Master's thesis - second talk:

Tableaux for Higher-Order Logic with If-Then-Else, Description and Choice

by Julian Backes on August 28, 2009

Advisor: Dr. Chad E. Brown

Supervisor: Prof. Dr. Gert Smolka

Contents

Recap from the first talk

Simply typed higher order logic and tableaux

manufacturer.com

Basics: Syntax/Semantics

- Context: Simply typed higher order logic
- Syntax:
  - Types (σ, τ, μ): $\tau ::= \iota \mid o \mid \tau\ \tau$
  - Terms (s, t, u, v): $t ::= x \mid c \mid tt \mid \lambda x.t$
  - Logical constants: $\neg$, $\wedge$, $\vee$, $\forall_\tau$, $\exists_\tau$, $=_\tau$, $\longrightarrow$, $\top$, $\bot$
- Typed terms as usual, we only consider well-typed terms
- Semantics:
  - o boolean sort, containing 1/true/top/$\top$ and 0/false/bottom/$\bot$
  - ι non-empty set of individuals
  - τ set of all total functions (standard interpretation) or subset of all total total functions (Henkin/non standard interpretation)

Basics: Tableau systems

- General idea: Proof by contradiction
- Instead of proving the validity of a formula, we show that the negation of the formula is unsatisfiable / refutable / yields $\bot$
- Tableau rules:

$$\frac{A}{A_1 \;\mid\; \dots \;\mid\; A_n}\; A \subsetneq A_i \qquad\qquad \textsc{Closed}\;\frac{A}{\bot}$$

- For simplicity: We only write what is needed in A to apply a rule and what is added in the A_i

Three fragments

- Three fragments of simply typed higher order logic by Brown and Smolka:
    - "Basic": No higher-order equations, no quantifiers, no λ; proof system is terminating and complete wrt standard models
    - "EFO": No higher-order equations, quantifiers at base types, supports λ; proof system not terminating but cut-free and complete wrt standard models
    - "Full": Higher-order equations, supports λ; proof system not terminating but cut-free and complete wrt non-standard models
- Goal of my thesis: Extend these fragments with more powerful logical constants while maintaining the existing properties (completeness, cut-freeness, termination)

First talk

- I already presented tablau rules for If-Then-Else which can be added to all three fragments while preserving all of their properties; proof straight forward Slides: http://www.ps.uni-sb.de/~julian/master

- Difference between Choice/Description and other well-known logical constants: There are several possible interpretations

- I presented rules for Choice based on a paper by Mints. Let C s and C s' occur as subterms on the branch:

$$\textsc{Choice}\ \frac{}{\neg(st)\ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a)\ \mid\ \neg(sa), s'a\ \mid\ Cs = Cs'}\ a\ \text{fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg\alpha t_1 \ldots t_n}{s_1 \neq t1\ \mid\ \ldots\ \mid\ s_n \neq t_n}\ \alpha\ \text{variable or some C s}$$

Choice

Were the rules ok?

ggregersen.eu

Problems with Mints' rules

$$\textsc{Choice}\ \frac{}{\neg(st) \ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a) \ \mid\ \neg(sa), s'a \ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg\alpha t_1 \ldots t_n}{s_1 \neq t1 \ \mid\ \ldots \ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

- There were four problems with these rules:

Problems with Mints' rules

$$\textsc{Choice}\ \frac{}{\neg(st)\ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a)\ \mid\ \neg(sa), s'a\ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg \alpha t_1 \ldots t_n}{s_1 \neq t1\ \mid\ \ldots\ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

- There were four problems with these rules:
    - Something was missing: Not all unsatisfiable branches could be refuted

Problems with Mints' rules

$$\textsc{Choice}\ \frac{}{\neg(st)\ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a)\ \mid\ \neg(sa), s'a\ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg\alpha t_1 \ldots t_n}{s_1 \neq t1\ \mid\ \ldots\ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

- There were four problems with these rules:
  - Something was missing: Not all unsatisfiable branches could be refuted
  - $\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches

Problems with Mints' rules

$$\textsc{Choice}\ \frac{}{\neg(st)\ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a)\ \mid\ \neg(sa), s'a\ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg\alpha t_1 \ldots t_n}{s_1 \neq t1\ \mid\ \ldots\ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

- There were four problems with these rules:
  - Something was missing: Not all unsatisfiable branches could be refuted
  - Choice$_{\text{Ext}}$ causes an exponential blow-up in the number of branches
  - Is C s at the "head" in MAT' not too restrictive?

