Calculi for the Gödel Logic

M. Ultlog

July 11, 2001

Abstract

This paper introduces the reader to the Gödel logic and to several related calculi.

1 Introduction

Within the last years multiple-valued logics, introduced in the 1920's independently by Łukasiewicz and Post, have attracted considerable attention by the computer science community due to their potential in the verification of soft- and hardware, in modelling epistemic states of knowledge in knowledge base systems etc. [7]. The practical success of a logic depends crucially on the availability of computational calculi for this logic. Without such calculi, the study of a logic in relation to its application in computer science and artificial intelligence is destined to remain purely theoretical. In the case of many-valued logics, computational calculi can be given.

Analytic calculi for many-valued logics have been known for quite some time. Sequent systems similar to those presented here have also been introduced by Schröter [13], Rousseau [10], Takahashi [15], and Carnielli [5]; equivalent tableaux formulations were given by Surma [14] and Carnielli [4] (Hähnle's work is based on the latter [6, 8, 7]). Many-valued natural deduction systems have been investigated in Baaz, Fermüller and Zach [3]. Calculi for automated theorem proving in many-valued logics are also legion. Apart from Hähnle's work on tableaux-based theorem proving, various resolution methods have been proposed, e.g., by O'Hearn and Stachniak [9] or Baaz and Fermüller [1, 2]. For more detailed surveys of the work done in these areas see [7, 16].

In this paper we present optimized calculi for the 3-valued Gödel logic, applying the algorithms developed in Salzer [12, 11]. After defining basic notions and notations in the next section, we define the Gödel logic in Section 3. The following sections are devoted to calculi for this logic: Section 4 describes a sequent calculus, and Section 5 gives a natural deduction system. The paper concludes with an outlook on future research topics.

2 The Syntax of the Gödel Logic

2.1. DEFINITION The *first order language* $\mathcal{L}$ for the Gödel logic consists of

1. free variables: $a_0, a_1, a_2, \ldots$
2. bound variables: $x_0, x_1, x_2, \ldots$
3. function symbols of arity $i (i \in \mathbf{N})$, including constants: $f_0^i, f_1^i, f_2^i, \ldots$
4. predicate constants of arity $i (i \in \mathbf{N})$: $P_0^i, P_1^i, P_2^i, \ldots$
5. propositional connectives, arity given in parenthesis: $\top(0)$, $\bot$ (0), $\neg$ (1), $\wedge$ (2), $\vee$ (2) and $\supset$ (2)
6. quantifiers: $\forall$ and $\exists$
7. auxiliary symbols: "(", ")" and ","

Terms are defined inductively: Every individual constant, free or bound variable is a term. If f^n is a function symbol of arity n, and $t_1, \ldots, t_n$ are terms, then $f^n(t_1, \ldots, t_n)$ is a term.

Formulae are also defined inductively:

1. If P^n is a predicate symbol of arity n, and $t_1, \ldots, t_n$ are terms, then $P^n(t_1, \ldots, t_n)$ is a formula. It is called *atomic* or an *atom*.
2. If $A_1, \ldots, A_n$ are formulae and $\Box^n$ is a propositional connective of arity n, then $\Box^n(A_1, \ldots, A_n)$ is a formula.
3. If A is a formula not containing the bound variable x, a is a free variable and Q is a quantifier, then $(\mathsf{Q}x)A(x)$, where $A(x)$ is obtained from A by replacing a by x at every occurrence of a in A, is a formula.

A formula is called *open*, if it contains free variables, and *closed* otherwise. A formula without quantifiers is called *quantifier-free*.

As a notational convention, lowercase letters from the beginning of the alphabet $(a, b, c, \ldots)$ will be used to denote free variables, $f, g, h, \ldots$ for function symbols and constants, $x, y, z, \ldots$ for bound variables, all possibly indexed. Uppercase letters $A, B, C, \ldots$ will stand for formulae, greek letters $\Gamma, \Delta, \Lambda, \ldots$ for sequences and sets of formulae, t and s for terms. The symbols $\Box$ and Q stand for general propositional connectives and quantifiers, respectively.

