Log. Univers. 16 (2022), 195–219

1661-8297/22/010195-25, *published online* February 16, 2022
https://doi.org/10.1007/s11787-022-00293-w

Logica Universalis

Tableaux for Logics of Content Relationship and Set-Assignment Semantics

Tomasz Jarmużek and Mateusz Klonowski

Abstract. In the paper, we examine tableau systems for R. Epstein's logics of content relationship: **D** (Dependence Logic), **DD** (Dual Dependence Logic), **Eq** (Logic of Equality of Content), **S** (Symmetric Relatedness Logic) and **R** (Nonsymmetric Relatedness Logic) (Epstein in The semantic foundations of Logic, Springer Science + Business Media, Dordrecht, (1990), cf. Epstein in Philos Stud 36:137–173, 1979, Epstein in Rep. Math. Logic 21:19–34, 1987, Klonowski in Logic Log Philos, accepted for publication, Krajewski in J Non Class Logic 8:7–33, 1991). The first tableau systems for those logics were defined by Carnielli (Rep Math Logic 21:35–46, 1987). However, his approach has some limitations, for example, it requires a proof of functional completeness and axiomatization. Notwithstanding the first two constraints, it does not include all Epstein logics, e.g., logic **Eq**. Unlike Carnielli's approach, here we use set-assignment semantics to determine those logics. Since syntax and semantics of a given logic usually determine a minimal syntax and structure of a tableau system for the logic along with other properties, we propose a uniform tableau framework for the logics determined by set-assignment semantics. What distinguishes our tableau systems is that they combine the features of tableaux for propositional logics and syllogistic logics when the problem of content of propositions is analysed in tableau proofs. To denote the content of propositions in the proofs, we use generalised labels (explored in the syllogistic context in Jarmużek and Goré (In: Fitting (ed.) Landscapes in Logic, College Publications, London, accepted)).

Mathematics Subject Classification. Primary 03-02; Secondary 03B20, 03B60, 03C50, 03C90, 03F03.

Keywords. Generalized labels, Logic of content relationship, Relating semantics, Set-assignment semantics, Tableaux for syllogistic, Uniform tableau framework.

The part of research made by Tomasz Jarmużek presented in the following article was financed by National Science Centre, Poland, number of grant: 2015/19/B/HS1/02478. The authors also would like to thank Maria Martinez-Ordaz for her remarks and suggestions.

1. Introduction

The aim of this paper is to introduce tableau systems for some logics of content relationship that were proposed by R. Epstein ([6], cf. [4,5,16,17]); these logics are: **D** (Dependence Logic), **DD** (Dual Dependence Logic), **Eq** (Logic of Equality of Content), **S** (Symmetric Relatedness Logic), and **R** (Nonsymmetric Relatedness Logic).

The significance of the paper lies in the fact that, while some tableau systems for Epstein logics have been previously proposed by W. Carnielli [1], his approach faces serious limitations.[1] For starters, it requires a proof of functional completeness and axiomatization. Furthermore, Carnielli's approach does not include logic **Eq**. And, in the case of tableau rules for **D**, there are some problems regarding completeness.[2]

In order to avoid these issues, here, we propose to examine Epstein's logics from the perspective of his *set-assignment semantics*. Since syntax and semantics of a given logic usually determine a minimal syntax and structure of a tableau system for the logic along with other properties, we propose a uniform tableau framework. What would be characteristic of the tableau systems that we present in this paper is that, when the problem of content of propositions is analysed in tableau proofs, our systems can combine the features of tableaux for propositional logics and syllogistic logics. In proofs, we use generalised labels (explored in the syllogistic context in [14]) to denote content of propositions. The combination of these features makes our proposal more robust and stronger than Carnielli's and other alternatives.

The paper consists of three parts. In the first one, we present Epstein's logics of content relationship. In the second part, we propose tableau systems for all considered logics and we provide the proof of the adequacy of each system. In the appendix, we consider some examples of tableau proofs to illustrate the applications of the introduced rules.

2. Epstein's Logics of Content Relationship

The interest in the problem of how to interpret the implication connective was already present in ancient times; but, nowadays, such an interest has increased to the point in which the formal analysis of conditional sentences constitutes the basis of many non-classical logics. The formal analysis of the implication often begins with the examination of a particular instance of this connective, the material implication and its paradoxical laws or properties; and it moves forward onto the many (non-classical) attempts to block or remove these paradoxes. It is worth noticing that the large majority of enterprises that

[1] Epstein's logics of content are also the logic **DPC** (Classically-Dependent Logic) and logics defined based on the **DPC** consequence relation.

[2] In Carnielli's proof of derivability of his rule (8) for logic **D** (see [1, pp. 44–45]) a fact (a counterpart for **D** of theorem 2 (Cut Rule), see [1, p. 41]) is used. But the proof of this fact seems to require application of the rule (8).

pursue this are motivated by phenomena other than the way in which classical logical values might influence the logical value of the implication.

An instance of these phenomena would be the content relationship of sentences. Take for example the case of a conditional sentence, it seems sensible to say that for such a sentence to be true it is not enough for either the antecedent to be false or the consequent to be true. As a matter of fact, we would expect that the truth of the sentence depends, at least partially, on the antecedent being related to the consequent content wise. And while this intuition has been well-accepted among contemporary logicians, there has been a significant discussion regarding how to formally represent and analyse this content relationship. We will focus on Richard Epstein's [6] formal approach to content relationship (cf. [4,5,16,17]).

2.1. Language and Semantics

Let $\mathbb{N}$ be the set of natural numbers. The language of Epstein's logic is the following propositional language $\mathcal{L}$:

$$\langle \mathsf{Var}, \{\neg, \wedge, \rightarrow\}, \{(,)\}\rangle,$$

where $\mathsf{Var} = \{p^i : i \in \mathbb{N}\}$ is a set of propositional variables. (Sometimes we will use simply the letters: $p, q, r, \ldots$)

The *set of formulas* is defined in a standard way and denoted by For. We will use the following abbreviation for any $A, B \in \mathsf{For}$:

$$A \nrightarrow B := A \rightarrow (B \rightarrow B).$$

We will also use some standard notations:

- the set of all subsets of Σ (the power set of Σ) will be denoted by $\mathcal{P}(\Sigma)$
- the set of propositional variables of formula $A \in \mathsf{For}$ will be denoted by $\mathsf{var}(A)$.

Moreover, let n be a function from $\mathcal{P}(\mathsf{Var})$ to $\mathcal{P}(\mathbb{N})$ defined as follows: $i \in \mathsf{n}(\Sigma)$ iff for some $p^i \in \mathsf{Var}$, $p^i \in \Sigma$, for all $\Sigma \subseteq \mathsf{Var}$, $i \in \mathbb{N}$.

In what follows, we define five Epstein's content relationship logics: three dependence logics (**D**, **DD**, **Eq**) and two relatedness logics (**S** and **R**) (see [6], cf. [4,5,16,17]). These formal systems can be defined in two ways: by means of set-assignment semantics (see [6], cf. [4,5,16,17]) and by means of relating semantics (see [7], cf. [10–12,16,19]).[3] We focus, however, on the first approach.

A *set-assignment structure* is an ordered pair $\langle v, s\rangle$ such that $v : \mathsf{Var} \longrightarrow \{1, 0\}$ is a classical valuation of propositional variables and $s : \mathsf{For} \longrightarrow \mathcal{P}(\Sigma)$, where Σ is any set, is the so-called set-assignment.[4] For the definition of logics **D**, **DD**, **Eq**, the set Σ might be empty but for the definition of relatedness logic **S**, we have to assume non-emptiness of $s(A)$, for any variable $A \in \mathsf{Var}$,

[3] In fact, Epstein [6] also presents for his logics a semantics based on binary relations over formulas. Such semantics might be considered to be the first example of relating semantics (see [16]).

[4] Let us note that by means of set-assignment structures, we can define various non-classical logics. This, however, requires to consider models with more than one set-assignment function (see [6,16,17]).

so non-emptiness of Σ as well.[5] For both cases, we use a union set-assignment, i.e., a set-assignment $s\colon \mathsf{For} \longrightarrow \mathcal{P}(\Sigma)$ such that for any $A \in \mathsf{For}$, we have:

$$s(A) = \bigcup\{s(B) \in \mathcal{P}(\Sigma) : B \in \mathsf{var}(A)\}. \tag{uSA}$$

By means of set-assignments, Epstein proposes to formally represent contents of sentences. In the truth-condition for implication in addition to the classical condition, it is required that some, a priori assumed, relation holds between the formal representations of contents.

Having a structure $\langle v, s\rangle$, in order to determine Epstein's logics, we assume the following definitions of $R_\Lambda \subseteq \mathsf{For}\times\mathsf{For}$, where $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$. For any $A, B \in \mathsf{For}$:

- for logic $\mathbf{D}$, $R_{\mathbf{D}}(A,B)$ iff $s(B) \subseteq s(A)$
- for logic $\mathbf{DD}$, $R_{\mathbf{DD}}(A,B)$ iff $s(A) \subseteq s(B)$
- for logic $\mathbf{Eq}$, $R_{\mathbf{Eq}}(A,B)$ iff $s(A) = s(B)$
- for logic $\mathbf{S}$, $R_{\mathbf{S}}(A,B)$ iff $s(A) \cap s(B) \neq \emptyset$
- for logic $\mathbf{R}$, $R_{\mathbf{R}}(A,B)$ iff $\begin{cases} A = B \text{ or } \mathsf{n}(A) \in s(B), & \text{if } A, B \in \mathsf{Var} \\ \exists_{x\in\mathsf{var}(A)}\exists_{y\in\mathsf{var}(B)} R_{\mathbf{R}}(x,y), & \text{otherwise.} \end{cases}$

We would like to make two comments on the logic $\mathbf{R}$. First, for the relating approach to the logic $\mathbf{R}$, Epstein defined the set of reflexive relations constructed in the following way. He defined reflexive relations on $\mathsf{Var} \times \mathsf{Var}$, and then extended them to $R \subseteq \mathsf{For} \times \mathsf{For}$ by the condition: $R(A,B)$ iff $\exists_{x\in\mathsf{var}(A)}\exists_{y\in\mathsf{var}(B)} R(x,y)$. We can call the relations *relatedness relations of content* (see [16]). It is quite easy to see that: (a) the relation $R_{\mathbf{R}}$ is a relatedness relation of content; (b) for any relatedness relation of content $R \subseteq \mathsf{For} \times \mathsf{For}$, there exists such a union set-assignment $s\colon \mathsf{For} \longrightarrow \mathcal{P}(\Sigma)$ that $R = R_{\mathbf{R}}$. We can consider $s\colon \mathsf{For} \longrightarrow \mathcal{P}(\mathbb{N})$ defined in the following way, for any $B \in \mathsf{Var}$, $s(B) = \{i \in \mathbb{N} : R(p^i, B)\}$. We extend the function on the whole set of formulas in the following way: $s(A) = \bigcup\{s(B) \in \mathcal{P}(\mathbb{N}) : B \in \mathsf{var}(A)\}$. This is a special case of (uSA), when $\Sigma = \mathbb{N}$.

