

**SOFTWARE VERIFICATION RESEARCH CENTRE
DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY OF QUEENSLAND**

**Queensland 4072
Australia**

TECHNICAL REPORT

No. 94-18

Tableau Style Proof Systems for Various Many-Valued Logics

Anthony Bloesch

April 1994

**Phone: +61 7 365 1003
Fax: +61 7 365 1533**

Tableau Style Proof Systems for Various Many-Valued Logics

Anthony Bloesch
email: anthonyb@cs.uq.oz.au.

April 1994

Abstract

Tableau style proof systems for many-valued logics have either been nonexistent or more complex than they need be. Here we demonstrate how good tableau style proof systems may be constructed for many-valued logics and give proof systems for the many-valued logics \mathbf{L}_3 , \mathbf{K}_3 , \mathbf{Q}_3 , \mathbf{G}_3 , \mathbf{S}_3 , \mathbf{S}_3'' , $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{T}}$. The techniques that are described may also be used to construct, among others, resolution and connection method style proof systems for many-valued logics.

1 Introduction

Many-valued logic grew out of the work of Lukasiewicz [34–39] and Post [51]. Lukasiewicz was motivated by philosophical issues such as the problem of future contingents. Post was motivated by mathematical issues such as generality. Briefly, whereas classical logic has just two truth values, many-valued logics have three or more truth values. Many-valued logics arise naturally out of many issues within logic. For example, while we might say that the proposition “Alice had tea with the Mad Hatter.” is true, what are we to say about the proposition “Alice had more hairs on her right arm than her left.”? Is it true, false, neither true nor false, or perhaps both true and false? If we choose to say it is neither true nor false since we simply cannot tell, then we need a new truth value \mathbf{i} for neither true nor false; and we need to modify our semantics to deal with \mathbf{i} .

Several many-valued logics have been developed. The most famous, and probably the earliest, is Lukasiewicz’s 3-valued logic (\mathbf{L}_3) [34–36, 38, 39, 62] which is intended to deal with future contingents (i.e. statements like “It is possible that I will be in Warsaw on the 21st of December.”). Lukasiewicz generalized the logic to n -valued and even an infinite-valued logic [37, 48]. Post defined n -valued logics where the truth values consist of the set $\{0, 1, \dots, n-1\}$, and the operators \vee and \neg are primitives defined so that $\nu(\alpha \vee \beta) = \min(\nu(\alpha), \nu(\beta))$ and $\nu(\neg \alpha) = \nu(\alpha + 1) \pmod{n}$ [51]. Bochvar developed a many-valued logic having a new value \mathbf{i} (meaningless) to avoid the logical paradoxes [11, 14, 15]. Sobociński developed a 3-valued logic (\mathbf{S}_3) [19, 59, 61, 65] that avoids many of the paradoxes of material implication. Kleene, motivated by consideration of the theory of recursive functions, developed two 3-valued logics called the systems of strong (\mathbf{K}_3) and weak connectives (Kleene’s weak 3-valued logic is identical to Bochvar’s logic) [30]. In developing the logic \mathbf{K}_3 , Kleene assumed parallel evaluation of the logical connectives. If instead we assume sequential evaluation (left-hand side first) then we have the logic commonly known as lisp-logic.

Often the development of a many-valued logic is motivated by nonphilosophical concerns. For example, Gödel, in proving that intuitionistic propositional logic is not finitely valued, developed a sequence of many-valued logics ($\mathbf{G}_2, \mathbf{G}_3, \dots$) [21]. Also, Shupecki developed a 3-valued logic \mathbf{S}_3'' [29] with the peculiarity that classical propositional logic (\mathbf{PC}) is part of it. And, as part of a survey of many-valued logics, Rescher [55] developed two 3-valued logics called T-split ($\tilde{\mathbf{T}}$) and F-split ($\tilde{\mathbf{F}}$).

Quantum mechanics poses special problems for logic—the truth or falsity of a predicate may be unknowable [53, 54]. For example, the act of measuring the truth of a predicate P may prevent the measurement of the truth of a *conjugate* proposition P' . Or, more strongly, in some interpretations of quantum mechanics, propositions P and Q may both be considered meaningful but the statement $P \wedge Q$ meaningless. In essence, quantum mechanics both limits the logical statements that can be

Table 2.1: Truth tables for the operators of \mathbf{L}_3 , where **t** denotes true, **f** false, and **i** indeterminate. The designated truth value is **t**.

			$\alpha \wedge \beta$	$\alpha \vee \beta$	$\alpha \rightarrow \beta$	$\alpha \leftrightarrow \beta$
α	$\neg \alpha$	$\alpha \backslash \beta$	f t i	f t i	f t i	f t i
f	t	f	f f f	f t i	t t t	t f i
t	f	t	f t i	t t t	f t i	f t i
i	i	i	f i i	i t i	i t t	i i t

simultaneously asserted and requires new kinds of assertions (e.g. P and P' are conjugate). Consideration of such issues led Reichenbach to developed a 3-valued logic called quantum logic (\mathbf{Q}_3) [53, 54] with 10 operators: cyclic negation ($\bar{\neg}$), diametrical negation (\neg), complete negation ($-$), conjunction (\wedge), disjunction (\vee), standard implication (\rightarrow), alternative implication (\Rightarrow), quasi-implication (\leftrightarrow), standard equivalence (\leftrightarrow) and alternative equivalence (\Leftrightarrow). These operators can be combined to make useful assertions about propositions. For example, $P \vee \bar{\neg} P$

n-Valued Logics

n-valued logics (taken from [55]); it should be obvious. Essentially the semantics is just a generalization of classical logic. The definitions for validity, invalidity and

$$\mathcal{T} = \{\mathbf{t}, \mathbf{f}, \mathbf{i}\}.$$

function that maps the set of formulae to the set of truth values. For example, in the logic \mathbf{L}_3 , $\mathcal{T} = \{\mathbf{t}, \mathbf{f}, \mathbf{i}\}$.

member of the set \mathcal{D} of ‘true’ truth values (for the logics \mathbf{L}_3 and \mathbf{S}_3 , $\mathcal{D} = \{\mathbf{t}, \mathbf{i}\}$). A formula α is said

valid (iff in every valuation ν , α is designated; this is written as $\models \alpha$) and invalid (iff in some valuation ν , α is not designated; this is written as $\not\models \alpha$).

consequence of the set of formulae Σ iff in every valuation ν that satisfies Σ , α is also the case that α is designated; this is written as $\Sigma \models \alpha$. If α is not a logical consequence of a set of formulae Σ , we write $\Sigma \not\models \alpha$.

