Analytic Proof Systems for Strong Kleene logic $\mathbf{K}_3$

M. Ultlog*

July 02, 2024

Abstract

We give sequent calculus, analytic tableaux, natural deduction, and clause translation systems for resolution for Strong Kleene logic $\mathbf{K}_3$.

1 Introduction

In this paper we present calculi for Strong Kleene logic $\mathbf{K}_3$. $\mathbf{K}_3$ has three truth values $f, *, t$ (with t designated), and connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$. Its syntax and semantics is detailed in section 2.

We first present a 3-sided sequent calculus in section 3. The fundamental idea for many-sided sequent calculi for finite-valued logics goes back to Schröter [11], Rousseau [8], Takahashi [13]. We follow the method given by Baaz, Fermüller, and Zach [4] and Zach [14] for constructing inference rules. This guarantees that our system automatically has soundness and completness theorems, cut-elimination theorem and Maehara lemma (interpolation). For proofs of these results see [4, 14].

Signed tableau systems for finite-valued logics were proposed by Surma [12] and Carnielli [6], and generalized by Hähnle [7]. In section 4, we present a signed tableau system for Strong Kleene logic.

Many-valued natural deduction systems for finite-valued logics have been investigated by Baaz, Fermüller, and Zach [3] and Zach [14]. We give the introduction and elimination rules for the natural deduction system for $\mathbf{K}_3$ in section 5.

In addition to Hähnle's work on tableaux-based theorem proving for finite-valued logic, Baaz and Fermüller [1] have studied resolution calculi for clauses of signed literals. In order for these calculi to be used to prove that formulas of Strong Kleene logic are valid or follow from some others, it is necessary to

*Cite as: Multlog (2024), "Analytic proof systems for Strong Kleene logic $\mathbf{K}_3$." [PDF generated by MULTLOG, v. 1.16a, `https://logic.at/multlog`]. Available at `https://logic.at/multlog/kleene.pdf`.

See appendices A and B for the specification of Strong Kleene logic.

produce sets of signed clauses. In section 6, we present a translation calculus that yields a set of clauses from a set of formulas.

The rules we provide are optimal in each case, and use the algorithms developed by Salzer [9, 10].

2 Syntax and semantics

Definition 1. The *propositional language* $\mathscr{L}$ for Strong Kleene logic consists of

1. propositional variables: $x_0, x_1, x_2, \ldots$
2. propositional connectives, arity given in parenthesis: $\neg$ (1) , $\wedge$ (2), $\vee$ (2), and $\rightarrow$ (2)
3. auxiliary symbols: "(", ")" and ","

Formulas are defined inductively:

1. Every propositional variable is a formula.
2. If A is a formula, so is $\neg A$.
3. If A and B are formulas, so is $(A \wedge B)$.
4. If A and B are formulas, so is $(A \vee B)$.
5. If A and B are formulas, so is $(A \rightarrow B)$.

As a notational convention, lowercase letters will be used to denote variables, possibly indexed. Uppercase letters $A, B, C, \ldots$ will stand for formulas, greek letters $\Gamma, \Delta, \Lambda, \ldots$ for sequences and sets of formulas. The symbol $\Box$ stands for general propositional connectives.

Definition 2. The *matrix* for Strong Kleene logic is given by:

1. the set of *truth values* $V = \{f, *, t\}$,
2. the set $V^+ = \{t\} \subseteq V$ of *designated truth values*,
3. the truth functions for connectives $\neg$, $\wedge$, $\vee$ and $\rightarrow$, as given below;

The set of *undesignated values* is $V^- = V \setminus V^+ = \{f, *\}$.

The truth functions for connectives $\neg$, $\wedge$, $\vee$ and $\rightarrow$ are defined by

$\widetilde{\neg}$	
f	t
$*$	$*$
t	f

$\widetilde{\wedge}$	f	$*$	t
f	f	f	f
$*$	f	$*$	$*$
t	f	$*$	t

$\widetilde{\vee}$	f	$*$	t
f	f	$*$	t
$*$	$*$	$*$	t
t	t	t	t

$\widetilde{\rightarrow}$	f	$*$	t
f	t	t	t
$*$	$*$	$*$	t
t	f	$*$	t

Definition 3. Let A be a formula and $x_0, x_1, \ldots, x_k$ the variables occurring in A. Then an *interpretation* $\mathfrak{I}$ of A is an assignment of truth values to the variables.

