Terminating Tableaux for the Basic Fragment of Simple Type Theory

Chad E. Brown and Gert Smolka
Saarland University

February 2, 2009
(revised April 21, 2009)

To appear in Tableaux 2009, LNCS, Springer

We consider the basic fragment of simple type theory, which restricts equations to base types and disallows lambda abstractions and quantifiers. We show that this fragment has the finite model property and that satisfiability can be decided with a terminating tableau system. Both results are with respect to standard models.

1 Introduction

We are interested in higher-order fragments of classical simple type theory [1, 6] for which it is decidable whether a formula is satisfied by a standard model. Only few such fragments are known:

- The propositional fragment, which is obtained by admitting no other base type but the type of truth values. In this case decidability follows from the fact that all types are interpreted as finite sets.
- The fragment consisting of disequations $s \neq t$ where s and t are pure terms that do not involve the type of truth values. The decidability follows from the completeness of lambda conversion [8].
- The fragments that correspond to propositional modal logics with inductive expressivity, for instance PDL [7] and the propositional μ-calculus [10].

In this paper we will show that the fragment of simple type theory that restricts equations to base types and disallows lambda abstraction and quantification is

decidable. We call the formulas of this fragment *basic*. Here are examples of unsatisfiable basic formulas:

1. $h(h\bot{=}h\neg\bot) \neq h\bot$ — $h : o\iota$
2. $h(f(f(fx))) \neq h(fx)$ — $x : o,\ f : oo,\ h : o\iota$
3. $x{\neq}y \wedge gx{=}y \wedge gy{=}x \wedge f(f(fx)){=}g(fx)$ — $a, x, y : o,\ f, g : oo$
4. $x{\neq}y \wedge gx{=}y \wedge gy{=}x \wedge pg \wedge \neg p(\neg)$ — $x, y : o,\ g : oo,\ p : (oo)o$
5. $qfx \wedge f(fx) \wedge f(qfx){\neq}fx$ — $x : o,\ f : oo,\ q : (oo)oo$

None of the formulas is a formula of standard first-order logic. Seen from the perspective of first-order logic, basic formulas are quantifier-free formulas where terms can be formulas and predicates and functions can be higher-order.

Most of the above formulas are out of the reach of the automated tactics of Isabelle [12] and the higher-order provers TPS [2] and LEO-II [4]. We hope that the techniques of this paper will contribute to better auto tactics for higher-order proof assistants.

Our decision procedure comes in the form of a terminating tableau system, which is a subsystem of a tableau system for full extensional type theory. The extended system is the dual of a Henkin-complete cut-free one-sided sequent calculus devised by Brown [5], which has been the starting point for the research reported in this paper. The most difficult part of the correctness proof for the terminating system is a model existence theorem, which we establish with the possible-values technique. The possible-values technique originated with cut elimination proofs [14, 13] and has been used by Brown [5] to obtain Henkin models. We seem to be the first to obtain standard models with the possible-values technique.

2 Basic Definitions

Types (σ, τ, μ) are obtained with the grammar $\sigma ::= \iota \mid o \mid \sigma\sigma$. The elements of o are the two truth values, ι is interpreted as a nonempty set, and a function type $\sigma\tau$ is interpreted as the set of all total functions from σ to τ.

We assume a countable set of **parameters** (x), where every parameter comes with a unique type, and where for every type there are infinitely many parameters of this type. We employ the **logical constants** $\bot : o$, $\neg : oo$, $\wedge : ooo$ and $=_\sigma : \sigma\sigma o$, where there is a logical constant $=_\sigma$ for every type σ. The logical constants take their standard interpretation. **Terms** (s, t, u, v) are defined inductively such that every term has a unique type: (1) every parameter is a term; (2) every logical constant is a term; (3) if s is a term of type $\tau\mu$ and t is a term of type τ, then st is a term of type μ; (4) if x is a name of type σ and t is a term of type τ, then $\lambda x.t$ is a term of type $\sigma\tau$. We write $s : \sigma$ to say that s is a term of type σ. If T is

a set of terms, T^σ denotes the set of all terms that are in T and have type σ.

The logical constants $=_\sigma$ are called **identities**, and terms of type o are called **formulas**. Formulas of the form $s =_\sigma t$ are called **equations**, and formulas of the form $\neg(s =_\sigma t)$ are called **disequations**. We write disequations as $s \neq_\sigma t$. We usually write equations and disequations without the type index σ.

A term is **basic** if it contains no other identity but $=_\iota$. We write Λ_σ for the set of all basic terms of type σ. A formula is **normal** if it is a basic formula or a disequation $s \neq t$ where s and t are basic terms. A **normal set** is a set of normal formulas.

The definition of normal formulas is asymmetric in that equations are restricted to type ι while disequations $s \neq_\sigma t$ are allowed at any type σ. The reason for this asymmetry is that the tableau system uses disequations as internal workhorse. Since $s \neq t$ and $ps \wedge \neg pt$ are equisatisfiable if p is fresh, normal formulas are not more expressive than basic formulas.