Second, in the considerations on the content relationship, it is assumed by Epstein that $R_{\mathbf{S}}(A,B)$ iff $s(A)\cap s(B) \neq \emptyset$. Also for the logic $\mathbf{R}$ it is assumed that two propositions should share the content but it requires introducing two set-assignments (see [6,16,17]). In the approach we propose, for two different variables A, B, $R_{\mathbf{R}}(A,B)$ does not have to imply $s(A)\cap s(B) \neq \emptyset$. It is because it is possible that, $\mathsf{n}(A) \in s(B)$, but $\mathsf{n}(A) \notin s(A)$, when A, B are different. For the moment we might think what happens if we also assume that $\mathsf{n}(A) \in s(A)$. So instead of the definition that we have proposed:

$$R_{\mathbf{R}}(A,B) \text{ iff } \begin{cases} A = B \text{ or } \mathsf{n}(A) \in s(B), & \text{if } A, B \in \mathsf{Var} \\ \exists_{x\in\mathsf{var}(A)}\exists_{y\in\mathsf{var}(B)} R_{\mathbf{R}}(x,y), & \text{otherwise,} \end{cases}$$

we could alternatively assume the following one:

$$R'_{\mathbf{R}}(A,B) \text{ iff } \begin{cases} A = B \text{ or } [\mathsf{n}(A) \in s(B) \text{ and } \mathsf{n}(A) \in s(A)], & \text{if } A, B \in \mathsf{Var} \\ \exists_{x\in\mathsf{var}(A)}\exists_{y\in\mathsf{var}(B)} R'_{\mathbf{R}}(x,y), & \text{otherwise.} \end{cases}$$

[5] Let us also note that for our definition of $\mathbf{R}$ the set Σ might be empty.

Both definitions, of $R_{\mathbf{R}}$ and $R'_{\mathbf{R}}$, are inductive. They differ only in the initial case for variables. Let us check this case. We take the structure: $\langle v, s\rangle$ and assume that we consider two different variables $A, B \in \mathsf{Var}$. Clearly, if $R'_{\mathbf{R}}(A, B)$, then $R_{\mathbf{R}}(A, B)$. The inverse side is much more interesting. Let us assume that it is not that $R'_{\mathbf{R}}(A, B)$ and simultaneously $\mathsf{n}(A) \in s(B)$, but $\mathsf{n}(A) \notin s(A)$. Since we assumed that A and B are different, so it does not matter if $\mathsf{n}(A) \in s(A)$. For both relations, $R_{\mathbf{R}}(A, B)$ and $R'_{\mathbf{R}}(A, B)$, if $A = B$.

Let us make a conclusion on the base of the last remark. From the logical point of view, whether we assume the first definition or the second one, it does not matter. Both provide the same logic. But it is worth noting that $R'_{\mathbf{R}}(A, B)$ implies $s(A) \cap s(B) \neq \emptyset$, and this results in the fact that $R'_{\mathbf{R}} \subseteq R_{\mathbf{S}}$. Independently, we know, however, that the logic $\mathbf{R}$ is a sublogic of $\mathbf{S}$ (see [4,6,16]).

Knowing these relationships, we want to simplify the approach to tableaux. So, in the further tableau analysis of $\mathbf{R}$, we assume the first definition, i.e., the definition of $R_{\mathbf{R}}$.

A set of structures becomes a set of models only if some special relation is set between them. It is usually a binary relation of satisfiability between a structure and any formula $A \in \mathsf{For}$.

Definition 2.1 (Λ–model). Let $\mathfrak{M} = \langle v, s\rangle$ be a structure. Let $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$. $\mathfrak{M}$ is *a model* of Λ (in short: Λ–model) iff for any $A, B \in \mathsf{For}$:

$$\begin{aligned}
\mathfrak{M} &\models A \text{ iff } v(A) = 1, \text{ if } A \in \mathsf{Var}\\
\mathfrak{M} &\models \neg A \text{ iff it is not that } \mathfrak{M} \models A \quad (\text{in symb.: } \mathfrak{M} \not\models A)\\
\mathfrak{M} &\models A \wedge B \text{ iff } \mathfrak{M} \models A \text{ and } \mathfrak{M} \models B\\
\mathfrak{M} &\models A \to B \text{ iff } \mathfrak{M} \not\models A \text{ or } \mathfrak{M} \models B, \text{ and } R_{\Lambda}(A, B).
\end{aligned}$$

We have the standard definition of semantic consequence and validity.

Definition 2.2 (Semantic consequence in Λ). Let $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ and $\Sigma \cup \{A\} \subseteq \mathsf{For}$:

- A is a *semantic consequence of* Σ *in* Λ (in symb.: $\Sigma \models_{\Lambda} A$) iff for any Λ-model $\mathfrak{M}$, if for any $B \in \Sigma$, $\mathfrak{M} \models B$, then $\mathfrak{M} \models A$
- A is a *valid formula of* Λ (in symb.: $\models_{\Lambda} A$) iff $\emptyset \models_{\Lambda} A$.

As we noticed above the logic $\mathbf{R}$ is a sublogic of the logic $\mathbf{S}$. Excluding the logic $\mathbf{S}$ or the logic $\mathbf{R}$ we get logics that are independent. To show such independence let us consider the following formulas:

$$((p \multimap q) \wedge (q \multimap r)) \to (p \multimap r) \tag{1}$$

$$p \to (q \to (p \wedge q)) \tag{2}$$

$$((p \multimap q) \wedge (q \multimap r)) \to ((p \multimap r) \wedge (p \multimap q)) \tag{3}$$

$$(p \multimap q) \to (q \multimap p) \tag{4}$$

$$(p \wedge q) \to p. \tag{5}$$

The following table specifies which formulas are valid (+) and which are not (−) for a given logic:

	D	**DD**	**Eq**	**S**	**R**
(1)	+	−	−	−	−
(2)	−	+	−	+	+
(3)	+	+	+	−	−
(4)	−	−	+	+	−
(5)	+	−	−	+	+

2.2. Variable Sharing and Paradoxes of Implication

Let us notice that Epstein's logics satisfy the variable sharing property, i.e., for any $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ and any $A, B \in \mathsf{For}$, if $\models_\Lambda A \to B$, then $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$. Indeed, if $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$, then $\mathsf{var}(B) \not\subseteq \mathsf{var}(A)$, $\mathsf{var}(A) \not\subseteq \mathsf{var}(B)$, $\mathsf{var}(A) \neq \mathsf{var}(B)$ and obviously $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$. Also for all variables $x \in \mathsf{var}(A), y \in \mathsf{var}(B)$, $\mathsf{n}(x) \notin \mathsf{var}(y)$.

Clearly, $\mathsf{var}\colon \mathsf{For} \longrightarrow \mathcal{P}(\mathsf{Var})$ is a union set-assignment, so with that, we can define Λ-model $\mathfrak{M}$ such that $\mathfrak{M} \not\models A \to B$.

The variable sharing property enables us to see that if a formula is not true in the model of the form $\langle v, \mathsf{var}\rangle$, then it is not valid in any of Epstein's logics. However in order to determine the analysed logics, we cannot consider only set-assignment structures of the form $\langle v, \mathsf{var}\rangle$. As Epstein noticed in [6], structures with var validate too much; for instance, consider the law of importation and the law of exportation.[6]

Let us also notice that for any of the analysed logics, none of the following paradoxes of implication is valid:

$$p \to (q \to p) \tag{6}$$

$$\neg p \to (p \to q) \tag{7}$$

$$p \to \neg(q \wedge \neg q) \tag{8}$$

$$(p \wedge \neg p) \to q. \tag{9}$$

In order to prove that neither (6) nor (7) is valid, it is enough to notice that the truth of q and the truth of $\neg p$ respectively do not entail that for any union set-assignment s, any of the considered relations between $s(p)$ and $s(q)$ hold. However, in order to prove that (8) and (9) are not valid, we can make use of a variable sharing property.

2.3. Proof Theory

In the previous section, we used the abbreviation of the form $A \nrightarrow B$. For all Epstein's logics, it enables one to express in the language that a proper relation between outputs of set-assignments holds. More specifically, for any $A, B \in \mathsf{For}$ and any set-assignment structure $\mathfrak{M} = \langle v, s\rangle$ and any $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$, if $\mathfrak{M}$ is Λ-model, then $\mathfrak{M} \models A \nrightarrow B$ iff $R_\Lambda(A, B)$.

[6]The situation is different for FDE-fragments of Epstein's logics (cf. [20,22]). In this case, we restrict models to structures with function var.

This is the basis of Epstein's axiomatic systems of **D**, **DD**, **Eq**, **S**, and **R** (see [6], cf. [4]). Indeed, formulas (1), (2), (3), (4) and (5) are examples of the axioms of the analysed logics: (1) of logic **D**, (2) of logics **DD**, **S** and **R**, (3) of logic **Eq**, but it is also a thesis of **D** and **DD**, (4) of logics **S** and **Eq**, (5) of logics **D**, **S** and **R**.[7]

In the 1980s and 1990s, other proof methods were specified for Epstein's logics and some of their modifications. Luis Fariñas del Cerro and Valérie Lugardon [3] defined sequent systems for certain modification of Epstein's dependence logics, specifically for modifications of **D**, **DD** and **Eq**, determined by set-assignment structures defined over a selected union set-assignment. In [2], logics of this kind were extended onto a first order language and sequent systems for these extensions were defined. Other interesting results were presented by Francesco Paoli, who focused on FDE-fragments of Epstein's logics.[8] In [20], Paoli presented the FDE-fragment of **S** from the algebraic, matrices, and axiomatic points of view. This analysis was extended onto other FDE-fragments of Epstein's logics in [22]. In that paper Paoli, also discussed tableaux for these FDE-fragments. In contrast to these approaches, in what follows, we focus on tableau systems that can account for the whole logics **D**, **DD**, **Eq**, **S** and **R**.

3. Tableaux

The first tableau systems for selected Epstein logics – being examples of relating logics – were introduced by W. Carnielli [1]. Carnielli in his paper focused on the relatedness logic **S** and suggested how one can modify the presented approach to get tableau systems for the relatedness logic **R**, the dependence logic **D**, and its dual version the logic **DD**. However, this approach, as we wrote in the introduction, has a number of limitations.[9]

But Carnielli's was not the only attempt to provide tableau systems for the logics of the considered type. Other tableau systems for a few relating logics, weaker than Epstein's logics, were given by Jarmużek and Klonowski in [11] (a tableau system for even weaker logic was presented in [9]). Furthermore, tableau systems for certain connexive logics defined on the basis of relating logics were defined by Jarmużek and Malinowski [12] (cf. [13]).