Tableaux for Various Many-Valued Logics

used for a variety of logics, both classical and nonclassical. The method is similar to modal logics [17, 18, 52]. As a proof technique, the method is similar to Beth’s [4–7]. Similar ideas to Beth’s can be seen in the work of Beth [4–7]. Similar ideas to Beth’s can be seen in the work of Beth [4–7]. Similar ideas to Beth’s can be seen in the work of Beth [4–7].

Table 2.2: Truth tables for the operators of \mathbf{K}_3 , where **t** denotes true, **f** false, and **i** indeterminate. The designated truth value is **t**.

α	$\neg\alpha$	$\alpha\backslash\beta$	$\alpha\wedge\beta$			$\alpha\vee\beta$			$\alpha\rightarrow\beta$			$\alpha\leftrightarrow\beta$		
f	t		f	t	i	f	t	i	f	t	i	f	t	i
f	t	f	f	f	f	f	t	i	t	t	t	t	f	i
t	f	t	f	t	i	t	t	t	f	t	i	f	t	i
i	i	i	f	i	i	i	t	i	i	t	i	i	i	i

Table 2.3: Truth tables for the operators of \mathbf{S}_3'' , where **t** denotes true, **f** false, and **i** unknown. The designated truth value is **t**.

α	$\neg\alpha$	$\exists\alpha$	$\alpha\backslash\beta$	$\alpha\rightarrow\beta$		
f	t	t		f	t	i
f	t	t	f	t	t	t
t	f	i	t	f	t	i
i	i	t	i	t	t	t

Table 2.4: Truth tables for the operators of \mathbf{Q}_3 , where **t** denotes true, **f** false, and **i** meaningless. The designated truth value is **t**.

α	$\neg\alpha$	$\exists\alpha$	$-\alpha$
f	t	t	t
t	f	i	i
i	i	f	t

$\alpha\backslash\beta$	$\alpha\wedge\beta$			$\alpha\vee\beta$			$\alpha\rightarrow\beta$			$\alpha\Rightarrow\beta$			$\alpha\leftrightarrow\beta$			$\alpha\leftrightarrow\beta$		
	f	t	i	f	t	i	f	t	i	f	t	i	f	t	i	f	t	i
f	f	f	f	f	t	i	t	t	t	i	i	i	t	f	i	t	f	f
t	f	t	i	t	t	t	f	t	i	f	t	f	f	t	i	f	t	f
i	f	i	i	i	t	i	i	t	t	t	i	i	i	i	t	f	f	t

Table 2.5: Truth tables for the operators of \mathbf{S}_3 , where **t** denotes true, **f** false, and **i** unknown. The designated truth values are **t** and **i**.

α	$\neg\alpha$	$\alpha\backslash\beta$	$\alpha\wedge\beta$			$\alpha\vee\beta$			$\alpha\rightarrow\beta$			$\alpha\leftrightarrow\beta$		
f	t		f	t	i	f	t	i	f	t	i	f	t	i
f	t	f	f	f	f	f	t	f	t	t	t	t	f	f
t	f	t	f	t	t	t	t	f	t	f	f	t	f	f
i	i	i	f	t	i	f	t	i	f	t	i	f	f	i

Table 2.6: Truth tables for the operators of $\tilde{\mathbf{T}}$, where **t** denotes true, **f** false, and **i** indeterminate. The designated truth values are **t** and **i**.

α	$\neg\alpha$	$\alpha \setminus \beta$	$\alpha \wedge \beta$			$\alpha \vee \beta$			$\alpha \rightarrow \beta$			$\alpha \leftrightarrow \beta$		
f	t		f	t	i	f	t	i	f	t	i	f	t	i
t	f	f	f	f	f	t	i	t	t	t	t	f	f	
i	f	t	f	t	i	t	t	t	f	t	i	f	t	i
		i	f	i	i	i	t	i	f	t	i	f	i	i

Table 2.7: Truth tables for the operators of $\tilde{\mathbf{F}}$, where **t** denotes true, **f** false, and **i** indeterminate. The designated truth value is **t**.

α	$\neg\alpha$	$\alpha \setminus \beta$	$\alpha \wedge \beta$			$\alpha \vee \beta$			$\alpha \rightarrow \beta$			$\alpha \leftrightarrow \beta$		
f	t		f	t	i	f	t	i	f	t	i	f	t	i
t	f	f	f	f	f	t	i	t	t	t	t	f	i	
i	t	t	f	t	i	t	t	t	f	t	i	f	t	i
		i	f	i	i	i	t	i	i	t	i	i	i	i

Table 2.8: Truth tables for the operators of \mathbf{G}_3 , where **t** denotes true, **f** false, and **i** indeterminate. The designated truth value is **t**.

α	$\neg\alpha$	$\alpha \setminus \beta$	$\alpha \wedge \beta$			$\alpha \vee \beta$			$\alpha \rightarrow \beta$			$\alpha \leftrightarrow \beta$		
f	t		f	t	i	f	t	i	f	t	i	f	t	i
t	f	f	f	f	f	t	i	t	t	t	t	f	f	
i	f	t	f	t	i	t	t	t	f	t	i	f	t	i
		i	f	i	i	i	t	i	f	t	t	f	i	t

Although, Smullyan [64] is often credited with the development of modern tableaux in fact Lis's [33] work predates it.

Traditional tableau methods can be seen as a search for an instantiation that satisfies premises of an argument together with the negation of the conclusion of the argument. If it can be shown that no such instantiation exists then the argument is taken to be valid in the relevant logic (i.e. tableau based proof systems are refutation systems). Many logics do not have, in the appropriate sense, the law of the excluded middle ($\neg\alpha\vee\alpha$). Thus we use a more general notion of a tableau proof. A tableau proof of an argument will be seen as a failed search for an instantiation in which the premises of the argument are designated true and the conclusion is not designated true. Clearly, the search technique must never report that no instantiation exists when in fact one does (i.e. the proof system must be sound). Conversely, it is desirable that if an instantiation exists the search procedure will be guaranteed to find it (i.e. the proof system should be complete).

3.2 A Formal Framework for Tableau

We characterize a tableau (dis)proof of, say, $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$ as follows:

- A tableau (dis)proof consists of an n-ary tree of *nodes*.
- The acyclic paths from the root to the leaves are the tableau's *branches*. Each branch may have associated with it certain simple auxiliary information such as an accessibility relation between worlds.
- Each node consists of a finite set of *signed formulae*.
- Each signed formula consists of a *truth sign* (drawn from a countable set of truth signs) and a finite formula of the underlying logic.
- Associated with each truth sign–operator pair is a *decomposition rule*. The decomposition rule states which new branches are to be created, which formulae are to appear on those branches, and how the auxiliary information is to change. In the case of an impossible truth sign–operator pair the decomposition rule may state that the branch closes immediately.
- Some branches of the tableau may be marked as *closed*. If all the branches are marked as closed then the entire tableau is closed and $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$ is proved.
- For a branch to be closed it must contain two signed formulae $\Lambda_1\alpha$ and $\Lambda_2\alpha$ where Λ_1 and Λ_2 are *opposite*, an impossible truth sign–operator pair, or inconsistent auxiliary information.
- If some branch of a tableau is not marked as closed and no decomposition rule can be applied to a formula on it, then we say that the branch is *open*. The formulae on such a branch form a counterexample to $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$.