2.2. DEFINITION We will use α as a variable for free variables (*eigenvariable*) and τ as a variable for terms (*term variable*). A formula consisting of some formula variables, eigenvariables and term variables is called a schema.

A *pre-instance* A' of a schema A is an actual formula from the formulae of $\mathcal{L}$ which contains occurrences of the eigenvariables and term variables of A.

An *instance* A'' of A is a pre-instance A' of A, where the eigenvariables and term variables have been replaced by free variables and terms not occurring in A'.

3 The Semantics of the Gödel Logic

3.1. Definition The *matrix* for the language $\mathcal{L}$ of the Gödel logic is given by:

1. the set of *truth values* $V = \{f, *, t\}$,
2. a subset $V^+ \subseteq V$ of *designated truth values*,
3. for every propositional connective $\Box$ an associated truth function, as given below.
4. for every quantifier Q an an associated truth function, as given below.

The truth functions for connectives $\top$, $\bot$, $\neg$, $\wedge$, $\vee$ and $\supset$ are defined by

$\top \equiv t \qquad \bot \equiv f$

$\neg$	
f	t
$*$	f
t	f

$\wedge$	f	$*$	t
f	f	f	f
$*$	f	$*$	$*$
t	f	$*$	t

$\vee$	f	$*$	t
f	f	$*$	t
$*$	$*$	$*$	t
t	t	t	t

$\supset$	f	$*$	t
f	t	t	t
$*$	f	t	t
t	f	$*$	t

The truth functions for quantifiers $\forall$ and $\exists$ are defined by

$\forall$	
$\{f, *, t\}$	f
$\{f, *\}$	f
$\{f, t\}$	f
$\{f\}$	f
$\{*, t\}$	$*$
$\{*\}$	$*$
$\{t\}$	t

$\exists$	
$\{f, *, t\}$	t
$\{f, *\}$	$*$
$\{f, t\}$	t
$\{f\}$	f
$\{*, t\}$	t
$\{*\}$	$*$
$\{t\}$	t

3.2. Definition A *structure* $\mathcal{M} = \langle D, \Phi_{\mathcal{M}} \rangle$ for a language $\mathcal{L}$ (an $\mathcal{L}$-structure) consists of the following:

1. A nonempty set D, called the *domain* (elements of D are called *individuals*).
2. A mapping $\Phi_{\mathcal{M}}$ satisfying the following:
   (a) Each free variable of $\mathcal{L}$ is mapped to an element of D.
   (b) Each n-ary function symbol f of $\mathcal{L}$ is mapped to a function $f_{\mathcal{M}} : D^n \to D$, or to an element of D if $n = 0$. Additionally, $\Phi_{\mathcal{M}}$ maps elements of D to themselves.
   (c) Each n-ary predicate symbol P of $\mathcal{L}$ is mapped to a function $P_{\mathcal{M}} : D^n \to V$, or to a element of V if $n = 0$.

3.3. Definition Let $\mathcal{M}$ be an $\mathcal{L}$-structure. An *assignment* s is a mapping from the free variables of $\mathcal{L}$to individuals.

An *interpretation* $\mathbf{I} = \langle \mathcal{M}, s \rangle$ is an $\mathcal{L}$-structure $\mathcal{M} = \langle D, \Phi_{\mathcal{M}} \rangle$ together with an assignment s. The mapping $\Phi_{\mathcal{M}}$ can be extended in the obvious way to a mapping $\Phi_{\mathbf{I}}$ from terms to individuals:

1. If t is a free variable, then $\Phi_{\mathbf{I}} := s(t)$.

2. If t is of the form $f(t_1, \ldots, t_k)$, where f is a k-ary function symbol and $t_1, \ldots, t_k$ are terms, then $\Phi_{\mathbf{I}} := f_{\mathcal{M}}\big(\Phi_{\mathbf{I}}(t_1), \ldots, \Phi_{\mathbf{I}}(t_k)\big)$.

