–189. Birkhäuser Verlag, Basel (2005), `http://wslc.math.ist.utl.pt/ftp/pub/CaleiroC/05-CCCM-dyadic.pdf`
7. Caleiro, C., Carnielli, W.A., Coniglio, M.E., Marcos, J.: How many logical values are there? Dyadic semantics for many-valued logics. Draft (forthcoming) (2005)
8. Carnielli, W.A.: Systematization of the finite many-valued logics through the method of tableaux. The Journal of Symbolic Logic 52(2), 473–493 (1987)
9. Fermüller, C.G., Leitsch, A., Hustadt, U., Tammet, T.: Resolution decision procedures. In: Handbook of Automated Reasoning, pp. 1791–1849. Elsevier, Amsterdam (2001)
10. Hähnle, R.: Tableaux for many-valued logics. In: D'Agostino, M., Gabbay, D., Hähnle, R., Posegga, J. (eds.) Handbook of Tableau Methods, pp. 529–580. Springer, Heidelberg (1999)
11. Hähnle, R.: Advanced many-valued logics. In: Gabbay, D.M., Guenthner, F. (eds.) Handbook of Philosophical Logic, 2nd edn., vol. 2, pp. 297–395. Kluwer, Dordrecht (2001)
12. Marcos, J.: Possible-translations semantics. In: Carnielli, W.A., Dionísio, F.M., Mateus, P. (eds.) Proceedings of the Workshop on Combination of Logics: Theory and applications (CombLog 2004), held in Lisbon, PT. 1049-001 Lisbon, PT, 2004. Departamento de Matemática, Instituto Superior Técnico. July 28–30, 2004, Lisbon, PT, July 28–30, 2004, pp. 119–128 (2004), `http://wslc.math.ist.utl.pt/ftp/pub/MarcosJ/04-M-pts.pdf`

13. Marcos, J., Mendonça, D.: Towards fully automated axiom extraction for finite-valued logics. In: Carnielli, W., Coniglio, M.E., D'Ottaviano, I.M.L. (eds.) The Many Sides of Logic, London. Studies in Logic. College Publications (2009), http://wslc.math.ist.utl.pt/ftp/pub/MarcosJ/08-MM-towards.pdf
14. Rosser, J.B., Turquette, A.R.: Many-Valued Logics. Studies in Logic and the Foundations of Mathematics. North-Holland, Amsterdam (1952)
15. Suszko, R.: The Fregean Axiom and Polish mathematical logic in the 1920s. Studia Logica 36, 373–380 (1977)
16. Wójcicki, R.: Theory of Logical Calculi. Kluwer, Dordrecht (1988)

Appendix

Illustration of a Tableau for Ł$_3$

The following example employs the simplified rules (Ł$_3$.#)*, together with the corresponding closure rules (Ł$_3$.C0) and (Ł$_3$.C1):

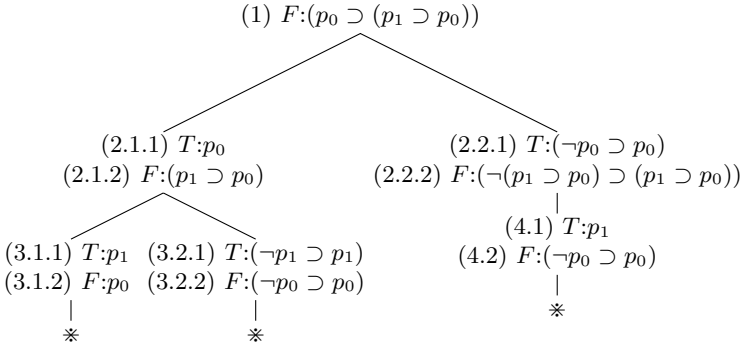

Fig. 1. A failed attempt to refute $p_0 \supset (p_1 \supset p_0)$

Proofs of the Main Results

Proof of Theorem 1. First, observe that if a valuation $\S : \mathbb{S} \to \mathcal{V}_n$ satisfies the head of a rule

$$X{:}\theta(\odot([\varphi_i]_{i=1}^j)) \implies || \, B_X^{\theta\odot}([\varphi_i]_{i=1}^j)$$

then, by construction, one may conclude that $\S$ also satisfies $\&[\overline{v_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^j$ for some sequence $[\overline{v_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^j$ of labeled formulas such that $t(\widehat{\theta}(\widehat{\odot}([v_i]_{i=1}^j))) = X$, that is, $b_\S(\varphi_i) = v_i$, for $1 \leq i \leq j$.

