Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 323 (2016) 253–270

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Classical Resolution for Many-Valued Logics

João Marcos[1]

LoLITA & Department of Informatics and Applied Mathematics, UFRN, Brazil

Cláudia Nalon

Departament of Computer Science, University of Brasília, Brazil

Abstract

We present a resolution-based proof method for finite-valued propositional logics based on an algorithmic reduction procedure that expresses these logics in terms of bivalent semantics. Our approach is hybrid in using some elements which are internal and others which are external to the many-valued logic under consideration, as we embed its original language into a more expressive metalanguage to deal with the satisfiability problem. In contrast to previous approaches to the same problem, our target language is fully classical, what turns the design of the resolution-based rules for a specific many-valued logic into a straightforward task. Correctness results, which are proved in detail in the present study, follow easily from results on classical resolution. We illustrate the application of the method with examples, and comment on its implementation, readily achievable by direct translation into classical propositional logic, making use of reliable existing automated provers.

Keywords: Multiple-valued Logics, Bivalent Semantics, Resolution Method

1 Introduction

Many-valued logics have just celebrated their centennial jubilee, and the computational proof method known as resolution is now commemorating its semicentennial birthday. While a competent extensive overview of the applicability of many-valued logics may be found in [2], and references therein, different automatic proof methods for such logics have been covered in depth elsewhere (see [12] and [6]). Among the variegated proof procedures available for dealing with many-valued logics, the resolution-based methods detailed in [1] and [7] are of particular interest and stand as close neighbours to the method introduced in the present paper. Briefly, both of the latter papers present clausal resolution-based procedures, taking as inputs a formula in the language of a particular many-valued logic and transforming it into a

[1] The research reported in this paper falls within the scope of the EU FP7 Marie Curie PIRSES-GA-2012-318986 project GeTFun: Generalizing Truth-Functionality. The first author further acknowledges partial support of CNPq.

http://dx.doi.org/10.1016/j.entcs.2016.09.001
1571-0661/© 2016 The Author(s). Published by Elsevier B.V.

labelled formula. Labels are used to mirror truth-values at the syntactic level, and are intended to represent the semantic conditions under which a formula is satisfied. The resolution inference rule is applied to clauses containing complementary literals and whose labels are unifiable (in the sense that they represent consistent semantic conditions). The main difference between those proof methods reside on the form of the labels: in [1] labels are singletons whilst in [7] labels are sets of signs.

In the present investigation, we take a similar route. Formulae to be tested for (un)satisfiability are also transformed into labelled clauses, that is, the inference rules are applied to a more expressive language in which the semantic notions are made explicit. However, labels take in all cases a very simple format, just one out of two possible signs. Thus, differently from the approaches in [1,7], unification on labels can be easily seen as equivalent to the ordinary application of classical propositional resolution. Also, the search for inconsistency (in the language of a given many-valued logic) takes the form of hyper-resolution inference rules in the classical metalanguage, allowing for a uniform classic-like approach to be algorithmically constructed for dealing with any finite-valued logic. The transformation of formulae into the labelled language relies on previous results by one of the authors [4,5], which are briefly surveyed here in order to make the presentation self-contained.

The paper is organised as follows. Section 2 introduces logics in abstract and from a semantic viewpoint, explains that they can all be characterised by bivalent semantics, and then focuses on the class of finite-valued truth-functional logics. Emphasis is put on the standard (two-sided) notion of *logical consequence*, rather than on approaches to many-valued logics that are based on mere combinatorial manipulation of finite-valued algebras. Section 3 explains the algorithm that allows one to describe finite-valued logics in terms of statements written in a fully classical metalanguage in which only two signs are employed. Providing a proof-theoretical perspective on this requires a generalisation of the way that syntactical complexity is measured, in order to allow for analytic calculi to be extracted from such a description. The subsequent section contains our main contribution. Section 4 is dedicated to setting up a generic resolution-based proof method for an arbitrary given finite-valued logic. Subsection 4.1 shows how to transform the mentioned bivalent descriptions into a clausal format that is more appropriate for applying resolution. The corresponding transformation adds new variable symbols that help encoding the structure of the original statements, preserving and reflecting their satisfiability. As output we obtain object-level expressions that take better advantage of the above mentioned generalised notion of complexity. Subsection 4.2 introduces the inference rules of a hyper-resolution proof method that applies to the clauses produced by the latter transformation. This method lies in between internal proof systems that capitalise on syntactic features of the original logics and external proof systems that formalise reasoning about the logics in a classical logical framework. We finish by some comments on what has been achieved and on how the present investigation may be further extended.

2 Every Many-valued Logic is Bivalent

This section contextualises our present study, and explains what we mean by a 'logic', from both an abstract and a semantic perspective. Special focus is put on the truth-functional case. We also introduce here an appropriate classical meta-language for describing a collection of valuations, and in the following section we discuss how to use it in implementing a computationally useful account of the so-called *Suszko's Thesis* (check [3] and references therein), according to which "every logic has but two logical values".

Definition 2.1 [Syntax] Let a *(propositional) signature* Σ be the union of a family $\{\Sigma_m\}_{m\in\mathbb{N}}$ of constructors, where each set Σ_m contains only function symbols of arity m, and where Σ_i and Σ_j are assumed disjoint whenever $i \neq j$. Let $\mathcal{A}$ be a denumerable collection of symbols called *(atomic) variables*, assumed to be disjoint from the signature Σ. The nullary symbols in Σ_0 are sometimes called *truth symbols*. A *(propositional) language* $\mathcal{S}$ is recursively generated in the usual way, by considering its *composite formulae* to be of the form $\odot(\varphi_0, \varphi_1, \ldots, \varphi_m)$, for some $\odot \in \Sigma_{m+1}$. In the latter case, the symbol $\odot$ is said to be the *head connective* and the formulae φ_k, for $0 \leqslant k \leqslant m$, are dubbed the *immediate subformulae* of $\odot(\varphi_0, \varphi_1, \ldots, \varphi_m)$. The formulae in $\mathcal{A} \cup \Sigma_0$ are said to be *noncomposite*, and have no proper subformulae. A *canonical notion of formula complexity* $\mathsf{cp} : \mathcal{S} \longrightarrow \mathbb{N}$ may then be defined by setting $\mathsf{cp}(\varphi) = 0$ if φ is noncomposite, and $\mathsf{cp}(\varphi) = 1 + \mathsf{Max}_{0\leqslant k\leqslant m}\,\mathsf{cp}(\varphi_k)$ if φ is composite and the φ_k, for $0 \leqslant k \leqslant m$, are its immediate subformulae. A *uniform substitution* is an endomorphism on the set of formulae that maps each constructor into itself; it is uniquely defined as soon as the variables of the language are mapped into formulae. By $\varphi[p \mapsto \psi]$ we will denote the result of uniformly substituting ψ for each occurrence of the variable p in the formula φ. □

In the present study we will only consider languages generated by finite signatures.

Definition 2.2 [Logic] For a fixed propositional language $\mathcal{S}$, a *logic* $\mathcal{L}$ is here defined as a structure $\langle \mathcal{S}, \succ_{\mathcal{L}} \rangle$, where the relation $\succ_{\mathcal{L}} \subseteq 2^{\mathcal{S}} \times \mathcal{S}$ respects the following four abstract axioms, for arbitrary $\Gamma \cup \Delta \cup \{\varphi\} \subseteq \mathcal{S}$:

(C1) if $\varphi \in \Gamma$, then $\Gamma \succ_{\mathcal{L}} \varphi$

(C2) if $\Gamma \succ_{\mathcal{L}} \varphi$, then $\Gamma \cup \Delta \succ_{\mathcal{L}} \varphi$

(C3) if $\Gamma \succ_{\mathcal{L}} \delta$, for every $\delta \in \Delta$, and $\Delta \succ_{\mathcal{L}} \varphi$, then $\Gamma \succ_{\mathcal{L}} \varphi$

(C4) if $\Gamma \succ_{\mathcal{L}} \varphi$, then $\varepsilon(\Gamma) \succ_{\mathcal{L}} \varepsilon(\varphi)$, for any uniform substitution ε

Any $\succ_{\mathcal{L}}$ respecting the first three axioms is called a *Tarskian consequence relation*. We call *substitution-invariance* the property described by the last axiom. We say that the formulae α and β are *$\mathcal{L}$-equivalent*, and denote this by $\alpha \approx_{\mathcal{L}} \beta$, if both $\alpha \succ_{\mathcal{L}} \beta$ and $\beta \succ_{\mathcal{L}} \alpha$. The following *replacement property* may or may not be respected by a given logic, where φ denotes an arbitrary formula to be used as context, and p denotes an arbitrary variable taken as placeholder:

(C5) if $\alpha \approx_{\mathcal{L}} \beta$, then $\varphi[p \mapsto \alpha] \approx_{\mathcal{L}} \varphi[p \mapsto \beta]$

