Computational Logic
320441 — Fall 2011
Fall 2011

Michael Kohlhase & Florian Rabe
School of Engineering & Science
Jacobs University, Bremen Germany
⟨first-initial⟩.⟨lastname⟩@jacobs-university.de
office: 103/168, East Wing, Research 1, phone: x3140/51

SOME RIGHTS RESERVED
©: Michael Kohlhase

1 Preface

This document contains the course notes for the graduate course "Computational Logic" held at Jacobs University, Bremen in the Fall Semsesters 2005, 2007, 2009, and 2011. The document mixes the slides presented in class with comments of the instructor to give students a more complete background reference.

This Document

1.1 This Document

Contents: The document mixes the slides presented in class with comments of the instructor to give students a more complete background reference.

Caveat: This document is made available for the students of this course only. It is still an early draft, and will develop over the course of the course. It will be developed further in coming academic years.

Licensing: This document is licensed under a Creative Commons license that requires attribution, forbids commercial use, and allows derivative works as long as these are licensed under the same license.

Knowledge Representation Experiment: This document is also an experiment in knowledge representation. Under the hood, it uses the sTEX package [Koh08, Koh10], a TEX/LATEX extension for semantic markup, which allows to export the contents into the eLearning platform PantaRhei.

Comments and extensions are always welcome, please send them to the author.

Comments: Comments and extensions are always welcome, please send them to the author.

Acknowledgments: The following students have submitted corrections and suggestions to this and earlier versions of the notes: Rares Ambrus, Florian Rabe, Deyan Ginev, Fulya Horozal.

Contents

1 Preface **2**
1.1 This Document . . . 2

2 Outline of the Course **4**

3 Administrativa **5**

4 A History of Ideas in Logic **9**

5 First-Order Logic **10**
5.1 First-Order Logic: Syntax and Semantics . . . 10
5.2 First-Order Substitutions . . . 13
5.3 Alpha-Renaming for First-Order Logic . . . 16

6 Inference in First-Order Logic **17**
6.1 Recap: General Properties of Logics and Calculi . . . 17
6.2 First-Order Calculi . . . 20
6.3 Abstract Consistency and Model Existence . . . 24
6.4 A Completeness Proof for First-Order ND . . . 32
6.5 Limits of First-Order Logic . . . 34

7 First-Order Inference with Tableaux **34**
7.1 First-Order Tableaux . . . 34
7.2 Free Variable Tableaux . . . 37
7.3 First-Order Unification . . . 38
7.4 Soundness and Completeness of First-Order Tableaux . . . 41

8 Higher-Order Logic and λ-Calculus **44**
8.1 Higher-Order Predicate Logic . . . 44
8.2 Simply Typed λ-Calculus . . . 51
8.3 Computational Properties of λ-Calculus . . . 53
8.3.1 Termination of β-reduction . . . 53
8.3.2 Confluence of $\beta\eta$ Conversion . . . 58
8.4 The Semantics of the Simply Typed λ-Calculus . . . 59
8.4.1 Soundness of the Simply Typed λ-Calculus . . . 59
8.4.2 Completeness of -Equality . . . 61
8.5 Simple Type Theory (Higher-Order Logic based on the Simply Typed λ-Calculus) 64
8.6 The Curry-Howard Isomorphism . . . 66
8.7 Simply Typed λ-Calculus as an Inference System . . . 71