3.4. Definition Given an interpretation $\mathbf{I} = \langle \mathcal{M}, s\rangle$, we define the *valuation* $\mathrm{val}_{\mathbf{I}}$ for formulae A to truth values as follows:

1. If A is atomic, $A = P(t_1, \ldots, t_n)$, where P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then let $\mathrm{val}_{\mathbf{I}}(A) = P_{\mathcal{M}}\big(\Phi_{\mathbf{I}}(t_1), \ldots, \Phi_{\mathbf{I}}(t_n)\big)$.

2. If $A = \Box(A_1, \ldots, A_n)$, where $A_1, \ldots, A_n$ are formulae, and $\widetilde{\Box}$ is the associated truth function to $\Box$, then $\mathrm{val}_{\mathbf{I}}(A) = \widetilde{\Box}\big(\mathrm{val}_{\mathbf{I}}(A_1), \ldots, \mathrm{val}_{\mathbf{I}}(A_n)\big)$.

3. If $A = (\mathsf{Q}x)(B(x))$, and $\widetilde{\mathsf{Q}}$ is the associated truth function to Q, then $\mathrm{val}_{\mathbf{I}}(A) = \widetilde{\mathsf{Q}}\big\{\mathrm{val}_{\mathbf{I}}(B(d)) | d \in D\big\}\big)$.

3.5. Definition A formula A is called *satisfiable* iff there is an interpretation $\mathbf{I}$ so that $\mathrm{val}_{\mathbf{I}}(A) \in V^+$, in symbols $\mathbf{I} \models A$, and *unsatisfiable* otherwise. A is called a *many-valued tautology* iff it is satisfiable for every interpretation $\mathbf{I}$.

Signed Formulae

The truth of many-valued formulae can be expressed by means of the truth of formulae in two-valued logic: We "sign" a formula A with a truth value v writing it as its exponent. The intended meaning is: A^v is considered true if A takes the truth value v and false otherwise.

3.6. Definition A *signed formula* is an expression of the form A^v, where A is a many-valued formula and v a truth value. A *signed formula expression* (sfe) is a formula built up from signed formulae using $\neg$, $\vee$, $\wedge$, $\forall$ and $\exists$.

A signed formula expression of the form A^v or $\neg A^v$ is called a *signed literal*. It is called an *atomic literal* iff A is atomic or *signed atom*.

Thus the logic of signed formula experssions is nothing but classical logic based on signed formulae.

4 A Sequent Calculus for the Gödel Logic

4.1. Definition (Syntax of Sequents) A *sequent* Γ of $\mathbf{L}$ is a triple $\Gamma_f|\Gamma_*|\Gamma_t$, of finite sequences Γ_v of formulae, where $v \in V$. The Γ_v are called the components of Γ.

For a sequence of formulae, 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 sequent $\Gamma_f, \Gamma'_f| \ldots |\Gamma_t, \Gamma'_t$.

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

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

4.3. Definition The *sequent calculus* **SC** for the Gödel is given by:

1. axiom schemas of the form $[V \colon A]$,
2. weakening rules for every truth value v:
$$\frac{\Gamma}{\Gamma, [v \colon A]}\ \mathrm{w}{:}v$$
3. exchange rules for every truth value v:
$$\frac{\Gamma, [v \colon A, B], \Delta}{\Gamma, [v \colon B, A], \Delta}\ \mathrm{x}{:}v$$
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,
7. an introduction rule $\mathsf{Q} \colon v$ for every quantifier Q and every truth value v, as specified below, where the free variables α occurring in the upper sequents satisfy the so-called *eigenvariable condition*: No α occurs in the lower sequent.

(2)–(5) are called *structural rules*. (6) and (7) are called *logical rules*.