Logics that respect the latter axiom are called *congruential*. □

A very natural kind of denotational semantics for a logical language is one that employs certain distinctive collections of truth-values in defining the associated notions of entailment:

Definition 2.3 [Semantics] A *valuation* for a language $\mathcal{S}$ is a mapping $w : \mathcal{S} \longrightarrow \mathcal{V}_w$, where $\mathcal{V}_w$ denotes a nonempty collection of *truth-values* containing a subset $\mathcal{D}_w$ of *designated values*. The values in $\mathcal{V}_w \backslash \mathcal{D}_w$ are called *undesignated*. As usual, a valuation w is said to *satisfy* a formula φ if the former assigns a designated value to the latter; otherwise, we say that w *falsifies* φ. We will call $\mathcal{V}(\mathbf{2}) = \{F, T\}$ the set of *logical values*, for which we fix $\mathcal{D}(\mathbf{2}) = \{T\}$. Any valuation over $\mathcal{V}_{\mathbf{2}}$ is referred to as a *bivaluation*. The restriction $w|_{\mathcal{A}}$ of the valuation w to $\mathcal{A} \subseteq \mathcal{S}$ is called an *assignment* of truth-values to the variables. A *semantics* for $\mathcal{S}$ is here simply a family Sem of valuations. A *valid* formula (a.k.a. *tautology*) of a given semantics is one that is not falsified by any valuation from this semantics; a formula is called *unsatisfiable* by a given semantics if all the corresponding valuations falsify it. For a fixed semantics Sem and given some $\Delta \subseteq \mathcal{S}$, denote by $\mathsf{Mod}(\Delta)$ the set of valuations of Sem that map all formulae of Δ into designated values. A canonical *entailment* relation $\models_{\mathsf{Sem}} \subseteq 2^{\mathcal{S}} \times \mathcal{S}$ is then defined by setting $\Gamma \models_{\mathsf{Sem}} \varphi$ iff $\mathsf{Mod}(\Gamma) \subseteq \mathsf{Mod}(\{\varphi\})$, i.e., we say that Γ *entails* φ iff there is no valuation in Sem that simultaneously satisfies all formulae in Γ and falsifies the formula φ. □

It is easy to check that an entailment relation always respects axioms (C1), (C2) and (C3) of a Tarskian consequence relation.

Semantics may come in different flavours, differing on the way they happen to collect the appropriate valuations for a given language. It is interesting to observe that there is a precise sense in which all semantics of the kind entertained above may be said to have a 'bivalent character'. Indeed, fix a semantics Sem, and for each $w \in \mathsf{Sem}$ define the total mapping $b_w : \mathcal{S} \longrightarrow \mathcal{V}(\mathbf{2})$ such that $b_w(\varphi) = T$ iff $w(\varphi) \in \mathcal{D}_w$. Let $\mathsf{Sem}(\mathbf{2})$ denote the family $\{b_w\}_{w \in \mathsf{Sem}}$ of bivaluations. It is straightforward now to check that $\Gamma \models_{\mathsf{Sem}(\mathbf{2})} \varphi$ iff $\Gamma \models_{\mathsf{Sem}} \varphi$. This means that every 'multiple-valued logic' may also be given a characteristic bivalent semantics. Hereupon, we shall call $\mathsf{Sem}(\mathbf{2})$ the *bivalent reduction* of Sem.

The following definition takes advantage of structural features of the language in defining the associated semantics:

Definition 2.4 [Truth-functionality] For some fixed signature Σ and some fixed set $\mathcal{V}$ of truth-values, a Σ-*algebra* over $\mathcal{V}$ is a structure in which each $\odot \in \Sigma_m$, with $m \in \mathbb{N}$, is interpreted as an operation $\widehat{\odot} : \mathcal{V}^m \longrightarrow \mathcal{V}$ with the same arity. The very language $\mathcal{S}$ may be seen as a Σ-algebra, if we take $\mathcal{V} = \mathcal{S}$: this defines the so-called *term algebra* generated by the variables in $\mathcal{A}$ over the propositional signature Σ. If a semantics Sem is given by the set of all homomorphisms from a given term algebra into a fixed Σ-algebra $\mathbb{V}$ with carrier $\mathcal{V}$, mapping each constructor $\odot$ into the corresponding operation $\widehat{\odot}$, such semantics is said to be *truth-functional*. An entailment relation $\models_{\mathsf{Sem}}$ is immediately defined, as before, as soon as a certain set $\mathcal{D} \subseteq \mathcal{V}$ of designated values is fixed throughout the valuations constituting the semantics Sem. $\square$

Note that each valuation of a truth-functional semantics is uniquely defined by some assignment of truth-values to the atomic variables in $\mathcal{A}$, and the value of a composite formula φ under a valuation w is defined by the values given by the same valuation to the immediate subformulae of φ, so that $w(\odot(\varphi_0, \varphi_1, \ldots, \varphi_m)) = \widehat{\odot}(w(\varphi_0), w(\varphi_1), \ldots, w(\varphi_m))$. This is indeed the way in which a truth-functional semantics implements the so-called 'Principle of Compositionality of Meaning'.

It is clear that a truth-functional semantics gives rise to a substitution-invariant consequence relation. We shall call a logic $\langle \mathcal{S}, \succ \rangle$ *genuinely κ-valued* if κ is the cardinality of the smallest set of truth-values $\mathcal{V}$ over which a truth-functional semantics Sem may be based to the effect that $\models_{\mathsf{Sem}}$ and $\succ$ are coextensional. A *finite-valued logic* is any genuinely κ-valued logic for which the cardinal κ is finite. It is clear that the bivalent reduction described above still applies in this particular context, and that any logic with a truth-functional semantics may be characterised thus by a collection of bivaluations. In case a logic $\mathcal{L}$ is genuinely κ-valued, for some $\kappa > 2$, it is clear, though, that its bivalent reduction cannot be a truth-functional semantics.

To associate a convenient proof system to a logic described in semantic terms, a wise choice of a metalanguage can make a great difference. One rather straightforward way of describing a truth-functional semantics proceeds by the use of labelled formulae, and a corresponding extension of the associated interpretation: given $\varphi \in \mathcal{S}$, we say that a valuation $w : \mathcal{S} \longrightarrow \mathcal{V}$ *satisfies* $X{:}\varphi$ if $w(\varphi) = X$ (do bear in mind, though, that while the X in $w(\varphi) = X$ denotes a value from $\mathcal{V}$, the same symbol plays the role of a purely syntactical sign in $X{:}\varphi$). This is the approach taken by the so-called 'singletons-as-signs' labelling discipline; according to an alternative discipline called 'sets-as-signs', w is said to satisfy $\mathcal{X}{:}\varphi$ if $w(\varphi) \in \mathcal{X}$, where $\mathcal{X} \subseteq \mathcal{V}$. In such a metalanguage we will also introduce a meta-conjunction represented by &, a meta-disjunction represented by $||$, a meta-implication represented by $\Longrightarrow$, all with the expected Boolean interpretations, and statements such as $(X_a{:}\varphi_a \ \&\ X_b{:}\varphi_b) \Longrightarrow (X_c{:}\varphi_c \ ||\ X_d{:}\varphi_d)$ will then have their obvious reading as axioms imposing restrictions on the class of valuations in Sem, namely: if a valuation $w \in \mathsf{Sem}$ happens to satisfy both $X_a{:}\varphi_a$ and $X_b{:}\varphi_b$, then this valuation should satisfy $X_c{:}\varphi_c$ or $X_d{:}\varphi_d$. The meta-conjunctions and the meta-disjunctions are generalised in the standard way to give support to any finite number of arguments.

We introduce in the metalanguage the symbols $\oplus$ and $\ominus$, respectively, for meta-*verum* (standing for an arbitrary labelled tautology, or a 0-ary conjunction) and meta-*falsum* (standing for an arbitrary labelled antilogy, or a 0-ary disjunction), and write a statement such as $X{:}\varphi \Longrightarrow \ominus$ to say that $X{:}\varphi$ is unsatisfiable, and a statement such as $\oplus \Longrightarrow X{:}\varphi$ to say that $X{:}\varphi$ is unfalsifiable. Metalinguistic statements in which all labels are restricted to one of only two possible signs are said to be *bivalent*. As we will see, any finite-valued logic may be semantically characterised in terms of bivalent statements. We will assume $\mathcal{V}(\mathbf{2}) = \{F, T\}$ to be the set of labels used in case we are talking about bivalent statements, and shall write $X^{\mathbf{c}}$ to denote the *conjugate* of X, defined by setting $F^{\mathbf{c}} = T$ and $T^{\mathbf{c}} = F$. A natural classical meta-negation may also be introduced as we are working with bivalent statements: we will write $\overline{D}$ to say that statement D fails to be the case. In particular, on what concerns the interaction between the meta-negation and the labelled formulae, we will assume that $\overline{X{:}\varphi} = X^{\mathbf{c}}{:}\varphi$. For a bivalent semantics, whose valuations are all total functions on $\{F, T\}$, it should be clear that the following statements are always satisfied: $\oplus \Longrightarrow (X{:}\varphi \mid\mid X^{\mathbf{c}}{:}\varphi)$ and $(X{:}\varphi \;\&\; X^{\mathbf{c}}{:}\varphi) \Longrightarrow \ominus$.