We characterize decomposition rules as follows:

α rules Rules of the form:

$$\frac{\alpha}{\beta_1}$$

$$\vdots$$

$$\beta_n$$

which mean: if α is on a branch then $\beta_1, \beta_2, \dots, \beta_n$ may be added to the branch.

β rules Rules of the form:

$$\frac{\alpha}{\begin{array}{c|c|c} \beta_{1,1} & \dots & \beta_{1,m} \\ \vdots & \ddots & \vdots \\ \beta_{n_1,1} & \dots & \beta_{n_m,m} \end{array}}$$

which mean: if α is on a branch then we may create $m \geq 2$ new subbranches where branch i has $\beta_{1,i}, \beta_{2,i}, \dots, \beta_{n_i,i}$ added to it.

We characterize a tableau construction procedure for, say, $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$ as follows:

1. Set the initial set of signed formulae in the root node of the tableau to:

$$\{\Lambda_1 \gamma_1, \Lambda_2 \gamma_2, \dots, \Lambda_n \gamma_n, \Lambda_{n+1} \delta\}.$$

Normally we expect that $\Lambda_1 = \Lambda_2 = \dots = \Lambda_n \neq \Lambda_{n+1}$. Set the current branch to the root node and the set of unexplored branches to the empty set.

2. Apply a decomposition rule to a formula chosen *fairly* from the set of signed formulae on the current branch. By fair choice we mean that we should choose formulae in such a way that eventually any given signed formula on the branch will be chosen.
3. As we add the signed formulae, generated by the decomposition rules, to the current branch we test to see if they are opposite to some signed formulae already on the branch. If we find such a formula we mark the branch as closed.
4. If the branch is closed then set the current branch to some unexplored branch (removing the branch from the list) and go to step 2 (if the set of unexplored branches is empty then $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$ is proved).
5. If subbranches were generated by the decomposition rule then set the current branch to one of them and add the other branch(es) to the list of unexplored branches.
6. Go to step 2.

3.3 The Tableau Proof Systems

By the use of signed formulae it is possible to construct tableau proof systems for $\mathbf{L}_3, \mathbf{K}_3, \tilde{\mathbf{F}}, \mathbf{Q}_3, \mathbf{G}_3, \mathbf{S}_3'', \tilde{\mathbf{T}}$ and \mathbf{S}_3 (figures 3.1–3.8). No tableau systems have previously been developed for $\tilde{\mathbf{F}}, \mathbf{Q}_3, \mathbf{G}_3, \mathbf{S}_3'', \tilde{\mathbf{T}}$ and \mathbf{S}_3 . The existing tableau systems for \mathbf{L}_3 [23, 60] and \mathbf{K}_3 [56] (actually a temporal variant of \mathbf{K}_3) produce larger and more complex tableaux than the new systems given here. Also, general schemes exist for constructing tableau based proof systems for many-valued logics [13]. These schemes amount to assigning a truth sign for every truth value. The resulting tableau proofs tend to be more complex (containing more branches and formulae) than the proofs generated using the tableau systems given here.

Definition 3.1 A *truth sign* is one of the set $\{\mathbf{T}, \mathbf{F}, \overline{\mathbf{T}}, \overline{\mathbf{F}}, \mathbf{I}\}$. Note: $\mathbf{I}, \overline{\mathbf{T}}$ and $\overline{\mathbf{F}}$ are interpreted as indeterminate/unknown (**i**), not true (**f** or **i**) and not false (**t** or **i**), respectively.

To prove that $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$, in logics $\mathbf{L}_3, \mathbf{K}_3, \tilde{\mathbf{F}}, \mathbf{Q}_3, \mathbf{G}_3$ and \mathbf{S}_3'' , we create a tableau branch where for each γ_i we add $\mathbf{T} \gamma_i$ to the branch and we add $\overline{\mathbf{T}} \delta$ to the end of the branch. To prove that $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$, in logics $\tilde{\mathbf{T}}$ and \mathbf{S}_3 , we create a tableau branch where for each γ_i we add $\overline{\mathbf{F}} \gamma_i$ to the branch and we add $\mathbf{F} \delta$ to the end of the branch. Then we apply the decomposition rules to each branch until it either becomes *closed* (contains two signed formulae $\Lambda_1 \alpha$ and $\Lambda_2 \alpha$ where Λ_1 and Λ_2 are opposite in the sense of table 3.1 or table 3.2; or, in the case of \mathbf{S}_3'' , if rule $\mathbf{F}_=$ is applied; or, in the case of \mathbf{Q}_3 , if rule \mathbf{F}_- is applied) or is *open* (a branch is not closed and no rule can be applied to a signed formulae on that branch to produce a signed formula new to that branch). Because of the finitary nature of the formulae and associated rules, a branch will always become either closed or open after a finite number of steps.

Figure 3.9 contains, for the logic \mathbf{K}_3 , both an example proof of $\text{PAR}, \text{P} \rightarrow \text{Q} \models \text{Q}$ and an open tableau showing $\not\models \text{PV} \rightarrow \text{P}$. The first proof is constructed as follows:

$\frac{\mathbf{T} \neg \alpha}{\mathbf{F} \alpha}$	$\frac{\mathbf{F} \neg \alpha}{\mathbf{T} \alpha}$	$\frac{\overline{\mathbf{T}} \neg \alpha}{\overline{\mathbf{F}} \alpha}$	$\frac{\overline{\mathbf{F}} \neg \alpha}{\overline{\mathbf{T}} \alpha}$	
$\frac{\mathbf{T} \alpha \wedge \beta}{\overline{\mathbf{F}} \alpha \mid \mathbf{T} \alpha \mid \mathbf{T} \beta \mid \overline{\mathbf{F}} \beta}$	$\frac{\mathbf{F} \alpha \wedge \beta}{\mathbf{F} \alpha \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \wedge \beta}{\mathbf{F} \alpha \mid \mathbf{F} \beta \mid \mathbf{I} \alpha \mid \mathbf{I} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \wedge \beta}{\overline{\mathbf{F}} \alpha \mid \overline{\mathbf{F}} \beta}$	
$\frac{\mathbf{T} \alpha \vee \beta}{\mathbf{T} \alpha \mid \mathbf{T} \beta}$	$\frac{\mathbf{F} \alpha \vee \beta}{\mathbf{F} \alpha \mid \overline{\mathbf{T}} \alpha \mid \overline{\mathbf{T}} \beta \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \vee \beta}{\overline{\mathbf{T}} \alpha \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \vee \beta}{\mathbf{T} \alpha \mid \mathbf{T} \beta \mid \mathbf{I} \alpha \mid \mathbf{I} \beta}$	
$\frac{\mathbf{T} \alpha \rightarrow \beta}{\mathbf{F} \alpha \mid \mathbf{T} \beta}$	$\frac{\mathbf{F} \alpha \rightarrow \beta}{\mathbf{T} \alpha \mid \overline{\mathbf{F}} \alpha \mid \overline{\mathbf{T}} \beta \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \rightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \rightarrow \beta}{\mathbf{F} \alpha \mid \mathbf{T} \beta \mid \mathbf{I} \alpha \mid \mathbf{I} \beta}$	
$\frac{\mathbf{T} \alpha \leftrightarrow \beta}{\mathbf{T} \alpha \mid \mathbf{F} \alpha \mid \mathbf{T} \beta \mid \mathbf{F} \beta}$	$\frac{\mathbf{F} \alpha \leftrightarrow \beta}{\mathbf{F} \alpha \mid \overline{\mathbf{T}} \alpha \mid \mathbf{T} \alpha \mid \overline{\mathbf{F}} \alpha \mid \overline{\mathbf{T}} \beta \mid \mathbf{T} \beta \mid \overline{\mathbf{T}} \beta \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \leftrightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \overline{\mathbf{T}} \alpha \mid \overline{\mathbf{T}} \beta \mid \overline{\mathbf{F}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \leftrightarrow \beta}{\mathbf{T} \alpha \mid \mathbf{F} \alpha \mid \mathbf{T} \beta \mid \mathbf{F} \beta \mid \mathbf{I} \alpha \mid \mathbf{I} \beta}$	
$\frac{\mathbf{I} \neg \alpha}{\mathbf{I} \alpha}$	$\frac{\mathbf{I} \alpha \wedge \beta}{\mathbf{I} \alpha \mid \mathbf{I} \beta}$	$\frac{\mathbf{I} \alpha \vee \beta}{\mathbf{I} \alpha \mid \mathbf{I} \beta}$	$\frac{\mathbf{I} \alpha \rightarrow \beta}{\mathbf{I} \alpha \mid \mathbf{I} \beta}$	$\frac{\mathbf{I} \alpha \leftrightarrow \beta}{\mathbf{I} \alpha \mid \mathbf{I} \beta}$

Figure 3.3: Tableau decomposition rules for \mathbf{S}_3 .

$\frac{\mathbf{T} \neg \alpha}{\mathbf{F} \alpha}$	$\frac{\mathbf{F} \neg \alpha}{\mathbf{T} \alpha}$	$\frac{\overline{\mathbf{T}} \neg \alpha}{\overline{\mathbf{F}} \alpha}$	$\frac{\overline{\mathbf{F}} \neg \alpha}{\overline{\mathbf{T}} \alpha}$
$\frac{\mathbf{T} \neg \alpha}{\overline{\mathbf{T}} \alpha}$	\times	$\frac{\overline{\mathbf{T}} \neg \alpha}{\mathbf{T} \alpha}$	$\frac{\overline{\mathbf{F}} \neg \alpha}{\overline{\mathbf{F}} \alpha}$
$\frac{\mathbf{T} \alpha \rightarrow \beta}{\overline{\mathbf{T}} \alpha \mid \mathbf{T} \beta}$	$\frac{\mathbf{F} \alpha \rightarrow \beta}{\mathbf{T} \alpha \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \rightarrow \beta}{\mathbf{T} \alpha \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \rightarrow \beta}{\overline{\mathbf{T}} \alpha \mid \overline{\mathbf{F}} \beta}$

Figure 3.4: Tableau decomposition rules for \mathbf{S}_3'' . Note: rule $\mathbf{F} \neg$ closes the tableau branch.

Table 3.1: The opposite truth signs of \mathbf{L}_3 , \mathbf{K}_3 , $\overline{\mathbf{F}}$, \mathbf{Q}_3 , \mathbf{G}_3 , \mathbf{S}_3'' and $\overline{\mathbf{T}}$. For example, since \mathbf{T} and \mathbf{F} are opposite truth signs, if both $\mathbf{T} \alpha$ and $\mathbf{F} \alpha$ appear on a branch then that branch is closed.

\mathbf{F}	\mathbf{T}
\mathbf{F}	$\overline{\mathbf{F}}$
\mathbf{T}	$\overline{\mathbf{T}}$

$\frac{\mathbf{T} \neg \alpha}{\mathbf{F} \alpha}$	$\frac{\mathbf{F} \neg \alpha}{\overline{\mathbf{F}} \alpha}$	$\frac{\overline{\mathbf{T}} \neg \alpha}{\overline{\mathbf{F}} \alpha}$	$\frac{\overline{\mathbf{F}} \neg \alpha}{\mathbf{F} \alpha}$
$\frac{\mathbf{T} \alpha \wedge \beta}{\mathbf{T} \alpha \mid \mathbf{T} \beta}$	$\frac{\mathbf{F} \alpha \wedge \beta}{\mathbf{F} \alpha \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \wedge \beta}{\overline{\mathbf{T}} \alpha \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \wedge \beta}{\overline{\mathbf{F}} \alpha \mid \overline{\mathbf{F}} \beta}$
$\frac{\mathbf{T} \alpha \vee \beta}{\mathbf{T} \alpha \mid \mathbf{T} \beta}$	$\frac{\mathbf{F} \alpha \vee \beta}{\mathbf{F} \alpha \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \vee \beta}{\overline{\mathbf{T}} \alpha \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \vee \beta}{\overline{\mathbf{F}} \alpha \mid \overline{\mathbf{F}} \beta}$
$\frac{\mathbf{T} \alpha \rightarrow \beta}{\mathbf{F} \alpha \mid \mathbf{T} \beta \mid \overline{\mathbf{T}} \alpha \mid \overline{\mathbf{F}} \beta}$	$\frac{\mathbf{F} \alpha \rightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \mathbf{F} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \rightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \mathbf{T} \alpha \mid \mathbf{F} \beta \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \rightarrow \beta}{\mathbf{F} \alpha \mid \overline{\mathbf{F}} \beta}$
$\frac{\mathbf{T} \alpha \leftrightarrow \beta}{\mathbf{T} \alpha \mid \mathbf{F} \alpha \mid \overline{\mathbf{T}} \alpha \mid \mathbf{T} \beta \mid \mathbf{F} \beta \mid \overline{\mathbf{F}} \alpha \mid \overline{\mathbf{T}} \beta \mid \overline{\mathbf{F}} \beta}$	$\frac{\mathbf{F} \alpha \leftrightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \mathbf{F} \alpha \mid \mathbf{F} \beta \mid \overline{\mathbf{F}} \beta}$	$\frac{\overline{\mathbf{T}} \alpha \leftrightarrow \beta}{\mathbf{F} \alpha \mid \overline{\mathbf{T}} \alpha \mid \overline{\mathbf{F}} \alpha \mid \mathbf{T} \alpha \mid \overline{\mathbf{F}} \beta \mid \mathbf{T} \beta \mid \mathbf{F} \beta \mid \overline{\mathbf{T}} \beta}$	$\frac{\overline{\mathbf{F}} \alpha \leftrightarrow \beta}{\overline{\mathbf{F}} \alpha \mid \mathbf{F} \alpha \mid \overline{\mathbf{F}} \beta \mid \mathbf{F} \beta}$