The definition of basic formulas can be extended with further propositional connectives including $=_o$. Since they can be expressed with the connectives we already have, this does not buy new expressivity.

For simplicity we provide only one base type ι different from o. Everything generalizes to countably many base types.

3 Tableau System

Figure 1 shows the rules of a terminating tableau system $\mathcal{B}$ that decides the satisfiability of finite normal sets. For the application constraint of Rule FE we supply the following definition. A disequation $s \neq_{\sigma\tau} t$ is **evident in** A if there exist $n \geq 1$ basic terms $u_1, \ldots, u_n$ such that $su_1 \ldots u_n \neq tu_1 \ldots u_n$ or $tu_1 \ldots u_n \neq su_1 \ldots u_n$ is in A. The names of the rules are derived as follows: MAT for Mating, FE for functional extensionality, BE for Boolean extensionality, DEC for decomposition, and CON for confrontation.

The rules in the first line of Figure 1 are the usual tableau rules deciding propositional logic. They also decide quantifier-free first-order logic without equality. In contrast to classical first-order logic, type theory allows *embedded formulas*, for instance $p(\neg x)$ where $p : oo$ and $x : o$. The rules MAT, DEC and BE handle embedded formulas. MAT decomposes "atomic" formulas into disequations, which are further decomposed with DEC. Embedded formulas are then fed back to the propositional rules by BE, as demonstrated by the following example.

$$\textbf{BOT}_\neg\ \frac{s\,,\ \neg s}{\bot} \qquad \textbf{DN}\ \frac{\neg\neg s}{s} \qquad \textbf{AND}\ \frac{s \wedge t}{s\,,\ t} \qquad \textbf{AND}_\neg\ \frac{\neg(s \wedge t)}{\neg s \mid \neg t}$$

$$\textbf{MAT}\ \frac{xs_1 \ldots s_n\,,\ \neg x t_1 \ldots t_n}{s_1 \neq t_1 \mid \cdots \mid s_n \neq t_n} \qquad \textbf{DEC}\ \frac{xs_1 \ldots s_n \neq_\iota x t_1 \ldots t_n}{s_1 \neq t_1 \mid \cdots \mid s_n \neq t_n}$$

$$\textbf{BOT}_\neq\ \frac{s \neq s}{\bot} \qquad \textbf{BE}\ \frac{s \neq_o t}{s\,,\ \neg t \ \mid\ \neg s\,,\ t}$$

$$\textbf{FE}\ \frac{s \neq_{\sigma\tau} t}{sx \neq tx}\ x : \sigma \text{ fresh and } s \neq t \text{ not evident in } A$$

$$\textbf{SYM}\ \frac{s =_\iota t}{t = s} \qquad \textbf{CON}\ \frac{s =_\iota t\,,\ u \neq_\iota v}{s \neq u \mid t \neq v}$$

A is the normal set to which the rule is applied
x fresh means that x does not occur in A
MAT and DEC assume $n \geq 1$

Figure 1: Tableau system $\mathcal{B}$

Example 3.1 The following tableau refutes an unsatisfiable normal set with embedded formulas.

$$p(fx(\neg\neg y)),\ \neg p(fxy)$$
MAT
$$fx(\neg\neg y) \neq fxy$$
DEC

	$\neg\neg y \neq y$ BE	
$x \neq x$		$\neg\neg\neg y,\ y$
$\textbf{BOT}_\neq$	$\neg\neg y,\ \neg y$	DN
$\bot$	$\textbf{BOT}_\neg$	$\neg y$
	$\bot$	$\textbf{BOT}_\neg$
		$\bot$

The types of the parameters are $p : \iota o$, $f : \iota o \iota$, $x : \iota$, and $y : o$. □

Example 3.2 Rules SYM and CON handle positive equations at ι. This is demonstrated by the following refutation.

$$\dfrac{a = b,\ fa = gb,\ fb \neq ga}{\text{CON}}$$

	$gb \neq ga$
$fa \neq fb$	DEC
DEC	$b \neq a$
$a \neq b$	SYM
BOT$_\neg$	$b = a$
$\bot$	BOT$_\neg$
	$\bot$

The types of the parameters are $a, b : \iota$ and $f, g : \iota\iota$. □

The confrontation rule CON does not exist in first-order systems. First-order systems typically employ the replacement rule

$$\text{REP}\ \dfrac{s =_\iota t\ ,\ C[s]}{C[t]}$$

Example 3.2 can also be refuted with the replacement rule instead of the confrontation rule. However, the confrontation rule is more powerful than the replacement rule since it supports the decomposition needed for embedded formulas. This is illustrated by the next example, which cannot be refuted with the replacement rule.

Example 3.3 Consider the normal set

$$\begin{array}{lll} fa = gb, & f'a = g'c, & f''b = g''c \\ fb \neq ga, & f'c \neq g'a, & f''c \neq g''b \end{array}$$

with the typing $a, b : o$ and $f, g, f', g', f'', g'' : o\iota$. The replacement rule REP cannot be applied to the set. The confrontation rule CON can be applied to 3 confrontation pairs. Application of DEC now yields 8 sets that all contain the unsatisfiable set

$$a \neq b,\ a \neq c,\ b \neq c$$

up to symmetry. This set can be refuted with BE and BOT$_\neg$. Thus the initial set can be refuted with $\mathcal{B}$. □

It remains to illustrate the use of the functional extensionality rule FE. FE is only needed if higher-order parameters are present.