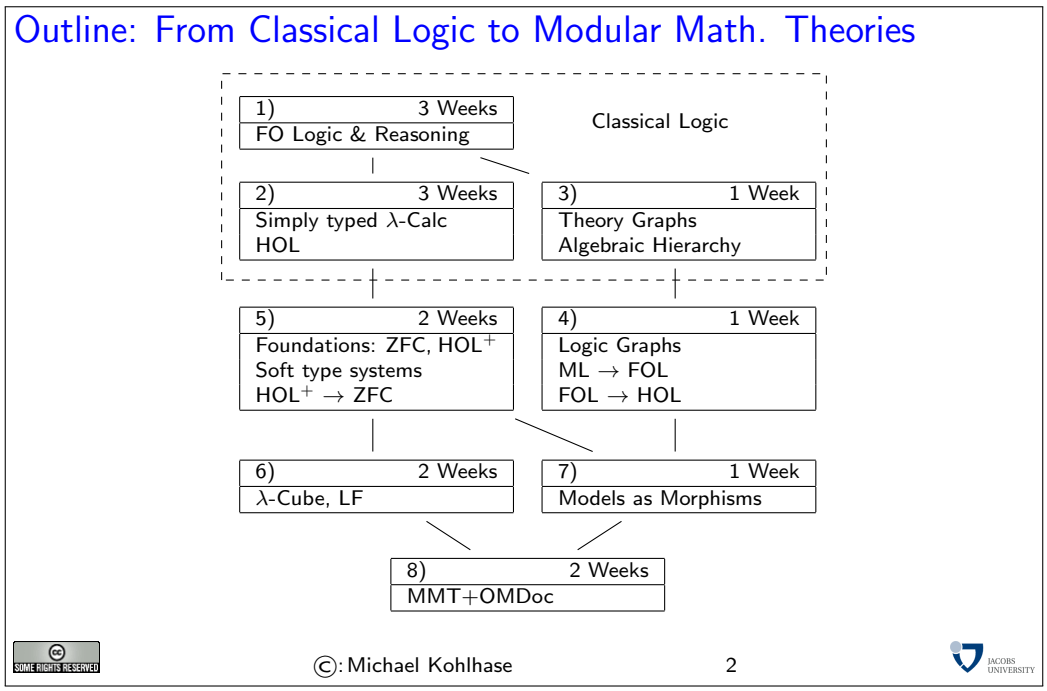
2 Outline of the Course

In this course, we want to achieve three things: we want to

1. expose you to various logics from a computational perspective, in particular
2. teach you how to build up logics and and express domain theories modularly, and
3. apply that to the foundations of mathematics.

Our motivation for that is mainly that we want to prepare you for research in the KWARC group in the process.

The organization of the course can best be seen in a diagram, where the boxes represent course modules and the lines the dependency relations between them.



The first part of the course (in the dashed box in the diagram) is a relatively classical, but very thorough introduction to mathematical logic. Starting with first-order logic, some of its calculi, and their properties. As first-order logic is not expressive enough to formalize important parts of mathematics, we extend first-order logic to higher-order logic which adds the critical ability to quantify over sets and functions. Most of the technical development of higher-order logics can be factored out into the simply typed λ-calculus, which we will use to introduce more modern, judgement-based treatments of logics. In a second thread in the course, we develop a modular regime of developing domain theories, where modularity is induced by theory morphisms.

In the remainder of the course, we develop these to threads further, in the fourth module, we extend the theory graphs by representations of logics (as logic graphs). Then we extend the discussion of first- and higher-order logic into a development of foundational systems for mathematics, culminating in Zermelo-Fraenkel Set theory, which later allows us to extend the theory/logic graphs by models (interpreted as special theory morphisms). Together with the development of logical frameworks in module six, we re-unite the two threads of the course by presenting the MMT and OMDoc representation formats that combine all of the issues discussed in the course. This brings us to the front of research and prepares us for the seminar in the following semester.

3 Administrativa

We will now go through the ground rules for the course. This is a kind of a social contract between the instructor and the students. Both have to keep their side of the deal to make learning and becoming Computer Scientists as efficient and painless as possible.

The Course/Lab Combo

- ▷ The course is accompanied by a Lab, which
  - ▷ During the semester: deepens the understanding of the course topics by practical exercise
  - ▷ Over the break: let's you experiment with them in a little research project
- ▷ Course and Lab are 5 credits each (⚠ about one day of work each!)
- ▷ They are graded independently

SOME RIGHTS RESERVED ©: Michael Kohlhase 3 JACOBS UNIVERSITY

Textbooks, Handouts and Information, Forum

- ▷ No required textbook, but course notes, posted slides
- ▷ Course notes will be posted at `http://kwarc.info/teaching/CompLog.html`
- ▷ Everything will be posted on PantaRhei (Notes+assignments+course forum)
  - ▷ announcements, contact information, course schedule and calendar
  - ▷ discussion among your fellow students (careful, we check for academic integrity!)
  - ▷ `http://panta.kwarc.info` (use your precourse login and pwd)
  - ▷ if there are problems send e-mail to `f.horozal@jacobs-university.de`

SOME RIGHTS RESERVED ©: Michael Kohlhase 4 JACOBS UNIVERSITY

No Textbook: Due to the special circumstances discussed above, there is no single textbook that covers the course. Instead we have a comprehensive set of course notes (this document). They are provided in two forms: as a large PDF that is posted at the course web page and on the PantaRhei system. The latter is actually the preferred method of interaction with the course materials, since it allows to discuss the material in place, to play with notations, to give feedback, etc. The PDF file is for printing and as a fallback, if the PantaRhei system, which is still under development, develops problems.