Example 2.5 Let $\Sigma_0^A = \{\bot\}$ and $\Sigma_2^A = \{\supset\}$. Let $\mathcal{V}(\mathbb{Q})$ be the set of all rational numbers in the real-valued interval [0,1], and let $\mathcal{D} = \{1\}$. Assume $\widehat{\bot} = 0$ and $\widehat{\supset}(x, y) = \mathsf{min}(1, 1-x+y)$. As a side effect of such truth-functional interpretations, one might venture describing the behaviour of the implication $\supset$ by various metalinguistic statements such as $0{:}\varphi \mid\mid 1{:}\psi \Longrightarrow 1{:}(\varphi \supset \psi)$ or $\frac{1}{2}{:}\varphi \;\&\; \frac{1}{4}{:}\psi \Longrightarrow \frac{3}{4}{:}(\varphi \supset \psi)$. Note, however, that a labelled composite statement such as $\oplus \Longrightarrow 1{:}(\varphi \supset (\varphi \supset \psi)) \supset (\varphi \supset \psi)$ is satisfied iff $w(\varphi) = 0$ or $w(\varphi) = 1$. Now, the original signature may be extended by considering $\Sigma_1^B = \{\mathsf{id}, \neg, \theta_a, \theta_b\}$ and $\Sigma_2^B = \{\wedge, \vee\}$ and taking these new symbols to be abbreviations of formulae written in the original signature, namely: $\neg\varphi \overset{\text{def}}{=} \varphi \supset \bot$; $\mathsf{id}(\varphi) \overset{\text{def}}{=} \neg\bot \supset \varphi$; $\theta_a(\varphi) \overset{\text{def}}{=} \neg\varphi \supset \varphi$; $\theta_b(\varphi) \overset{\text{def}}{=} \varphi \supset \neg\varphi$; $\varphi \vee \psi \overset{\text{def}}{=} (\varphi \supset \psi) \supset \psi$; $\varphi \wedge \psi \overset{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi)$. As illustrations, notice, that while the metalinguistic statement $X{:}(\varphi \wedge \psi) \;\&\; 1{:}\varphi \Longrightarrow X{:}\psi$ is satisfied for every $X \in \mathbb{Q} \cap [0, 1]$ the statement $(X{:}\theta_a(\varphi) \mid\mid Y{:}\theta_b(\varphi)) \Longrightarrow (X{:}\varphi \;\&\; Y{:}\neg\varphi)$ is satisfied only for $X, Y \in \{0, 1\}$. One way of enforcing the validity of the latter statement is by replacing $\mathcal{V}(\mathbb{Q})$ by $\{0, 1\}$ (restricting thus the set of undesignated values to the singleton $\{0\}$). Łukasiewicz's logics Ł$_n$, for $n \geqslant 2$, are obtained if we replace $\mathcal{V}(\mathbb{Q})$ by $\mathcal{V}(n) = \left\{\frac{m}{n-1} : 0 \leqslant m \leqslant n-1\right\}$. Equivalently, to the same effect one could impose on the semantics the bivalent axiom $\oplus \Longrightarrow 1{:}\varphi \mid\mid 1{:}\neg\varphi$. In the above hierarchy of logics, Classical Logic (CL) corresponds to Ł$_2$ — thus, it not only has a bivalent semantics (one may set $b_w(\varphi) = T$ if $w(\varphi) = 1$, and $b_w(\varphi) = F$ otherwise) but is indeed a genuinely 2-valued logic. □

It is interesting to notice how, in the case of Ł$_2$, every bivalent statement may be rewritten in a useful way with the help of the negation connective: to that effect one just has to substitute '1:¬' for every '0:' that appears as prefix of a labelled formula, and confirm by induction that the resulting metalinguistic expression is satisfied iff the original metalinguistic expression is satisfied — so, in particular, $(0{:}\varphi \mid\mid 1{:}\psi) \Longrightarrow 1{:}(\varphi \supset \psi)$ becomes $(1{:}\neg\varphi \mid\mid 1{:}\psi) \Longrightarrow 1{:}(\varphi \supset \psi)$. This could give support to a natural argument for claiming that the addition of signs containing 'semantic information' to formulae is 'superfluous' (in the above example, one might consider just omitting the prefix '1:' that now appears in front of every object language expression), in view of the expressivity that the classical object language displays in internalising the classical metalinguistic information given by the labels. In general, of course, this argument runs unaltered only for CL. At any rate, there is nothing really special about *negation* in the preceding argument: the exact same impression of superfluity would in fact be caused by invoking the connective θ_b, whose interpretation only happens to coincide with that of negation over $\mathcal{V}(\mathbf{2})$.

It is not too hard to provide a combinatorial argument to show that Łukasiewicz's logics are so related that $\models_{Ł_m} \subseteq \models_{Ł_n}$ iff $n-1$ divides $m-1$. From the above example, however, it might be hard to tell the exact difference, say, between the 3-valued implication and the 5-valued implication just by looking at the statements that describe them, not least because of the use of different collections of labels to describe the semantics of $Ł_3$ and of $Ł_5$. A clever and generic way of solving this particular difficulty, in fact, would be by describing both logics under a classic-like logical framework, using a common signature and a bivalent semantics. The real problem, in that case, would then be how best to use our metalanguage to describe the corresponding non-truth-functional semantics in a computationally useful way.

There is a well-studied algorithmic approach ([4,5]) to producing a description of the 'bivalent reduction' of any finite-valued logic and to using the latter description so as to provide uniform tableau-theoretic characterisations of such a logic, with associated proof strategies that ensure, in each case, termination of the proof-search tasks, constructing either a proof or a counter-example to any given conjecture. This consists in a mechanised procedure in four steps that receives as input the full specification (syntax and semantics) of a finite-valued logic $\mathcal{L}$ and:

(A1) check if the object language of $\mathcal{L}$ is sufficiently expressive for the task;

(A2) produce a sufficiently expressive conservative extension of $\mathcal{L}$, if necessary;

(A3) axiomatise the semantics of $\mathcal{L}$ using bivalent statements expressed in the classical metalanguage under a particular (disjunctive) normal form;

(A4) use the bivalent statements to produce a sound and complete (tableau) proof system for $\mathcal{L}$.

In the present paper we will have our logics presented already in a language that is expressive enough for our purposes, thus bypassing steps **(A1)** and **(A2)**, where we have no contribution here to make. Next, we will import the original step **(A3)**, and then introduce a transformation function that helps in massaging the bivalent statements into a 'clause form' that is more appropriate to the use by the resolution method. Finally, we will redesign step **(A4)** in order to produce sound and complete uniform classic-like resolution calculi for all finite-valued logics.

3 The Bivalent Reduction

We briefly describe, in what follows, the bivalent statements produced by the above mentioned algorithm that outputs the bivalent reduction of any finite-valued logic. The key to the procedure is to find, first, a way of using the language of the given logic to distinguish between each pair of designated truth-values, and similarly to distinguish between each pair of undesignated truth-values. In that way, each value of the original truth-functional semantics will turn out to have a unique 'binary print' that sets it apart from all the other values. Such binary print will be expressed in terms of a suitable combination of logical values.

For the following definitions we fix a (sufficiently expressive) genuinely κ-valued logic $\mathcal{L}$, for finite κ, whose semantics Sem has $\mathcal{V}$ and $\mathcal{D}$, respectively, as its sets of truth-values and designated values. We recall that b_w denotes the bivaluation induced by the valuation $w \in \mathsf{Sem}$.

Definition 3.1 [Binary print] A *separating sequence* $\overrightarrow{\theta} = \langle \theta_0, \theta_1, \ldots, \theta_s \rangle$ for $\mathcal{L}$ is a sequence of unary connectives (to be referred to as *separators*) with the property that for every pair of distinct valuations $w_1, w_2 \in \mathsf{Sem}$, that is, every pair of valuations such that $w_1(p) \neq w_2(p)$ for some variable p, there is some θ_r, for $0 \leqslant r \leqslant s$, such that $b_{w_1}(\theta_r(p)) \neq b_{w_2}(\theta_r(p))$. Given some $x \in \mathcal{V}$ assigned by some valuation w to a given formula φ, by the *binary print* $\overrightarrow{\theta}(x)$ we mean the unique sequence $\overrightarrow{X} = \langle b_w(\theta_0(\varphi)), b_w(\theta_1(\varphi)), \ldots, b_w(\theta_s(\varphi)) \rangle \in \{F, T\}^{s+1}$. If an $(s+1)$-long sequence of F's and T's does not coincide with any binary print of a truth-value in $\mathcal{V}$, we say that such sequence *represents an absurd*. $\square$

Intuitively, an absurd corresponds to a *semantically unobtainable* scenario: an alleged logical description of an 'algebraic value' that is not to be found among the available truth-values of the Σ-algebra interpreting the language.