Definition 4. Given an interpretation $\mathfrak{I}$, we define the *valuation* $\mathrm{val}_{\mathfrak{I}}$ for formulas A to truth values as follows:

1. If A is atomic, then $\mathrm{val}_{\mathfrak{I}}(A)$ simply is the interpretation of A.
2. If $A = \Box(A_1, \ldots, A_n)$, where $A_1, \ldots, A_n$ are formulas, and $\widetilde{\Box}$ is the associated truth function to $\Box$, then $\mathrm{val}_{\mathfrak{I}}(A) = \widetilde{\Box}\big(\mathrm{val}_{\mathfrak{I}}(A_1), \ldots, \mathrm{val}_{\mathfrak{I}}(A_n)\big)$.

Definition 5. An interpretation $\mathfrak{I}$ *satisfies* a formula A, in symbols: $\mathfrak{I} \models A$, iff $\mathrm{val}_{\mathfrak{I}}(A) \in V^+$.

Definition 6. Δ *entails* A iff $\mathfrak{I} \models A$ for every interpretation $\mathfrak{I}$ such that $\mathfrak{I} \models B$ for all $B \in \Delta$. A is a *tautology* iff it is satisfied by every interpretation $\mathfrak{I}$.

3 Sequent calculus for Strong Kleene logic

Definition 7 (Syntax of Sequents)**.** A *sequent* Γ is a triple

$$\Gamma_f \mid \Gamma_* \mid \Gamma_t$$

of finite sequences Γ_v of formulas, where $v \in V$. The Γ_v are called the *components* of Γ.

For a sequence of formulas Δ, and $W \subseteq V$, let $[W\colon \Delta]$ denote the sequent whose component Γ_v is Δ if $v \in W$ and empty otherwise. For $[\{w_1, \ldots, w_k\}\colon \Delta]$ we also write $[w_1, \ldots, w_k\colon \Delta]$. If Γ and Γ' are sequents, then Γ, Γ' denotes the component-wise union, i.e., the v-component of Γ, Γ' is Γ_v, Γ'_v.

Definition 8. Let $\mathfrak{I}$ be an interpretation. $\mathfrak{I}$ *satisfies* a sequent Γ iff there is a $v \in V$ so that for some formula $A \in \Gamma_v$, $\mathrm{val}_{\mathfrak{I}}(F) = v$. $\mathfrak{I}$ is called a *model* of Γ, in symbols $\mathfrak{I} \models \Gamma$.

Γ is called *satisfiable* iff there is an interpretation $\mathfrak{I}$ so that $\mathfrak{I} \models \Gamma$ and *valid* iff for every interpretation $\mathfrak{I}$, $\mathfrak{I} \models \Gamma$.

Proposition 9. *$\Delta \models A$ iff the sequent $[f, *\colon \Delta], [t\colon A]$ is valid.*

Definition 10. The *sequent calculus* for Strong Kleene logic is given by:

1. axiom schemas of the form $[V\colon A]$,
2. weakening rules for every truth value v:

$$\frac{\Gamma}{\Gamma, [v\colon A]}\ \textsc{w}{:}v$$

3. exchange rules for every truth value v:

$$\frac{\Gamma, [v\colon A, B], \Delta}{\Gamma, [v\colon B, A], \Delta}\ \textsc{x}{:}v$$

4. contraction rules for every truth value v:

$$\frac{\Gamma, [v\colon A, A]}{\Gamma, [v\colon A]}\ \mathrm{C}{:}v$$

5. cut rules for every two truth values $v \neq w$:

$$\frac{\Gamma, [v\colon A] \quad \Delta, [w\colon A]}{\Gamma, \Delta}\ \mathrm{CUT}{:}vw$$

6. an introduction rule $\Box\colon v$ for every connective $\Box$ and every truth value v, as specified below.

(2)–(5) are called *structural rules*. (6) are called *logical rules*.