Now we come to a topic that is always interesting to the students: the grading scheme.

Prerequisites, Requirements, Grades

- ▷ Prerequisites: Motivation, Interest, Curiosity, hard work (mainly,...)
  - ▷ exposure to discrete Math, possibly category theory
  - ▷ experience in (some) logics

  You can do this course if you want! (even without those, but they help)
- ▷ Grades: (plan your work involvement carefully)

Course		Lab	
Attendance and Wakefulness	20%	Attendance and Wakefulness	10%
Graded Theory Assignments	60%	Graded Lab Assignments	50%
		Intersession Project	20%
Discussion	20%	Discussion	20%

In particular, no midterm, and no final, but attendance is mandatory! (excuses possible)

- ▷ Note that for the grades, the percentages of achieved points are added with the weights above, and only then the resulting percentage is converted to a grade.

SOME RIGHTS RESERVED ©: Michael Kohlhase 5 JACOBS UNIVERSITY

Our main motivation in this grading scheme is that I want to entice you to learn continuously. This means that you will have to stay involved, do all your homework assignments, and keep abreast with the course. This means that your continued involvement may be higher than other (graduate) courses, but you are free to concentrate on these during exam week.

Homework assignments

- ▷ Goal: Reinforce and apply what is taught in class.
- ▷ Homeworks: will be small individual problem/programming/proof assignments (but take time to solve) group submission if and only if explicitly permitted
- ▷ Admin: To keep things running smoothly
  - ▷ Homeworks will be posted on PantaRhei
  - ▷ Homeworks are handed in electronically in `grader` (plain text, Postscript, PDF,...)
  - ▷ go to the tutorials, discuss with your TA (they are there for you!)
  - ▷ materials: sometimes posted ahead of time; then read before class, prepare questions, bring printout to class to take notes
- ▷ Homework Discipline:
  - ▷ start early! (many assignments need more than one evening's work)
  - ▷ Don't start by sitting at a blank screen
  - ▷ Humans will be trying to understand the text/code/math when grading it.

SOME RIGHTS RESERVED ©: Michael Kohlhase 6 JACOBS UNIVERSITY

Homework assignments are a central part of the course, they allow you to review the concepts covered in class, and practice using them.

The next topic is very important, you should take this very seriously, even if you think that this is just a self-serving regulation made by the faculty.

All societies have their rules, written and unwritten ones, which serve as a social contract among its members, protect their interestes, and optimize the functioning of the society as a whole. This is also true for the community of scientists worldwide. This society is special, since it balances intense cooperation on joint issues with fierce competition. Most of the rules are largely unwritten; you are expected to follow them anyway. The code of academic integrity at Jacobs is an attempt to put some of the aspects into writing.

It is an essential part of your academic education that you learn to behave like academics, i.e. to function as a member of the academic community. Even if you do not want to become a scientist in the end, you should be aware that many of the people you are dealing with have gone through an academic education and expect that you (as a graduate of Jacobs) will behave by these rules.

The Code of Academic Integrity

- ▷ Jacobs has a "Code of Academic Integrity"
  - ▷ this is a document passed by the faculty (our law of the university)
  - ▷ you have signed it last week (we take this seriously)
- ▷ It mandates good behavior and penalizes bad from both faculty and students
  - ▷ honest academic behavior (we don't cheat)
  - ▷ respect and protect the intellectual property of others (no plagiarism)
  - ▷ treat all Jacobs members equally (no favoritism)
- ▷ this is to protect you and build an atmosphere of mutual respect
  - ▷ academic societies thrive on reputation and respect as primary currency
- ▷ The Reasonable Person Principle (one lubricant of academia)
  - ▷ we treat each other as reasonable persons
  - ▷ the other's requests and needs are reasonable until proven otherwise

SOME RIGHTS RESERVED ©: Michael Kohlhase 7 JACOBS UNIVERSITY

To understand the rules of academic societies it is central to realize that these communities are driven by economic considerations of their members. However, in academic societies, the primary good that is produced and consumed consists in ideas and knowledge, and the primary currency involved is academic reputation[1]. Even though academic societies may seem as altruistic — scientists share their knowledge freely, even investing time to help their peers understand the concepts more deeply — it is useful to realize that this behavior is just one half of an economic transaction. By publishing their ideas and results, scientists sell their goods for reputation. Of course, this can only work if ideas and facts are attributed to their original creators (who gain reputation by being cited). You will see that scientists can become quite fierce and downright nasty when confronted with behavior that does not respect other's intellectual property.