Example 3.4 The following tableau has two branches both of which contain the refutable set $\{a \neq b,\ a \neq c,\ b \neq c\}$. Thus the set in the first line is refutable.

$$a \neq b,\ fa,\ fb,\ ga,\ gb,\ pf,\ \neg pg$$
MAT
$$f \neq g$$
FE
$$fc \neq gc$$
BE

$fc,\ \neg gc$	$\neg fc,\ gc$
MAT	MAT
$a \neq c$	$a \neq c$
MAT	MAT
$b \neq c$	$b \neq c$

Note the crucial use of the functional extensionality rule FE. The types of the parameters are $a, b, c : o$, $f, g : oo$ and $p : (oo)o$. □

In summary we can now say that $\mathcal{B}$ extends the classical propositional system with the rules MAT, DEC, $\text{BOT}_{\neq}$, BE, FE and CON to account for embedded formulas. MAT and DEC decompose formulas that are atomic for the classical rules. This way BE can finally lift embedded formulas to the top level. To deal with equality, the traditional replacement rule is replaced by the confrontation rule. All rules so far are already needed for first-order normal formulas. For higher-order parameters a single rule FE incorporating functional extensionality is needed.

The only higher-order tableau system we could find in the literature is the calculus of Kohlhase [9]. Kohlhase's calculus does not have equality, but there are unification constraints that play the role of our top level disequations. For unification constraints Kohlhase has rules that are similiar to MAT, DEC, FE, and BE. Our tableau rules also have similarities with the rules in Benzmüller's [3] higher-order RUE-resolution calculus, which employs primitive equality. In particular, the RUE calculus allows resolution of positive equations against negative equations (which play the role of unification constraints). Combining this form of resolution with decomposition, one obtains a rule

$$\frac{C \vee s = t \qquad D \vee u \neq v}{C \vee D \vee s \neq u \vee t \neq v}$$

which is essentially the same as our confrontation rule CON.

4 Soundness and Termination

The rules in Figure 1 apply to normal sets and produce one or several normal sets by adding normal formulas. More precisely, if a rule applies to a normal set A, it yields $n \geq 1$ normal sets $A_1, \ldots, A_n$ called **alternatives** such that $A \subseteq A_i$ for all $i \in \{1, \ldots, n\}$. If $n \geq 2$, the rule applied is called **branching**. To obtain a terminating system, we admit only applications where $\bot \notin A$ and the alternatives $A_1, \ldots, A_n$ are all proper supersets of A (i.e., $A \subsetneq A_i$). The tableau system $\mathcal{B}$ is **sound** if for every application of a rule the set A is satisfiable if and only if one of the alternatives $A_1, \ldots, A_n$ is satisfiable. For the termination of $\mathcal{B}$ we consider the relation $A \to A'$ on normal sets that holds if and only if A' can be obtained as an alternative by a rule that applies to A. We say that $\mathcal{B}$ **terminates** if this relation terminates on finite normal sets. Finally, we call a normal set A **evident** if $\bot \notin A$ and no rule of $\mathcal{B}$ applies to A. We will show the following:

- $\mathcal{B}$ is sound.
- $\mathcal{B}$ terminates on finite normal sets.
- Evident sets are satisfiable, and finite evident sets are finitely satisfiable.

Together, soundness, termination and model existence yield a decision procedure for the satisfiability of finite normal sets.

Proposition 4.1 (Soundness) $\mathcal{B}$ is sound.

Proof Let $A_1, \ldots, A_n$ be obtained from A by application of a rule. It suffices to show that for every model of A there exists an interpretation that is a model of at least one of the alternatives $A_1, \ldots, A_n$. For $\text{BOT}_\neg$ this follows from the fact that the implication $x \wedge \neg x \to \bot$ is valid. For $\text{AND}_\neg$ the validity of $\neg(x \wedge y) \to \neg x \vee \neg y$ suffices, and for FE the validity of $f \neq g \to \exists x. fx \neq gx$ does the job. Note that this is in fact equivalent to functional extensionality $(\forall x. fx = gx) \to f = g$. The soundness of the other rules follows with similar arguments. ∎

For the termination proof we distinguish between Rule FE and the other rules. FE is special in that it introduces new parameters. We first show that the subsystem $\mathcal{B}_0$ of $\mathcal{B}$ obtained by removing FE terminates. The proof will exhibit an upper bound function $\mathcal{U}$ from sets of terms to sets of terms such that the following holds for every derivation $A_1 \to \cdots \to A_n$ in $\mathcal{B}_0$:

1. $\mathcal{U}A_1 = \cdots = \mathcal{U}A_n$
2. $\mathcal{U}A_i$ is a finite set such that $A_i \subseteq \mathcal{U}A_i$ for all $i = 1, \ldots, n$.