Figure 3.8: Tableau decomposition rules for \mathbf{G}_3 .

Table 3.2: The opposite truth signs of \mathbf{S}_3 . For example, since \mathbf{T} and \mathbf{F} are opposite truth signs, if both $\mathbf{T} \alpha$ and $\mathbf{F} \alpha$ appear on a branch then that branch is closed.

\mathbf{F}	\mathbf{T}
\mathbf{F}	$\overline{\mathbf{F}}$
\mathbf{F}	\mathbf{I}
\mathbf{T}	\mathbf{I}
\mathbf{T}	$\overline{\mathbf{T}}$

1. $\mathbf{T} P \wedge R$		1. $\overline{\mathbf{T}} P \vee \neg P$
2. $\mathbf{T} P \rightarrow Q$		2. $\overline{\mathbf{T}} P$ (1)
3. $\overline{\mathbf{T}} Q$		3. $\overline{\mathbf{T}} \neg P$ (1)
4. $\mathbf{T} P$ (1)		4. $\overline{\mathbf{F}} P$ (3)
5. $\mathbf{T} R$ (1)		
6. $\mathbf{F} P$ (2)	┌──────────┐	
×		7. $\mathbf{T} Q$ (2)
	└──────────┘	×

Figure 3.9: Example proof of $P \wedge R, P \rightarrow Q \models_{\mathbf{K}_3} Q$ (left) and an open tableau showing $\not\models_{\mathbf{K}_3} P \vee \neg P$ (right). Note: the numbers in parentheses indicate the line from which the formula was generated.

1. Add the formulae $\mathbf{T} P \wedge R$, $\mathbf{T} P \rightarrow Q$ and $\overline{\mathbf{T}} Q$ to the branch to give lines 1, 2 and 3.
2. Decompose the formula $\mathbf{T} P \wedge R$ to give lines 4 and 5.
3. Decompose the formula $\mathbf{T} P \rightarrow Q$ to give the branches at lines 6 and 7.
4. Mark both branches closed since they both contain, in the sense of table 3.1, opposite signed formulae (i.e. $\mathbf{T} P$ and $\mathbf{F} P$; and $\overline{\mathbf{T}} Q$ and $\mathbf{T} Q$).

The open tableau is constructed in a similar way except that since it does not contain a pair of opposite formulae it cannot be closed and hence contains a counterexample. In this case, the presence of $\overline{\mathbf{T}} P$ tells us that $\nu(P) \in \{\mathbf{f}, \mathbf{i}\}$ and $\overline{\mathbf{F}} P$ tells us that $\nu(P) \in \{\mathbf{t}, \mathbf{i}\}$ thus we have a counterexample when $\nu(P) = \mathbf{i}$ which we can confirm by examining table 2.2.

4 Proof of Soundness

We now prove the soundness of the proof system for \mathbf{K}_3 . The proof can be easily modified to give a proof of soundness for the other logics.

Definition 4.1 A *truth sign* is one of the set $\{\mathbf{T}, \mathbf{F}, \overline{\mathbf{T}}, \overline{\mathbf{F}}\}$.

Definition 4.2 A *signed formula* is a formula together with a truth sign (e.g. $\mathbf{T} P$ means P is true).

Definition 4.3 The signed formula $\mathbf{T} \alpha$ is said to *hold* in a valuation ν iff $\nu(\alpha) \in \{\mathbf{t}\}$. Similarly, the signed formulae $\mathbf{F} \alpha$, $\overline{\mathbf{T}} \alpha$ and $\overline{\mathbf{F}} \alpha$ hold in a valuation ν iff $\nu(\alpha) \in \{\mathbf{f}\}$, $\nu(\alpha) \in \{\mathbf{f}, \mathbf{i}\}$ and $\nu(\alpha) \in \{\mathbf{t}, \mathbf{i}\}$, respectively.

Definition 4.4 A signed formula \mathcal{F} is *satisfiable* iff a valuation ν exists such that \mathcal{F} holds in ν . Similarly, a set of formulae is *satisfiable* iff all the formulae in it hold in a valuation ν .

Definition 4.5 A tableau branch is *satisfiable* iff a valuation ν exists such that each signed formula on the branch holds in ν .

Definition 4.6 A tableau is *satisfiable* iff at least one branch on it is satisfiable.

Definition 4.7 A pair of truth signs are *opposite* iff either one is \mathbf{F} and the other \mathbf{T} , or one is \mathbf{F} and the other $\overline{\mathbf{F}}$, or one is \mathbf{T} and the other $\overline{\mathbf{T}}$. Similarly, two signed formulae are opposite iff their formulae are identical and their truth signs are opposite.

Definition 4.8 A tableau is *closed* iff each branch on it contains at least two opposite signed formulae.

Definition 4.9 A tableau is *open* (i.e. unclosable) iff it contains a branch such that (1) no decomposition rule of figure 3.2 can be applied to a signed formula on that branch to yield a signed formula not already on that branch and (2) no two signed formulae on that branch are opposite.

Lemma 4.1 If a tableau \mathcal{T} is satisfiable then it will also be satisfiable after the application of any of the tableau formation rules of figure 3.2.