The introduction rules for connective $\neg$ are given by

$$\frac{\Gamma, [t\colon A]}{\Gamma, [f\colon \neg A]}\ \neg{:}f \qquad \frac{\Gamma, [*\colon A]}{\Gamma, [*\colon \neg A]}\ \neg{:}* \qquad \frac{\Gamma, [f\colon A]}{\Gamma, [t\colon \neg A]}\ \neg{:}t$$

The introduction rules for connective $\wedge$ are given by

$$\frac{\Gamma, [f\colon A, B]}{\Gamma, [f\colon A \wedge B]}\ \wedge{:}f \qquad \frac{\Gamma, [*, t\colon B] \quad \Gamma, [*\colon A, B] \quad \Gamma, [*, t\colon A]}{\Gamma, [*\colon A \wedge B]}\ \wedge{:}*$$

$$\frac{\Gamma, [t\colon B] \quad \Gamma, [t\colon A]}{\Gamma, [t\colon A \wedge B]}\ \wedge{:}t$$

The introduction rules for connective $\vee$ are given by

$$\frac{\Gamma, [f\colon B] \quad \Gamma, [f\colon A]}{\Gamma, [f\colon A \vee B]}\ \vee{:}f \qquad \frac{\Gamma, [f, *\colon B] \quad \Gamma, [*\colon A, B] \quad \Gamma, [f, *\colon A]}{\Gamma, [*\colon A \vee B]}\ \vee{:}*$$

$$\frac{\Gamma, [t\colon A, B]}{\Gamma, [t\colon A \vee B]}\ \vee{:}t$$

The introduction rules for connective $\to$ are given by

$$\frac{\Gamma, [f\colon B] \quad \Gamma, [t\colon A]}{\Gamma, [f\colon A \to B]}\ \to{:}f \qquad \frac{\Gamma, [f, *\colon B] \quad \Gamma, [*\colon A, B] \quad \Gamma, [*, t\colon A]}{\Gamma, [*\colon A \to B]}\ \to{:}*$$

$$\frac{\Gamma, [f\colon A], [t\colon B]}{\Gamma, [t\colon A \to B]}\ \to{:}t$$

Definition 11. An upward tree of sequents is called a *proof* in the sequent calculus iff every leaf is an axiom, and all other sequents in it are obtained from the ones standing immediately above it by applying one of the rules. The sequent at the root of P is called its *end-sequent*. A sequent Γ is called *provable* iff it is the end-sequent of some proof.

Theorem 12 (Soundness and Completeness)**.** *A sequent is provable if and only if it is valid.*

Proof. See Theorems 3.1 and 3.2 of Baaz et al. [4] or Theorems 3.3.8 and 3.3.10 of Zach [14]. □

Corollary 13. *In Strong Kleene logic,* $\models A$ *iff* $[t\colon A]$ *has a sequent proof, and* $\Delta \models A$ *iff* $[f, *\colon \Delta], [t\colon A]$ *has a proof.*

Theorem 14 (Cut-elimination)**.** *The cut rule is eliminable in the sequent calculus for Strong Kleene logic.*

Proof. See Theorem 4.1 of Baaz et al. [4] or Theorem 3.5.3 of Zach [14]. □

Theorem 15 (Maehara lemma)**.** *The Maehara lemma holds for the sequent calculus for Strong Kleene logic.*

Proof. See Theorem 3.8.1 of Zach [14]. □

4 Tableaux for Strong Kleene logic

Although the method of Surma [12] and Carnielli [6] for obtaining signed analytic tableaux systems applies to Strong Kleene logic, it has a drawback. As Hähnle [7] pointed out, to show that a formula is valid, it is required to provide as many closed tableaux as there are non-designated values. This is usually not desirable; the generalized approach by Hähnle [7] solves this problem. Below we give a tableau system for Strong Kleene logic using the sets of signs $V \setminus \{v\}$, i.e., the tableau system exactly dual to that of Carnielli (in the sense of [2]).

Definition 16. A *signed formula* is an expression of the form $v\colon A$ where $v \in V$ and A is a formula.

Definition 17. A *tableau* for a set of signed formulas Δ is a downward rooted tree of signed formulas where each one is either an element of Δ or results from a signed formula in the branch above it by a branch expansion rule. A tableau is *closed* if every branch contains, for some formula A, the signed formulas $v\colon A$ for all $v \in V$, or a signed formula $v\colon A$ with a branch expansion rule that explicitly closes the branch ($\otimes$).