Problems with Mints' rules

$$\textsc{Choice}\ \frac{}{\neg(st)\ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a)\ \mid\ \neg(sa), s'a\ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg \alpha t_1 \ldots t_n}{s_1 \neq t1\ \mid\ \ldots\ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

- There were four problems with these rules:
  - Something was missing: Not all unsatisfiable branches could be refuted
  - $\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches
  - Is C s at the "head" in MAT' not too restrictive?
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?

Unrefutable, unsatisfiable branch

- Consider the following unsatisfiable branch/set:
    - $\{s =_{\iota o} t, C\ s\ !=_{\iota}\ C\ t\}$

Unrefutable, unsatisfiable branch

- Consider the following unsatisfiable branch/set:
  - $\{s =_{\iota o} t, C\ s\ !=_{\iota}\ C\ t\}$
- Using $\text{Choice}_{\text{Ext}}$ and Confrontation yields
  - $\{s =_{\iota o} t, C\ s \neq_{\iota} C\ t, C\ s =_{\iota} C\ t, C\ s \neq_{\iota} C\ s, C\ t \neq_{\iota} C\ t\}$
  - this branch cannot be refuted

Unrefutable, unsatisfiable branch

- Consider the following unsatisfiable branch/set:
  - {s $=_{\iota o}$ t, C s !$=_\iota$ C t}
- Using Choice$_{\text{Ext}}$ and Confrontation yields
  - {s $=_{\iota o}$ t, C s $\neq_\iota$ C t, C s $=_\iota$ C t, C s $\neq_\iota$ C s, C t $\neq_\iota$ C t}
  - this branch cannot be refuted
- We need "MAT' at type ι": DEC'

$$\text{DEC'}\ \frac{\alpha s_1 \ldots s_n \neq_\iota \alpha t_1 \ldots t_n}{s_1 \neq t_1 \;\mid\; \ldots \;\mid\; s_n \neq t_n}\ \alpha \text{ variable or some C s}$$

Unrefutable, unsatisfiable branch ctd

- $\{s =_{\iota o} t, C\ s \neq_\iota C\ t, C\ s =_\iota C\ t, C\ s \neq_\iota C\ s, C\ t \neq_\iota C\ t\}$

Unrefutable, unsatisfiable branch ctd

- $\{s =_{\iota o} t, C\ s \neq_\iota C\ t, C\ s =_\iota C\ t, C\ s \neq_\iota C\ s, C\ t \neq_\iota C\ t\}$
- One question remaining: Is C s not too restrictive?

Unrefutable, unsatisfiable branch ctd

- $\{s =_{\iota o} t, C\, s \neq_\iota C\, t, C\, s =_\iota C\, t, C\, s \neq_\iota C\, s, C\, t \neq_\iota C\, t\}$
- One question remaining: Is C s not too restrictive?
  - Answer: probably not :-) We don't know...