Proof Let \mathcal{T}' be the new tableau. Let \mathcal{B}^* be the branch to which the rule is applied. Since \mathcal{T} is satisfiable it must have a branch \mathcal{B} that is satisfiable. We have two cases:

1. $\mathcal{B}^* \neq \mathcal{B}$. Since no rule was applied to \mathcal{B} , \mathcal{B} is still a branch of \mathcal{T}' and thus \mathcal{T}' is satisfiable.
2. $\mathcal{B}^* = \mathcal{B}$. We show that for at least one branch produced by a rule all the new signed formulae hold. Let \mathcal{F} be the signed formula to which the rule is applied. Since \mathcal{B} is satisfiable \mathcal{F} must hold for some valuation ν . We have twenty cases. For brevity, we show only two:
 - (a) $\mathcal{F} = \mathbf{T}\alpha \wedge \beta$. The formulae added to the branch are $\mathbf{T}\alpha$ and $\mathbf{T}\beta$. Since $\mathbf{T}\alpha \wedge \beta$ holds we know that $\nu(\alpha \wedge \beta) \in \{\mathbf{t}\}$ and hence, by truth tables, that $\nu(\alpha) \in \{\mathbf{t}\}$ and $\nu(\beta) \in \{\mathbf{t}\}$. Thus $\mathbf{T}\alpha$ and $\mathbf{T}\beta$ must also hold and thus \mathcal{T}' is satisfiable.
 - (b) $\mathcal{F} = \mathbf{T}\alpha \rightarrow \beta$. The formulae added to the two new branches are $\mathbf{F}\alpha$ and $\mathbf{T}\beta$. We show that at least one of them holds. Since $\mathbf{T}\alpha \rightarrow \beta$ holds we know that either $\nu(\alpha) \in \{\mathbf{f}\}$ or $\nu(\beta) \in \{\mathbf{t}\}$. Thus either $\mathbf{F}\alpha$ or $\mathbf{T}\beta$ must hold and thus \mathcal{T}' is satisfiable. \square

Lemma 4.2 If there is a closed tableau for a set of signed formulae Γ then Γ is not satisfiable.

Proof Assume that Γ is satisfiable and a closed tableau exists for Γ . We show a contradiction. Let \mathcal{T} be a tableau consisting of the signed formulae of Γ on a single branch. Since Γ is satisfiable, \mathcal{T} is satisfiable. By lemma 4.1 every new tableau we construct from \mathcal{T} is also satisfiable, including the final closed tableau. But examination of the closure conditions for \mathbf{K}_3 (table 3.1) shows that no closed tableau can be satisfiable. Thus we have a contradiction. \square

Theorem 4.1 The above tableau system is sound.

Proof We show that if $\Gamma = \{\mathbf{T}\gamma_1, \mathbf{T}\gamma_2, \dots, \mathbf{T}\gamma_n, \overline{\mathbf{T}}\delta\}$ has a corresponding tableau proof then $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$. From lemma 4.2 if Γ leads to a closed tableau then Γ is not satisfiable and hence $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$. \square

5 Proof of Completeness

We now prove the completeness of the proof system for \mathbf{K}_3 . The proof can be easily modified to give a proof of completeness for the other logics.

Definition 5.1 A *signed Hintikka set* \mathbf{H} is a set of signed formulae such that:

1. For any propositional letter ρ it is neither the case that both $\mathbf{T}\rho \in \mathbf{H}$ and $\overline{\mathbf{T}}\rho \in \mathbf{H}$, nor that both $\mathbf{F}\rho \in \mathbf{H}$ and $\overline{\mathbf{F}}\rho \in \mathbf{H}$, nor that both $\mathbf{F}\rho \in \mathbf{H}$ and $\mathbf{T}\rho \in \mathbf{H}$.
2. If $\mathbf{T}\neg\alpha \in \mathbf{H}$ then $\mathbf{F}\alpha \in \mathbf{H}$.
3. etc. (according to table 2.2).

For example, $\{\mathbf{T}P \vee (Q \wedge R), \mathbf{T}P\}$ and $\{\mathbf{T}P \vee (Q \wedge R), \mathbf{T}Q \wedge R, \mathbf{T}Q, \mathbf{T}R\}$ are both Hintikka sets. Note that a signed Hintikka set can never contain both $\mathbf{T}\alpha$ and $\overline{\mathbf{T}}\alpha$, or both $\mathbf{F}\alpha$ and $\overline{\mathbf{F}}\alpha$, or both $\mathbf{T}\alpha$ and $\mathbf{F}\alpha$.

Lemma 5.1 Every signed Hintikka set is satisfiable.

Proof Let \mathbf{H} be a signed Hintikka set. We show that a valuation ν can always be constructed from \mathbf{H} in such a way that every signed formula in \mathbf{H} holds. Let ν be constructed as follows:

1. If for some propositional letter ρ , $\mathbf{T} \rho \in \mathbf{H}$ then $\nu(\rho) = \mathbf{t}$.
2. If for some propositional letter ρ , $\mathbf{F} \rho \in \mathbf{H}$ then $\nu(\rho) = \mathbf{f}$.
3. If for some propositional letter ρ , neither $\mathbf{T} \rho \in \mathbf{H}$ nor $\mathbf{F} \rho \in \mathbf{H}$ then $\nu(\rho) = \mathbf{i}$.
4. If $\nu(\alpha) = \mathbf{f}$ then $\nu(\neg\alpha) = \mathbf{t}$.
5. etc. (according to table 2.2).

From condition 1 of definition 5.1 it follows that ν is well defined over predicate letters. By structural induction it follows that every signed formula in \mathbf{H} holds in ν . \square

Lemma 5.2 Every open branch \mathcal{B} of a tableau has a corresponding signed Hintikka set containing all the signed formulae of \mathcal{B} .

Proof Since \mathcal{B} is open, none of the closure conditions of table 3.1 apply for any proposition ρ and thus condition 1 of definition 5.1 must be satisfied. The other conditions follow by structural induction over signed formulae. \square

Theorem 5.1 The above tableau system is complete.

Proof We need to show that if $\gamma_1, \gamma_2, \dots, \gamma_n \models \delta$ then $\Gamma = \{\mathbf{T} \gamma_1, \mathbf{T} \gamma_2, \dots, \mathbf{T} \gamma_n, \overline{\mathbf{T}} \delta\}$ produces a closed tableau. We show the contrapositive (i.e. if Γ produces an open tableau then $\gamma_1, \gamma_2, \dots, \gamma_n \not\models \delta$). Assume Γ produces an open tableau. Then it must have an open branch \mathcal{B} and, by lemma 5.2, \mathcal{B} has a corresponding Hintikka set \mathbf{H} . Thus, by lemma 5.1, that set must be satisfiable and hence Γ must be satisfiable, thus $\gamma_1, \gamma_2, \dots, \gamma_n \not\models \delta$. \square

6 Conclusions and Directions for Further Research

We have presented a promising approach to automated theorem proving in many-valued logics based on the idea of using signed formulae where the truth signs reflect useful categories of truth values. We have demonstrated the approach by generating new tableau style proof systems for the many-valued logics \mathbf{L}_3 , \mathbf{K}_3 , \mathbf{Q}_3 , \mathbf{G}_3 , \mathbf{S}_3 , \mathbf{S}_3'' , $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{T}}$.