One special case of academic rules that affects students is the question of cheating, which we will cover next.

[1] Of course, this is a very simplistic attempt to explain academic societies, and there are many other factors at work there. For instance, it is possible to convert reputation into money: if you are a famous scientist, you may get a well-paying job at a good university,...

Cheating [adapted from CMU:15-211 (P. Lee, 2003)]

- ▷ There is no need to cheat in this course!! (hard work will do)
- ▷ cheating prevents you from learning (you are cutting your own flesh)
- ▷ if you are in trouble, come and talk to me (I am here to help you)
- ▷ We expect you to know what is useful collaboration and what is cheating
  - ▷ you will be required to hand in your own original code/text/math for all assignments
  - ▷ you may discuss your homework assignments with others, but if doing so impairs your ability to write truly original code/text/math, you will be cheating
  - ▷ copying from peers, books or the Internet is plagiarism unless properly attributed (even if you change most of the actual words)
  - ▷ more on this as the semester goes on ...
- ▷ ⚠ There are data mining tools that monitor the originality of text/code. ⚠

SOME RIGHTS RESERVED

We are fully aware that the border between cheating and useful and legitimate collaboration is difficult to find and will depend on the special case. Therefore it is very difficult to put this into firm rules. We expect you to develop a firm intuition about behavior with integrity over the course of stay at Jacobs.

Next we come to a special project that is going on in parallel to teaching the course. I am using the coures materials as a research object as well. This gives you an additional resource, but may affect the shape of the coures materials (which now server double purpose). Of course I can use all the help on the research project I can get.

Experiment: E-Learning with OMDoc/PantaRhei

- ▷ My research area: deep representation formats for (mathematical) knowledge
- ▷ Application: E-learning systems (represent knowledge to transport it)
- ▷ Experiment: Start with this course (Drink my own medicine)
  - ▷ Re-Represent the slide materials in OMDoc (Open Math Documents)
  - ▷ Feed it into the PantaRhei system (`http://trac.mathweb.org/planetary`)
  - ▷ Try it on you all (to get feedback from you)
- ▷ Tasks (Unfortunately, I cannot pay you for this; maybe later)
  - ▷ help me complete the material on the slides (what is missing/would help?)
  - ▷ I need to remember "what I say", examples on the board. (take notes)
- ▷ Benefits for you (so why should you help?)
  - ▷ you will be mentioned in the acknowledgements (for all that is worth)
  - ▷ you will help build better course materials (think of next-year's freshmen)

SOME RIGHTS RESERVED

4 A History of Ideas in Logic

History of Ideas (abbreviated): Propositional Logic

▷ General Logic ([ancient Greece, e.g. Aristotle])

+ conceptual separation of syntax and semantics
+ system of inference rules ("Syllogisms")
– no formal language, no formal semantics

▷ Propositional Logic [Boole ~ 1850]

+ functional structure of formal language (propositions + connectives)
+ mathematical semantics ($\rightsquigarrow$Boolean Algebra)
– abstraction from internal structure of propositions

SOME RIGHTS RESERVED ©: Michael Kohlhase 10 JACOBS UNIVERSITY

History of Ideas (continued): Predicate Logic

▷ Frege's "Begriffsschrift" [Fre79]

+ functional structure of formal language(terms, atomic formulae, connectives, quantifiers)
– weird graphical syntax, no mathematical semantics
– paradoxes e.g. Russell's Paradox [R. 1901] (the set of sets that do not contain themselves)

▷ modern form of predicate logic [Peano $\sim$ 1889]

+ modern notation for predicate logic ($\vee, \wedge, \Rightarrow, \forall, \exists$)

SOME RIGHTS RESERVED ©: Michael Kohlhase 11 JACOBS UNIVERSITY

History of Ideas (continued): First-Order Predicate Logic

▷ Types ([Russell 1908])