Unrefutable, unsatisfiable branch ctd

- $\{s =_{\iota o} t, C\ s \neq_{\iota} C\ t, C\ s =_{\iota} C\ t, C\ s \neq_{\iota} C\ s, C\ t \neq_{\iota} C\ t\}$
- One question remaining: Is C s not too restrictive?
    - Answer: probably not :-) We don't know...
- But remember: $\text{Choice}_{\text{Ext}}$ is not a nice rule (exponential blow-up)

Unrefutable, unsatisfiable branch ctd

- $\{s =_{\iota o} t, C\ s \neq_\iota C\ t, C\ s =_\iota C\ t, C\ s \neq_\iota C\ s, C\ t \neq_\iota C\ t\}$
- One question remaining: Is C s not too restrictive?
  - Answer: probably not :-) We don't know...
- But remember: $\text{Choice}_{\text{Ext}}$ is not a nice rule (exponential blow-up)
- We showed: Relaxing the C s restriction and just require C in MAT' and DEC'

$$\text{DEC'}\ \frac{\alpha s_1 \ldots s_n \neq_\iota \alpha t_1 \ldots t_n}{s_1 \neq t_1 \mid \ldots \mid s_n \neq t_n}\ \alpha \text{ variable or C} \qquad \text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg\alpha t_1 \ldots t_n}{s_1 \neq t_1 \mid \ldots \mid s_n \neq t_n}\ \alpha \text{ variable or C}$$

makes $\text{Choice}_{\text{Ext}}$ unnecessary

4 Problems, 3 Solutions

- There were four problems with these rules:
  - Something was missing: Not all unsatisfiable branches could be refuted
  - $\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches
  - Is C s at the "head" in MAT'/DEC' not too restrictive?
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?
- For the last problem, we need some definitions...

4 Problems, 3 Solutions

- There were four problems with these rules:

We need DEC'!

  - ~~Something was missing: Not all unsatisfiable branches could be refuted~~
  - $\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches
  - Is C s at the "head" in MAT'/DEC' not too restrictive?
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?

- For the last problem, we need some definitions...

4 Problems, 3 Solutions

- There were four problems with these rules:
  - We need DEC'!
    ~~Something was missing: Not all unsatisfiable branches could be refuted~~
  - We don't need it!
    ~~$\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches~~
  - Is C s at the "head" in MAT'/DEC' not too restrictive?
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?
- For the last problem, we need some definitions...

4 Problems, 3 Solutions

- There were four problems with these rules:
  - ~~Something was missing: Not all unsatisfiable branches could be refuted~~ We need DEC'!
  - ~~$\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches~~ We don't need it!
  - ~~Is C s at the "head" in MAT'/DEC' not too restrictive?~~ Now, it is!
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?
- For the last problem, we need some definitions...

4 Problems, 3 Solutions

- There were four problems with these rules:
  - ~~Something was missing: Not all unsatisfiable branches could be refuted~~ We need DEC'!
  - ~~$\text{Choice}_{\text{Ext}}$ causes an exponential blow-up in the number of branches~~ We don't need it!
  - ~~Is C s at the "head" in MAT'/DEC' not too restrictive?~~ Now, it is!
  - C s and C s' must occur as subterms on the branch. What about the subterm λx.C x?
- For the last problem, we need some definitions...

Accessibility and the subterm problem

- Def: Let E be a branch. A term s is **discriminating** in E if and only if there is a term t such that $(s \neq_\iota t) \in E$ or $(t \neq_\iota s) \in E$

Accessibility and the subterm problem

- Def: Let E be a branch. A term s is **discriminating** in E if and only if there is a term t such that $(s \neq_\iota t) \in E$ or $(t \neq_\iota s) \in E$

- Def: Let E be a branch. A term s is **accessible** in E if and only if there is a context $C = [] \; t_1 \ldots t_n$ such that

Accessibility and the subterm problem

- Def: Let E be a branch. A term s is **discriminating** in E if and only if there is a term t such that $(s \neq_\iota t) \in E$ or $(t \neq_\iota s) \in E$

- Def: Let E be a branch. A term s is **accessible** in E if and only if there is a context $C = [] \, t_1 \ldots t_n$ such that
  - $C[s]$ is discriminating in E for $C[s]$ of type ι or