The following should be read having Ex. 2.5 on the background.

Example 3.2 Very short separation sequences are promptly available for Classical Logic. There are indeed in this case no pair of (un)designated values to distinguish. Thus, one could consider 1-long sequences made of any connective $\odot \in \Sigma_1^A \cup \Sigma_1^B$, in view of the fact that $b_{w_1}(\odot(p)) \neq b_{w_2}(\odot(p))$ whenever $w_1(p) = 0$ and $w_2(p) = 1$. Note, however, that $\langle \mathsf{id} \rangle$ is by itself *not* a separating sequence for Ł$_3$, for it does not tell the two undesignated values apart. Any 2-long sequence of distinct unary connectives from our extended signature would however do the job equally well for Ł$_3$. Separating sequences for other Ł$_n$'s are forcibly longer. For Ł$_5$, for instance, one could make do with the 4-long sequence $\langle \mathsf{id}, \neg, \theta_a, \theta_b \rangle$. $\square$

To simplify matters, we will choose henceforth to set $\theta_0(p) = p$. Strictly speaking, in this case θ_0 will not really denote a unary *connective*, but it will be just as good, as its interpretation coincides in fact with the interpretation of the 'identity connective' id. Obviously, $b_{w_1}(\theta_0(p)) \neq b_{w_2}(\theta_0(p))$ whenever $w_1(p)$ denotes a designated value, and $w_2(p)$ denotes an undesignated value. Thus, the other θ_r's (for $r > 0$) in the separating sequence have the role of allowing us to distinguish between two truth-values that are both designated, or between two truth-values that are both undesignated. The very existence of a separating sequence means that congruentiality (axiom (C5) in Def. 2.2) *always* fail for sufficiently expressive non-boolean truth-functional logics. Indeed, one can in general only be assured that two equivalent formulae may be replaced one by another *salva veritate* when every valuation assigns to them *exactly* the same truth-value, for otherwise there will be some sentential context that distinguishes these equivalent formulae. However, the replacement property is obviously fully enjoyed by our classical metalanguage, and we will in the future be using it often, at the meta-level, in replacing a labelled formula by any classically equivalent labelled formula.

The above definitions lead us now very naturally to an extension of the notion of formula complexity (contrast with the canonical complexity in Def. 2.1):

Definition 3.3 [Generalized measure of complexity] Formulae of the form $\theta_r(\varphi)$, where θ_r is a separator and φ is noncomposite, are called *basic*. A *generalised notion of formula complexity* $\mathsf{gcp} : \mathcal{S} \longrightarrow \mathbb{N}$ is defined by setting $\mathsf{gcp}(\varphi) = 0$ if φ is basic, and $\mathsf{gcp}(\varphi) = 1 + \mathsf{Max}_{0 \leqslant k \leqslant m} \mathsf{gcp}(\varphi_k)$, if φ has the form $\theta_r(\odot(\varphi_0, \varphi_1, \ldots, \varphi_m))$, for $\odot \in \Sigma_{m+1}$ and $0 \leqslant r \leqslant s$. Formulae with positive complexity gcp are said to be *analysable*. Such complexity measure may be extended to labelled formulae by setting $\mathsf{gcp}(X{:}\varphi) = \mathsf{gcp}(\varphi)$. □

Note that, in the present approach, neither labels nor separators contribute to an increase in formula complexity, and in general we have that $\mathsf{gcp}(\varphi) \leqslant \mathsf{cp}(\varphi)$.

Let $\overrightarrow{\theta}(X) = \langle X_0, X_1, \ldots, X_s\rangle$ be the binary print of some truth-value $X \in \mathcal{V}$. Note that there is a sense in which the generic metalinguistic statement $X{:}\varphi$ may be assumed to be *described* by the bivalent statement $\&_{r=0}^{s} X_r{:}\theta_r(\varphi)$. Indeed, the former statement is satisfied iff the latter is satisfied, modulo the respective labellings. In other words: $w(\varphi) = X$ iff $b_w(\theta_r(\varphi)) = X_r$ for every $0 \leqslant r \leqslant s$. We shall refer to $\&_{r=0}^{s} X_r{:}\theta_r(\varphi)$ as $V(\varphi, \overrightarrow{X})$. Considering next an analysable formula φ of the form $\theta_r(\odot(\varphi_0, \varphi_1, \ldots, \varphi_m))$ and a logical value X, we shall call $R_X^{\theta_r\odot}$ the set of all tuples of values from $\mathcal{V}$ that the subformulae $\varphi_0, \varphi_1, \ldots, \varphi_m$ may be assigned by a valuation $w \in \mathsf{Sem}$ in order to guarantee that $b_w(\varphi) = X$.

Aided by the separators, we now set ourselves the goal of describing the behaviour of analysable formulae in terms of formulae with lower generalised complexity measure. In particular, to each label $X \in \{F, T\}$, each separator $\theta_r \in \Sigma^\theta$ and each m-ary non-separator connective $\odot$ from the signature of $\mathcal{L}$ we will associate the following *B-statement*:

$$X{:}\theta_r(\odot(p_1, p_2, \ldots, p_m)) \quad \Longrightarrow \quad ||_{\langle x_1, x_2 \ldots, x_m\rangle \in R_X^{\theta_r\odot}} \, (\&_{k=1}^{m} V(p_k, \overrightarrow{\theta}(x_k))) \qquad (B_X^{\theta_r\odot})$$

In addition to the above, we should note that all the $(s+1)$-long binary sequences of F's and T's that do *not* correspond to binary prints of truth-values in $\mathcal{V}$ in fact describe *impossible* semantic scenarios. As it happens, however, it is often the case that partial knowledge about a given binary sequence provides enough information for us to conclude that it represents an absurd. In other words, maybe we do not know all elements of a given binary sequence at a certain stage of development of our reasoning, but we know enough to be able to conclude that any way of completing this sequence leads to an absurd. Accordingly, it is useful to entertain certain 'minimal unobtainable partial binary sequences'. The idea is to add a symbol for an 'undefined' logical value $\uparrow$ and call some sequence $\overrightarrow{Y} \in \{F, T, \uparrow\}^{s+1}$ *unobtainable* already when any $(s+1)$-long fully defined sequence of F's and T's that coincides with the former sequence in all positions that do not contain undefined symbols represents an absurd, that is, when no extension of $\overrightarrow{Y}$ represents an algebraic value from the original semantics. As expected, a *minimal* unobtainable such sequence is one that does not properly extend another unobtainable sequence.

Example 3.4 To illustrate the issue about partial binary sequences, note that if Ł$_5$ is considered over the separating sequence $\langle \mathsf{id}, \neg, \theta_a, \theta_b \rangle$, as in Ex. 3.2, its truth-values will be individualised by 5 out of 16 possible tuples of F's and T's. One (minimal) description of the 11 unobtainable tuples might be given by the following partial binary sequences: $\langle \uparrow, \uparrow, F, F \rangle$, $\langle \uparrow, T, \uparrow, F \rangle$, $\langle \uparrow, T, T, \uparrow \rangle$, $\langle T, \uparrow, F, \uparrow \rangle$, and $\langle T, \uparrow, \uparrow, T \rangle$. □

Given a minimal unobtainable sequence $\overrightarrow{Y} \in \{F, T, \uparrow\}^{s+1}$, by $\mathsf{dom}(\overrightarrow{Y})$ we denote the set $\{0 \leqslant r \leqslant s : Y_r \neq \uparrow\}$. To any such sequence $\overrightarrow{Y}$ we will associate the following *U-statement*:

$$\&_{r \in \mathsf{dom}(\overrightarrow{Y})} Y_r{:}\theta_r(p_0) \quad \Longrightarrow \quad \ominus \tag{U_Y}$$

We may say that a minimal unobtainable binary sequence $\overrightarrow{Y}$ *covers* any $(s+1)$-long binary sequence $\overrightarrow{X}$ such that $\mathsf{dom}(\overrightarrow{Y}) \subseteq \mathsf{dom}(\overrightarrow{X})$.

The above B-statements and U-statements allow us to describe a very convenient list of bivalent statements to use in characterising a given finite-valued logic.