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

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

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

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

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

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

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 $\supset$ are given by

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

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

The introduction rules for quantifier $\forall$ are given by

$$\frac{\Gamma, [f\colon A(\tau)]}{\Gamma, [f\colon (\forall\, x)A(x)]}\ \forall{:}f \qquad \frac{\Gamma, [*\colon A(\tau)] \quad \Gamma, [*, t\colon A(\alpha)]}{\Gamma, [*\colon (\forall\, x)A(x)]}\ \forall{:}*$$

$$\frac{\Gamma, [t\colon A(\alpha)]}{\Gamma, [t\colon (\forall\, x)A(x)]}\ \forall{:}t$$

The introduction rules for quantifier $\exists$ are given by

$$\frac{\Gamma, [f\colon A(\alpha)]}{\Gamma, [f\colon (\exists\, x)A(x)]}\ \exists{:}f \qquad \frac{\Gamma, [*\colon A(\tau)] \quad \Gamma, [f, *\colon A(\alpha)]}{\Gamma, [*\colon (\exists\, x)A(x)]}\ \exists{:}*$$

$$\frac{\Gamma, [t\colon A(\tau)]}{\Gamma, [t\colon (\exists\, x)A(x)]}\ \exists{:}t$$

4.4. Definition (Proof) An upward tree P of sequents is called a *proof* in the sequent calculus **SC** iff every leaf is an instance of an axiom in **SC**, and all other sequents in it are obtained from the ones standing immediately above it by applying one of the rules of **SC**. The sequent at the root of P is called its *end-sequent*. A sequent Γ is called *provable* in **SC**, in symbols: $\vdash^{\mathbf{SC}} \Gamma$ iff it is the end-sequent of some proof in **SC**.

4.5. Theorem (Soundness and Completeness for **SC** of the Gödel logic) *A sequent is provable, if and only if it is valid.*

For a proof of these theorems see Zach [16].

5 Natural Deduction

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:

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

5.1. Definition The *natural deduction system* for the Gödel logic is given by:

1. Assumptions of the form $[V^-\colon A]$ where A is any formula.
2. A weakening rule for all $v \in V^+$:

$$\frac{\begin{array}{c}\Gamma^- \\ \Gamma^+\end{array}}{\Gamma^+, [v\colon A]}\ \mathrm{w}\colon v$$

   Weakenings are considered as introduction rules.

3. For every connective $\square$ and every truth value v an introduction rule $\square$:I$_v$ (if $v \in V^+$) or an elimination rule $\square$:E$_v$ (if $v \in V^-$).
4. For every quantifier Q and every truth value v_i an introduction rule Q:I$_i$ (if $v_i \in V^+$) or an elimination rule Q:E$_i$ (if $v_i \in V^-$).

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

$$\frac{\begin{array}{cc}\Gamma_0^-, \lceil[*\colon \top]\rceil & \Gamma_1^- \\ \Gamma_0^+, [t\colon \top] & \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \top{:}E_f \qquad \frac{\begin{array}{cc}\Gamma_0^-, \lceil[f\colon \top]\rceil & \Gamma_1^- \\ \Gamma_0^+, [t\colon \top] & \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \top{:}E_* \qquad \frac{}{[t\colon \top]}\ \top{:}I_t$$

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

$$\frac{\begin{array}{c}\Gamma_0^-, \lceil[*\colon \perp]\rceil \\ \Gamma_0^+, [t\colon \perp]\end{array}}{\Gamma_0^+}\ \perp{:}E_f \qquad \frac{\begin{array}{cc}\Gamma_0^-, \lceil[f\colon \perp]\rceil & \Gamma_1^- \\ \Gamma_0^+, [t\colon \perp] & \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \perp{:}E_* \qquad \frac{\begin{array}{c}\Gamma_1^- \\ \Gamma_1^+\end{array}}{\Gamma_1^+, [t\colon \perp]}\ \perp{:}I_t$$