In what follows, we present a tableau approach that allows to define adequate tableau systems for various logics determined by set-assignment semantics. We focus on logics: **D**, **DD**, **Eq**, **S** and **R**.

[7] Let us emphasize that axioms introduced by Epstein are founded on relational properties presented within the second approach based on relating semantics.

[8] It is important to notice that Paoli also presented results concerning not only the fragments of Epstein's logics. In [21], he presented the constructive completeness of **S** (for an alternative proof see [15]) and determined the limited principle of substitution of equivalent formulas for **S**. In cooperation with Antonio Ledda and Michele Pra Baldi, he examined in [18] various algebraic interpretations of a demodalized analytic implication understood as an implication of the logic **D** (cf. [5]).

[9] It is worth noting that the tableaux for a generally defined relating logic are given in [9], while in [10], the tableau systems for deontic logics determined by the relating semantics are defined.

3.1. Tableau Language

The language of tableau system is the language of $\mathcal{L}$ extended by auxiliary expressions. We assume an additional symbol: $\sim$ (negation for new expressions). The set of auxiliary expressions Ae is the union of the following sets:

- $\mathbb{N} \times \mathsf{For}$
- $\{\langle \sim i, A\rangle \colon \langle i, A\rangle \in \mathbb{N} \times \mathsf{For}\}$.

Expressions of the form $\langle i, A\rangle$ and $\langle \sim i, A\rangle$ are supposed to represent that object i belongs to the content of A or i does not belong to the content of A, respectively. If it is not misleading, we will omit the brackets: $\langle\ \rangle$. So for example, instead of $\langle \sim i, A\rangle$ or $\langle i, A\rangle$, we will just write i, A, and $\sim i, A$, etc.

You can see, we are using labels here to denote the content of the sentences. This is part of the tableaux strategy for syllogistic logic, where generalized labels can denote many different things [14].

The *set of tableau expressions* is the set $\mathsf{Ex} = \mathsf{For} \cup \mathsf{Ae}$. This is a language we conduct tableau proofs in it. Finally, we define a branch inconsistent set of expressions.

Definition 3.1 (Branch inconsistent set of expressions). Let $\Sigma \subseteq \mathsf{Ex}$.

- Σ is *branch inconsistent* iff it contains at least two expressions that form a complementary pair of any of the following forms:
  - $A, \neg A$
  - $\langle i, A\rangle, \langle \sim i, A\rangle$,

  for any $A \in \mathsf{For}$, $i \in \mathbb{N}$.
- Σ is *branch consistent* iff Σ is not branch inconsistent.

3.2. Tableau Rules

We assume some general set of tableau rules for all logics we examine here. For the elimination of connectives, we have:

$$(\mathrm{R}_\wedge)\ \frac{A \wedge B}{\begin{array}{c} A \\ B \end{array}} \qquad (\mathrm{R}_\rightarrow)\ \frac{A \rightarrow B}{\neg A \,|\, B} \qquad (\mathrm{R}_{\neg\wedge})\ \frac{\neg(A \wedge B)}{\neg A \,|\, \neg B} \qquad (\mathrm{R}_{\neg\neg})\ \frac{\neg\neg A}{A}$$

The above rules do not need any comments. They reflect the classical truth conditions for $\wedge$, $\neg$ and partially $\rightarrow$. For particular logics, we will complete the rule for implication. For the reduction of auxiliary expressions, we assume the general rules:

$$(\mathrm{R}_i)\ \frac{i, A}{i, A_1 \,|\, \ldots \,|\, i, A_n} \qquad (\mathrm{R}_{\sim i})\ \frac{\sim i, A}{\begin{array}{c} \sim i, A_1 \\ \vdots \\ \sim i, A_n \end{array}}$$

where $\mathsf{var}(A) = \{A_1, \ldots, A_n\}$ where $\mathsf{var}(A) = \{A_1, \ldots, A_n\}$

Both rules correspond to the uniform set assignment condition (uSA). (R_i) says that if object i belongs to the content of formula A, then object i belongs

to the content of at least one variable of A. In turn, ($\mathrm{R}_{\sim i}$) postulates that if object i does not belong to the content of A, then object i does not belong to the content of any atomic proposition of A.[10]

Now we introduce specific rules for the particular logics. For logic **D**, we assume:

$$(\mathrm{R}_{\mathbf{D}\rightarrow})\ \frac{\begin{array}{c}A \rightarrow B\\ i, C\end{array}}{i, A} \qquad (\mathrm{R}_{\mathbf{D}\neg\rightarrow})\ \frac{\neg(A \rightarrow B)}{\begin{array}{c|c}A & i, B\\ \neg B & \sim i, A\end{array}}$$

where $\mathsf{var}(C) \subseteq \mathsf{var}(B)$ where i is new on the branch

Both presented tableau rules reflect the truth-condition for implication in logic **D**. In case of ($\mathrm{R}_{\mathbf{D}\rightarrow}$), the content of consequence is included in the content of antecedent. In case of ($\mathrm{R}_{\mathbf{D}\neg\rightarrow}$), A is true, but B false, or the content of consequence is not included in the content of antecedent.[11]

The specific tableau rules for logic **DD** are:

$$(\mathrm{R}_{\mathbf{DD}\rightarrow})\ \frac{\begin{array}{c}A \rightarrow B\\ i, C\end{array}}{i, B} \qquad (\mathrm{R}_{\mathbf{DD}\neg\rightarrow})\ \frac{\neg(A \rightarrow B)}{\begin{array}{c|c}A & i, A\\ \neg B & \sim i, B\end{array}}$$

where $\mathsf{var}(C) \subseteq \mathsf{var}(A)$ where i is new on the branch

The ideas behind ($\mathrm{R}_{\mathbf{DD}\rightarrow}$) and ($\mathrm{R}_{\mathbf{DD}\neg\rightarrow}$) are similar to the latter tableau rules. However, now inversely, the content of antecedent is included in the content of consequence.

For logic **Eq** we assume the tableau rules ($\mathrm{R}_{\mathbf{D}\rightarrow}$) and ($\mathrm{R}_{\mathbf{DD}\rightarrow}$), since the content of antecedent is equal to the content of consequence. But for negation of implication, we combine the rules ($\mathrm{R}_{\mathbf{D}\neg\rightarrow}$) and ($\mathrm{R}_{\mathbf{DD}\neg\rightarrow}$) into one rule[12]:

$$(\mathrm{R}_{\mathbf{Eq}\neg\rightarrow})\ \frac{\neg(A \rightarrow B)}{\begin{array}{c|c|c}A & i, A & \sim i, A\\ \neg B & \sim i, B & i, B\end{array}}$$

where i is new on the branch

For logic **S**, we assume specific tableau rules:

$$(\mathrm{R}_{\mathbf{S}\rightarrow})\ \frac{A \rightarrow B}{\begin{array}{c}i, A\\ i, B\end{array}} \qquad (\mathrm{R}_{\mathbf{S}\neg\rightarrow})\ \frac{\neg(A \rightarrow B)}{\begin{array}{c|c}A & i, A\\ \neg B & j, B\end{array}}$$

where i is new on the branch where i, j are new on the branch and $i \neq j$

[10] Examples of applications of (R_i) and ($\mathrm{R}_{\sim i}$) are presented in Appendix, see Figs. 4, 5, 6, 7, 8 and 9.

[11] Examples of applications of ($\mathrm{R}_{\mathbf{D}\rightarrow}$) and ($\mathrm{R}_{\mathbf{D}\neg\rightarrow}$) are presented in Appendix, see Figs. 4 and 5.

[12] Examples of applications of ($\mathrm{R}_{\mathbf{Eq}\neg\rightarrow}$) are presented in Appendix, see Figs. 1, 6 and 7.

Both rules (without and with negation) introduce new labels since content of propositions is possibly supposed to be non-empty for **S**. However, while in the case of ($R_{\mathbf{S}\rightarrow}$) both propositions have the same content, in ($R_{\mathbf{S}\neg\rightarrow}$), the content is different and so the labels must be different. Additionally, we assume the tableau rule ($R_{\mathbf{S}}$):

$$(R_{\mathbf{S}})\ \frac{\begin{array}{c} i, A \\ j, B \\ k, A \end{array}}{\sim k, B}$$

where i, j are introduced by the rule($R_{\mathbf{S}\neg\rightarrow}$)

The rule ($R_{\mathbf{S}}$) guarantees that two propositions do not share any label if they have labels introduced by application of ($R_{\mathbf{S}\neg\rightarrow}$). All the rules correspond to the truth condition for implication in **S**: the intersection of the content of antecedent and the content of consequence is non-empty.[13]

Finally, for logic **R**, we assume specific tableau rules:

$$(R_{\mathbf{R}\rightarrow})\ \frac{A \rightarrow B}{k_1, B_1 \,|\, \ldots \,|\, k_m, B_1 \,|\, \ldots \,|\, k_1, B_n \,|\, \ldots \,|\, k_m, B_n}$$

if $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$,

where $\mathsf{n}(\mathsf{var}(A)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(B) = \{B_1, \ldots, B_n\}$

The rule ($R_{\mathbf{R}\rightarrow}$) is applicable when $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$. This rule expresses this part of the truth-condition for relating implication which requires that for some $A_i \in \mathsf{var}(A)$, $\mathsf{n}(A_i) \in s(B_j)$, for some $B_j \in \mathsf{var}(B)$. When $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$, it is only needed to use the rule ($R_{\rightarrow}$), since A and B share at least one variable.

$$(R_{\mathbf{R}\neg\rightarrow}(1))\ \frac{\neg(A \rightarrow B)}{\begin{array}{c|c} A & \sim k_1, B_1 \\ \neg B & \vdots \\ & \sim k_m, B_1 \\ & \vdots \\ & \sim k_1, B_n \\ & \vdots \\ & \sim k_m, B_n \end{array}}$$

if $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$,

where $\mathsf{n}(\mathsf{var}(A)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(B) = \{B_1, \ldots, B_n\}$

The rule ($R_{\mathbf{R}\neg\rightarrow}(1)$) is applicable when $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$. This rule splits a proof. The left side corresponds to the traditional negative condition for implication: so the antecedent is true, while the consequence is false. The right side negatively expresses this part of the truth-condition for relating

[13] Examples of applications of ($R_{\mathbf{S}\rightarrow}$), ($R_{\mathbf{S}\neg\rightarrow}$) and ($R_{\mathbf{S}}$) are presented in Appendix, see Figs. 2, 8 and 9.

implication, which requires that for some $A_i \in \mathsf{var}(A)$, $\mathsf{n}(A_i) \in s(B_j)$, for some $B_j \in \mathsf{var}(B)$. So we have a string: $\sim k_1, B_1, \ldots, \sim k_m, B_1, \ldots, \sim k_1, B_n, \ldots, \sim k_m, B_n$, where $\mathsf{n}(\mathsf{var}(A)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(B) = \{B_1, \ldots, B_n\}$. In the tableau language, it just states that for all $A_i \in \mathsf{var}(A)$, $\mathsf{n}(A_i) \notin s(B_j)$, for all $B_j \in \mathsf{var}(B)$.