The above tableau style proof systems were generated by hand. Since there are only a finite number of ways of partitioning a logic's truth-values it should be possible to construct a system that automatically generates tableau style proof systems and presents to the user the best candidate systems. Such a system should be further investigated.

Just as we can easily construct tableau style proof systems, by using truth signs to partition truth values into useful categories, we can also make use of truth signs to construct resolution [57, 58] and connection method [1, 10] style proof systems—we need only replace the notion of contradictory formulae, used in resolution and the connection method, with the concept of opposite signed formulae.

Existing, resolution style proof systems for many-valued logics [2, 50] have used, effectively, one truth sign per truth value thus increasing the size of the original problem and hence the size and complexity of the proofs. When formulae are split into simpler components (say during the generation of a clausal form) the use of truth signs that reflect natural categories of truth values can greatly reduce the number of formulae generated. For example, if we were to attempt to prove $\models \mathbf{P} \vee \neg \mathbf{P}$ in \mathbf{K}_3 using resolution we would carry out resolution on the formula $\overline{\mathbf{T}} \mathbf{P} \vee \neg \mathbf{P}$ but Baaz and Fermüller [2] would use the pair of formulae $\mathbf{F} \mathbf{P} \vee \neg \mathbf{P}$ and $\mathbf{I} \mathbf{P} \vee \neg \mathbf{P}$ (the interested reader will find that Baaz and Fermüller's approach leads to a rapidly growing set of clauses but ours generates only two formulae).

Some many valued logics do not have appropriate normal forms for either resolution or the connection method. This need not concern us since if we use signed formulae then metatheoretic analogues of these normal forms can easily be constructed. For example, in \mathbf{K}_3 we do not have an equivalent of the resolution rule $(\alpha \vee \beta) \wedge (\neg \alpha \vee \gamma) \rightarrow (\beta \vee \gamma)$ but we do have the metatheoretic equivalent $((\mathbf{T}\alpha) \vee \beta) \wedge ((\overline{\mathbf{T}}\alpha) \vee \gamma) \rightarrow (\beta \vee \gamma)$ (where \wedge , \vee and \rightarrow are read classically).

A potentially fruitful line of research would be an investigation, based on the above ideas, of resolution and connection method style proof systems for many-valued logics.

Acknowledgments

My thanks to Terry Halpin and Rodney Beard for their comments on an earlier draft of this report; and Ed Kazmierczak for reviewing it. This work was supported by an APRA and ARC special research centre funding.

References

- [1] Peter B. Andrews. Theorem proving via general matings. *Journal of the ACM*, 28(2):191–214, 1981.
- [2] Matthias Baaz and Christian G. Fermüller. Resolution for many-valued logics. In A. Voronkov, editor, *Logic Programming and Automated Reasoning*, number 624 in Lecture Notes in Artificial Intelligence, pages 107–118, Berlin, 1992. Springer-Verlag.
- [3] A. H. Basson and D. J. O’Connor. *Introduction to Symbolic Logic*. University Tutorial Press, London, 3rd edition, 1959.
- [4] E. W. Beth. Semantic entailment and formal derivability. *Mededelingen van de Koninklijke Nederlandse Akademie van Wetenschappen, Afdeling Letterkunde*, 18(13):309–342, 1955. Reprinted in [8].
- [5] E. W. Beth. Completeness results for formal systems. In J. A. Todd, editor, *Proceedings of the International Congress of Mathematicians*, pages 281–288. Cambridge University Press, August 1958.
- [6] E. W. Beth. On machines which prove theorems. *Wisen Naturkundig Tijdschrift*, 32:49–60, 1958. Reprinted in [9].
- [7] E. W. Beth. *Formal Methods*. Synthese library. D. Reidel, Dordrecht, 1962.
- [8] E. W. Beth. Semantic entailment and formal derivability. In Jaakko Hintikka, editor, *The Philosophy of Mathematics*, chapter 1, pages 9–41. Oxford University Press, London, 1969.
- [9] E. W. Beth. On machines which prove theorems. In Jörg Siekmann and Graham Wrightson, editors, *Automation of Reasoning*, volume 1 of *Symbolic Computation*, pages 79–90. Springer-Verlag, Berlin, 1983. Reprint of [6].
- [10] Wolfgang Bibel. On matrices with connections. *Journal of the ACM*, 28(4):633–645, 1981.
- [11] D. A. Bochvar. Ob odnum trézhznačnom isčislénii i égo priménénii k analizu paradoksov klassičéskogo rassirennogo funkcionalnogo isčisléniiá. *Matématičeskij Sbornik*, 4:287–308, 1939.
- [12] L. Borkowski, editor. *Jan Lukasiewicz Selected Works*. Studies in Logic and the Foundations of Mathematics. North-Holland, Amsterdam, 1970.
- [13] Walter A. Carnielli. Systematization of finite many-valued logics through the method of tableaux. *Journal of Symbolic Logic*, 52(2):473–493, June 1987.