Definition 3.5 [Bivalent statements induced by $\mathcal{L}$] Let $\mathcal{L}$ be a finite-valued logic over a signature Σ and a set of truth-values $\mathcal{V}$, let $\overrightarrow{\theta}$ be a separating sequence for $\mathcal{L}$, call $\Sigma^\theta \subseteq \Sigma_1$ the set of separators in $\overrightarrow{\theta}$, and let $\{Y_j\}_{j \in \lambda}$, for some $\lambda \in \mathbb{N}$, be a family of minimal unobtainable binary sequences that jointly cover all the binary prints that do not correspond to truth-values in $\mathcal{V}$. The bivalent semantics that we shall call $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ is described by the collection of all B-statements $B_X^{\theta_r \odot}$, for each label $X \in \{F, T\}$, for each $\theta_r \in \Sigma^\theta$ and each $\odot \in \Sigma \backslash \Sigma^\theta$, together with the collection of all U_{Y_j}-statements, for $j \in \lambda$. □

Note that $R_F^{\theta_r \odot} \cup R_T^{\theta_r \odot} = \mathcal{V}^{m+1}$, for $\odot \in \Sigma_m$ (and also that $R_F^{\theta_r \odot} \cap R_T^{\theta_r \odot} = \varnothing$), but it may occur that one of these two sets is empty. In that case, we will have an empty disjunction on the right-hand side of a B-statement $B_X^{\theta_r \odot}$, and this obviously amounts to a 'degenerate' B-statement that looks more like a U-statement, given that it describes a semantically unobtainable scenario.

$(U\langle T, F \rangle)$	$(T{:}p_0 \;\&\; F{:}\theta(p_0))$	$\Longrightarrow$	$\ominus$
$(B_T^{\perp})$	$T{:}\perp$	$\Longrightarrow$	$\ominus$
$(B_T^{\theta\perp})$	$T{:}\theta\perp$	$\Longrightarrow$	$\ominus$
$(B_F^{\theta\theta})$	$F{:}\theta(\theta(p_0))$	$\Longrightarrow$	$F{:}\theta(p_0)$
$(B_T^{\theta\theta})$	$T{:}\theta(\theta(p_0))$	$\Longrightarrow$	$T{:}\theta(p_0)$
$(B_F^{\supset})$	$F{:}p_0 \supset p_1$	$\Longrightarrow$	$(T{:}p_0 \;\&\; F{:}p_1) \mid\mid (T{:}\theta(p_0) \;\&\; F{:}\theta(p_1))$
$(B_T^{\supset})$	$T{:}p_0 \supset p_1$	$\Longrightarrow$	$(F{:}p_0 \;\&\; T{:}\theta(p_1)) \mid\mid F{:}\theta(p_0) \mid\mid T{:}p_1$
$(B_F^{\theta\supset})$	$F{:}\theta(p_0 \supset p_1)$	$\Longrightarrow$	$(T{:}p_0 \;\&\; F{:}\theta(p_1))$
$(B_T^{\theta\supset})$	$T{:}\theta(p_0 \supset p_1)$	$\Longrightarrow$	$F{:}p_0 \mid\mid T{:}\theta(p_1)$

Fig. 1. Set of bivalent axioms for Ł$_3$.

The following are among the main results of [5]:

Theorem 3.6 (Soundness & Completeness) *The bivalent semantics described by $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ contains the same bivaluations included in the bivalent reduction* $\mathsf{Sem}(\mathbf{2})$ *of $\mathcal{L}$.*

Theorem 3.7 (Effectiveness) *Let $b \in \mathsf{Sem}(\mathbf{2})$, and let φ be an analysable formula. Let $\mathcal{A}(\varphi)$ be the set of variables that occur in φ. Then the value of $b(\varphi)$ is uniquely determined from the values of $b(\theta_r(p))$ for $0 \leqslant r \leqslant s$ and $p \in \mathcal{A}(\varphi)$. Moreover, the value of $b(\varphi)$ may be effectively computed using the $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ statements.*

The perspicacious reader will surely have suspected that this provides a way of extracting from $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ the rules characterising an analytic tableau system. In the next section we will show how such description may be transformed into a description that is more appropriate for working with resolution calculi.

The following illustration builds on previous examples, and will be further explored in the next section.

Example 3.8 Fig. 1 contains a set of bivalent axioms for Ł$_3$ produced by the above described algorithm, using $\langle \theta_0, \theta_a \rangle$ as a separating sequence. For the sake of legibility, we omit the subscript in θ_a. □

4 Resolution Calculus

This section introduces a proof method for finite-valued logics through backward reasoning in the form of a clausal resolution-based proof method, a refutational procedure applied to formulae written in a specific Conjunctive Normal Form. Standard implementations of clausal resolution take a suitable normal form for the classical negation of a formula to be tested for satisfiability and then (a set of) rules based on the Resolution Principle are applied until the empty clause is found or no new clauses can be generated. If the empty clause is found, the original formula is unsatisfiable; otherwise, the formula is satisfiable and we can build a model that witnesses its satisfiability. We also take here the clausal approach, but instead of working from the original semantics of a given finite-valued logic, we use the bivalent semantics for such a logic, described in Section 3, to produce the normal form. This way, the calculus we shall present here works over a set of clauses in the metalanguage representing the bivalent semantics. The calculus is parametrised by the language and the separating sequence used to produce the metalinguistic statements for a given logic. We will denote by $\mathsf{RES}_{\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})}$ our resolution calculus for the bivalent statements $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ that describe the many-valued logic $\mathcal{L}$ with the separating sequence $\overrightarrow{\theta}$.

4.1 Normal Form

Let $\mathcal{L} = \langle \mathcal{S}, \succ_{\mathcal{L}} \rangle$, where $\mathcal{S}$ is recursively defined over a finite signature $\Sigma = \bigcup_{k \in \mathbb{N}} \Sigma_k$, be a fixed finite-valued logic with a truth-functional semantics Sem. Let $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ be the description of the bivalent semantics $\mathsf{Sem}(\mathbf{2})$ corresponding to our bivalent reduction of Sem.

From Def. 3.5 we note that the bivalent statements describing a given semantics have their right-hand side in Disjunctive Normal Form. However, clausal resolution works over meta-conjunctions of clauses (meta-disjunctions of labelled basic formulae) representing the Conjunctive Normal Form of a formula. More precisely, the normal form that is obtained from our bivalent statements, henceforth to be called CNF_{BS}, consists in metalinguistic statements of the form $\&_{i=0}^{a}(||_{j=0}^{b_i} X_{ij}{:}\varphi_{ij})$ where $a, b_i \in \mathbb{N}$, $X_{ij} \in \{T, F\}$ and each φ_{ij} is a basic formula. We note from Def. 3.3 that the disjuncts $X_{ij}{:}\varphi_{ij}$ cannot be further analysed, and play thus the role of 'literals' in classical resolution. Furthermore, as the meta-conjunction is associative, commutative and idempotent, we may treat a formula in CNF_{BS} more simply as a *set* of clauses.

There are alternative ways of producing the conjunctive normal form starting from the B-statements $B_X^{\theta_r \odot}$. One of these is to apply the usual distribution rules to such a metalinguistic formula, but that might give rise to an exponential increase in the size of the formula. Another way of transforming a formula into CNF_{BS} is to apply the meta-negation to formulae in $B_{X^{\mathbf{c}}}^{\theta_r \odot}$ followed by applications of De Morgan at the level of the metalanguage. However, it has been shown that 'language-preserving transformations' like these might lead to larger sets of clauses (cf. Ex. 4.6 in [2]) than the so-called 'structure-preserving transformations'. In contrast, the latter kind of transformations introduce abbreviations, in the form of new atomic variables and bi-implications, in order to help producing a normal form. This is the main approach chosen, for instance, in [7], and also entertained in [2]. In the present study, we combine both kinds of transformation, but taking advantage of the structure of B-statements, where the right-hand side of meta-implications are already meta-disjunctions, and we use *renaming* to replace analysable formulae appearing as disjuncts. Applying at the metalinguistic level the results from [9], if a formula φ is a subformula of ψ, that is, if ψ has the form $\psi[p \mapsto \varphi]$, then the labelled formula $X{:}\psi' \;\&\; X{:}(t_\varphi \Longrightarrow \varphi) \;\&\; X{:}(\varphi \Longrightarrow t_\varphi)$, where $\psi' = \psi[p \mapsto t_\varphi]$ for some fresh atomic variable t_φ, can be used to replace the formula $X{:}\psi$ whilst preserving satisfiability. The 'meta-bi-implication' $X{:}(t_\varphi \Longrightarrow \varphi) \;\&\; X{:}(\varphi \Longrightarrow t_\varphi)$ is often referred to as the *definition* of φ. In the present study, as in [7], we restrict renaming to formulae φ of positive polarity, that is, formulae that occur in the scope of an even number of meta-negations. Thus, only one side of the definition of φ is needed, namely the meta-implication $X{:}(t_\varphi \Longrightarrow \varphi)$. Again, satisfiability is preserved and the resulting normal form is shorter (cf. [9]). Differently from [7] and [2], the transformation into CNF_{BS} does not produce many-valued labelled formulae (either following the efficient labelling discipline 'sets-as-signs', or the straightforward 'singletons-as-signs'), but directly produces classic-like *two-signed formulae*. To some extent, our approach is closer to that in [8], where the satisfiability problem for many-valued logics is reduced to the satisfiability problem in classical propositional logic, but we confine our signs to the two labels that represent the underlying logical values. As a pleasant consequence, existing satisfiability procedures may be used to test a set of clauses without further ado. We note, however, that here we also need to take into consideration the U-statements, which provide the restrictions under which a meta-conjunction of labelled-formulae is meaningful. The U-statements will originate appropriate resolution-based rules, in Section 4.2, and the latter will turn out to be essential for the completeness of our method. In what follows we define the necessary prior transformation into CNF_{BS}.