When $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$, the rule $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$ is not applicable, since A and B share at least one variable. In this case, we have only the left side of $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$, so we must use the rule $(\mathrm{R}_{\neg\rightarrow}(2))$[14]:

$$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2)) \quad \frac{\neg(A \rightarrow B)}{\begin{array}{c} A \\ \neg B \end{array}}$$

if $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$.

It is worth underlining that in our formulation of tableau rules for the logic **R**, we do not need to use the rules (R_i) and $(\mathrm{R}_{\sim i})$. All rules we have defined for the relating implication in **R** just reduce the indexes to the level of variables.

By $\mathsf{tr}(\Lambda)$ we will denote the set of tableau rules for logic Λ, where $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$.

For any tableau rules expressions, in the numerator will be called *input*, while expressions from the denominator will be called *output*. As an example, let us take the rule: $(\mathrm{R}_{\wedge})$. One of its inputs is $\{p \wedge q\}$ and then the corresponding output is set $\{p, q\}$. Notice that this rule is a non-branching one, i.e., for any input, it has got only one output (one set of formulas). On the other hand, $(\mathrm{R}_{\mathbf{D}\neg\rightarrow})$ is a branching rule and in this case we have two outputs, for example: $\{p, \neg q\}$, and $\{\langle i, \sim p\rangle, \langle i, q\rangle\}$ for some new i, if the input is $\{\neg(p \rightarrow q)\}$.

Once we have the notion of input, we can define the notion of applicability of a rule. Let (R) be a tableau rule and $\Sigma \in \mathsf{Ex}$. (R) is *applicable to* Σ iff an input of the (R) is a subset of Σ and Σ is a branch consistent set.

It is worth noting that our rules for set-assignment semantics combine two aspects. On the one hand, these are rules that contain the classic truth aspect: it takes place in case of pure formulas in the tableau rules. However, on the other hand, we have a syllogistic aspect. In case of expressions with labels, we treat formulas as names and labels as denotation. Thus, formula A is treated as a name of the content in expressions like $\langle \sim i, A\rangle$ or $\langle i, A\rangle$ (cf. [14]).

3.3. Tableau Consequence Relation

We define the relation of tableau consequence by referring to the concept of closure under tableau rules.[15] This is the equivalent of $\models_{\Lambda}$ for the individual logics Λ we are considering. Of course, any particular tableau consequence relation depends on the rules for a given logic used in the proof.

[14]Examples of applications of $(\mathrm{R}_{\mathbf{R}\rightarrow})$, $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$ and $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$ are presented in Appendix, see Figs. 3, 10 and 11.

[15]The concept of closure under tableau rules was introduced in [11]. The story of the notion of tableau consequence relation was told in the book [8].

Definition 3.2 (Closure under tableau rules). Let $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ and $\Sigma, \Gamma \subseteq \mathsf{Ex}$. Γ is *a closure of* Σ *under tableau rules of* $\mathsf{tr}(\Lambda)$ (for short: $\mathsf{tr}(\Lambda)$-*closure of* Σ) iff there exists such a subset of natural numbers K that:

- $K = \mathbb{N}$ or $K = \{1, \ldots, n\}$ for some $n \in \mathbb{N}$
- there exists such an injective sequence $f\colon K \longrightarrow \mathcal{P}(\mathsf{Ex})$ that:
  - $X_1 = \Sigma$
  - for all $i, i+1 \in K$, there exists such a tableau rule (R) of $\mathsf{tr}(\Lambda)$ that its input is included in X_i, while one of its corresponding outputs is equal to $X_{i+1} \setminus X_i$
  - for all $i, i+1 \in K$, for any tableau rule (R) of $\mathsf{tr}(\Lambda)$, if (R)'s input is included in X_i and one of (R)'s corresponding outputs is equal to $X_{i+1} \setminus X_i$, then there are no such $j, j+1 \in K$ that $j > i$ and one of the remaining outputs of (R) is equal to $X_{j+1} \setminus X_j$
  - for any tableau rule (R) of $\mathsf{tr}(\Lambda)$, if (R)'s input is included in $\bigcup_{i \in K} X_i$, then one of the (R)'s corresponding outputs is in $\bigcup_{i \in K} X_i$
- $\Gamma = \bigcup_{i \in K} X_i$.

In practice, we can treat the notion of closure given in the definition of the closure as the notion of a complete branch (closed or open). In fact, it is the union of all elements that are on a complete branch.

Definition 3.3 (Tableau consequence relation). Let $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ and $\Sigma \cup \{A\} \subseteq \mathsf{For}$.

- A is *a tableau consequence of* Σ *in* Λ (in symb.: $\Sigma \rhd_\Lambda A$) iff there is a finite set $\Gamma \subseteq \Sigma$ such that any $\mathsf{tr}(\Lambda)$-closure of $\Gamma \cup \{\neg A\}$ is branch inconsistent.
- A is *a thesis of* Λ (in symb.: $\rhd_\Lambda A$) iff $\emptyset \rhd_\Lambda A$.

Here we have proposed a quite formal definition of tableau proof, reduced to the notion of closure under the set of tableau rules. However, in Appendix we present the examples of tableau proofs written in the standard, less formal way.

3.4. Soundness, Completeness and Decidability

Now, we would like to be able to select labels from sets of expressions. We define a function:

Definition 3.4 (Function choosing labels). The function choosing labels is the function $\circ\colon \mathsf{Ex} \cup \mathcal{P}(\mathsf{Ex}) \longrightarrow \mathcal{P}(\mathbb{N})$ defined by conditions:

- $\circ(A) = \emptyset$
- $\circ(\langle i, A\rangle) = \circ(\langle \sim i, A\rangle) = \{i\}$
- $\circ(\Sigma) = \bigcup\{\circ(x)\colon x \in \Sigma\}$

for all $A \in \mathsf{For}$, $\Sigma \subseteq \mathsf{For}$, and $i \in \mathbb{N}$.

We can now generalize the notion of interpretation of a set of expressions in a model.

Definition 3.5 (Model suitable for a set of expressions). Let $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$, $\mathfrak{M} = \langle v, s\rangle$ be Λ-model with the domain Γ for the set assignment function s, $\Sigma \subseteq \mathsf{Ex}$, and let $\Gamma \subseteq \Gamma'$. $\mathfrak{M}$ *is suitable for* Σ iff there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ such that:

- if $A \in \Sigma$ then $\mathfrak{M} \models A$
- if $\langle i, A\rangle \in \Sigma$ then $\mathsf{f}(i) \in s(A)$
- if $\langle \sim i, A\rangle \in \Sigma$ then $\mathsf{f}(i) \notin s(A)$

for all $A, B \in \mathsf{For}$ and $i \in \mathbb{N}$.

One comment to the definition is needed. We assume the extended domain Γ' for the case of logic $\mathbf{R}$ where we use the superscripts of variables as the part of content (see the part of the proof of the Lemma 3.6 for the logic $\mathbf{R}$, the case for the rule $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$).

It is clear that for a branch inconsistent set of expressions no model is suitable.

The following lemma enables us to prove soundness theorems for considered logics.

Lemma 3.6. (Soundness lemma.) *Let* $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$, $\Sigma \subseteq \mathsf{Ex}$ *and* $\mathfrak{M} = \langle v, s\rangle$ *be a model* Λ*-model suitable for* Σ*. For any tableau rule* (R) *of* $\mathsf{tr}(\Lambda)$*, if* (R) *has been applied to* Σ*, then* $\mathfrak{M}$ *is suitable for the union of* Σ *and at least one output obtained by the application of rule* (R)*.*

Proof. Assume all the hypotheses. For the cases in which the general elimination rules for connectives $((\mathrm{R}_\wedge), (\mathrm{R}_\rightarrow), (\mathrm{R}_{\neg\wedge}), (\mathrm{R}_{\neg\neg}))$ are applied, the proof is obvious because they behave classically. We consider only two general cases for connectives.

- Suppose $(\mathrm{R}_\wedge)$ has been applied to Σ . Then, $A \wedge B \in \Sigma$ and we get output $\{A, B\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $\mathfrak{M} \models A \wedge B$. Thus, by Definition 2.1 of Λ-model, $\mathfrak{M} \models A$ and $\mathfrak{M} \models B$. Hence, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{A, B\}$.
- Suppose $(\mathrm{R}_\rightarrow)$ has been applied to Σ. Then, $A \rightarrow B \in \Sigma$ and we get two outputs $\{\neg A\}$ and $\{B\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $\mathfrak{M} \models A \rightarrow B$. Thus, by Definition 2.1 of Λ-model, $\mathfrak{M} \models \neg A$ or $\mathfrak{M} \models B$. Hence, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\neg A\}$ or for the set $\Sigma \cup \{B\}$.

Now we examine general cases of the tableau rules for reduction of auxiliary expressions. We have two cases (R_i) and $(\mathrm{R}_{\sim i})$:

- Suppose (R_i) has been applied to Σ. Then, $\langle i, A\rangle \in \Sigma$ and we get n-outputs $\{\langle i, A_1\rangle\}, \ldots, \{\langle i, A_n\rangle\}$, where $\mathsf{var}(A) = \{A_1, \ldots, A_n\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ such that $\mathsf{f}(i) \in s(A)$. But by the uniform set assignment condition (uSA), it means that $\mathsf{f}(i) \in s(A_k)$, for some $1 \leq k \leq n$. So, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle i, A_k\rangle\}$.

- Suppose $(R_{\sim i})$ has been applied to Σ. Then, $\langle \sim i, A\rangle \in \Sigma$ and we get the output $\{\langle \sim i, A_1\rangle, \ldots, \langle \sim i, A_n\rangle\}$, where $\mathsf{var}(A) = \{A_1, \ldots, A_n\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ such that $\mathsf{f}(i) \notin s(A)$. But by the uniform set assignment condition (uSA), it means that $\mathsf{f}(i) \notin s(A_k)$, for all $1 \leq k \leq n$. So, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle \sim i, A_1\rangle, \ldots, \langle \sim i, A_n\rangle\}$.

Let us assume that $\Lambda = \mathbf{D}$. We must consider two cases: $(R_{\mathbf{D}\rightarrow})$ and $(R_{\mathbf{D}\neg\rightarrow})$.