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

$$\frac{\begin{array}{cc}\Gamma_0^-, \lceil[*\colon \neg A]\rceil & \Gamma_1^-, \lceil[*\colon A]\rceil \\ \Gamma_0^+, [t\colon \neg A] & \Gamma_1^+, [t\colon A]\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \neg{:}E_f \qquad \frac{\begin{array}{cc}\Gamma_0^-, \lceil[f\colon \neg A]\rceil & \Gamma_1^- \\ \Gamma_0^+, [t\colon \neg A] & \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \neg{:}E_*$$

$$\frac{\begin{array}{c}\Gamma_1^-, \lceil[f\colon A]\rceil \\ \Gamma_1^+\end{array}}{\Gamma_1^+, [t\colon \neg A]}\ \neg{:}I_t$$

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

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[*\colon A \wedge B]\rceil \\ \Gamma_0^+, [t\colon A \wedge B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f\colon A, B]\rceil \\ \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \wedge{:}E_f$$

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[f\colon A \wedge B]\rceil \\ \Gamma_0^+, [t\colon A \wedge B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[*\colon B]\rceil \\ \Gamma_1^+, [t\colon B]\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[*\colon A, B]\rceil \\ \Gamma_2^+\end{array} \quad \begin{array}{c}\Gamma_3^-, \lceil[*\colon A]\rceil \\ \Gamma_3^+, [t\colon A]\end{array}}{\Gamma_0^+, \ldots, \Gamma_3^+}\ \wedge{:}E_*$$

$$\dfrac{\begin{array}{c}\Gamma_1^- \\ \Gamma_1^+, [t\colon B]\end{array} \quad \begin{array}{c}\Gamma_2^- \\ \Gamma_2^+, [t\colon A]\end{array}}{\Gamma_1^+, \Gamma_2^+, [t\colon A \wedge B]}\ \wedge{:}I_t$$

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

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[*\colon A \vee B]\rceil \\ \Gamma_0^+, [t\colon A \vee B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f\colon B]\rceil \\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[f\colon A]\rceil \\ \Gamma_2^+\end{array}}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+}\ \vee{:}E_f$$

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[f\colon A \vee B]\rceil \\ \Gamma_0^+, [t\colon A \vee B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f, *\colon B]\rceil \\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[*\colon A, B]\rceil \\ \Gamma_2^+\end{array} \quad \begin{array}{c}\Gamma_3^-, \lceil[f, *\colon A]\rceil \\ \Gamma_3^+\end{array}}{\Gamma_0^+, \ldots, \Gamma_3^+}\ \vee{:}E_*$$

$$\dfrac{\begin{array}{c}\Gamma_1^- \\ \Gamma_1^+, [t\colon A, B]\end{array}}{\Gamma_1^+, [t\colon A \vee B]}\ \vee{:}I_t$$

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

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[*\colon A \supset B]\rceil \\ \Gamma_0^+, [t\colon A \supset B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f\colon B]\rceil \\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[*\colon A]\rceil \\ \Gamma_2^+, [t\colon A]\end{array}}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+}\ \supset{:}E_f$$

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[f\colon A \supset B]\rceil \\ \Gamma_0^+, [t\colon A \supset B]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[*\colon B]\rceil \\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^- \\ \Gamma_2^+, [t\colon A]\end{array}}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+}\ \supset{:}E_*$$

$$\dfrac{\begin{array}{c}\Gamma_1^-, \lceil[f\colon A]\rceil, \lceil[*\colon B]\rceil \\ \Gamma_1^+, [t\colon B]\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[f, *\colon A]\rceil \\ \Gamma_2^+, [t\colon B]\end{array}}{\Gamma_1^+, \Gamma_2^+, [t\colon A \supset B]}\ \supset{:}I_t$$

The introduction and elimination rules for quantifier $\forall$ are given by

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[*\colon (\forall x)A(x)]\rceil \\ \Gamma_0^+, [t\colon (\forall x)A(x)]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f\colon A(\alpha)]\rceil \\ \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \forall{:}E_f$$