– restriction to well-types expression
+ paradoxes cannot be written in the system
+ Principia Mathematica ([Whitehead, Russell 1910])

▷ Identification of first-order Logic ([Skolem, Herbrand, Gödel $\sim$ 1920 – '30])

– quantification only over individual variables (cannot write down induction principle)
+ correct, complete calculi, semi-decidable
+ set-theoretic semantics ([Tarski 1936])

SOME RIGHTS RESERVED ©: Michael Kohlhase 12 JACOBS UNIVERSITY

History of Ideas (continued): Foundations of Mathematics

- ▷ Hilbert's Program: find logical system and calculus, ([Hilbert $\sim$ 1930])
  - ▷ that formalizes all of mathematics
  - ▷ that admits sound and complete calculi
  - ▷ whose consistence is provable in the system itself
- ▷ Hilbert's Program is impossible! ([Gödel 1931])

  Let $\mathcal{L}$ be a logical system that formalizes arithmetics $(\mathbb{N}, +, *)$,
  - ▷ then $\mathcal{L}$ is incomplete
  - ▷ then the consistence of $\mathcal{L}$ cannot be proven in $\mathcal{L}$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 13 JACOBS UNIVERSITY

History of Ideas (continued): λ-calculus, set theory

- ▷ Simply typed λ-calculus ([Church 1940])
  - + simplifies Russel's types, λ-operator for functions
  - + comprehension as β-equality (can be mechanized)
  - + simple type-driven semantics (standard semantics $\rightsquigarrow$ incompleteness)
- ▷ Axiomatic set theory
  - +– type-less representation (all objects are sets)
  - + first-order logic with axioms
  - + restricted set comprehension (no set of sets)
  - – functions and relations are derived objects

SOME RIGHTS RESERVED ©: Michael Kohlhase 14 JACOBS UNIVERSITY

5 First-Order Logic

5.1 First-Order Logic: Syntax and Semantics

First-Order Predicate Logic (PL^1)

- ▷ Coverage: We can talk about (*All humans are mortal*)
  - ▷ individual things and denote them by variables or constants
  - ▷ properties of individuals, (e.g. being human or mortal)
  - ▷ relations of individuals, (e.g. *sibling_of* relationship)
  - ▷ functions on individuals, (e.g. the *father_of* function)

  We can also state the existence of an individual with a certain property, or the universality of a property.
- ▷ But we cannot state assertions like
  - ▷ *There is a surjective function from the natural numbers into the reals.*
- ▷ First-Order Predicate Logic has many good properties (complete calculi, compactness, unitary, linear unification,...)
- ▷ But too weak for formalizing: (at least directly)
  - ▷ natural numbers, torsion groups, calculus, ...
  - ▷ generalized quantifiers (*most, at least three, some,...*)

SOME RIGHTS RESERVED

PL^1 Syntax (Signature and Variables)

- ▷ **Definition 1** First-order logic (PL^1), is a formal logical system extensively used in mathematics, philosophy, linguistics, and computer science. It combines propositional logic with the ability to quantify over individuals
- ▷ PL^1 talks about two kinds of objects: (so we have two kinds of symbols)
  - ▷ o for truth values (like in PL^0)
  - ▷ Type ι for individuals (numbers, foxes, Pokémon,...)
- ▷ **Definition 2** A first-order signature consists of (all disjoint; $k \in \mathbb{N}$)
  - ▷ connectives: $\Sigma^o = \{T, F, \neg, \vee, \wedge, \Rightarrow, \Leftrightarrow, \ldots\}$ (functions on truth values)
  - ▷ function constants: $\Sigma^f_k = \{f, g, h, \ldots\}$ (functions on individuals)
  - ▷ predicate constants: $\Sigma^p_k = \{p, q, r, \ldots\}$ (relations among inds.)
  - ▷ (Skolem constants: $\Sigma^{sk}_k = \{f^k_1, f^k_2, \ldots\}$) (witness constants; countably ∞)
  - ▷ We take the signature Σ to be all of these together: $\Sigma := \Sigma^o \cup \Sigma^f \cup \Sigma^p \cup \Sigma^{sk}$, where $\Sigma^* := \bigcup_{k\in\mathbb{N}} \Sigma^*_k$.
- ▷ We assume a set of individual variables: $\mathcal{V}_\iota = \{X_\iota, Y_\iota, Z, X^1{}_\iota, X^2\}$ (countably ∞)