Since $A_1 \subsetneq \cdots \subsetneq A_n$, the bound function suffices for termination of $\mathcal{B}_0$.

Let T range over sets of terms. We define the bound function as $\mathcal{U}T := C(S(\mathcal{E}T))$ where the functions S (subterms), $\mathcal{E}$ (elements) and C (compounds) are defined as follows:

- $\mathcal{E}T$ is the least set of terms such that:
  1. $\bot \in \mathcal{E}T$
  2. If $(s =_\iota t) \in T$, then $s, t \in \mathcal{E}T$.
  3. If $(s \neq t) \in T$, then $s, t \in \mathcal{E}T$.
  4. If $\neg s \in T$ and s is not an equation, then $s \in \mathcal{E}T$.
  5. If $s \in T$ and s is neither a negated term nor an equation, then $s \in \mathcal{E}T$.
- ST is the set of all subterms of the terms in T.
- CT is the least set of terms such that:
  1. $T \subseteq CT$.
  2. If $s, t \in T^\iota$, then $(s =_\iota t) \in CT$.
  3. If $s, t \in T^\sigma$, then $(s \neq t) \in CT$.
  4. If $s \in T^o$, then $\neg s \in CT$.

All three functions are monotone functions from set of terms to set of terms that preserve finiteness. The following properties are easy to verify:

1. If $A \to A'$ in $\mathcal{B}_0$, then $S(\mathcal{E}A) = S(\mathcal{E}A')$.
2. $T \subseteq C(\mathcal{E}T) \subseteq C(S(\mathcal{E}T))$.

Hence $\mathcal{U}$ has the required properties and $\mathcal{B}_0$ terminates.

We now extend the termination result to $\mathcal{B}$. We define the **power of** A as the multiset that contains for each function type σ as many copies of σ as there are 2-element subsets $\{s, t\} \subseteq (S(\mathcal{E}A))^\sigma$ such that $s \neq t$ is not evident in A. It is not difficult to verify the following for normal sets A:

1. Application of Rule FE decreases the power of A.
2. Application of a rule other than FE does not increase the power of A.

Hence we have proved the termination of $\mathcal{B}$.

Proposition 4.2 (Termination) $\mathcal{B}$ terminates.

5 Model Existence

Recall that an evident set is a normal set that does not contain $\bot$ and to which no rule of $\mathcal{B}$ can add a formula.

Theorem 5.1 (Model Existence) Every evident set has a model, and every finite evident set has a finite model.

Before proving the theorem we state two important consequences.

Corollary 5.2 The tableau system $\mathcal{B}$ constitutes a decision procedure for the satisfiability of normal formulas.

Proof Follows from Theorem 5.1 since $\mathcal{B}$ is sound and terminating. ∎

Corollary 5.3 Normal formulas have the finite model property.

Proof Let s be a satisfiable normal formula. Since $\mathcal{B}$ is sound and terminating, we can obtain a finite evident set E that contains s. Now Theorem 5.1 gives us a finite model of E and hence of s. ∎

We now begin the proof of the model existence theorem. Let E be an evident set. We will construct a model $\mathcal{I}$ of E that is finite if E is finite. We arrange the following:

- $\mathcal{I}o := \{0,1\}$
- $\mathcal{I}(\sigma\tau)$:= set of all total functions from $\mathcal{I}\sigma$ to $\mathcal{I}\tau$
- $\mathcal{I}\bot := 0$
- $\mathcal{I}(\neg)$, $\mathcal{I}(\wedge)$, and $\mathcal{I}(=_\sigma)$ are defined as usual.

It remains to define $\mathcal{I}$ for the type ι and the parameters.

Discriminants

We prepare the definition of $\mathcal{I}\iota$. We write $s \sharp t$ if E contains $s \neq t$ or $t \neq s$. A term $s \in \Lambda_\iota$ is **discriminating** if there is a term t such that $s \sharp t$. We write Δ for the set of all discriminating terms. A **discriminant** is a maximal subset $D \subseteq \Delta$ such that for all $s \sharp t$ either $s \notin D$ or $t \notin D$. We define $\mathcal{I}\iota$ to be the set of all discriminants.

- $\mathcal{I}\iota := \{ D \mid D \text{ is a discriminant} \}$

Example 5.4 Suppose $E = \{x \neq y, x \neq z, y \neq z\}$ and $x, y, z : \iota$. Then there are 3 discriminants: $\{x\}$, $\{y\}$, $\{z\}$. □

Example 5.5 Suppose $E = \{x \neq f(fx), fx \neq f(f(fx))\}$ and $f : \iota\iota$. Then there are 4 discriminants: $\{x, fx\}$, $\{x, f(f(fx))\}$, $\{f(fx), fx\}$, $\{f(fx), f(f(fx))\}$. □

Proposition 5.6 If E contains exactly n disequations at ι, then there are at most 2^n discriminants. If E contains no disequation at ι, then $\emptyset$ is the only discriminant.