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[f\colon (\forall x)A(x)]\rceil \\ \Gamma_0^+, [t\colon (\forall x)A(x)]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[*\colon A(\tau)]\rceil \\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[f, *\colon A(\alpha)]\rceil \\ \Gamma_2^+\end{array}}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+}\ \forall{:}E_*$$

$$\dfrac{\begin{array}{c}\Gamma_1^- \\ \Gamma_1^+, [t\colon A(\tau)]\end{array}}{\Gamma_1^+, [t\colon (\forall x)A(x)]}\ \forall{:}I_t$$

The introduction and elimination rules for quantifier $\exists$ are given by

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[*: (\exists x)A(x)]\rceil\\ \Gamma_0^+, [t: (\exists x)A(x)]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[f: A(\alpha)]\rceil\\ \Gamma_1^+\end{array}}{\Gamma_0^+, \Gamma_1^+}\ \exists{:}E_f$$

$$\dfrac{\begin{array}{c}\Gamma_0^-, \lceil[f: (\exists x)A(x)]\rceil\\ \Gamma_0^+, [t: (\exists x)A(x)]\end{array} \quad \begin{array}{c}\Gamma_1^-, \lceil[*: A(\tau)]\rceil\\ \Gamma_1^+\end{array} \quad \begin{array}{c}\Gamma_2^-, \lceil[f, *: A(\alpha)]\rceil\\ \Gamma_2^+\end{array}}{\Gamma_0^+, \Gamma_1^+, \Gamma_2^+}\ \exists{:}E_*$$

$$\dfrac{\begin{array}{c}\Gamma_1^-\\ \Gamma_1^+, [t: A(\tau)]\end{array}}{\Gamma_1^+, [t: (\exists x)A(x)]}\ \exists{:}I_t$$

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

1. Let A be any formula. Then

$$\dfrac{[V^-: A]}{[V^+: A]}$$

   is a derivation of A from the assumption $[V^-: A]$ (an *initial derivation*).

2. If D_k are derivations of Γ_k^+, Δ_k^+ from the assumptions $\Gamma_k^-, \hat{\Delta}_k^-$, and

$$\dfrac{\left\langle \begin{array}{c}\Gamma_k^-, \lceil\Delta_k^-\rceil\\ \Gamma_k^+, \Delta_k^+\end{array} \right\rangle_{k\in K}}{\Pi^+}$$

   is an instance of a deduction rule with $\hat{\Delta}_k^-$ a subsequent of Δ_k^-, and all eigenvariable conditions are satisfied, then

$$\dfrac{\langle D_k\rangle_{k\in K}}{\Pi^+}$$

   is a derivation of Π^+ from the assumptions $\bigcup_{k\in K}\Gamma_k^-$. The formulae in $\hat{\Delta}_k^-$ which do not occur in $\bigcup_{k\in K}\Gamma_k^-$ are said to be *discharged* at this inference.

5.3. Definition We call a formula occurrence A

1. the *conclusion formula* of an introduction, if it is the formula being introduced, i.e., it is F in the conclusion $[i: F]$;

2. a *premise formula* of an introduction, if it is one of the formulae in $\Delta'_{f:i}(j)^+$ in that introduction;

3. a *major premise formula* of an elimination, if it is among the formula being eliminated, i.e., in the major premise $[V^+: F]$;

4. a *minor premise formula* of an elimination, if it is among the formulae in $\Delta'_{f:i}(j)^+$ in that elimination,

5. a *discharged assumption formula* of an elimination, if it stands immediately below an assumption which contains the formulae in $\Delta'_{f:i}(j)^-$ being discharged at that elimination.

A formula occurrence A is said to *follow* A', if both are of the same form and A' stands immediately above A at the same position.

5.4. Theorem (Soundness) *If a partial sequent Γ^+ can be derived from the assumptions Γ^-, then the following holds for every interpretation $\mathbf{I}$: If no formula in Γ^-_v $(v \in V^-)$ evaluates to the truth value v, then there is a $w \in V^+$ and a formula in Γ^+_w that evaluates to w.*

5.5. Theorem (Completeness) *Natural deduction systems are complete.*

For a proof see Zach [16].