The branch expansion rules for connective $\neg$ are given by

$$\frac{f\colon \neg A}{t\colon A} \qquad \frac{*\colon \neg A}{*\colon A} \qquad \frac{t\colon \neg A}{f\colon A}$$

The branch expansion rules for connective $\wedge$ are given by

$$\frac{f\colon A \wedge B}{\begin{array}{l} f\colon A \\ f\colon B \end{array}} \qquad \frac{*\colon A \wedge B}{\begin{array}{lll} *\colon B & *\colon A & *\colon A \\ t\colon B & *\colon B & t\colon A \end{array}} \qquad \frac{t\colon A \wedge B}{\begin{array}{ll} t\colon B & t\colon A \end{array}}$$

The branch expansion rules for connective $\vee$ are given by

$$\frac{f\colon A \vee B}{\begin{array}{ll} f\colon B & f\colon A \end{array}} \qquad \frac{*\colon A \vee B}{\begin{array}{lll} f\colon B & *\colon A & f\colon A \\ *\colon B & *\colon B & *\colon A \end{array}} \qquad \frac{t\colon A \vee B}{\begin{array}{c} t\colon A \\ t\colon B \end{array}}$$

The branch expansion rules for connective $\rightarrow$ are given by

$$\frac{f\colon A \rightarrow B}{\begin{array}{ll} f\colon B & t\colon A \end{array}} \qquad \frac{*\colon A \rightarrow B}{\begin{array}{lll} f\colon B & *\colon A & *\colon A \\ *\colon B & *\colon B & t\colon A \end{array}} \qquad \frac{t\colon A \rightarrow B}{\begin{array}{c} f\colon A \\ t\colon B \end{array}}$$

Definition 18. An interpretation $\mathfrak{I}$ *satisfies* a signed formula $v\colon A$ iff $\mathrm{val}_{\mathfrak{I}}(A) \neq v$. A set of signed formulas is *satisfiable* if some interpretation $\mathfrak{I}$ satisfies all signed formulas in it.

Theorem 19. *A set of signed formulas is unsatisfiable iff it has a closed tableau.*

Proof. Apply Theorems 4.14 and 4.21 of Hähnle [7]; interpreting $v\colon A$ as $\mathsf{S}\ A$ where $\mathsf{S} = V \setminus \{v\}$. □

Corollary 20. *In Strong Kleene logic, $\models A$ iff $\{v\colon A \mid v \in V^+\}$ has a closed tableau. $\Delta \models A$ iff $\{v\colon B \mid v \in V^-, B \in \Delta\} \cup \{v\colon A \mid v \in V^+\}$ has a closed tableau.*

5 Natural deduction for Strong Kleene logic

Let Γ be a (set) sequent, $V^+ \subseteq V$ the set of *designated truth values*. The set of non-designated truth values is then $V^- = V \setminus V^+$. We divide the sequent Γ into its designated part Γ^+ and its non-designated part Γ^- in the obvious way:

$$\begin{aligned} \Gamma^+ &:= \langle \Gamma_v \mid v \in V^+ \rangle \\ \Gamma^- &:= \langle \Gamma_v \mid v \in V^- \rangle \end{aligned}$$

Definition 21. The *natural deduction rules* for Strong Kleene logic are given by:

1. A weakening rule for all $v \in V^+$:

$$\frac{\Gamma^+}{\Gamma^+, [v\colon A]}\ \textsc{w}{:}v$$

2. For every connective $\Box$ and every truth value v an introduction rule $\Box\textsc{i}{:}v$ (if $v \in V^+$) or an elimination rule $\Box\textsc{e}{:}v$ (if $v \in V^-$).