Our transformation function τ takes a labelled formula $X{:}\varphi$ as input and converts it into a normal form through recursive applications of rewriting and renaming. To each B-statement of the form $X_0{:}\varphi_0 \Longrightarrow (||_{i=1}^{n} X_i{:}\varphi_i)$ we associate the following rewrite rule: $\tau(X_0{:}\varphi_0) \longmapsto \tau(||_{i=1}^{n} X_i{:}\varphi_i)$. If $\odot$ is the head connective in φ_0, we indicate the corresponding rewrite rule as $\tau^{\odot}_{X_0}$. The transformation distributes over meta-conjunctions and over meta-disjunctions, as follows:

$$\tau(\&_{i=1}^{n} X_i{:}\varphi_i) \longmapsto \&_{i=1}^{n} \tau(X_i{:}\varphi_i). \qquad (\tau^{\&})$$

$$\tau(||_{i=1}^{n} X_i{:}\varphi_i) \longmapsto ||_{i=1}^{n} \tau(X_i{:}\varphi_i). \qquad (\tau^{||})$$

The next rule renames meta-conjunctions that appear within meta-disjunctions:

$$\tau(\psi \mid\mid \&_{i=1}^{n} X_i{:}\varphi_i) \longmapsto \tau(\psi \mid\mid T{:}t) \;\&\; \tau(T{:}t \Longrightarrow \&_{i=1}^{n} X_i{:}\varphi_i) \qquad (\tau^{ren})$$

where ψ is a labelled formula, $\&_{i=1}^{n} X_i{:}\varphi_i$ is a meta-conjunction and t is a fresh atomic variable. Note that meta-disjunctions are associative and commutative. Thus, τ^{ren} applies to any disjunct that is not a labelled basic formula. Meta-conjunctions on the right-hand side of meta-implications are rewritten as usual:

$$\tau(T{:}t \Longrightarrow \&_{i=1}^{n} X_i{:}\varphi_i) \longmapsto \&_{i=1}^{n} \tau(T{:}t \Longrightarrow \tau(X_i{:}\varphi_i)) \qquad (\tau^{\Longrightarrow\&})$$

The final rewrite rule transforms meta-implications into meta-disjunctions:

$$\tau(T{:}t \Longrightarrow D) \longmapsto \begin{cases} F{:}t \mid\mid D, & \text{if } D \text{ is a clause} \\ \tau(T{:}p \Longrightarrow \tau(D)), & \text{otherwise} \end{cases} \qquad (\tau^{\Longrightarrow})$$

As the base case for the transformation function, we set:

$$\tau(X{:}\varphi) \longmapsto X{:}\varphi, \text{ if } \varphi \text{ is a basic formula.} \qquad (\tau^{b})$$

Clauses are kept in simplified form, i.e. the following simplification rules apply at all steps of the transformation (where D is a clause and φ is a basic formula):

$$\begin{array}{lllllll} \sigma(D \mid\mid X{:}\varphi \mid\mid X{:}\varphi) & \longmapsto & \sigma(D \mid\mid X{:}\varphi) & \quad & \sigma(D \mid\mid \oplus) & \longmapsto & \oplus \\ \sigma(D \mid\mid X{:}\varphi \mid\mid X^{\mathbf{c}}{:}\varphi) & \longmapsto & \oplus & & \sigma(D \mid\mid \ominus) & \longmapsto & \sigma(D) \end{array}$$

The following lemma shows that the transformation into CNF_{BS} is correct.

Lemma 4.1 *Let $\mathcal{L}$ be a finite-valued logic and $\mathcal{S}$ be its language. Let $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ be the set of bivalent statements associated with $\mathcal{L}$ and the separating sequence $\overrightarrow{\theta}$, and let φ be a formula in $\mathcal{S}$. Then, φ is satisfiable if, and only if, $\tau(T{:}\varphi)$ is satisfiable.*

Proof. The rewrite rules given by τ are based on the equivalences given in [5] (case of $\tau^{\odot}_{X_0}$), on classical equivalences for all other rewrite rules ($\tau^{\&}$, $\tau^{||}$, $\tau^{\Longrightarrow\&}$, and $\tau^{\Longrightarrow}$), and on classical renaming for the transformation rule τ^{ren}. As both kinds of transformation, namely replacement and renaming, preserve satisfiability (cf. [9]), we conclude that φ is satisfiable if and only if $\tau(T{:}\varphi)$ is also satisfiable. □

1. $T{:}t_1 \parallel T{:}t_9$
2. $F{:}t_1 \parallel T{:}t_2 \parallel T{:}t_8 \parallel T{:}p$
3. $F{:}t_1 \parallel F{:}p$
4. $F{:}t_2 \parallel T{:}t_3 \parallel T{:}t_6$
5. $F{:}t_2 \parallel T{:}\theta(p)$
6. $F{:}t_3 \parallel T{:}p$
7. $F{:}t_3 \parallel T{:}t_4 \parallel T{:}t_5$
8. $F{:}t_4 \parallel T{:}p$
9. $F{:}t_5 \parallel \theta(p)$
10. $F{:}t_6 \parallel \theta(p)$
11. $F{:}t_6 \parallel T{:}t_7$
12. $F{:}t_7 \parallel T{:}p$
13. $F{:}t_8 \parallel T{:}p$
14. $F{:}t_8 \parallel T{:}t_7$
15. $F{:}t_9 \parallel T{:}t_3 \parallel T{:}t_6 \parallel \theta(p)$
16. $F{:}t_9 \parallel F{:}\theta(p)$

where

New Variable	Renames
t_1	$T{:}((p \supset (p \supset \bot)) \supset p)\ \&\ F{:}p$
t_2	$F{:}(p \supset (p \supset \bot))\ \&\ T{:}\theta(p)$
t_3	$T{:}p\ \&\ F{:}(p \supset \bot)$
t_4	$T{:}p\ \&\ F{:}\bot$
t_5	$T{:}\theta(p)\ \&\ F{:}\theta(\bot)$
t_6	$T{:}\theta(p)\ \&\ F{:}\theta(p \supset \bot)$
t_7	$T{:}p\ \&\ F{:}\theta(\bot)$
t_8	$T{:}p\ \&\ F{:}\theta(p \supset \bot)$
t_9	$T{:}\theta((p \supset (p \supset \bot)) \supset p)\ \&\ F{:}\theta(p)$

Fig. 2. The clausal form of $(((p \supset (p \supset \bot)) \supset p) \supset p)$.

Example 4.2 The formula $\varphi_0 = (((p \supset (p \supset \bot)) \supset p) \supset p)$ is valid in Ł$_3$. Fig. 2 shows the set of clauses resulting from the transformation of $F{:}\varphi_0$ into CNF$_{BS}$, where the transformation function is based on the set of bivalent statements for Ł$_3$ with separating sequence $\langle \theta_0, \theta_a \rangle$, given in Fig. 1. Tautologies have been suppressed. On the right-hand side, we present the definitions of the new atomic variables introduced along the translation. $\square$

4.2 Inference Rules

Let $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ be the bivalent description of a finite-valued logic $\mathcal{L}$ on a language $\mathcal{S}$, and let $\{U_j\}_{j \in \lambda}$, for some $\lambda \in \mathbb{N}$, be the family of U-statements in this description. Let $X{:}\varphi$ be a labelled formula with $\varphi \in \mathcal{S}$. Let Φ be the set of clauses obtained by transforming $X{:}\varphi$ into the corresponding CNF$_{BS}$ based on $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$. The resolution calculus for $\mathcal{L}$ will comprise a binary resolution rule (RES), which is a syntactical variation of the usual classical (binary) resolution rule [11], and a set of hyper-resolution inference rules, named (RES$_{U_j}$), for $j \in \lambda$, to deal with the valuation restrictions related to the corresponding U_j-statements. Hyper-resolution (cf. [10]) is a refinement of the resolution method, which combines several binary resolution steps, thus avoiding the generation of intermediate clauses (and the interaction between them) when searching for a proof. Recall that U-statements express the binary sequences which are unobtainable as binary prints of the original truth-values of $\mathcal{L}$. Hence, the meta-conjunction of labelled formulae on the left-hand side of the meta-implication in a U-statement corresponds to a logical absurdity in the semantics of the metalanguage. The resolution rules are given in Fig. 3, where D_i is a clause, $X_i \in \{T, F\}$, and φ, φ_i are basic formulae (where $1 \leqslant i \leqslant n_j$, $j \in \lambda$, and n_j is the number of meta-conjuncts in U_j). Premises of the resolution-based inference rules are called *parent clauses*. Conclusions in such rules are called *resolvents*. Also, we refer to $\{T{:}\varphi, F{:}\varphi\}$ (resp. $\{X_1{:}\varphi_1, \ldots, X_{n_j}{:}\varphi_{n_j}\}$) occurring in the parent clauses of (RES) (resp. (RES$_{U_j}$)) as an *inconsistent set of formulae*. Labelled formulae in an inconsistent set of formulae are said to be *resolved* by the respective inference rule.