6 Clause Formation Rules

6.1. Definition A *(many-valued) clause* $C = \{A_1^{w_1}, \ldots, A_n^{w_n}\}$ 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 formulae.

6.2. Definition (Semantics of Clauses and Sets of Clauses) We translate clauses (clause sets) into signed formula expression by defining:

$$\mathrm{sfe}(C) \stackrel{\mathrm{def}}{=} \forall \vec{x} \bigvee_{L \in C} L$$

$$\mathrm{sfe}(\mathcal{C}) \stackrel{\mathrm{def}}{=} \bigwedge_{C \in \mathcal{C}} \mathrm{sfe}(C)$$

Satisfiability and validity can then be expressed in terms of sfes: A clause C (clause set $\mathcal{C}$) is satisfiable (valid), if and only if $\mathrm{sfe}(C)$ ($\mathrm{sfe}(\mathcal{C})$) is satisfiable (valid).

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

$$\frac{\mathcal{C} \cup \{C \cup \{\top^f\}\}}{\mathcal{C} \cup \{C\}}\ \top{:}f \qquad \frac{\mathcal{C} \cup \{C \cup \{\top^*\}\}}{\mathcal{C} \cup \{C\}}\ \top{:}* \qquad \frac{\mathcal{C} \cup \{C \cup \{\top^t\}\}}{\mathcal{C}}\ \top{:}t$$

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

$$\frac{\mathcal{C} \cup \{C \cup \{\bot^f\}\}}{\mathcal{C}}\ \bot{:}f \qquad \frac{\mathcal{C} \cup \{C \cup \{\bot^*\}\}}{\mathcal{C} \cup \{C\}}\ \bot{:}* \qquad \frac{\mathcal{C} \cup \{C \cup \{\bot^t\}\}}{\mathcal{C} \cup \{C\}}\ \bot{:}t$$

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\mathcal{C} \cup \{C \cup \{(A \supset B)^f\}\}}{\mathcal{C} \cup \{C \cup \{B^f\},\ C \cup \{A^*, A^t\}\}}\ \supset{:}f \qquad \frac{\mathcal{C} \cup \{C \cup \{(A \supset B)^*\}\}}{\mathcal{C} \cup \{C \cup \{B^*\},\ C \cup \{A^t\}\}}\ \supset{:}*$$

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

The clause formation rules for quantifier $\forall$ are given by

$$\frac{\mathcal{C} \cup \{C \cup \{((\forall x)A(x))^f\}\}}{\mathcal{C} \cup \{C \cup \{A(f(\vec{y}))^f\}\}}\ \forall{:}f$$

$$\frac{\mathcal{C} \cup \{C \cup \{((\forall x)A(x))^*\}\}}{\mathcal{C} \cup \{C \cup \{A(f(\vec{y}))^*\},\ C \cup \{A(b)^*, A(b)^t\}\}}\ \forall{:}*$$

$$\frac{\mathcal{C} \cup \{C \cup \{((\forall x)A(x))^t\}\}}{\mathcal{C} \cup \{C \cup \{A(b)^t\}\}}\ \forall{:}t$$

The clause formation rules for quantifier $\exists$ are given by

$$\frac{\mathcal{C} \cup \{C \cup \{((\exists x)A(x))^f\}\}}{\mathcal{C} \cup \{C \cup \{A(b)^f\}\}}\ \exists{:}f$$

$$\frac{\mathcal{C} \cup \{C \cup \{((\exists x)A(x))^*\}\}}{\mathcal{C} \cup \{C \cup \{A(f(\vec{y}))^*\},\ C \cup \{A(b)^*, A(b)^f\}\}}\ \exists{:}*$$

$$\frac{\mathcal{C} \cup \{C \cup \{((\exists x)A(x))^t\}\}}{\mathcal{C} \cup \{C \cup \{A(f(\vec{y}))^t\}\}}\ \exists{:}t$$