SOME RIGHTS RESERVED

PL^1 Syntax (Formulae)

▷ **Definition 3** terms: $\mathbf{A} \in \mathit{wff}_\iota(\Sigma_\iota)$ (denote individuals: type ι)

- ▷ $\mathcal{V}_\iota \subseteq \mathit{wff}_\iota(\Sigma_\iota)$,
- ▷ if $f \in \Sigma_k^f$ and $\mathbf{A}^i \in \mathit{wff}_\iota(\Sigma_\iota)$ for $i \leq k$, then $f(\mathbf{A}^1, \ldots, \mathbf{A}^k) \in \mathit{wff}_\iota(\Sigma_\iota)$.

▷ **Definition 4** propositions: $\mathbf{A} \in \mathit{wff}_o(\Sigma)$ (denote truth values: type o)

- ▷ if $p \in \Sigma_k^p$ and $\mathbf{A}^i \in \mathit{wff}_\iota(\Sigma_\iota)$ for $i \leq k$, then $p(\mathbf{A}^1, \ldots, \mathbf{A}^k) \in \mathit{wff}_o(\Sigma)$,
- ▷ if $\mathbf{A}, \mathbf{B} \in \mathit{wff}_o(\Sigma)$, then $T, \mathbf{A} \wedge \mathbf{B}, \neg\mathbf{A}, \forall X.\mathbf{A} \in \mathit{wff}_o(\Sigma)$.

▷ **Definition 5** We define the connectives $F, \vee, \Rightarrow, \Leftrightarrow$ via the abbreviations $\mathbf{A} \vee \mathbf{B} := \neg(\neg\mathbf{A} \wedge \neg\mathbf{B})$, $\mathbf{A} \Rightarrow \mathbf{B} := \neg\mathbf{A} \vee \mathbf{B}$, $\mathbf{A} \Leftrightarrow \mathbf{B} := (\mathbf{A} \Rightarrow \mathbf{B}) \wedge (\mathbf{B} \Rightarrow \mathbf{A})$, and $F := \neg T$. We will use them like the primary connectives $\wedge$ and $\neg$

▷ **Definition 6** We use $\exists X.\mathbf{A}$ as an abbreviation for $\neg(\forall X.\neg\mathbf{A})$. (existential quantifier)

▷ **Definition 7** Call formulae without connectives or quantifiers atomic else complex.

SOME RIGHTS RESERVED ©: Michael Kohlhase 17 JACOBS UNIVERSITY

Semantics of PL^1

▷ We fix the Universe $\mathcal{D}_o = \{\mathsf{T}, \mathsf{F}\}$ of truth values

▷ We assume an arbitrary universe $\mathcal{D}_\iota \neq \emptyset$ of individuals (this choice is a parameter to the semantics)

▷ **Definition 8** An interpretation $\mathcal{I}$ assigns values to constants, e.g.

- ▷ $\mathcal{I}(\neg)\colon \mathcal{D}_o \to \mathcal{D}_o$ with $\mathsf{T} \mapsto \mathsf{F}$, $\mathsf{F} \mapsto \mathsf{T}$, and $\mathcal{I}(\wedge) = \ldots$ (as in PL^0)
- ▷ $\mathcal{I}\colon \Sigma_k^f \to \mathcal{F}({\mathcal{D}_\iota}^k; \mathcal{D}_\iota)$ (interpret function symbols as arbitrary functions)
- ▷ $\mathcal{I}\colon \Sigma_k^p \to \mathcal{P}({\mathcal{D}_\iota}^k)$ (interpret predicates as arbitrary relations)

▷ **Definition 9** A variable assignment $\varphi\colon \mathcal{V}_\iota \to \mathcal{D}_\iota$ maps variables into the universe.