Proposition 5.7 Let D_1 and D_2 be different discriminants. Then:

1. D_1 and D_2 are separated by a disequation in E, that is, there exist terms $t_1 \in D_1$ and $t_2 \in D_2$ such that $t_1 \sharp t_2$.
2. D_1 and D_2 are not connected by an equation in E, that is, there exist no terms $t_1 \in D_1$ and $t_2 \in D_2$ such that $(t_1 = t_2) \in E$.

Proof The first claim follows by contradiction. Suppose there are no terms $t_1 \in D_1$ and $t_2 \in D_2$ such that $t_1 \not\sharp t_2$. Let $t \in D_1$. Then $t \in D_2$ since D_2 is a maximal compatible set of discriminating terms. Thus $D_1 \subseteq D_2$. A symmetric argument yields $D_2 \subseteq D_1$. Contradiction.

The second claim also follows by contradiction. Suppose there is an equation $(s_1 \approx s_2) \in E$ such that $s_1 \in D_1$ and $s_2 \in D_2$. By the first claim we have terms $t_1 \in D_1$ and $t_2 \in D_2$ such that $t_1 \not\sharp t_2$. Since E is closed under CON, we have $s_1 \not\sharp t_1$ or $s_2 \not\sharp t_2$. Contradiction since D_1 and D_2 are discriminants. ∎

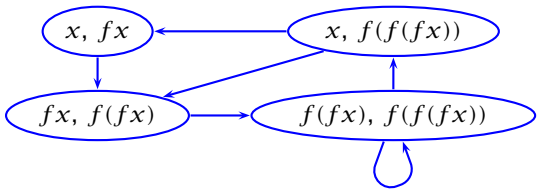
Possible Values

It remains to define $\mathcal{I}$ for the parameters. Given the presence of higher-order parameters this is not straightforward. We base the definition on a family of relations $\rhd_\sigma \subseteq \Lambda_\sigma \times \mathcal{I}\sigma$ defined by induction on types:

$$\begin{aligned}
s \rhd_o 0 &:\Longleftrightarrow s \notin E \\
s \rhd_o 1 &:\Longleftrightarrow \neg s \notin E \\
s \rhd_\iota D &:\Longleftrightarrow s \in [D] \\
s \rhd_{\sigma\tau} f &:\Longleftrightarrow st \rhd_\tau fa \text{ whenever } t \rhd_\sigma a \\
[D] &:= D \cup \{\, s \in \Lambda_\iota \mid s \text{ not discriminating} \,\}
\end{aligned}$$

We read $s \rhd a$ as "s can be a" or "a is a **possible value** for s". Note that if s is a basic formula such that $s \notin E$ and $\neg s \notin E$, then both 0 and 1 are possible values for s. We will show that every basic term has a possible value and that we obtain a model of E if we define $\mathcal{I}x$ as a possible value for x for every parameter x. Such a model will satisfy $s \rhd \hat{\mathcal{I}}s$ for every basic term s. Note that $\hat{\mathcal{I}}s$ denotes the value the term s evaluates to in the model $\mathcal{I}$.

Example 5.8 Suppose $E = \{x \not\approx f(fx),\ fx \not\approx f(f(fx))\}$ and $f : \iota\iota$. The following graph shows the discriminants and the possible pairs for possible values of f.



There are 2 possible values for x. To obtain a possible value for f, we must choose for every node exactly one departing edge. Hence there are 4 possible values for f. For each choice of a value for x and f we obtain a model of E. Altogether we obtain 8 models this way. Four of the obtained models have a junk value at ι (i.e., a value that is not denoted by a basic term). From the models

with a junk value we can obtain three-valued models. There is no two-valued model. □

Compatibility

We define a family of relations $\|_\sigma \subseteq \Lambda_\sigma \times \Lambda_\sigma$ by induction on types:

$$\begin{aligned} s \|_o t &:\Longleftrightarrow \{s, \neg t\} \notin E \text{ and } \{\neg s, t\} \notin E \\ s \|_\iota t &:\Longleftrightarrow \text{not } s \sharp t \\ s \|_{\sigma\tau} t &:\Longleftrightarrow su \|_\tau tv \text{ whenever } u \|_\sigma v \end{aligned}$$

We say that s and t are **compatible** if $s \| t$. A set T of terms is **compatible** if $s \| t$ for all terms $s, t \in T$. If $T \subseteq \Lambda_\sigma$, we write $T \triangleright a$ if a is a common possible value for all terms $s \in T$. We will show that a set of equi-typed terms is compatible if and only if all its terms have a common possible value.

Proposition 5.9 The compatibility relations $\|_\sigma$ are symmetric.

The compatibility relations are also reflexive. Showing this fact will take some effort. For the induction to go through we will strengthen the hypothesis. First we prove the following lemma.

Lemma 5.10 Let s be a basic term. Then:

1. If $s : o$, then $s \| s$.
2. If $s = cs_1 \dots s_n$ and c is a logical constant, then $s \| s$.

Proof The first claim follows by contradiction. Suppose $s \nparallel_o s$. Then $s, \neg s \in E$, contradicting the assumption that E is evident.