The introduction and elimination rules for connective $\neg$ are given by

$$\frac{\Gamma_0^-, \lceil[*\colon \neg A]\rceil \quad \Gamma_1^- \qquad \Gamma_0^-, \lceil[f\colon \neg A]\rceil \quad \Gamma_1^-, \lceil[*\colon A]\rceil \\ \Gamma_0^+, [t\colon \neg A] \quad \Gamma_1^+, [t\colon A] \qquad \Gamma_0^+, [t\colon \neg A] \quad \Gamma_1^+}{\Gamma_0^+, \Gamma_1^+} \; \neg\text{E}\colon f \qquad \frac{}{\Gamma_0^+, \Gamma_1^+} \; \neg\text{E}\colon *$$

$$\frac{\Gamma_1^-, \lceil[f\colon A]\rceil \\ \Gamma_1^+}{\Gamma_1^+, [t\colon \neg A]} \; \neg\text{I}\colon t$$

The introduction and elimination rules for connective $\wedge$ are given by

$$\frac{\Gamma_0^-, \lceil[*\colon A \wedge B]\rceil \quad \Gamma_1^-, \lceil[f\colon A, B]\rceil \\ \Gamma_0^+, [t\colon A \wedge B] \qquad \Gamma_1^+}{\Gamma_0^+, \Gamma_1^+} \; \wedge\text{E}\colon f$$

$$\frac{\Gamma_0^-, \lceil[f\colon A \wedge B]\rceil \quad \Gamma_1^-, \lceil[*\colon B]\rceil \quad \Gamma_2^-, \lceil[*\colon A, B]\rceil \quad \Gamma_3^-, \lceil[*\colon A]\rceil \\ \Gamma_0^+, [t\colon A \wedge B] \quad \Gamma_1^+, [t\colon B] \qquad \Gamma_2^+ \qquad \Gamma_3^+, [t\colon A]}{\Gamma_0^+, \ldots, \Gamma_3^+} \; \wedge\text{E}\colon *$$

$$\frac{\Gamma_1^- \qquad \Gamma_2^- \\ \Gamma_1^+, [t\colon B] \quad \Gamma_2^+, [t\colon A]}{\Gamma_1^+, \Gamma_2^+, [t\colon A \wedge B]} \; \wedge\text{I}\colon t$$

The introduction and elimination rules for connective $\vee$ are given by

$$\frac{\Gamma_0^-, \lceil[*\colon A \vee B]\rceil \quad \Gamma_1^-, \lceil[f\colon B]\rceil \quad \Gamma_2^-, \lceil[f\colon A]\rceil \\ \Gamma_0^+, [t\colon A \vee B] \qquad \Gamma_1^+ \qquad \Gamma_2^+}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+} \; \vee\text{E}\colon f$$

$$\frac{\Gamma_0^-, \lceil[f\colon A \vee B]\rceil \quad \Gamma_1^-, \lceil[f, *\colon B]\rceil \quad \Gamma_2^-, \lceil[*\colon A, B]\rceil \quad \Gamma_3^-, \lceil[f, *\colon A]\rceil \\ \Gamma_0^+, [t\colon A \vee B] \qquad \Gamma_1^+ \qquad \Gamma_2^+ \qquad \Gamma_3^+}{\Gamma_0^+, \ldots, \Gamma_3^+} \; \vee\text{E}\colon *$$

$$\frac{\Gamma_1^- \\ \Gamma_1^+, [t\colon A, B]}{\Gamma_1^+, [t\colon A \vee B]} \; \vee\text{I}\colon t$$

The introduction and elimination rules for connective $\to$ are given by

$$\frac{\Gamma_0^-, \lceil[*\colon A \to B]\rceil \quad \Gamma_1^-, \lceil[f\colon B]\rceil \qquad \Gamma_2^- \\ \Gamma_0^+, [t\colon A \to B] \qquad \Gamma_1^+ \qquad \Gamma_2^+, [t\colon A]}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+} \; \to\text{E}\colon f$$

$$\frac{\Gamma_0^-, \lceil[f\colon A \to B]\rceil \quad \Gamma_1^-, \lceil[f, *\colon B]\rceil \quad \Gamma_2^-, \lceil[*\colon A, B]\rceil \quad \Gamma_3^-, \lceil[*\colon A]\rceil \\ \Gamma_0^+, [t\colon A \to B] \qquad \Gamma_1^+ \qquad \Gamma_2^+ \qquad \Gamma_3^+, [t\colon A]}{\Gamma_0^+, \ldots, \Gamma_3^+} \; \to\text{E}\colon *$$

$$\frac{\begin{array}{c}\Gamma_1^-, \lceil [f\colon A] \rceil \\ \Gamma_1^+, [t\colon B]\end{array}}{\Gamma_1^+, [t\colon A \to B]}\ {\to}\mathrm{I}\colon t$$

Definition 22. A *natural deduction derivation* is defined inductively as follows:

1. Let A be any formula. Then
$$\frac{[V^-\colon A]}{[V^+\colon A]}$$
is a derivation of A from the assumption $[V^-\colon A]$ (an *initial derivation*).