SOME RIGHTS RESERVED ©: Michael Kohlhase 18 JACOBS UNIVERSITY

Semantics (PL^1 continued)

▷ The value function $\mathcal{I}_\varphi$ is recursively defined:

- ▷ $\mathcal{I}_\varphi\colon \mathit{wff}_\iota(\Sigma_\iota) \to \mathcal{D}_\iota$ assigns values to terms.
  - ▷ $\mathcal{I}_\varphi(X) := \varphi(X)$ and
  - ▷ $\mathcal{I}_\varphi(f(\mathbf{A}_1,\ldots,\mathbf{A}_k)) := \mathcal{I}(f)(\mathcal{I}_\varphi(\mathbf{A}_1),\ldots,\mathcal{I}_\varphi(\mathbf{A}_k))$
- ▷ $\mathcal{I}_\varphi\colon \mathit{wff}_o(\Sigma) \to \mathcal{D}_o$ assigns values to formulae:
  - ▷ $\mathcal{I}_\varphi(T) = \mathcal{I}(T) = \mathsf{T}$, $\mathcal{I}_\varphi(\neg\mathbf{A}) = \mathcal{I}(\neg)(\mathcal{I}_\varphi(\mathbf{A}))$ $\mathcal{I}_\varphi(\mathbf{A}\wedge\mathbf{B}) = \mathcal{I}(\wedge)(\mathcal{I}_\varphi(\mathbf{A}),\mathcal{I}_\varphi(\mathbf{B}))$ (just as in PL^0)
  - ▷ $\mathcal{I}_\varphi(p(\mathbf{A}^1,\ldots,\mathbf{A}^k)) := \mathsf{T}$, iff $\langle\mathcal{I}_\varphi(\mathbf{A}^1),\ldots,\mathcal{I}_\varphi(\mathbf{A}^k)\rangle \in \mathcal{I}(p)$
  - ▷ $\mathcal{I}_\varphi(\forall X.\mathbf{A}) := \mathsf{T}$, iff $\mathcal{I}_{\varphi,[a/X]}(\mathbf{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$.

▷ Model: $\mathcal{M} = \langle\mathcal{D}_\iota,\mathcal{I}\rangle$ varies in $\mathcal{D}_\iota$ and $\mathcal{I}$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 19 JACOBS UNIVERSITY

Free and Bound Variables

▷ **Definition 10** We call an occurrence of a variable X bound in a formula $\mathbf{A}$, iff it occurs in a sub-formula $\forall X.\mathbf{B}$ of $\mathbf{A}$. We call a variable occurrence free otherwise.

For a formula $\mathbf{A}$, we will use $\mathbf{BVar}(\mathbf{A})$ (and $\mathbf{free}(\mathbf{A})$) for the set of bound (free) variables of $\mathbf{A}$, i.e. variables that have a free/bound occurrence in $\mathbf{A}$.

▷ **Definition 11** We define the set $\mathbf{free}(\mathbf{A})$ of free variables of a formula $\mathbf{A}$ inductively:

$$\begin{aligned}
&\mathbf{free}(X) := \{X\}\\
&\mathbf{free}(f(\mathbf{A}_1,\ldots,\mathbf{A}_n)) := \textstyle\bigcup_{1\leq i\leq n}\mathbf{free}(\mathbf{A}_i)\\
&\mathbf{free}(p(\mathbf{A}_1,\ldots,\mathbf{A}_n)) := \textstyle\bigcup_{1\leq i\leq n}\mathbf{free}(\mathbf{A}_i)\\
&\mathbf{free}(\neg\mathbf{A}) := \mathbf{free}(\mathbf{A})\\
&\mathbf{free}(\mathbf{A}\wedge\mathbf{B}) := \mathbf{free}(\mathbf{A})\cup\mathbf{free}(\mathbf{B})\\
&\mathbf{free}(\forall X.\mathbf{A}) := \mathbf{free}(\mathbf{A})\backslash\{X\}
\end{aligned}$$

▷ **Definition 12** We call a formula $\mathbf{A}$ closed or ground, iff $\mathbf{free}(\mathbf{A}) = \emptyset$. We call a closed proposition a sentence, and denote the set of all ground terms with $\mathit{cwff}_\iota(\Sigma_\iota)$ and the set of sentences with $\mathit{cwff}_o(\Sigma_\iota)$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 20 JACOBS UNIVERSITY