Accessibility and the subterm problem

- Def: Let E be a branch. A term s is **discriminating** in E if and only if there is a term t such that $(s \neq_\iota t) \in E$ or $(t \neq_\iota s) \in E$
- Def: Let E be a branch. A term s is **accessible** in E if and only if there is a context $C = [] \; t_1 \ldots t_n$ such that
  - $C[s]$ is discriminating in E for $C[s]$ of type ι or
  - $C[s] \in E$ or $\neg C[s] \in E$ for $C[s]$ of type o

Accessibility and the subterm problem

- Def: Let E be a branch. A term s is **discriminating** in E if and only if there is a term t such that $(s \neq_\iota t) \in E$ or $(t \neq_\iota s) \in E$

- Def: Let E be a branch. A term s is **accessible** in E if and only if there is a context $C = [] \; t_1 \dots t_n$ such that
  - $C[s]$ is discriminating in E for $C[s]$ of type ι or
  - $C[s] \in E$ or $\neg C[s] \in E$ for $C[s]$ of type o

- Solution to our subterm problem: "is accessible in E" instead of "is a subterm in E" does the job

Comparison of the rules

- These were the rules we started with:

$$\textsc{Choice}\ \frac{}{\neg(st) \ \mid\ s(Cs)}\ \text{t term of suitable type}$$

$$\textsc{Choice}_{Ext}\ \frac{}{sa, \neg(s'a) \ \mid\ \neg(sa), s'a \ \mid\ Cs = Cs'}\ a \text{ fresh}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg \alpha t_1 \ldots t_n}{s_1 \neq t_1 \ \mid\ \ldots \ \mid\ s_n \neq t_n}\ \alpha \text{ variable or some } Cs$$

- And these are the new rules:

$$\textsc{Choice}\ \frac{}{\neg(st) \ \mid\ s(Cs)}\ Cs \text{ accessible}$$

$$\text{DEC'}\ \frac{\alpha s_1 \ldots s_n \neq_\iota \alpha t_1 \ldots t_n}{s_1 \neq t_1 \ \mid\ \ldots \ \mid\ s_n \neq t_n}\ \alpha \text{ variable or C}$$

$$\text{MAT'}\ \frac{\alpha s_1 \ldots s_n, \neg \alpha t_1 \ldots t_n}{s_1 \neq t_1 \ \mid\ \ldots \ \mid\ s_n \neq t_n}\ \alpha \text{ variable or C}$$

Completeness with Choice

A proof sketch

Photo: myspacegeek.net

Completeness proof

- I will not explain the whole completeness proof here
- The hard part reduces to the Model Existence Theorem:
  - Def: A set E of formulas (representing a branch) is called **evident** if it does not contain $\perp$ and is closed under the tablau rules
  - Model Existence Theorem: If a set E is evident, then there exists an interpretation which satifies all formulas in E
- The proof of this Theorem is long, I only present the general ideas here
- But first, we need one more definition....

Model existence theorem

Model existence theorem

- Given an evident set E, define **possible values relation** by induction on types:
  - s $\triangleright_o$ 0 :<=> [s] $\notin$ E
  - s $\triangleright_o$ 1 :<=> ¬[s] $\notin$ E
  - s $\triangleright_{\sigma\tau}$ f :<=> st $\triangleright_\tau$ fa whenever t $\triangleright_\sigma$ a
  - (We skip $\triangleright_\iota$ here, it is defined using discriminants)
- Fact: Any term has a possible value (proof uses MAT' and DEC')
- For the proof of the model existence theorem, we need to show that for any term s, s $\triangleright$ $\mathfrak{I}$ s (where $\mathfrak{I}$ is a corresponding interpretation); proof is done by induction on s
  - Question: For the case C $\triangleright$ $\mathfrak{I}$ C, what is $\mathfrak{I}$ C? We said that the interpretation of C is not unique so we have to give one...