Suppose now that a valuation $\S$ satisfies a given sequence of 2-signed formulas. Then, by the above observation, it is clear that $\S$ satisfies all the formulas in some branch of any tableau built from these initial formulas by applying rules as above. To see that any such branch must be open, just note that no closure rule may be applied. To that effect, the branch would have to contain labeled

formulas that match the head $F{:}\varphi \,\&\, T{:}\varphi$ of the classic-like closure rule (C0), or else the head $\&(\widetilde{c}^{\mathbb{S}}_R(\varphi))$ of a closure rule (C#), for some given minimal closuring partial binary print $\widetilde{c}_R$. It would follow then that any valuation that satisfies this branch would either have to assign two different values to the formula φ or have to associate to φ a closuring partial binary print.

Proof of Lemma 1. We first check that every rule applied according to the specified requirement of analiticity does indeed reduce the complexity.

- Consider $\varphi = \theta(\odot([\varphi_i]_{i=1}^j)) \in \Theta$ with $\ell(\varphi) = 1 + \ell(\varphi_1) + \ldots + \ell(\varphi_j)$, and the corresponding θ-rule:

$$X{:}\theta(\odot([\varphi_i]_{i=1}^j)) \implies \,||\, B_X^{\theta\odot}([\varphi_i]_{i=1}^j).$$

  Then, for each $1 \leq i \leq j$ and $0 \leq r \leq s$, we have that:
  $\ell(\varphi) = 1 + \ell(\varphi_1) + \ldots + \ell(\varphi_j) > \ell(\varphi_i) \geq \ell(\theta_r(\varphi_i))$.
- Consider now $\varphi = \odot([\varphi_i]_{i=1}^j) \notin \Theta$, and the corresponding non-θ-rule:

$$X{:}\odot([\varphi_i]_{i=1}^j) \implies \,||\, B_X^{\odot}([\varphi_i]_{i=1}^j).$$

  Then, for each $1 \leq i \leq j$ and $0 \leq r \leq s$, we again have that:
  $\ell(\varphi) = 1 + \ell(\varphi_1) + \ldots + \ell(\varphi_j) > \ell(\varphi_i) \geq \ell(\theta_r(\varphi_i))$.

In either case the complexity of every signed formula in the conclusion of the rule is lower than the complexity of the head signed formula. Hence, given an initial finite set with m many 2-signed formulas of complexity bounded by g, the tableau will be exhausted after the application of at most $m \times u^g$ rules, where u is the maximum number of formulas in the conclusion of a rule.

Proof of Theorem 2. Given an open branch of an exhausted tableau, consider any assignment $\rho : \mathcal{A} \to \mathcal{V}_n$ such that, for every $p \in \mathcal{A}$, the binary print $\overline{\rho(p)} = [X_i]_{i=0}^s$ of $\rho(p)$ agrees with the information available, that is:

- either $X_i{:}\theta_i(p)$ occurs in the branch, or
- neither $T{:}\theta_i(p)$ nor $F{:}\theta_i(p)$ occur in the branch.

Accordingly, if $X_i = T$ then $F{:}\theta_i(p)$ does not appear in the branch; *mutatis mutandis*, if $X_i = F$ then $T{:}\theta_i(p)$ does not appear in the branch. Note that since none of the closure rules can be applied to the branch, we have non-closuring information about every atom p, and such an assignment ρ can always be defined. To prove that the homomorphic extension $\S_\rho : \mathbb{S} \to \mathbb{V}$ of such assignment indeed satisfies all the formulas in the branch it is sufficient to prove that if $\S_\rho$ satisfies all the formulas in some branch of a tableau rule, then it also satisfies the head of the rule. To such purpose, we need only consider the generic case of a rule

$$X{:}\theta(\odot([\varphi_i]_{i=1}^j)) \implies \,||\, B_X^{\theta\odot}([\varphi_i]_{i=1}^j).$$

Assume that $\S_\rho$ satisfies one of the disjunctions in the conclusion of the rule, that is, $\S_\rho$ satisfies some $B_X^{\theta\odot}([\varphi_i]_{i=1}^j) = \{\&[\overline{v_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^j : t(\widehat{\theta}(\widehat{\odot}([v_i]_{i=1}^j))) = X\}$. Then, for each $1 \leq i \leq j$, $\S_\rho$ satisfies $\overline{v_i}^{\mathbb{S}}(\varphi_i)$ or, equivalently, $\S_\rho(\varphi_i) = v_i$. Therefore, it follows that $t(\S_\rho(\theta(\odot([\varphi_i]_{i=1}^j)))) = X$, and thus $\S_\rho$ satisfies the head of the rule $X{:}\theta(\odot([\varphi_i]_{i=1}^j))$.