- Suppose $(R_{\mathbf{D}\rightarrow})$ has been applied to Σ. Then, $A \rightarrow B$ and $\langle C, i\rangle \in \Sigma$, where $\mathsf{var}(C) \subseteq \mathsf{var}(B)$, and we get output $\{\langle i, A\rangle\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $\mathfrak{M} \models A \rightarrow B$ and $s(B) \subseteq s(A)$ ($\mathfrak{M}$ is **D**-model). Moreover, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ such that $\mathsf{f}(i) \in s(C)$. But by the uniform set assignment condition (uSA), it means that $\mathsf{f}(i) \in \bigcup s(\mathsf{var}(C))$ and, since $\mathsf{var}(C) \subseteq \mathsf{var}(B)$, $\mathsf{f}(i) \in s(A)$. Hence, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle i, A\rangle\}$.
- Suppose $(R_{\mathbf{D}\neg\rightarrow})$ has been applied to Σ. Then, $\neg(A \rightarrow B) \in \Sigma$, and we get two outputs $\{A, \neg B\}$ and $\{\langle \sim i, A\rangle, \langle i, B\rangle\}$, where i is a new label on the branch. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, (1) $\mathfrak{M} \not\models A$ and $\mathfrak{M} \models B$, or (2) $s(B) \not\subseteq s(A)$ ($\mathfrak{M}$ is **D**-model). If (1) happens, by Definition of 2.1 and Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{A, \neg B\}$. If (2) happens, there exists such $a \in \Gamma'$ that $a \in s(B)$ and $a \notin s(A)$. Additionally, by Definition 3.5 of suitable model, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$. However, i is a new label on the branch. So we define the extension of f, the function $\mathsf{f}'\colon \circ(\Sigma) \cup \{i\} \longrightarrow \Gamma'$ with $\mathsf{f}'(i) = a$. Thus, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle \sim i, A\rangle, \langle i, B\rangle\}$.

Let us assume that $\Lambda = \mathbf{DD}$. The cases are similar to the former ones. In models for **DD**, we have just the inverse inclusion and in $\mathsf{tr}(\mathbf{DD})$, inverse tableau rules $(R_{\mathbf{DD}\rightarrow})$ and $(R_{\mathbf{DD}\neg\rightarrow})$.

Let us assume that $\Lambda = \mathbf{Eq}$. The cases are also similar to the former ones. In models for **Eq**, we have just both inclusions and in $\mathsf{tr}(\mathbf{Eq})$, the tableau rules $(R_{\mathbf{D}\rightarrow})$ and $(R_{\mathbf{DD}\rightarrow})$. The tableau rule $(R_{\mathbf{Eq}\neg\rightarrow})$ is a combination of the rules: $(R_{\mathbf{D}\neg\rightarrow})$ and $(R_{\mathbf{DD}\neg\rightarrow})$, which were checked for the proper inclusions.

Let us assume that $\Lambda = \mathbf{S}$. We must consider three cases: $(R_{\mathbf{S}\rightarrow})$, $(R_{\mathbf{S}\neg\rightarrow})$, and $(R_{\mathbf{S}})$.

- Suppose $(R_{\mathbf{S}\rightarrow})$ has been applied to Σ. Then, $A \rightarrow B \in \Sigma$, $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$, and we get output $\{\langle i, A\rangle, \langle i, B\rangle\}$, where i is new on the branch. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $\mathfrak{M} \models A \rightarrow B$ and $s(B) \cap s(A) \neq \emptyset$ ($\mathfrak{M}$ is **S**-model). So, there is some $a \in s(B) \cap s(A)$. Moreover, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$. Since i is a new label on the branch, we define the extension of f, the function $\mathsf{f}'\colon \circ(\Sigma) \cup \{i\} \longrightarrow \Gamma'$ with $\mathsf{f}'(i) = a$. Thus, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle i, A\rangle, \langle i, B\rangle\}$.

- Suppose ($R_{\mathbf{S}\neg\rightarrow}$) has been applied to Σ. Then, $\neg(A \rightarrow B) \in \Sigma$ and we get two outputs $\{A, \neg B\}$ and $\{\langle i, A\rangle, \langle j, B\rangle\}$, where i, j are new labels on the branch and are different. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, (1) $\mathfrak{M} \not\models A$ and $\mathfrak{M} \models B$, or (2) $s(B) \cap s(A) = \emptyset$ ($\mathfrak{M}$ is **S**-model). If (1) happens, by definition of 2.1 and Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{A, \neg B\}$. If (2) happens, there exist such $a, b \in \Gamma'$ that $a \in s(A)$, $a \notin s(B)$, and $b \notin s(A)$, $b \in s(B)$, $s(B) \cap s(A) = \emptyset$ and in **S**-models the content of propositions is assumed to be non-empty. Additionally, by Definition 3.5 of suitable model, there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$. However, i, j are new labels on the branch. So we define the extension of f, the function $\mathsf{f}'\colon \circ(\Sigma) \cup \{i, j\} \longrightarrow \Gamma'$ with $\mathsf{f}'(i) = a$, $\mathsf{f}'(j) = b$. Thus, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle i, A\rangle, \langle j, B\rangle\}$.
- Suppose ($R_{\mathbf{S}}$) has been applied to Σ. Then, $\neg(A \rightarrow B), \langle i, A\rangle, \langle j, B\rangle, \langle k, A\rangle \in \Sigma$, where i, j are introduced by the rule ($R_{\mathbf{S}\neg\rightarrow}$), we get the output $\{\langle{\sim}k, B\rangle\}$ and thus, i and j are different. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $s(B) \cap s(A) = \emptyset$ ($\mathfrak{M}$ is **S**-model) and there exists a function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ with $\mathsf{f}(i) \in s(A)$, $\mathsf{f}(i) \notin s(B)$, $\mathsf{f}(j) \notin s(A)$, $\mathsf{f}(j) \in s(B)$, $\mathsf{f}(k) \in s(A)$, $\mathsf{f}(k) \notin s(B)$. Thus, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle{\sim}k, B\rangle\}$.

Let us finally assume that $\Lambda = \mathbf{R}$. We must consider three cases: ($R_{\mathbf{R}\rightarrow}$), ($R_{\mathbf{R}\neg\rightarrow}(1)$), and ($R_{\mathbf{R}\neg\rightarrow}(2)$).

- Suppose ($R_{\mathbf{R}\rightarrow}$) has been applied to Σ. Then, $A \rightarrow B \in \Sigma$, and we get the number of outputs equal to: m multiplied by n, where $\mathsf{n}(\mathsf{var}(A)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(B) = \{B_1, \ldots, B_n\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, $\mathfrak{M} \models A \rightarrow B$ and for some $A_i \in \mathsf{var}(A)$, $\mathsf{n}(A_i) \in s(B_j)$, for some $B_j \in \mathsf{var}(B)$, since $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ ($\mathfrak{M}$ is **R**-model). Let us assume than $k_i \in s(B_j)$. If k_i is a new label on the branch, we define the extension of f, the function $\mathsf{f}'\colon \circ(\Sigma) \cup \{k_i\} \longrightarrow \Gamma'$ with $\mathsf{f}'(k_i) = k_i$ and then $\mathsf{f}'(k_i) \in s(B_j)$. If for some reasons k_i is not a new label on the branch, the function f is sufficient to state that $\mathsf{f}(k_i) \in s(B_j)$. In both cases, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{\langle k_i, B_j\rangle\}$.
- Suppose ($R_{\mathbf{R}\neg\rightarrow}(1)$) has been applied to Σ. Then, $\neg(A \rightarrow B) \in \Sigma$, $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$, and we get two outputs: (a) $\{A, \neg B\}$ and (b) $\{{\sim}k_1, B_1, \ldots, {\sim}k_m, B_1, \ldots, {\sim}k_1, B_n, \ldots, {\sim}k_m, B_n\}$, where $\mathsf{n}(\mathsf{var}(A)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(B) = \{B_1, \ldots, B_n\}$. Because the model $\mathfrak{M}$ is suitable for Σ, by Definition 3.5 of suitable model, either (a)' $\mathfrak{M} \models A$ and $\mathfrak{M} \models \neg B$ or (b)' for all $A_i \in \mathsf{var}(A)$, $\mathsf{n}(A_i) \notin s(B_j)$, for all $B_j \in \mathsf{var}(B)$, since $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ ($\mathfrak{M}$ is **R**-model). If (a)' happens, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{A, \neg B\}$. If (b)' happens, by Definition 3.5 of suitable model, the model $\mathfrak{M}$ is suitable for the set $\Sigma \cup \{{\sim}k_1, B_1, \ldots, {\sim}k_m, B_1, \ldots, {\sim}k_1, B_n, \ldots, {\sim}k_m, B_n\}$, since either we use the existing function $\mathsf{f}\colon \circ(\Sigma) \longrightarrow \Gamma'$ and then $\mathsf{f}(k_1) \notin s(B_1), \ldots, \mathsf{f}(k_m) \notin s(B_1), \ldots, \mathsf{f}(k_1) \notin s(B_n), \ldots, \mathsf{f}(k_m) \notin s(B_n)$,

or we extend it to the function $\mathsf{f}'\colon \circ(\Sigma) \cup \{k_1, \ldots, k_m\} \longrightarrow \Gamma'$, under which $\mathsf{f}'(k_1) = k_1, \ldots, \mathsf{f}'(k_m) = k_m$, and then $\mathsf{f}'(k_1) \notin s(B_1)$, ..., $\mathsf{f}'(k_m) \notin s(B_1)$, ..., $\mathsf{f}'(k_1) \notin s(B_n)$, ..., $\mathsf{f}'(k_m) \notin s(B_n)$.

- Suppose $(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$ has been applied to Σ. Then, $\neg(A \rightarrow B) \in \Sigma$, $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$, and we get one output $\{A, \neg B\}$. Since the model $\mathfrak{M}$ is suitable to $\{\neg(A \rightarrow B)\}$ and $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$ ($\mathfrak{M}$ is **R**-model), by Definition 3.5 of suitable model, $\mathfrak{M}$ is also suitable to the set $\Sigma \cup \{A, \neg B\}$.

□

When we have a closure under $\mathsf{tr}(\Lambda)$ tableau rules, we can try to generate Λ-model. Here we introduce the definition of generated structure.

Definition 3.7. (Structure generated) Let $\Sigma \subseteq \mathsf{Ex}$. *A structure generated by* Σ (for short: Σ *-structure*) is $\langle v_\Sigma, s_\Sigma \rangle$ such that:

- $v_\Sigma(A) = 1$ iff $A \in \Sigma$, for any $A \in \mathsf{Var}$
- $s_\Sigma\colon \mathsf{For} \longrightarrow \mathcal{P}(\{i\colon \langle i, A\rangle \in \Sigma\})$
- $i \in s_\Sigma(A)$ iff for some $B \in \mathsf{var}(A)$: $\langle i, B\rangle \in \Sigma$, for any $A \in \mathsf{For}$.

In the case of $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}\}$ a structure can be a model, if Σ is a branch consistent $\mathsf{tr}(\Lambda)$-closure, but the case of **S** needs a little modification.