- [14] Alonzo Church. D. A. Bochvar. Ob odnum tréhznačnom isčislénii i égo priménénii k analizu paradoksov klassičéskogo rassirennoho funkcionalnogo isčisléniá. *Journal of Symbolic Logic*, 4(2):98–99, June 1939.
- [15] Alonzo Church. D. A. Bochvar. Über einen Aussagenkalkül mit abzählbaren logischen Summen und Produkten. *Journal of Symbolic Logic*, 5(3):119, September 1940.
- [16] J. P. Cleave. Comment on “Does many-valued logic have any use?”. In Körner [31], pages 88–91. Comment on [62].
- [17] Melvin Fitting. Tableau methods of proof for modal logics. *Notre Dame Journal of Formal Logic*, 13(2):237–247, April 1972.
- [18] Melvin Fitting. First-order modal tableaux. *Journal of Automated Reasoning*, 4:191–213, 1988.
- [19] Orrin Frink, Jr. Sobociński, Bolesław. Axiomatization of a partial system of three-value [sic] calculus of propositions. *Mathematical Reviews*, 14(9):834, October 1953.
- [20] R. Giles. Comment on “Does many-valued logic have any use?”. In Körner [31], pages 92–95. Comment on [62].
- [21] Kurt Gödel. Zum intuitionistischen Aussagenkalkül. *Anzeiger der Akademie der Wissenschaften Wien, mathematisch, naturwissenschaftliche Klasse*, 69:65–66, 1932. English translation available in [22].
- [22] Kurt Gödel. On the intuitionistic propositional calculus. In Solomon Feferman, John W. Dawson, Jr., Stephen C. Kleene, Gregory H. Moore, Robert M. Solovay, and Jean van Heijenoort, editors, *Kurt Gödel Collected Works*, volume 1, pages 223–225. Oxford University Press, New York, 1986. English translation of [21].
- [23] Reiner Hähnle. Towards an efficient tableau proof procedure for multiple-valued logics. In E. Börger, H. Kleine Büning, M.M. Richter, and W. Schönfeld, editors, *4rd Workshop on Computer Science Logic*, volume 533 of *Lecture Notes in Computer Science*, pages 248–260, Berlin, October 1990. Springer-Verlag.
- [24] Jaakko Hintikka. Form and content in quantification theory. Number 8 in *Acta Philosophica Fennica*. Helsinki, 1955.
- [25] Jaakko Hintikka. Existential presuppositions and their elimination. In *Models for Modalities* [27], pages 23–44.
- [26] Jaakko Hintikka. Modality and quantification. In *Models for Modalities* [27], pages 57–70.
- [27] Jaakko Hintikka, editor. *Models for Modalities*. D. Reidel, Dordrecht, 1969.
- [28] Jaakko Hintikka. The modes of modality. In *Models for Modalities* [27], pages 71–86.
- [29] H. Hiž. Ślupecki, Jersy. Le calcul complet de propositions à trois valeurs logiques. *Mathematical Reviews*, 10(1):1, January 1949.
- [30] Stephen Cole Kleene. On notation for ordinal numbers. *Journal of Symbolic Logic*, 3(4):150–155, December 1938.
- [31] Stephan Körner, editor. *Philosophy of Logic: Proceedings of the Third Bristol Conference on Critical Philosophy*, Oxford, 1976. Basil Blackwell.
- [32] Zbigniew Lis. Logical consequence, semantic and formal. *Studia Logica*, 10:58–60, 1960.

- [33] Zbigniew Lis. Wynikanie semantyczne a wynikanie formalne. *Studia Logica*, 10:39–54, 1960. English summary available in [32].
- [34] Jan Łukasiewicz. Farewell lecture by Professor Jan Łukasiewicz, delivered in the warsaw university lecture hall on March 7, 1918. In Borkowski [12], pages 84–86.
- [35] Jan Łukasiewicz. O logice trójwartościowej. *Ruch Filozoficzny*, 5:170–171, 1920. English translations available in [46] and [42].
- [36] Jan Łukasiewicz. O pojęciu możliwości. *Ruch Filozoficzny*, 5:169–170, 1920. English translation available in [41].
- [37] Jan Łukasiewicz. Interpretacja liczbowa teorii zdań. *Ruch Filozoficzny*, 7:92–93, 1923. English translation available in [45].
- [38] Jan Łukasiewicz. Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalküls. *Comptes Rendus des Séances de la Société des Sciences et des Lettres de Varsovie*, cl. iii, 23:51–77, 1930. English translation available in [47] and [43].
- [39] Jan Łukasiewicz. O determinizmie. In J. Słupecki, editor, *Z Zagadnień Logiki i Filozofii*. 1961. English translation available in [40].
- [40] Jan Łukasiewicz. On determinism. In McCall [49], chapter 2, pages 19–39.
- [41] Jan Łukasiewicz. On the notion of possibility. In McCall [49], pages 15–16.
- [42] Jan Łukasiewicz. On three-valued logic. In McCall [49], pages 16–17.
- [43] Jan Łukasiewicz. Philosophical remarks on many-valued systems of propositional logic. In McCall [49], pages 40–65.
- [44] Jan Łukasiewicz. Investigations into the sentential calculus. In Borkowski [12], pages 131–152.
- [45] Jan Łukasiewicz. A numerical interpretation of the theory of propositions. In Borkowski [12], pages 129–130.
- [46] Jan Łukasiewicz. On three-valued logic. In Borkowski [12], pages 87–88.
- [47] Jan Łukasiewicz. Philosophical remarks on many-valued systems of propositional logic. In Borkowski [12], pages 153–178.
- [48] Jan Łukasiewicz and Alfred Tarski. Untersuchungen über den Aussagenkalkül. *Comptes Rendus des Séances de la Société des Sciences et des Lettres de Varsovie*, cl. iii, 23:1–21, 1930. English translation available in [44].
- [49] Storrs McCall, editor. *Polish Logic: 1920–1939*. Oxford University Press, Oxford, 1967.
- [50] Peter O’Hearn and Zbigniew Stachniak. Note on theorem proving strategies for resolution counterparts of non-classical logics. In *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation*, pages 364–372, New York, July 1989. Association for Computing Machinery, ACM Press.
- [51] Emil L. Post. Introduction to a general theory of elementary propositions. *American Journal of Mathematics*, 43:163–185, 1921.
- [52] Steve Reeves. A note on functional reflexivity with partial unification. Department of Computer Science and Statistics Technical Report 359, University of London, June 1985.
- [53] Hans Reichenbach. *Philosophical Foundations of Quantum Mechanics*. University of California Press, Berkeley, 1944.

- [54] Hans Reichenbach. Reply to Ernest Nagel’s criticism of my views on quantum mechanics. *Journal of Philosophy*, 43:239–247, 1946.
- [55] Nicholas Rescher. *Many-valued Logic*. McGraw Hill, New York, 1969.
- [56] Nicholas Rescher and Alasdair Urquhart. *Temporal Logic*, volume 3 of *Library of Exact Philosophy*. Springer-Verlag, Vienna, 1971.
- [57] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, January 1965.
- [58] J. A. Robinson. The generalized resolution principle. *Machine Intelligence*, 3:77–93, 1968.
- [59] Gene F. Rose. Sobociński, Bolesław. Axiomatization of a partial system of three-value [sic] calculus of propositions. *Journal of Symbolic Logic*, 18(3):283, September 1953.
- [60] Peter H. Schmitt. Perspectives in multiple-valued logic. In R. Studer, editor, *International Scientific Symposium on Natural Language and Logic*, volume 459 of *Lecture Notes in Artificial Intelligence*, pages 206–220, Hamburg, May 1989. Springer-Verlag.
- [61] H. Scholz. Sobociński, Bolesław: Axiomatization of a partial system of three-value [sic] calculus of propositions. *Zentralblatt für Mathematik und ihre Grenzgebiete*, 49(7):292, April 1959.
- [62] Dana Scott. Does many-valued logic have any use? In Körner [31], pages 64–74. For comments see [16, 20, 63].
- [63] T. J. Smiley. Comment on “Does many-valued logic have any use?”. In Körner [31], pages 74–88. Comment on [62].
- [64] Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, Berlin, 1968.
- [65] Bolesław Sobociński. Axiomatization of a partial system of three-value [sic] calculus of propositions. *Journal of Computing Systems*, 1(1):23–55, 1952.