Interpretation of C

- Goal: Prove C $\triangleright$ $\Im$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\Im$ C but....
- Problem 2: $\Im$ C is not a choice function: what if the set is empty?
- This interpretation does the job!
- The proof that $\Im$ C is a choice function requires the Choice rule
  - And we need to know that any value is a possible value for some term
    => we need non-standard models for Choice at functional types

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is

- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....
- Problem 2: $\mathfrak{I}$ C is not a choice function: what if the set is empty?
- This interpretation does the job!
- The proof that $\mathfrak{I}$ C is a choice function requires the Choice rule
  - And we need to know that any value is a possible value for some term
    => we need non-standard models for Choice at functional types

Interpretation of C

- We define $\Im$ C to be a function such that ($\Im$ C) f is
  - some a if {C s | s $\triangleright$ f} $\triangleright$ a

- Goal: Prove C $\triangleright$ $\Im$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\Im$ C but....
- Problem 2: $\Im$ C is not a choice function: what if the set is empty?
- This interpretation does the job!
- The proof that $\Im$ C is a choice function requires the Choice rule
  - And we need to know that any value is a possible value for some term
    => we need non-standard models for Choice at functional types

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a   if {C s | s $\triangleright$ f} $\triangleright$ a

- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
    - some a if {C s | C s is accessible and s ▹ f} ▹ a

- Goal: Prove C ▹ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a   if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a

- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a   if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a

- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....
- Problem 2: $\mathfrak{I}$ C is not a choice function: what if the set is empty?

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a
  - some b if f b = 1

- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....
- Problem 2: $\mathfrak{I}$ C is not a choice function: what if the set is empty?

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a
  - some b if f b = 1
  - some c otherwise
- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....
- Problem 2: $\mathfrak{I}$ C is not a choice function: what if the set is empty?

Interpretation of C

- We define $\Im$ C to be a function such that ($\Im$ C) f is
  - some a if {C s | C s is accessible and s ▹ f} ▹ a
  - some b if f b = 1
  - some c otherwise
- Goal: Prove C ▹ $\Im$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C ▹ $\Im$ C but....
- Problem 2: $\Im$ C is not a choice function: what if the set is empty?
- This interpretation does the job!

Interpretation of C

- We define $\Im$ C to be a function such that ($\Im$ C) f is
    - some a   if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a
    - some b   if f b = 1
    - some c   otherwise
- Goal: Prove C $\triangleright$ $\Im$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\Im$ C but....
- Problem 2: $\Im$ C is not a choice function: what if the set is empty?
- This interpretation does the job!
- The proof that $\Im$ C is a choice function requires the Choice rule

Interpretation of C

- We define $\mathfrak{I}$ C to be a function such that ($\mathfrak{I}$ C) f is
  - some a if {C s | C s is accessible and s $\triangleright$ f} $\triangleright$ a
  - some b if f b = 1
  - some c otherwise
- Goal: Prove C $\triangleright$ $\mathfrak{I}$ C
- Problem 1: This is not enough, the proof requires accessible terms
- Now, we can prove C $\triangleright$ $\mathfrak{I}$ C but....
- Problem 2: $\mathfrak{I}$ C is not a choice function: what if the set is empty?
- This interpretation does the job!
- The proof that $\mathfrak{I}$ C is a choice function requires the Choice rule
  - And we need to know that any value is a possible value for some term
    => we need non-standard models for Choice at functional types

Results/Recap

- We have proven completeness
  - for "EFO" together with Choice at base types wrt standard models
  - for "Full" together with Choice at any type wrt non-standard models

Results/Recap

- We have proven completeness
  - for "EFO" together with Choice at base types wrt standard models
  - for "Full" together with Choice at any type wrt non-standard models
- We have not proven completeness for "Basic" with Choice at type ι:
  - The problem: Preserve termination
  - Possible solution: Only instantiate discriminating terms

Results/Recap

- We have proven completeness
  - for "EFO" together with Choice at base types wrt standard models
  - for "Full" together with Choice at any type wrt non-standard models
- We have not proven completeness for "Basic" with Choice at type ι:
  - The problem: Preserve termination
  - Possible solution: Only instantiate discriminating terms
- What about description?