Given that the first claim holds, for the second claim it suffices to consider terms of the forms $\neg$, $\wedge$, $=_\iota$, $(\wedge)t$, and $(=_\iota)t$. In all cases the claim follows by contradiction. We show $=_\iota \| =_\iota$. The other cases are similar.

Suppose $=_\iota \nparallel =_\iota$. Then there exist terms such that $s_1 \|_\iota s_2$, $t_1 \|_\iota t_2$, and $s_1{=_\iota}t_1 \nparallel_o s_2{=_\iota}t_2$. Then either $s_1{=}t_1$ and $s_2{\neq}t_2$ are both in E or $s_1{\neq}t_1$ and $s_2{=}t_2$ are both in E. Since E is closed under CON, we have either $s_1 \sharp s_2$ (contradicting $s_1 \| s_2$) or $t_1 \sharp t_2$ (contradicting $t_1 \| t_2$). ■

Lemma 5.11 (Reflexivity) For every type σ and all basic terms s, t, $xs_1 \dots s_n$, $xt_1 \dots t_n$ of type σ:

1. We never have both $s \|_\sigma t$ and $s \sharp t$.
2. We always have either $xs_1 \dots s_n \|_\sigma xt_1 \dots t_n$ or $s_i \sharp t_i$ for some $i \in \{1, \dots, n\}$.
3. We always have $s \|_\sigma s$.

Proof By mutual induction on σ. The base cases for Claim (1) follow easily from the definition of compatibility and closure of E under BE. The base cases for Claim (2) use closure of E under MAT and DEC, and the base cases for Claim (3) use closure of E under $\text{BOT}_\neg$ and $\text{BOT}_\neq$. Next we show the claims for $\sigma = \tau\mu$.

1. By contradiction. Suppose $s \parallel_\sigma t$ and $s \sharp t$. Since E is closed under FE there exist $n \geq 1$ terms $u_1, \ldots, u_n$ such that $su_1 \ldots u_n \sharp tu_1 \ldots u_n$. By inductive hypothesis (3) we have $u_i \parallel u_i$ for $i = 1, \ldots, n$. Hence $su_1 \ldots u_n \parallel tu_1 \ldots u_n$ since $s \parallel_\sigma t$. This contradicts inductive hypothesis (1) since $su_1 \ldots u_n \sharp tu_1 \ldots u_n$.

2. Suppose $xs_1 \ldots s_n \nparallel_\sigma xt_1 \ldots t_n$. Then there exist terms $u \parallel_\tau v$ such that $xs_1 \ldots s_n u \nparallel_\mu xt_1 \ldots t_n v$. By inductive hypotheses (2) and (1) we have $s_i \sharp t_i$ for some $i \in \{1, \ldots, n\}$.

3. Suppose $s \nparallel_\sigma s$. By Lemma 5.10 we have $s = xs_1 \ldots s_n$. By Claim (2), which we have already established for σ, we have $s_i \sharp s_i$ for some $i \in \{1, \ldots, n\}$. Contradiction since E is closed under $\text{BOT}_\neq$. ∎

Lemma 5.12 (Common Value) Let $T \subseteq \Lambda_\sigma$. Then T is compatible if and only if there exists a value a such that $T \rhd_\sigma a$.

Proof By induction on σ.

$\sigma = o$, $\Rightarrow$. By contraposition. Suppose $T \not\rhd 0$ and $T \not\rhd 1$. Then there are terms $s, t \in T$ such that $s, \neg t \in E$. Thus $s \nparallel t$. Hence T is not compatible.

$\sigma = o$, $\Leftarrow$. By contraposition. Suppose $s \nparallel_o t$ for $s, t \in T$. Then $s, \neg t \in E$ without loss of generality. Hence $s \not\rhd 0$ and $t \not\rhd 1$. Thus $T \not\rhd 0$ and $T \not\rhd 1$.

$\sigma = \iota$, $\Rightarrow$. Let T be compatible. Then there exists a discriminant D that contains all the discriminating terms in T. Thus $T \rhd D$.

$\sigma = \iota$, $\Leftarrow$. By contradiction. Suppose $T \rhd D$ and T is not compatible. Then there are terms $s, t \in T$ such that $s \sharp t$. Thus s and t cannot be both in D. This contradicts $s, t \in T \rhd D$.

$\sigma = \tau\mu$, $\Rightarrow$. Let T be compatible. We define $T_a := \{ ts \mid t \in T,\ s \rhd_\tau a \}$ for every value $a \in \mathcal{I}\tau$ and show that T_a is compatible. Let $t_1, t_2 \in T$ and $s_1, s_2 \rhd_\tau a$. It suffices to show $t_1 s_1 \parallel t_2 s_2$. By the inductive hypothesis $s_1 \parallel_\tau s_2$. Since T is compatible, $t_1 \parallel t_2$. Hence $t_1 s_1 \parallel t_2 s_2$.