Corollary 3.8. *Let* $\Sigma \subseteq \mathsf{Ex}$ *be a branch consistent* $\mathsf{tr}(\Lambda)$*-closure, where* $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$. *Then there exists a structure* $\langle v_\Sigma, s_\Sigma \rangle$ *that is a* Λ*-model.*

Proof. If $\Sigma \subseteq \mathsf{Ex}$ is a $\mathsf{tr}(\Lambda)$-closure of some set, where $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{R}\}$, and Σ is branch consistent, then the structure $\langle v_\Sigma, s_\Sigma \rangle$ is a Λ-model. It is because truth conditions for $\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{R}$ require only set-theoretical relations/operations (Definition 2.1). Also the condition (uSA) is satisfied, which is a result of application of the tableau rules (R_i) and $(\mathrm{R}_{\sim i})$ in cases of $\mathbf{D}, \mathbf{DD}, \mathbf{Eq}$. One exception is logic **S**, because the **S**-models require non-emptiness of assigned sets. So in this case, we add to the set $\{i\colon \langle i, A\rangle \in \Sigma\}$ a new object a and redefine function s_Σ to function $s'_\Sigma\colon \mathsf{For} \longrightarrow \mathcal{P}(\{i\colon \langle i, A\rangle \in \Sigma\} \cup \{a\})$ in the following way:

$$s'_\Sigma(A) = \begin{cases} s_\Sigma(A) & \text{if } s_\Sigma(A) \neq \emptyset \\ \{a\} & \text{otherwise.} \end{cases}$$

Then, the structure $\langle v_\Sigma, s'_\Sigma \rangle$ is an **S**-model. □

The following lemma enables us to prove completeness theorems for considered logics.

Lemma 3.9. (Completeness lemma.) *Let* $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ *and* Σ *be a branch consistent closure under* $\mathsf{tr}(\Lambda)$. *Then there is a* Λ*-model such that for any* $A \in \mathsf{For}$*, if* $A \in \Sigma$*, then* $\mathfrak{M} \models A$.

Proof. Assume all the hypotheses. For $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ there exists a structure $= \langle v_\Sigma, s_\Sigma \rangle$ that is a Λ-model $\mathfrak{M}$ (Corollary 3.8). We make an inductive proof of the thesis.

Base case. Let $A \in \mathsf{Var}$. Suppose $A \in \Sigma$. Then, by Definition 3.7 $\mathfrak{M} \models A$. Suppose that $\neg A \in \Sigma$. Then, by Definition 3.7 $\mathfrak{M} \not\models A$, since Σ is branch consistent. So $\mathfrak{M} \models \neg A$.

Inductive hypothesis. Let $n \in \mathbb{N}$. Let c be a function of the complexity of formulas. Suppose that for any $A \in \mathsf{For}$ such that $\mathsf{c}(A) \leq n$, if $A \in \Sigma$, then $\mathfrak{M} \models A$.

Inductive step. Let $A \in \mathsf{For}$ and and $\mathsf{c}(A) = n + 1$.

The cases for formulas built by means of classical connectives and their negations, i.e., $B \wedge C$, $\neg(B \wedge C)$, $\neg\neg B$, follow from the inductive hypothesis and applications of the tableau rules ($\mathrm{R}_{\wedge}$), ($\mathrm{R}_{\neg\wedge}$), ($\mathrm{R}_{\neg\neg}$), since Σ is a closure under tableau rules $\mathsf{tr}(\Lambda)$. The cases for implication and negation depends on the logic Λ.

Let us assume $\Lambda = \mathbf{DD}$.

Let $A := B \rightarrow C$. Since Σ is a closure under tableau rules, ($\mathrm{R}_{\rightarrow}$) was applied and (1) $\neg B$ or (2) $C \in \Sigma$. If (1) happens, by the inductive hypothesis $\mathfrak{M} \models \neg B$. If (2) happens, then by the inductive hypothesis $\mathfrak{M} \models C$. Regardless of whether (1) or (2) happens, we have: $\mathfrak{M} \models B$ implies $\mathfrak{M} \models C$. However, the rule ($\mathrm{R}_{\mathbf{DD}\rightarrow}$) still could be applied to $B \rightarrow C$, if some $\langle i, D\rangle \in \Sigma$, where $\mathsf{var}(D) \subseteq \mathsf{var}(B)$. If there is no such expression, then $s_{\Sigma}(B) = \emptyset$ and $s_{\Sigma}(B) \subseteq s_{\Sigma}(C)$. If there is such expression, then $i \in s_{\Sigma}(B)$, and by the rule ($\mathrm{R}_{\mathbf{DD}\rightarrow}$), $i \in s_{\Sigma}(C)$, so $s_{\Sigma}(B) \subseteq s_{\Sigma}(C)$. Both by the definition of generated model 3.7. Putting these together with: $\mathfrak{M} \models B$ implies $\mathfrak{M} \models C$, we get $\mathfrak{M} \models B \rightarrow C$.

Let $A := \neg(B \rightarrow C)$. Since Σ is a closure under rules, so the rule ($\mathrm{R}_{\mathbf{DD}\neg\rightarrow}$) was applied. So (1) $B, \neg C \in \Sigma$ or (2) $\langle i, B\rangle, \langle \sim i, C\rangle \in \Sigma$ where i is a new label on the branch. If (1) happens, then by the inductive hypothesis $\mathfrak{M} \models B$ and $\mathfrak{M} \models \neg C$. So, $\mathfrak{M} \models \neg(B \rightarrow C)$. If (2) happens, then by the inductive hypothesis $i \in s_{\Sigma}(C)$ and $i \notin s_{\Sigma}(B)$, so $s_{\Sigma}(B) \not\subseteq s_{\Sigma}(C)$. Thus in both cases $\mathfrak{M} \models \neg(B \rightarrow C)$, by the definition of model 2.1.

Let us assume $\Lambda = \mathbf{D}$. The proofs of the cases for implication and negated implication are similar to the proofs for $\Lambda = \mathbf{DD}$. We examine the tableau rules ($\mathrm{R}_{\rightarrow}$), ($\mathrm{R}_{\mathbf{DD}\rightarrow}$), and ($\mathrm{R}_{\mathbf{DD}\neg\rightarrow}$), under assumption that in the **D**-models we have the condition of the inverse inclusion of contents for the implication.

Let us assume $\Lambda = \mathbf{Eq}$. The case for implication in **Eq** includes cases for **D** and **DD**. In **Eq**-models, in the truth condition for implication we have the identity of contents, so two inclusions. So, we would have to check rules: ($\mathrm{R}_{\mathbf{D}\rightarrow}$) and ($\mathrm{R}_{\mathbf{DD}\rightarrow}$). For the case of negated implication, we have the tableau rule ($\mathrm{R}_{\mathbf{Eq}\neg\rightarrow}$). The checking of the tree possibilities reduces to what we have already done for the negated implication case in **DD** plus the inverse rule (and the inverse inclusion) in the case of **D**.

Let us assume $\Lambda = \mathbf{S}$.

Let $A := B \rightarrow C$. Since Σ is a closure under tableau rules, ($\mathrm{R}_{\mathbf{S}\rightarrow}$) was applied and $\langle i, B\rangle, \langle i, C\rangle \in \Sigma$, where i is new on the branch. By the definition of generated model 3.7, $i \in s_{\Sigma}(B)$ and $i \in s_{\Sigma}(C)$, so $i \in s_{\Sigma}(B) \cap s_{\Sigma}(C)$ which fulfills one part of the truth condition for the implication in **S**-models: $s_{\Sigma}(B) \cap s_{\Sigma}(C) \neq \emptyset$. Since Σ is a closure under tableau rules, so also ($\mathrm{R}_{\rightarrow}$) was also applied and (1) $\neg B$ or (2) $C \in \Sigma$. If (1) happens, then by the inductive

hypothesis $\mathfrak{M} \models \neg B$. If (2) happens, then by the inductive hypothesis $\mathfrak{M} \models C$. Regardless of whether (1) or (2) happens, putting all facts together, we have $\mathfrak{M} \models B \to C$.

Let $A := \neg(B \to C)$. Since Σ is a closure under tableau rules, ($R_{\mathbf{S}\neg\to}$) was applied and (1) $B, \neg C \in \Sigma$ or (2) $\langle i, B\rangle, \langle j, \neg C\rangle \in \Sigma$, where i, j are new on the branch and $i \neq j$. The case (1) was examined for **DD**. For case (2) let us notice that i and j are different and if $\langle k, B\rangle \in \Sigma$ (or $\langle k, C\rangle \in \Sigma$) then $\langle \sim k, C\rangle \in \Sigma$ (or $\langle k, \sim B\rangle \in \Sigma$), by the tableau rule ($R_{\mathbf{S}}$). So, by the definition of generated model 3.7, $s_\Sigma(B) \cap s_\Sigma(C) = \emptyset$. Whether (1) or (2), $\mathfrak{M} \models \neg(B \to C)$.

Let us assume $\Lambda = \mathbf{R}$.

Let $A := B \to C$ and $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ (if $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ then the case is similar to the former logics, since then ($R_\to$) was applied). Since Σ is a closure under tableau rules, ($R_{\mathbf{R}\to}$) was applied and $\langle k_i, C_j\rangle \in \Sigma$, where $k_i \in \mathsf{n}(\mathsf{var}(A))$ and $B_j \in \mathsf{var}(C)$. By the definition of generated model 3.7, $k_i \in s_\Sigma(C)$, which fulfills one part of the truth condition for the implication in **R**-models. Since Σ is a closure under tableau rules, so also ($R_\to$) was also applied and (1) $\neg B$ or (2) $C \in \Sigma$. If (1) happens, then by the inductive hypothesis $\mathfrak{M} \models \neg B$. If (2) happens, then by the inductive hypothesis $\mathfrak{M} \models C$. Regardless of whether (1) or (2) happens, putting all facts together, we have $\mathfrak{M} \models B \to C$.

Let $A := \neg(B \to C)$. We have two possibilities: (a) $\mathsf{var}(A) \cap \mathsf{var}(B) = \emptyset$ and the rule ($R_{\mathbf{R}\neg\to}(1)$) was applied, (b) $\mathsf{var}(A) \cap \mathsf{var}(B) \neq \emptyset$ and the rule ($R_{\mathbf{R}\neg\to}(2)$) was applied, since Σ is a closure under rules. In case of (a) we have two outputs: (a1) B, $\neg C$ and (a2) $\sim k_1, C_1, \ldots \sim k_m, C_1, \ldots \sim k_1, C_n, \ldots \sim k_m, C_n$, where $\mathsf{n}(\mathsf{var}(B)) = \{k_1, \ldots, k_m\}$ and $\mathsf{var}(C) = \{C_1, \ldots, C_n\}$. In case of (a1), by the inductive hypothesis, $\mathfrak{M} \models B$ and $\mathfrak{M} \models \neg C$. So, $\mathfrak{M} \models \neg(B \to C)$. In case of (a2), by the definition of generated model 3.7, $k_i \notin s_\Sigma(C)$, for all $k_i \in \mathsf{n}(\mathsf{var}(B))$, so it is not that $R_{\mathbf{R}}(B, C)$. Thus, $\mathfrak{M} \models \neg(B \to C)$. In case of (b) by the inductive hypothesis $\mathfrak{M} \models B$ and $\mathfrak{M} \models \neg C$. So, $\mathfrak{M} \models \neg(B \to C)$. Hence, in all cases $\mathfrak{M} \models \neg(B \to C)$, by the definition of model 2.1. □

Finally, we can prove the main theorem.