6.3. THEOREM *Let D be the result of exhaustively applying the translation rules to a clause set $\mathcal{C}$. Then D is a set of proper clauses, i.e. D contains only signed atoms (all many-valued connectives and quantifiers are eliminated). Furthermore, D is equivalent to $\mathcal{C}$ with respect to satisfiability.*

7 Future directions of research

In this paper we presented an optimal sequent calculus, an optimal natural deduction system, and optimal clause formation rules. Currently we are searching for optimized versions of other calculi for the Gödel logic, like tableau systems or negative variants of sequent calculus. Another interesting topic are calculi based on sets as signs [7]. On the practical side, we intend to implement the calculi in a prototype using Prolog.

Acknowledgements

I am deeply indebted to my creators. In particular I would like to thank M. Baaz, C. Fermüller and R. Zach for laying the foundations, and G. Salzer for calling me into existence. Last but not least, I am grateful to A. Leitgeb, W. Nix and M. Schranz for their cosmetical surgery, greatly enhancing my outward appearance.

References

[1] M. Baaz and C. G. Fermüller. Resolution for many-valued logics. In A. Voronkov, editor, *Logic Programming and Automated Reasoning (LPAR'92)*, LNCS 624 (LNAI), pages 107–118. Springer, 1992.

[2] M. Baaz and C. G. Fermüller. Resolution-based theorem proving for many-valued logics. *J. Symbolic Computation*, 19:353–391, 1995.

[3] M. Baaz, C. G. Fermüller, and R. Zach. Systematic construction of natural deduction systems for many-valued logics. In *Proc. 23rd International Symposium on Multiple-valued Logic*, pages 208–215. IEEE Computer Society Press, Los Alamitos, May 24–27 1993.

[4] W. A. Carnielli. Systematization of finite many-valued logics through the method of tableaux. *J. Symbolic Logic*, 52(2):473–493, 1987.

[5] W. A. Carnielli. On sequents and tableaux for many-valued logics. *J. Non-Classical Logic*, 8(1):59–76, 1991.

[6] R. Hähnle. Uniform notation of tableaux rules for multiple-valued logics. In *Proc. 21st International Symposium on Multiple-valued Logic*, pages 238–245, 1991.

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

[8] Reiner Hähnle. *Tableaux-Based Theorem Proving in Multiple-Valued Logics.* PhD thesis, Universität Karlsruhe, Institut für Logik, Komplexität und Deduktionssysteme, may 1992.

[9] P. O'Hearn and Z. Stachniak. A resolution framework for finitely-valued first-order logics. *J. Symbolic Computation*, 13:235–254, 1992.

[10] G. Rousseau. Sequents in many valued logic I. *Fund. Math.*, 60:23–33, 1967.

[11] G. Salzer. MUltlog: an expert system for multiple-valued logics. In *Collegium Logicum: Annals of the Kurt-Gödel-Society*, volume 2. Springer, 1996.

[12] G. Salzer. Optimal axiomatizations for multiple-valued operators and quantifiers based on semi-lattices. In *13th Int. Conf. on Automated Deduction (CADE'96)*, LNCS (LNAI). Springer, 1996.

[13] K. Schröter. Methoden zur Axiomatisierung beliebiger Aussagen- und Prädikatenkalküle. *Z. Math. Logik Grundlag. Math.*, 1:241–251, 1955.

[14] S. J. Surma. An algorithm for axiomatizing every finite logic. In D. C. Rine, editor, *Computer Science and Multiple-valued Logic: Theory and Applications*, pages 137–143. North-Holland, Amsterdam, 1977.

[15] M. Takahashi. Many-valued logics of extended Gentzen style I. *Sci. Rep. Tokyo Kyoiku Daigaku Sect A*, 9(231):95–110, 1967.

[16] R. Zach. *Proof Theory of Finite-valued Logics.* Diplomarbeit, Technische Universität Wien, 1993.