By the inductive hypothesis we now know that for every $a \in \mathcal{I}\tau$ there is a $b \in \mathcal{I}\mu$ such that $T_a \rhd_\mu b$. Hence there is a function $f \in \mathcal{I}\sigma$ such that $T_a \rhd_\mu fa$. Thus $T \rhd_\sigma f$.

$\sigma = \tau\mu$, $\Leftarrow$. Let $T \rhd_\sigma f$ and $s, t \in T$. We show $s \parallel_\sigma t$. Let $u \parallel_\tau v$. It suffices to show $su \parallel_\mu tv$. By the inductive hypothesis $u, v \rhd_\tau a$ for some value a. Hence $su, tv \rhd_\mu fa$. Thus $su \parallel_\mu tv$ by the inductive hypothesis. ∎

By Lemmas 5.11 and 5.12 we have a possible value for every parameter x (since $x \parallel x$). Hence we can define $\mathcal{I}x$ such that $x \rhd \mathcal{I}x$ for all parameters x. This completes the definition of the interpretation $\mathcal{I}$.

Lemma 5.13 (Admissibility) For all basic terms s: $s \rhd \hat{\mathcal{I}}s$.

Proof By induction on s. Let s be a basic term.

If $s = x$ is a parameter, we haven chosen $\mathcal{I}x$ such that $x \rhd \mathcal{I}x$.

If $s = tu$ is an application, we have $t \rhd \hat{\mathcal{I}}t$ and $u \rhd \hat{\mathcal{I}}u$ by the inductive hypothesis. Hence $tu \rhd (\hat{\mathcal{I}}t)(\hat{\mathcal{I}}u) = \hat{\mathcal{I}}(tu)$.

Let $s = (\wedge)$. Assume $t_1 \rhd_o a$ and $t_2 \rhd_o b$. We show $t_1 \wedge t_2 \rhd \mathcal{I}(\wedge)ab$ by contradiction. Suppose $t_1 \wedge t_2 \not\rhd \mathcal{I}(\wedge)ab$. Case analysis.

1. $a = b = 1$. Then $\neg t_1, \neg t_2 \notin E$ and $\neg(t_1 \wedge t_2) \in E$. Contradiction since E is closed under $\text{AND}_\neg$.
2. $a = 0$ or $b = 0$. Then $t_1 \notin E$ or $t_2 \notin E$, and $(t_1 \wedge t_2) \in E$. Contradiction since E is closed under AND.

Let $s = (=_\iota)$. Assume $t_1 \rhd_\iota D_1$ and $t_2 \rhd_\iota D_2$. We show $(t_1 = t_2) \rhd_o \mathcal{I}(=_\iota)D_1D_2$ by contradiction. Suppose $(t_1 = t_2) \not\rhd_o \mathcal{I}(=_\iota)D_1D_2$. Case analysis.

1. $D_1 = D_2$. Then $(t_1 \neq t_2) \in E$. Hence t_1, t_2 are discriminating and thus $t_1 \in D_1$ and $t_2 \in D_2 = D_1$. Contradiction by the definition of discriminants.
2. $D_1 \neq D_2$. Then $(t_1 = t_2) \in E$. Contradiction by Proposition 5.7 (2).

The case $s = \neg$ follows with the closure of E under DN. The case $s = \bot$ is straightforward. ∎

Lemma 5.14 For all formulas $s \in E$: $\hat{\mathcal{I}}s = 1$.

Proof Let $s \in E$. Then s is either basic or a disequation between basic terms.

Suppose s is basic. By Lemma 5.13, $s \rhd \hat{\mathcal{I}}s$. On the other hand, $s \not\rhd 0$ since $s \in E$. Hence $\hat{\mathcal{I}}s = 1$.

Suppose $s = (s_1 \neq s_2)$ where s_1 and s_2 are basic. Then $s_1 \sharp s_2$. Assume $\hat{\mathcal{I}}s_1 = \hat{\mathcal{I}}s_2$. Then $s_1, s_2 \rhd \hat{\mathcal{I}}s_1$ by Lemma 5.13 and hence $s_1 \parallel s_2$ by Lemma 5.12. Contradiction by Lemma 5.11 (1) since $s_1 \sharp s_2$. ∎

This completes the proof of Theorem 5.1.

6 Extensions

A Henkin-complete cut-free tableau system $\mathcal{T}$ for extensional type theory can be obtained as the dual of Brown's one-sided sequent system [5]. Our system $\mathcal{B}$ is

in fact a subsystem of this system. $\mathcal{B}$ contains all the distinctive rules of $\mathcal{T}$ (MAT, DEC, CON). In the following, we consider the additional rules of $\mathcal{T}$ and discuss their impact on termination.

General Equality

$\mathcal{B}$ restricts equality to the base type ι. If we admit all identities in basic terms, two additional rules are needed:

$$\text{EQB}\ \frac{s =_o t}{s, t \mid \neg s, \neg t} \qquad \text{EQF}\ \frac{s =_{\sigma\tau} t}{su = tu}$$

Rule EQF destroys termination. However, we can preserve termination if we restrict EQF such that u must be a term that already occurs as a subterm. It is open whether the resulting system is complete.