$$(\mathsf{RES})\quad \dfrac{\begin{array}{l} D_1 \mid\mid T{:}\varphi \\ D_2 \mid\mid F{:}\varphi \end{array}}{D_1 \mid\mid D_2}
\qquad\qquad
(\mathsf{RES}_{U_j})\quad \dfrac{\begin{array}{c} D_1 \;\mid\mid\; X_1{:}\varphi_1 \\ \vdots \\ D_{n_j} \;\mid\mid\; X_{n_j}{:}\varphi_{n_j} \end{array}}{D_1 \mid\mid \ldots \mid\mid D_{n_j}}$$

$$\text{for each } U_j = X_1{:}\varphi_1 \& \ldots \& X_{n_j}{:}\varphi_{n_j} \Longrightarrow \ominus$$

Fig. 3. Resolution-based Rules for $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$.

Definition 4.3 [Derivations & Refutations] Let Φ be a set of clauses in CNF_{BS}. A *derivation for* Φ is a sequence $\Phi_0, \Phi_1, \ldots$ of sets of clauses in CNF_{BS}, with $\Phi_0 = \Phi$ and for all $i > 0$, $\Phi_{i+1} = \Phi_i \cup \{D\}$, where D is a clause (in simplified form) obtained by an application of either (RES) or (RES_{U_j}) in $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$ to clauses in Φ_i. Repeated formulae are not added to the next step in a derivation, i.e. we require that $(\Phi_{i+1} \backslash \Phi) \neq \varnothing$, and that D is not a tautology. A *refutation* for a set of clauses Φ is a finite derivation for Φ, $\Phi_0, \ldots, \Phi_m$, with $m \in \mathbb{N}$, such that $\ominus \in \Phi_m$. □

Next, we show an example of the application of the method.

Example 4.4 The formula $(((p \supset (p \supset \bot)) \supset p) \supset p)$ is valid in Ł$_3$. Fig. 4 shows a refutation for the set of clauses resulting from the transformation of $F{:}(((p \supset (p \supset \bot)) \supset p) \supset p)$ into CNF_{BS} given in Example 4.2. As shown in Table 1, there is only one U-statement for the bivalent semantics of Ł$_3$, namely, $(T{:}\varphi \;\&\; F{:}\theta(\varphi)) \Longrightarrow \ominus$. Thus, the corresponding hyper-resolution rule has the form: from $D \mid\mid T{:}\varphi$ and $D' \mid\mid F{:}\theta(\varphi)$ infer $D \mid\mid D'$, where D and D' are clauses and φ is a formula of Ł$_3$. In the following, we only show the clauses that are relevant in the refutation. Justification is given on the right-hand side: Numbers refer to the corresponding parent clauses and the application of the hyper-resolution rule is indicated by U (otherwise, (RES) is applied). We also indicate in each case the list of labelled formulae being resolved.

A derivation *terminates* iff either the empty clause is derived or no new clauses can be derived by further application of the resolution rules of $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$. From Def. 4.3, recall that we do not add repeated clauses nor tautologies to the generated set of clauses. These restrictions, together with simplification, are usual in resolution-based proof methods and help to establish termination, as follows.

Theorem 4.5 (Termination) *Let Φ be a set of clauses in CNF_{BS}. Then any derivation for Φ in $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$ terminates.*

Proof. Firstly, note that none of the inference rules introduce new labelled basic formulae in the clause set. As there are only a finite number of such formulae occurring in Φ, only a finite number of CNF_{BS} clauses may be built. Formally, if n is the number of basic formulae occurring in Φ, there are only 3^n possible different meta-disjunctions (modulo reordering) that can be built: either the labelled basic formula $X{:}\varphi$ does not occur in the clause, or it occurs in the form $T{:}\varphi$, or else it occurs in the form $F{:}\varphi$ (as meta-disjunctions are in simplified form). Also, for any derivation $\Phi_0, \Phi_1, \ldots$ for Φ, by the definition of a derivation, we have that Φ_{i+1} must contain a new clause as compared to Φ_i. Thus, there must be some $m \in \mathbb{N}$

1.	$T{:}t_1 \mid\mid T{:}t_9$	$[assumption]$
2.	$F{:}t_1 \mid\mid T{:}t_2 \mid\mid T{:}t_8 \mid\mid T{:}p$	$[assumption]$
3.	$F{:}t_1 \mid\mid F{:}p$	$[assumption]$
4.	$F{:}t_2 \mid\mid T{:}t_3 \mid\mid T{:}t_6$	$[assumption]$
5.	$F{:}t_2 \mid\mid T{:}\theta(p)$	$[assumption]$
6.	$F{:}t_3 \mid\mid T{:}p$	$[assumption]$
10.	$F{:}t_6 \mid\mid T{:}\theta(p)$	$[assumption]$
11.	$F{:}t_6 \mid\mid T{:}t_7$	$[assumption]$
12.	$F{:}t_7 \mid\mid T{:}p$	$[assumption]$
13.	$F{:}t_8 \mid\mid T{:}p$	$[assumption]$
15.	$F{:}t_9 \mid\mid T{:}t_3 \mid\mid T{:}t_6 \mid\mid T{:}\theta(p)$	$[assumption]$
16.	$F{:}t_9 \mid\mid F{:}\theta(p)$	$[assumption]$
17.	$F{:}t_7 \mid\mid F{:}t_9$	$[12, 16, U, T{:}p, F{:}\theta(p)]$
18.	$F{:}t_3 \mid\mid F{:}t_9$	$[6, 16, U, T{:}p, F{:}\theta(p)]$
19.	$T{:}t_2 \mid\mid T{:}t_8 \mid\mid T{:}p \mid\mid T{:}t_9$	$[2, 1, t_1]$
20.	$F{:}p \mid\mid T{:}t_9$	$[3, 1, t_1]$
21.	$F{:}\theta(p) \mid\mid T{:}t_1$	$[16, 1, t_9]$
22.	$T{:}t_2 \mid\mid T{:}p \mid\mid T{:}t_9$	$[19, 13, t_8]$
23.	$T{:}t_2 \mid\mid T{:}t_9$	$[22, 20, p]$
24.	$T{:}t_9 \mid\mid T{:}\theta(p)$	$[23, 5, t_2]$
25.	$T{:}t_2 \mid\mid F{:}\theta(p)$	$[23, 16, t_9]$
26.	$T{:}\theta(p) \mid\mid T{:}t_3 \mid\mid T{:}t_6$	$[24, 15, t_9]$
27.	$T{:}\theta(p) \mid\mid F{:}t_3$	$[24, 18, t_9]$
28.	$T{:}\theta(p) \mid\mid T{:}t_6$	$[26, 27, t_3]$
29.	$T{:}\theta(p)$	$[28, 10, t_6]$
30.	$T{:}t_2$	$[29, 25, \theta(p)]$
31.	$T{:}t_1$	$[29, 21, \theta(p)]$
32.	$T{:}t_3 \mid\mid T{:}t_6$	$[30, 4, t_2]$
33.	$F{:}p$	$[31, 3, t_1]$
34.	$T{:}t_6 \mid\mid T{:}p$	$[32, 6, t_3]$
35.	$T{:}t_6$	$[34, 33, p]$
36.	$T{:}t_7$	$[35, 11, t_6]$
37.	$T{:}p$	$[36, 12, t_7]$
38.	$T{:}t_9$	$[37, 20, p]$
39.	$F{:}t_7$	$[38, 17, t_9]$
40.	$\ominus$	$[39, 36, t_7]$

□

Fig. 4. A refutation for $(((p \supset (p \supset \bot)) \supset p) \supset p)$.

such that either the empty clause is in Φ_m, or $|\Phi_m| \leqslant 3^n$ and no new further clauses can be generated. Therefore, any derivation for Φ in $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$ terminates. □

The next result establishes soundness of the resolution method.