2. If D_k are derivations of Γ_k^+, Δ_k^+ from the assumptions $\Gamma_k^-, \hat{\Delta}_k^-$, and
$$\frac{\left\langle \begin{array}{c}\Gamma_k^-, \lceil \Delta_k^- \rceil \\ \Gamma_k^+, \Delta_k^+\end{array} \right\rangle_{k\in K}}{\Pi^+}$$
is an instance of a deduction rule with $\hat{\Delta}_k^-$ a subsequent of Δ_k^-, and all eigenvariable conditions are satisfied, then
$$\frac{\langle D_k \rangle_{k\in K}}{\Pi^+}$$
is a derivation of Π^+ from the assumptions $\bigcup_{k\in K} \Gamma_k^-$. The formulas in $\hat{\Delta}_k^-$ which do not occur in $\bigcup_{k\in K} \Gamma_k^-$ are said to be *discharged* at this inference.

Theorem 23. *A partial sequent Γ^+ can be derived from the assumptions Γ^- in the natural deduction system for Strong Kleene logic iff, for every interpretation $\mathfrak{I}$, either some formula in Γ_v^- ($v \in V^-$) evaluates to the truth value v, or there is a $w \in V^+$ and a formula in Γ_w^+ that evaluates to w.*

Proof. See Theorems 4.7 and 5.4 of Baaz et al. [3] or Theorems 4.2.8 and 4.3.4 of Zach [14]. □

Corollary 24. *$\Gamma \models A$ iff there is a natural deduction derivation of $[V^+\colon A]$ from $[V^-\colon \Gamma]$.*

6 Resolution and clause formation rules for Strong Kleene logic

The many-valued resolution calculus of Baaz and Fermüller [1] applies to Strong Kleene logic. We present the framework here, as well as a language preserving clause translation system for Strong Kleene logic.

Definition 25 (Signed formula)**.** A *signed formula* is an expression of the form A^v, where A is a formula and $v \in V$. If A is a propositonal variable, A^v is a *signed atom*.

Definition 26 (Signed clause)**.** A (signed) *clause* $C = \{A_1^{v_1}, \ldots, A_n^{v_n}\}$ is a finite set of signed atoms (proper clause). The empty clause is denoted by $\Box$.

An *extended clause* is a finite set of signed formulas.

Definition 27 (Semantics of clause sets)**.** Let $\mathfrak{I}$ be an interpretation. $\mathfrak{I}$ *satisfies* a clause C iff there is some signed formula $A^v \in C$, so that $\mathrm{val}_{\mathfrak{I}}(A) = v$. $\mathfrak{I}$ satisfies a clause set $\mathscr{C}$ iff it satisfies every clause in $\mathscr{C}$. $\mathscr{C}$ is called *satisfiable* iff some structure satisfies it, and *unsatisfiable* otherwise.

The clause formation rules for connective $\neg$ are given by

$$\frac{\mathscr{C} \cup \{C \cup \{(\neg A)^f\}\}}{\mathscr{C} \cup \{C \cup \{A^t\}\}}\ \neg{:}f \qquad \frac{\mathscr{C} \cup \{C \cup \{(\neg A)^*\}\}}{\mathscr{C} \cup \{C \cup \{A^*\}\}}\ \neg{:}*$$

$$\frac{\mathscr{C} \cup \{C \cup \{(\neg A)^t\}\}}{\mathscr{C} \cup \{C \cup \{A^f\}\}}\ \neg{:}t$$