Lambda Abstraction

$\mathcal{B}$ disallows lambda abstractions. If we admit lambda abstractions in basic terms, an additional rule incorporating β-reduction is needed:

$$\text{BETA}\ \frac{s}{t}\quad t \text{ is obtainable from } s \text{ by } \beta\text{-reduction}$$

Example 6.1 $\mathcal{B}$ extended with BETA can prove the η-law:

$(\lambda x.fx) \not\approx f$	initial formula
$(\lambda x.fx)a \not\approx fa$	FE
$fa \not\approx fa$	BETA
$\bot$	$\text{BOT}_{\not\approx}$

□

Example 6.2 $\mathcal{B}$ extended with BETA does not terminate:

$p(\lambda x.p(fx)),\ \neg p(fa)$	initial formulas	$x, a : \sigma,\ f : \sigma\sigma o,\ p : (\sigma o)o$
$(\lambda x.p(fx)) \not\approx fa$	MAT	
$(\lambda x.p(fx))b \not\approx fab$	FE	
$p(fb) \not\approx fab$	BETA	
$\neg p(fb),\ fab$	BE	
$(\lambda x.p(fx)) \not\approx fb$	MAT	
...		

Note that σ can be any type. The problem are the new parameters introduced by FE and the disequations introduced by MAT. There seems to be no easy fix. For $\sigma = \iota$ the initial formulas do have a finite model: $\mathcal{I}\iota = \mathcal{I}o,\ \mathcal{I}f = \lambda xy.y$, $\mathcal{I}p(\lambda x.x) = 0,\ \mathcal{I}p(\lambda x.0) = 1$. □

Quantifiers

The extension of $\mathcal{B}$ with general equality and lambda abstraction can express quantification and thus already covers full simple type theory. If desired, quantification can be directly accounted for by additional logical constants. For universal quantification, we may have the constants $\forall_\sigma : (\sigma o)o$ and the rules

$$\text{ALL}\ \frac{\forall_\sigma s}{st}\ t:\sigma \qquad\qquad \text{ALL}_\neg\ \frac{\neg\forall_\sigma s}{\neg sx}\ x:\sigma \text{ fresh}$$

7 Conclusion

We have presented a terminating tableau system that decides satisfiability of basic formulas with respect to standard models. This contributes a new decidability and completeness result for higher-order logic with standard semantics. Our model existence proof relies on the possible-values technique, which for the first time is used to construct standard models. We are confident that our results can be extended. On the one side, the addition of equations at functional types may preserve decidability. On the other side, the addition of first-order quantifiers may preserve completeness.

Besides theoretical curiosity, there is a practical interest behind our research. We feel that the decision technique presented in this paper will lead to stronger auto tactics for interactive theorem provers. Even with a naive implementation, our decision technique can decide many problems that are out of the reach of current systems. As is, decomposition and confrontation may cause combinatorial explosion. An idea for further research is the integration of congruence closure techniques [11], which could efficiently replace most applications of the branching confrontation rule.

References

[1] Peter B. Andrews. Classical type theory. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 15, pages 965–1007. Elsevier Science, 2001.

[2] Peter B. Andrews and Chad E. Brown. TPS: A hybrid automatic-interactive system for developing proofs. *Journal of Applied Logic*, 4(4):367–395, 2006.

[3] Christoph Benzmüller. Extensional higher-order paramodulation and RUE-resolution. In *Proceedings of the 16th International Conference on Automated Deduction*, volume 1632 of *LNAI*, pages 399–413. Springer-Verlag, 1999.

[4] Christoph Benzmüller, Frank Theiss, Larry Paulson, and Arnaud Fietzke. LEO-II — A cooperative automatic theorem prover for higher-order logic. In *Fourth International Joint Conference on Automated Reasoning (IJCAR'08)*, volume 5195 of *LNAI*. Springer, 2008.

[5] Chad E. Brown. *Automated Reasoning in Higher-Order Logic: Set Comprehension and Extensionality in Church's Type Theory*. College Publications, 2007.

[6] William M. Farmer. The seven virtues of simple type theory. *J. Appl. Log.*, 6(3):267–286, 2008.

[7] Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211, 1979.

[8] Harvey Friedman. Equality between functionals. In R. Parikh, editor, *Proc. Logic Colloquium 1972-73*, volume 453 of *Lectures Notes in Mathematics*, pages 22–37. Springer, 1975.

[9] Michael Kohlhase. Higher-order tableaux. In Peter Baumgartner, Reiner Hähnle, and Joachim Posegga, editors, *TABLEAUX*, volume 918 of *LNCS*, pages 294–309. Springer, 1995.

[10] Dexter Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[11] Greg Nelson and Derek C. Oppen. Fast decision procedures based on congruence closure. *J. ACM*, 27(2):356–364, 1980.

[12] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[13] Dag Prawitz. Hauptsatz for higher order logic. *J. Symb. Log.*, 33:452–457, 1968.

[14] Moto-o Takahashi. A proof of cut-elimination theorem in simple type theory. *Journal of the Mathematical Society of Japan*, 19:399–410, 1967.