Theorem 4.6 (Soundness) *The resolution calculus $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$ for $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ based on the rules* (RES) *and* (RES_{U_j})*, for each U_j-statement in $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$, is sound.*

Proof. We start by establishing the soundness of each inference rule in $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$. Soundness of (RES) follows from the results in [11]. For the soundness of (RES_{U_j}), assume all premises hold. Then, by definition of satisfiability, the meta-conjunction of those premises is also satisfiable. It is a straightforward exercise to show that $\&_{i=1}^{n_j}(D_i \mid\mid X_i{:}\varphi_i)$ is semantically equivalent to $(\&_{i=1}^{n_j} \overline{D_i}) \Longrightarrow (\&_{i=1}^{n_j} X_i{:}\varphi_i)$, where $\overline{D_i}$ is the meta-negation of D_i. From the results in [5], by the U_j-statement $\&_{i=1}^{n_j} X_i{:}\varphi_i \Longrightarrow \ominus$, we have that $\&_{i=1}^{n_j} X_i{:}\varphi_i$ is unsatisfiable. Thus, by the semantics of the meta-implication, we conclude that $\&_{i=1}^{n_j} \overline{D_i}$ is unsatisfiable, and by the semantics of the meta-conjunction and meta-negation, we obtain that $\mid\mid_{i=1}^{n_j} D_i$, the resolvent of (RES_{U_j}), is satisfiable. Now, let Φ be a satisfiable set of clauses and let $\Phi_0, \Phi_1, \ldots$ be a derivation for Φ. As all inference rules are sound, by an easy induction on the length of a derivation, every Φ_i, $i \geqslant 0$, is also satisfiable. Thus, the resolution calculus based on (RES) and (RES_{U_j}) is sound. □

As expected, the completeness result only depends on showing that the hyper-resolution rule can be simulated by binary resolution.

Lemma 4.7 *Let $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ be the bivalent semantics of $\mathcal{L}$ with separating sequence $\overrightarrow{\theta}$ and $\mathsf{RES}_{\mathcal{B}(\mathcal{L},\overrightarrow{\theta})}$ the resolution calculus based on $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$. Let (RES_{U_j}) be the hyper-resolution rule corresponding to the U_j-statement $\&_{i=1}^{n_j} X_i{:}\varphi_i \Longrightarrow \ominus$ in $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$. Then, (RES_{U_j}) can be simulated by binary resolution.*

Proof. The proof is straightforward. Assume Φ is the set of clauses $\{(D_1 \mid\mid X_1{:}\varphi_1), (D_2 \mid\mid X_2{:}\varphi_2), \ldots, (D_{n_j} \mid\mid X_{n_j}{:}\varphi_{n_j})\}$, i.e. the premises in (RES_{U_j}). Note that the meta-negation of $\&_{i=1}^{n_j} X_i{:}\varphi_i$ corresponds to the unfalsifiable meta-disjunction $\mid\mid_{i=1}^{n_j} X_i^{\mathbf{c}}{:}\varphi_i$. Call the latter statement D. Take $\Phi' = \Phi \cup \{D\}$. Construct a derivation $\Phi'_0, \Phi'_1, \ldots$ for Φ' by taking $\Phi'_0 = \Phi'$, $\Phi'_1 = \Phi'_0 \cup \{D'_1\}$, where D'_1 is the resolvent of D and $(D_1 \mid\mid X_1{:}\varphi_1)$ by (RES). For the remainder of the construction, take $\Phi'_{i+1} = \Phi'_i \cup \{D'_{i+1}\}$, where D'_{i+1} is the resolvent of D'_i and $(D_{i+1} \mid\mid X_{i+1}{:}\varphi_{i+1})$ by (RES). Thus, Φ'_{n_j} contains $\mid\mid_{i=1}^{n_j} D_i$, the same clause as the resolvent of (RES_{U_j}) applied to Φ. □

Theorem 4.8 (Completeness) *Let $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$ be the bivalent semantics of $\mathcal{L}$ with separating sequence $\overrightarrow{\theta}$. The resolution calculus, $\mathsf{RES}_{\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})}$, based on (RES) and the family of rules (RES_{U_j}), for each U_j-statement in $\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})$, is complete.*

Proof. Immediate from Lemma 4.7 and the completeness of the resolution calculus for propositional logic [11]. □

5 Final Remarks

We have developed a sound, complete, and terminating resolution-based proof method for finite-valued logics. In particular, the correctness results were easily achieved as they rely on the satisfiability preservation of the normal form based on bivalent semantics, and on syntactical variations of usual resolution-based rules. Also, the fact that a set of formulae $\Gamma = \{\gamma_0, \gamma_1, \ldots, \gamma_n\}$ entails φ in the particular many-valued logic under consideration can be translated into the unsatisfiability of $\&_{i=0}^{n} \tau(T{:}\gamma_i) \;\&\; \tau(F{:}\varphi)$ in our metalanguage. Hence, strong completeness of our method easily follows from the results offered in the present study.

From our correctness results, a proof in $\mathsf{RES}_{\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})}$ may be simulated by ordinary propositional resolution, by using the unfalsifiable formulae originated from the unobtainable partial binary sequences (see Lemma 4.7) and the translation of the set of clauses into the ordinary propositional language, and by then removing labels from metalinguistic formulae and introducing classical meta-negations where appropriate. Also, formulae of the form $\theta(\varphi)$, where θ is a separator and φ is basic, are not analysable and may be regarded as *ground terms*, translatable into new atomic symbols. It should thus be clear that automatisation of $\mathsf{RES}_{\mathcal{B}(\mathcal{L}, \overrightarrow{\theta})}$ only requires representation into the language of an existing propositional theorem-prover.

The efficiency of our transformation procedure, as compared, for instance, to those of [1] and [7], depends strongly on the size of the B-statements, which in turn depends on the set of separating sequences chosen to generate such statements as well as the number and arities of connectives in the object-language. Like the transformation of many-valued formulae into singleton-as-signs given in [1], the procedure given here leads to a clause set that is exponential on the size of the original formula. On the bright side, as argued above, the transformation we present produces truly classical formulae in the metalanguage and there is no need to perform unification on labels. The more parsimonious set of clauses obtained in [7] under the discipline sets-as-signs takes advantage of the 'inverse tableau' procedure (where a branch represents the disjunction of formulae occurring therewith) in order to generate the clausal form. Believing that such procedure can be derived from the optimized linear cut-based tableau construction given in [5], we leave this here as matter for further investigation. The main distinguishing feature of the method introduced in the present paper lies in the target language to which the formulae are translated. As noted before, the resolution-based procedure presented here might produce clauses which turn out inconsistent in the many-valued logic given as source, but which are satisfiable in the target classical metalanguage. Accordingly, our method requires the hyper-resolution rules corresponding to the U-statements in order to make the proof method complete. The procedures presented in [1,7] do not produce such clauses, as unification on labels ensures that resolvents are always labelled by "meaningful" sets.

The extension of the above study to first-order logics endowed with distribution quantifiers does not seem so much of a technical challenge, but rather depends on previously extending the underlying reduction algorithm to the first-order case. A more interesting challenge —which we know how to solve, but the solution does not fit here in the margin— is the extension of the above study to cover logics with finite-valued nondeterministic matrices.

References

[1] Baaz, M. and C. G. Fermüller, *Resolution for many-valued logics*, Lecture Notes in Computer Science **624** (1992), pp. 107–118.

[2] Baaz, M., C. G. Fermüller and G. Salzer, *Automated deduction for many-valued logics*, in: J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, **II**, Elsevier Science, 2001 pp. 1355–1402.

[3] Caleiro, C., W. A. Carnielli, M. E. Coniglio and J. Marcos, *Suszko's Thesis and dyadic semantics*, Research report, CLC/IST, 2003.
URL http://sqig.math.ist.utl.pt/pub/CaleiroC/03-CCCM-dyadic1.pdf

[4] Caleiro, C. and J. Marcos, *Many-valuedness meets bivalence: Using logical values in an effective way*, Journal of Multiple-Valued Logic and Soft Computing **19** (2012), pp. 51–70.

[5] Caleiro, C., J. Marcos and M. Volpe, *Bivalent semantics, generalized compositionality and analytic classic-like tableaux for finite-valued logics*, Theoretical Computer Science **603** (2015), pp. 84–110.

[6] Hähnle, R., "Automated Deduction in Multiple-Valued Logics," Oxford University Press, 1993.

[7] Hähnle, R., *Short conjunctive normal forms in finitely valued logics*, Journal of Logic and Computation **4** (1994), pp. 905–927.

[8] Kovács, L., A. Mantsivoda and A. Voronkov, *The inverse method for many-valued logics*, in: F. Castro, A. Gelbukh and M. González, editors, *Advances in Artificial Intelligence and Its Applications*, Lecture Notes in Computer Science **8265**, Springer Berlin Heidelberg, 2013 pp. 12–23.

[9] Plaisted, D. A. and S. A. Greenbaum, *A Structure-Preserving Clause Form Translation*, Journal of Logic and Computation **2** (1986), pp. 293–304.

[10] Robinson, J. A., *Automatic deduction with hyper-resolution*, International Jounal of Computer Mathematics **1** (1965), pp. 227–234.

[11] Robinson, J. A., *A Machine–Oriented Logic Based on the Resolution Principle*, Journal of the ACM **12** (1965), pp. 23–41.

[12] Zach, R., "Proof Theory of Finite-Valued Logics," Ph.D. thesis, TU-Wien (1993).