The clause formation rules for connective $\wedge$ are given by

$$\frac{\mathscr{C} \cup \{C \cup \{(A \wedge B)^f\}\}}{\mathscr{C} \cup \{C \cup \{A^f, B^f\}\}}\ \wedge{:}f$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \wedge B)^*\}\}}{\mathscr{C} \cup \{C \cup \{B^*, B^t\},\ C \cup \{A^*, B^*\},\ C \cup \{A^*, A^t\}\}}\ \wedge{:}*$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \wedge B)^t\}\}}{\mathscr{C} \cup \{C \cup \{B^t\},\ C \cup \{A^t\}\}}\ \wedge{:}t$$

The clause formation rules for connective $\vee$ are given by

$$\frac{\mathscr{C} \cup \{C \cup \{(A \vee B)^f\}\}}{\mathscr{C} \cup \{C \cup \{B^f\},\ C \cup \{A^f\}\}}\ \vee{:}f$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \vee B)^*\}\}}{\mathscr{C} \cup \{C \cup \{B^f, B^*\},\ C \cup \{A^*, B^*\},\ C \cup \{A^f, A^*\}\}}\ \vee{:}*$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \vee B)^t\}\}}{\mathscr{C} \cup \{C \cup \{A^t, B^t\}\}}\ \vee{:}t$$

The clause formation rules for connective $\rightarrow$ are given by

$$\frac{\mathscr{C} \cup \{C \cup \{(A \rightarrow B)^f\}\}}{\mathscr{C} \cup \{C \cup \{B^f\},\ C \cup \{A^t\}\}}\ \rightarrow{:}f$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \rightarrow B)^*\}\}}{\mathscr{C} \cup \{C \cup \{B^f, B^*\},\ C \cup \{A^*, B^*\},\ C \cup \{A^*, A^t\}\}}\ \rightarrow{:}*$$

$$\frac{\mathscr{C} \cup \{C \cup \{(A \rightarrow B)^t\}\}}{\mathscr{C} \cup \{C \cup \{A^f, B^t\}\}}\ \rightarrow{:}t$$

Theorem 28. *Let $T(\mathscr{C})$ be the result of exhaustively applying the translation rules to a clause set $\mathscr{C}$. Then $T(\mathscr{C})$ is a set of proper clauses, i.e., $T(\mathscr{C})$ contains only signed atoms (all connectives are eliminated). Furthermore, $T(\mathscr{C})$ is satisfiable iff $\mathscr{C}$ is.*

Proposition 29. *Let A be a sentence and Δ be a set of sentences. Then*

1. *$\models A$ iff $\{\{A^v \mid v \in V^-\}\}$ is unsatisfiable.*
2. *$\Delta \models A$ iff*

$$\{\{B^w \mid w \in V^+\} \mid B \in \Delta\} \cup \{\{A^v \mid v \in V^-\}\}$$

is unsatisfiable.

Definition 30. A clause R is a *resolvent* of clauses C_1, C_2 if $R = (C_1 \setminus \{A^{v_1}\}) \cup (C_2 \setminus \{A^{v_2}\})$ where $v_1 \neq v_2$

Definition 31. A *resolution refutation* of a clause set $\mathscr{C}$ is a sequence of clauses $C_1, \ldots, C_n$ so that for every i, $C_i \in \mathscr{C}$ or C_i is a resolvent of C_j, C_k with $j, k < i$, and $C_n = \emptyset$.

Theorem 32. *A clause set $\mathscr{C}$ is unsatisfiable iff it has a resolution refutation.*

Proof. See Theorems 3.14 and 3.19 of Baaz and Fermüller [1] or Theorems 2.5.5 and 2.5.8 of Zach [14]. □

Corollary 33. *$\Delta \models A$ iff*

$$T(\{\{B^w \mid w \in V^+\} \mid B \in \Delta\} \cup \{\{A^v \mid v \in V^-\}\})$$

has a resolution refutation.

References

[1] Matthias Baaz and Christian G. Fermüller. Resolution-based theorem proving for many-valued logics. *Journal of Symbolic Computation*, 19(4):353–391, 1995. DOI 10.1006/jsco.1995.1021.

[2] Matthias Baaz, Christian G. Fermüller, and Richard Zach. Dual systems of sequents and tableaux for many-valued logics. *Bulletin of the EATCS*, 51:192–197, 1993. DOI 10.11575/PRISM/38908.