Theorem 3.10 (Adequacy theorem). *Let* $\Sigma \cup \{A\} \subseteq \mathsf{For}$ *and* $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$. *Then,* $\Sigma \rhd_\Lambda A$ *iff* $\Sigma \models_\Lambda A$.

Proof. Assume all the hypotheses.

($\Rightarrow$) Suppose $\Sigma \rhd_\Lambda A$. Hence, by Definition 3.3, there is a finite $\Gamma \subseteq \Sigma$ such that any $\mathsf{tr}(\Lambda)$-closure of $\Gamma \cup \{\neg A\}$ is branch inconsistent. Suppose there is a Λ-model $\mathfrak{M} = \langle v, s\rangle$ such that $\mathfrak{M} \models \Sigma$ and $\mathfrak{M} \models \neg A$. By Definition 3.5, $\mathfrak{M}$ is suitable for $\Gamma \cup \{\neg A\}$. By Lemma 3.6 there is $\mathsf{tr}(\Lambda)$-closure Δ of $\Gamma \cup \{\neg A\}$ for which $\mathfrak{M}$ is suitable. However, such a closure must be branch inconsistent. Thus in the closure either there is $B \in \mathsf{For}$ such that $\mathfrak{M} \models B$ and $\mathfrak{M} \not\models B$ or there are $\langle i, B\rangle, \langle \sim i, B\rangle \in \mathsf{Ae}$ and under some function f – since $\mathfrak{M}$ is suitable to Δ (Definition 3.5) – $\mathsf{f}(i) \in s(B)$ and $\mathsf{f}(i) \notin s(B)$. Hence, for any Λ-model $\mathfrak{M}$: $\mathfrak{M} \models \Sigma$ implies $\mathfrak{M} \models A$.

($\Leftarrow$) Suppose $\Sigma \not\rhd_\Lambda A$. So, for any finite $\Gamma \subseteq \Sigma$ there is a branch consistent $\mathsf{tr}(\Lambda)$-closure Δ of $\Gamma \cup \{\neg A\}$, such that $\Gamma \cup \{\neg A\} \subseteq \Delta$. Hence, there is a branch consistent $\mathsf{tr}(\Lambda)$-closure Δ' such that $\Sigma \cup \{\neg A\} \subseteq \Delta'$. Otherwise, any of such a closure would consist some branch inconsistency. But by Definition 3.3 this would mean that for some finite $\Gamma \subseteq \Sigma$ no $\mathsf{tr}(\Lambda)$-closure of $\Gamma \cup \{\neg A\}$ is branch consistent. As a consequence, by Lemma 3.9, there is a Γ-model for Λ such that $\mathfrak{M} \models \Sigma \cup \{\neg A\}$. Therefore, by Definition 2.1, $\Sigma \models_\Lambda A$. □

Applying the previous theorem, we can obtain the decidability of the logics under study. Since a logic $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$ is adequate to the tableau system defined by $\mathsf{tr}(\Lambda)$ (Theorem 3.11), it is sufficient to prove that $\rhd_\Lambda$ is decidable. Assuming that the Λ-logic tableau rules can be applied a finite number of times, we get only finite closures under these rules that need to be examined. So, we have the theorem.

Theorem 3.11. (Decidability theorem). *Let* $\Lambda \in \{\mathbf{D}, \mathbf{DD}, \mathbf{Eq}, \mathbf{S}, \mathbf{R}\}$. *Then* Λ *is decidable.*

4. Conclusion

We proposed tableau systems for five logical systems defined by set-assignment semantics. Our approach combines two aspects: the classic truth aspect and a syllogistic aspect. In case of expressions with labels we treat formulas as names and labels as denotations. Additionally, our tableaux are finite. But different non-classical logics, like modal logics **K**, **T**, **KTB**, **S4**, **S5** and **S4Grz**; intuitionistic logics **IPL** and Johansson's **MIL**; paraconsistent logic $\mathbf{J_3}$; and many-valued logics $\mathbf{Ł_3}$, $\mathbf{Ł}_n$, $\mathbf{Ł}_{\aleph_0}$, $\mathbf{K}_3$, $\mathbf{G}_3$ and $\mathbf{G}_{\aleph_0}$ have set-assignment semantics as well (see [6,16,17]). Thus, the question of future research is to extend the proposed framework to these systems and to face the problem of infinite tableaux, which, for example, occurs in the case of transitivity of accessibility relation.

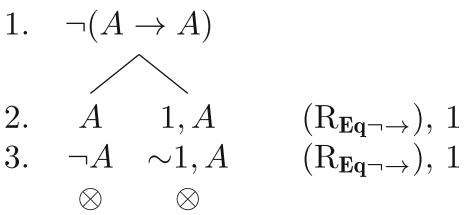

FIGURE 1. For any $A \in \mathsf{For}$, $\rhd_{\mathbf{Eq}} A \rightarrow A$

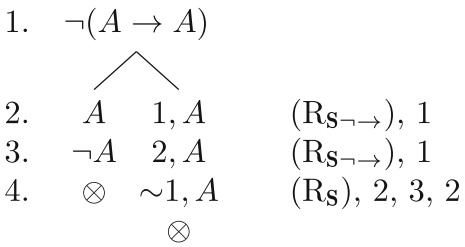

FIGURE 2. For any $A \in \mathsf{For}$, $\rhd_{\mathbf{S}} A \rightarrow A$

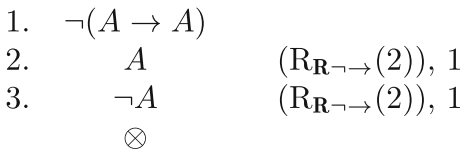

FIGURE 3. For any $A \in \mathsf{For}$, $\rhd_{\mathbf{R}} A \rightarrow A$

Appendix

In Appendix we would like to present examples of tableau proofs for the considered logics. Let us focus on the formulas (1) and (4) introduced in the section devoted to semantic presentation of Epstein's logics. For starters, let us note that any formula of the form $A \rightarrow A$ is a valid formula for any of Epstein's logic. We can easily check that any time we get $\neg(A \rightarrow A)$ on the branch we can close it. We consider this situation only for **Eq**, **S** and **R** (see Figs. 1, 2 and 3).

In what follows we present tableaux:

- for (1): Figs. 4, 6, 8 and 10 (we can read off the tableau proof in the system of **DD** from the tableau proof in the system of **Eq**)
- for (4): Figs. 5, 7, 9 and 11 (the tableau proof in the system of **DD** is similar to the tableau proof in the system of **D**).

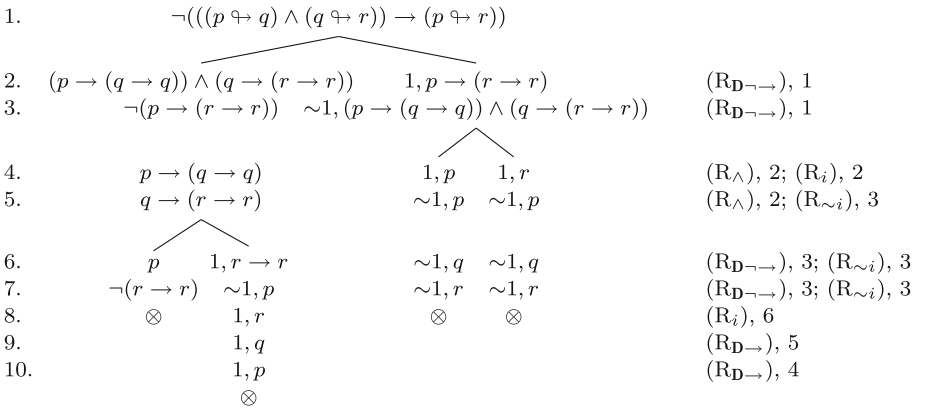

FIGURE 4. $\rhd_{\mathbf{D}}((p \curvearrowright q) \wedge (q \curvearrowright r)) \rightarrow (p \curvearrowright r)$

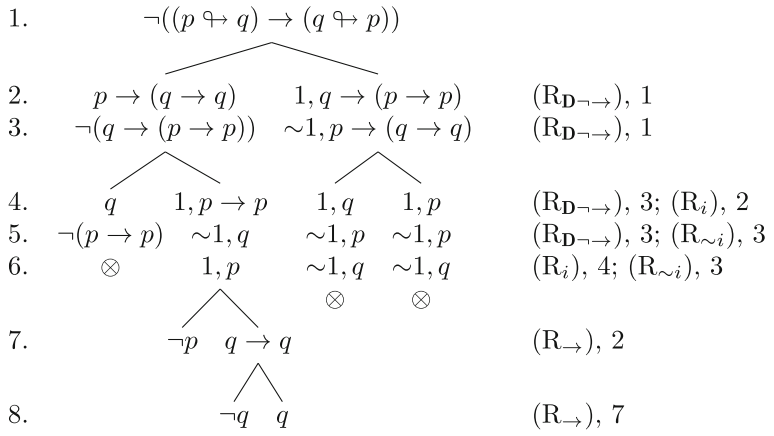

FIGURE 5. $\not\rhd_{\mathbf{D}}(p \curvearrowright q) \rightarrow (q \curvearrowright p)$

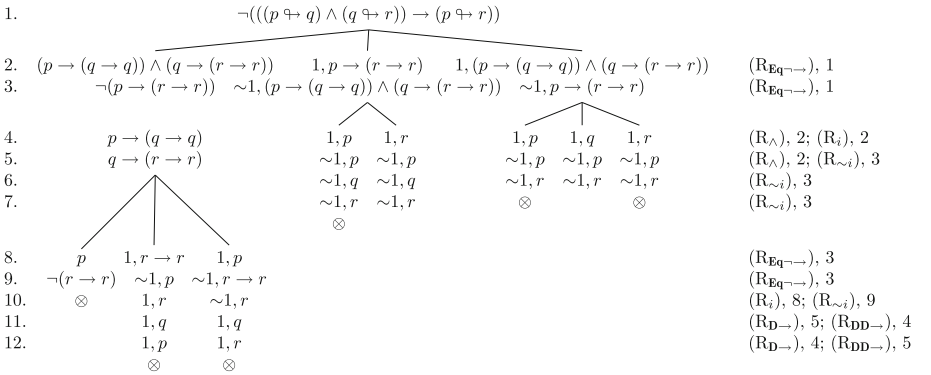

FIGURE 6. $\not\rhd_{\mathbf{Eq}}((p \curvearrowright q) \wedge (q \curvearrowright r)) \rightarrow (p \curvearrowright r)$