Description

- Given the tableau rule for Choice, it is easy to give one for Description:

$$\textsc{Desc}\ \frac{}{\neg(st)\ \mid\ a \neq b,\ sa,\ sb\ \mid\ s(Ds)}\ Ds \text{ accessible; } a,\ b \text{ fresh}$$

Description

- Given the tableau rule for Choice, it is easy to give one for Description:

$$\textsc{Desc}\ \frac{}{\neg(st)\ \mid\ a \neq b,\ sa,\ sb\ \mid\ s(Ds)}\ Ds \text{ accessible; } a,\, b \text{ fresh}$$

- Using this rule, the completeness proof works analogously to the completeness proof with Choice
  - This includes the non-standardness at functional types

Description

- Given the tableau rule for Choice, it is easy to give one for Description:

$$\textsc{Desc}\ \frac{}{\neg(st) \ \mid\ a \neq b,\ sa,\ sb \ \mid\ s(Ds)}\ Ds \text{ accessible; } a,\, b \text{ fresh}$$

- Using this rule, the completeness proof works analogously to the completeness proof with Choice
  - This includes the non-standardness at functional types
- Chad suspects that it is possible to add Description at any type to EFO while maintaining completeness

Description

- Given the tableau rule for Choice, it is easy to give one for Description:

$$\textsc{Desc}\ \frac{}{\neg(st) \ \mid\ a \neq b,\ sa,\ sb \ \mid\ s(Ds)}\ Ds \text{ accessible; } a, b \text{ fresh}$$

- Using this rule, the completeness proof works analogously to the completeness proof with Choice
  - This includes the non-standardness at functional types
- Chad suspects that it is possible to add Description at any type to EFO while maintaining completeness
- Actually, we don't know...

Bonus: Independence results

- In the first talk, at least Prof. Smolka wanted to know:
  - Is If-Then-Else really more powerful, i.e. is it not possible to express If-Then-Else using other logical constants?

Bonus: Independence results

- In the first talk, at least Prof. Smolka wanted to know:
  - Is If-Then-Else really more powerful, i.e. is it not possible to express If-Then-Else using other logical constants?
- Result: If-Then-Else *is* more powerful

Bonus: Independence results

- In the first talk, at least Prof. Smolka wanted to know:
    - Is If-Then-Else really more powerful, i.e. is it not possible to express If-Then-Else using other logical constants?
- Result: If-Then-Else *is* more powerful
- Proof works by constructing a (non-standard) model which contains all well-known logical constants but not If-Then-Else
    - Construction uses a binary logical relation: Equality

Bonus: Independence results

- In the first talk, at least Prof. Smolka wanted to know:
  - Is If-Then-Else really more powerful, i.e. is it not possible to express If-Then-Else using other logical constants?
- Result: If-Then-Else *is* more powerful
- Proof works by constructing a (non-standard) model which contains all well-known logical constants but not If-Then-Else
  - Construction uses a binary logical relation: Equality
- We also constructed a model which containts all well-known logical constants, including If-Then-Else, but not Description
  - Construction uses an infinitary logical relation

Thank you!

And stay tuned for Sigurd's talk...

References

- Chad E. Brown, G. Smolka: "Terminating Tableaux for the Basic Fragment of Simple Type Theory". TABLEAUX 2009:138-151, Springer LNCS 5607.
- Chad E. Brown, G. Smolka: "Extended First-Order Logic". TPHOLs 2009, Springer LNCS 5674.
- Chad E. Brown, G. Smolka: "Complete Cut-Free Tableaux for Equational Simple Type Theory". 2009.
- G. Mints: Cut-Elimination for Simple Type Theory with an Axiom of Choice. The Journal of Symbolic Logic, Vol. 64, No. 2., pp. 479-485. 1999.
- G. Takeuti: "Proof Theory (Studies in Logic and the Foundations of Mathematics)". Elsevier Science Ltd, 2 Rev Sub edition. 1987.