[3] Matthias Baaz, Christian G. Fermüller, and Richard Zach. Systematic construction of natural deduction systems for many-valued logics. In *23rd International Symposium on Multiple-Valued Logic. Proceedings*, pages 208–213. IEEE Press, 1993. DOI 10.1109/ISMVL.1993.289558.

[4] Matthias Baaz, Christian G. Fermüller, and Richard Zach. Elimination of cuts in first-order finite-valued logics. *Journal of Information Processing and Cybernetics EIK*, 29(6):333–355, 1993. DOI 10.11575/PRISM/38801.

[5] Matthias Baaz, Christian G. Fermüller, Gernot Salzer, and Richard Zach. Labeled calculi and finite-valued logics. *Studia Logica*, 61(1):7–33, 1998. DOI 10.1023/A:1005022012721.

[6] Walter A. Carnielli. Systematization of finite many-valued logics through the method of tableaux. *The Journal of Symbolic Logic*, 52(2):473–493, 1987. DOI 10.2307/2274395.

[7] Reiner Hähnle. *Automated Deduction in Multiple-Valued Logics*. Oxford University Press, 1993.

[8] George Rousseau. Sequents in many valued logic I. *Fundamenta Mathematicae*, 60:23–33, 1967. URL `http://matwbn.icm.edu.pl/ksiazki/fm/fm60/fm6012.pdf`.

[9] Gernot Salzer. MUltlog: An expert system for multiple-valued logics. In *Collegium Logicum*, pages 50–55. Springer, 1996. DOI 10.1007/978-3-7091-9461-4_3.

[10] Gernot Salzer. Optimal axiomatizations of finitely valued logics. *Information and Computation*, 162(1–2):185–205, 2000. DOI 10.1006/inco.1999.2862.

[11] Karl Schröter. Methoden zur Axiomatisierung beliebiger Aussagen- und Prädikatenkalküle. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 1(4):241–251, 1955. DOI 10.1002/malq.19550010402.

[12] Stanislaw J. Surma. An algorithm for axiomatizing every finite logic. In David C. Rine, editor, *Computer Science and Multiple-Valued Logic: Theory and Applications*, pages 137–143. North-Holland, 1977.

[13] Moto-o Takahashi. Many-valued logics of extended Gentzen style I. *Science Reports of the Tokyo Kyoiku Daigaku, Section A*, 9(231):271–292, 1968. URL `https://www.jstor.org/stable/43699119`.

[14] Richard Zach. Proof theory of finite-valued logics. Diplomarbeit, Technische Universität Wien, 1993. DOI 10.11575/PRISM/38803.

A kleene.lgc – specification of Strong Kleene logic

```
logic "Strong Kleene".
truth_values {f,*,t}.
designated_truth_values {t}.
ordering(linear, "f < * < t").
operator(neg  /1, mapping { (t): f,
                            (*): *,
                            (f): t
                          }
        ).
```

```
operator(and  /2, mapping { (t,t): t,
                            (t,*): *,
                            (t,f): f,
                            (*,t): *,
                            (*,*): *,
                            (*,f): f,
                            (f,t): f,
                            (f,*): f,
                            (f,f): f
                          }
        ).
operator(or   /2, mapping { (t,t): t,
                            (t,*): t,
                            (t,f): t,
                            (*,t): t,
                            (*,*): *,
                            (*,f): *,
                            (f,t): t,
                            (f,*): *,
                            (f,f): f
                          }
        ).
operator(imp  /2, mapping { (t,t): t,
                            (t,*): *,
                            (t,f): f,
                            (*,t): t,
                            (*,*): *,
                            (*,f): *,
                            (f,t): t,
                            (f,*): t,
                            (f,f): t
                          }
        ).
```

B kleene.cfg – LaTeX translations

```
texName(and,    \\land  ).
texName(or,     \\lor   ).
texName(neg,    \\lnot  ).
texName(imp,    \\to).
texInfix(and).
texInfix(or).
texInfix(imp).
texPrefix(neg).

texExtra("ShortName", "\\mathbf{K}_3").
texExtra("Link", "https://logic.at/multlog/kleene.pdf").
```