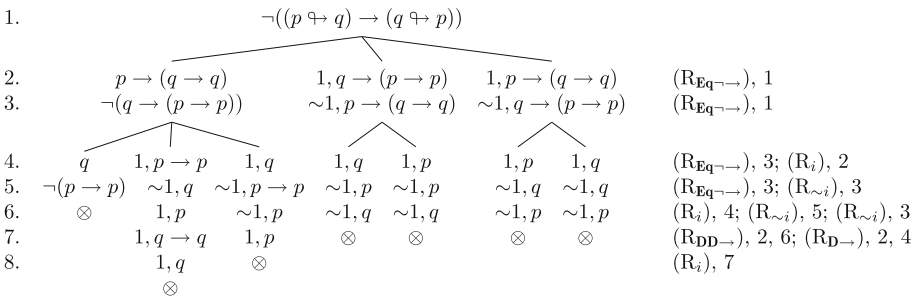

FIGURE 7. $\rhd_{\mathbf{Eq}}(p \looparrowright q) \rightarrow (q \looparrowright p)$

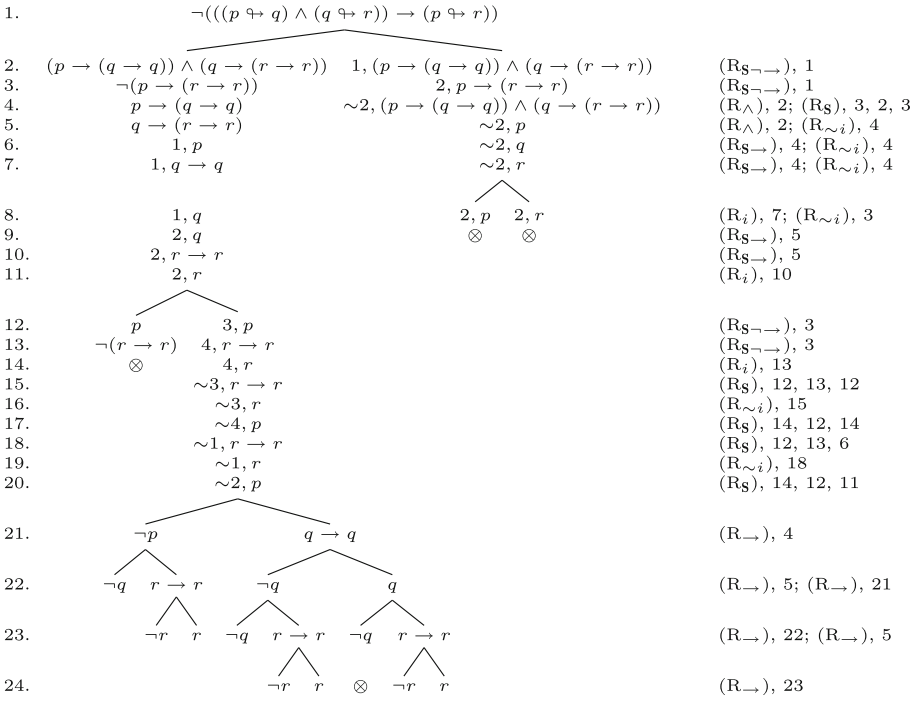

FIGURE 8. $\not\rhd_{\mathbf{S}}((p \looparrowright q) \wedge (q \looparrowright r)) \rightarrow (p \looparrowright r)$

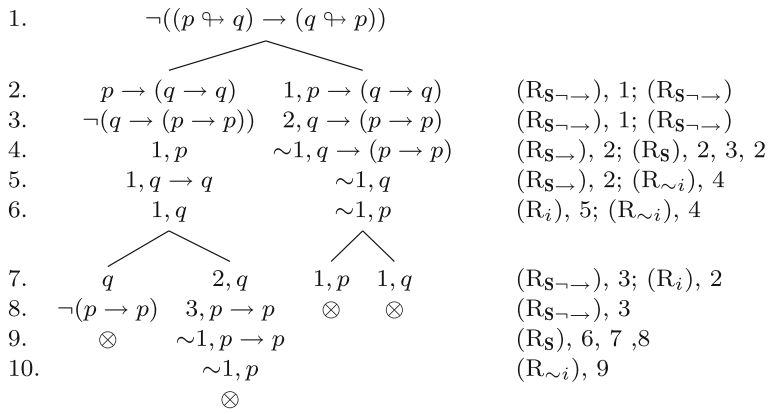

FIGURE 9. $\rhd_{\mathbf{S}}(p \looparrowright q) \rightarrow (q \looparrowright p)$

1.	$\neg(((p \looparrowright q) \wedge (q \looparrowright r)) \rightarrow (p \looparrowright r))$	
2.	$(p \looparrowright q) \wedge (q \looparrowright r)$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$, 1
3.	$\neg(p \rightarrow (r \rightarrow r))$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$, 1
4.	$(p \rightarrow (q \rightarrow q))$	$(\mathrm{R}_{\wedge})$, 2
5.	$(q \rightarrow (r \rightarrow r))$	$(\mathrm{R}_{\wedge})$, 2
6.	$\mathsf{n}(p), q$	$(\mathrm{R}_{\mathbf{R}\rightarrow})$, 4
7.	$\mathsf{n}(q), r$	$(\mathrm{R}_{\mathbf{R}\rightarrow})$, 5
8.	$p \quad \sim\mathsf{n}(p), r$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$, 3
9.	$\neg(r \rightarrow r)$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$, 3
	$\otimes$	

FIGURE 10. $\not\rhd_{\mathbf{R}}((p \looparrowright q) \wedge (q \looparrowright r)) \rightarrow (p \looparrowright r)$

1.	$\neg((p \looparrowright q) \rightarrow (q \looparrowright p))$	
2.	$p \rightarrow (q \rightarrow q)$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$, 1
3.	$\neg(q \rightarrow (p \rightarrow p))$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(2))$, 1
4.	$\mathsf{n}(p), q$	$(\mathrm{R}_{\mathbf{R}\rightarrow})$, 2
5.	$q \quad \sim\mathsf{n}(q), p$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$, 3
6.	$\neg(p \rightarrow p)$	$(\mathrm{R}_{\mathbf{R}\neg\rightarrow}(1))$, 3
	$\otimes$	

FIGURE 11. $\not\rhd_{\mathbf{R}}(p \looparrowright q) \rightarrow (q \looparrowright p)$

References

[1] Carnielli, W.A.: Methods of proofs of relatedness and dependence logic. Rep. Math. Logic **21**, 35–46 (1987)

[2] Del Cerro, L.F., Lugardon, V.: Quantification and dependence logics. In: Akama, S. (ed.) Logic, Language and Computation. Applied Logic Series, vol. 5, pp. 179–190. Springer Science+Business Media, Dordrecht (1997)

[3] Del Cerro, L.F., Lugardon, V.: Sequents for dependence logics. Logique Anal. **133/134**, 57–71 (1991)

[4] Epstein, R.L.: Relatedness and implication. Philos. Stud. **36**, 137–173 (1979)

[5] Epstein, R.L.: The algebra of dependence logic. Rep. Math. Logic **21**, 19–34 (1987)

[6] Epstein, R.L.: The Semantic Foundations of Logic. Volume 1: Propositional Logics. Springer Science+Business Media, Dordrecht (1990)

[7] Jarmużek, T.: Relating semantics as fine-grained semantics for intensional propositional logics. In: Giordani, A., Malinowski, J. (eds.) Logic in High Definition, Trends in Logical Semantics, vol. 56, pp. 13–30. Springer, Cham (2021)

[8] Jarmużek, T.: Tableau Methods for Propositional Logic and Term Logic. Peter Lang GmbH Internationaler Verlag der Wissenschaften, Berlin (2020)

[9] Jarmużek, T., Kaczkowski, B.: On some logic with a relation imposed on formulae: Tableau system $\mathcal{F}$. Bull. Sect. Logic **43**(1/2), 53–72 (2014)

[10] Jarmużek, T., Klonowski, M.: On logics of strictly-deontic modalities. A semantic and Tableau approach. Logic Log. Philos. **29**(3), 335–380 (2020)

[11] Jarmużek, T., Klonowski, M.: Some intensional logics defined by relating semantics and tableau systems. In: Giordani, A., Malinowski, J. (eds.) Logic in High Definition, Trends in Logical Semantics, vol. 56, pp. 31–48. Springer, Cham (2021)

[12] Jarmużek, T., Malinowski, J.: Boolean connexive logics: semantics and tableau approach. Logic Log. Philos. **28**(3), 427–448 (2019)

[13] Jarmużek, T., Malinowski, J.: Modal Boolean connexive logics: semantics and tableau approach. Bull Sect. Logic **48**(3), 213–243 (2019)

[14] Jarmużek, T., Goré, R.: Tableau metatheory for syllogistic logics. In: Fitting, M. (ed.) Selected topics from contemporary logics, pp. 539–582. College Publications, London (2021)

[15] Klonowski, M.: A Post-style proof of completeness theorem for Symmetric Relatedness Logic S. Bull. Sect. Logic **47**(3), 201–214 (2018)

[16] Klonowski, M.: History of relating logic. The origin and research directions. Logic Log. Philos. **30**(4), 579–629 (2021)

[17] Krajewski, S.: One or many logics? (Epstein's set-assignement semantics for logical calculi). J. Non-Classical Logic **8**(1), 7–33 (1991)

[18] Ledda, A., Paoli, F., Baldi, M.P.: Algebraic analysis of demodalised analytic implication. J. Philos. Log. **48**, 957–979 (2019)

[19] Malinowski, J., Palczewski, R.: Relating semantics for connexive logic. In: Giordani, A., Malinowski, J. (eds.) Logic in High Definition, Trends in Logical Semantics, vol. 56, pp. 49–65. Springer, Cham (2021)

[20] Paoli, F.: Semantics for first degree relatedness logic. Rep. Math. Logic **27**, 81–94 (1993)

[21] Paoli, F.: S is constructively complete. Rep. Math. Logic **30**, 31–47 (1996)

[22] Paoli, F.: Tautological entailments and their rivals. In: Béziau, J.Y., Carnielli, W.A., Gabbay, D.M. (eds.) Handbook of Paraconsistency, pp. 153–175. College Publications, London (2007)

Tomasz Jarmużek and Mateusz Klonowski
Department of Logic, Nicolaus Copernicus University
Moniuszki 16
87-100 Toruń
Poland
e-mail: jarmuzek@umk.pl;
mateusz.klonowski@umk.pl

Received: December 4, 2021.

Accepted: December 20, 2021.