5.2 First-Order Substitutions

Substitutions

- ▷ Intuition: If $\mathbf{B}$ is a term and X is a variable, then we denote the result of systematically replacing all occurrences of X in a term $\mathbf{A}$ by $\mathbf{B}$ with $[\mathbf{B}/X](\mathbf{A})$.
- ▷ Problem: What about $[Z/Y],[Y/X](X)$, is that Y or Z?
- ▷ Folklore: $[Z/Y],[Y/X](X) = Y$, but $[Z/Y]([Y/X](X)) = Z$ of course. (Parallel application)
- ▷ **Definition 13** We call $\sigma\colon \mathit{wff}_\iota(\Sigma_\iota) \to \mathit{wff}_\iota(\Sigma_\iota)$ a substitution, iff $\sigma(f(\mathbf{A}_1,\ldots,\mathbf{A}_n)) = f(\sigma(\mathbf{A}_1),\ldots,\sigma(\mathbf{A}_n))$ and the support $\mathbf{supp}(\sigma) := \{X \mid \sigma(X) \neq X\}$ of σ is finite.
- ▷ **Notation 14** Note that a substitution σ is determined by its values on variables alone, thus we can write σ as $\sigma\big|_{\mathcal{V}_\iota} = \{[\sigma(X)/X] \mid X \in \mathbf{supp}(\sigma)\}$.
- ▷ **Example 15** $[a/x],[f(b)/y],[a/z]$ instantiates $g(x,y,h(z))$ to $g(a,f(b),h(a))$.
- ▷ **Definition 16** We call $\mathbf{intro}(\sigma) := \bigcup_{X\in\mathbf{supp}(\sigma)} \mathbf{free}(\sigma(X))$ the set of variables introduced by σ.

SOME RIGHTS RESERVED

Substitution Extension

- ▷ **Notation 17 (Substitution Extension)** Let σ be a substitution, then we denote with $\sigma,[\mathbf{A}/X]$ the function $\{\langle Y,\mathbf{A}\rangle \in \sigma \mid Y \neq X\} \cup \{\langle X,\mathbf{A}\rangle\}$. ($\sigma,[\mathbf{A}/X]$ coincides with σ of X, and gives the result $\mathbf{A}$ there.)
- ▷ Note: If σ is a substitution, then $\sigma,[\mathbf{A}/X]$ is also a substitution.
- ▷ **Definition 18** If σ is a substitution, then we call $\sigma,[\mathbf{A}/X]$ the extension of σ by $[\mathbf{A}/X]$.

SOME RIGHTS RESERVED

Substitutions on Propositions

▷ Problem: We want to extend substitutions to propositions, in particular to quantified formulae: What is $\sigma(\forall X.\mathbf{A})$?

▷ Idea: σ should not instantiate bound variables.

▷ **Definition 19** $\sigma(\forall X.\mathbf{A}) := \forall X.\sigma_{-X}(\mathbf{A})$, where $\sigma_{-X} := \sigma, [X/X]$

▷ Problem: This can lead to variable capture: $[f(X)/Y](\forall X.p(X,Y))$ would evaluate to $\forall X.p(X, f(X))$, where the second occurrence of X is bound after instantiation, whereas it was free before.

▷ **Definition 20** Let $\mathbf{B} \in \mathit{wff}_\iota(\Sigma_\iota)$ and $\mathbf{A} \in \mathit{wff}_o(\Sigma)$, then we call $\mathbf{B}$ substitutable for X in $\mathbf{A}$, iff $\mathbf{A}$ has no occurrence of X in a subterm $\forall Y.\mathbf{C}$ with $Y \in \mathbf{free}(\mathbf{B})$.

▷ Solution: Forbid substitution $[\mathbf{B}/X]\mathbf{A}$, when $\mathbf{B}$ is not substitutable for X in $\mathbf{A}$.

▷ Better Solution: Rename away the bound variable X in $\forall X.p(X,Y)$ before applying the substitution. (see alphabetic renaming later.)

SOME RIGHTS RESERVED ©: Michael Kohlhase

Substitution Value Lemma for Terms

▷ **Lemma 21** *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be terms, then* $\mathcal{I}_\varphi([\mathbf{B}/X]\mathbf{A}) = \mathcal{I}_\psi(\mathbf{A})$, *where* $\psi = \varphi, [\mathcal{I}_\varphi(\mathbf{B})/X]$.

▷ Proof: by induction on the depth of $\mathbf{A}$:

P.1.1 depth=0:

P.1.1.1 Then $\mathbf{A}$ is a variable (say Y), or constant, so we have three cases

P.1.1.1.1 $\mathbf{A} = Y = X$: then $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\varphi([\mathbf{B}/X](X)) = \mathcal{I}_\varphi(\mathbf{B}) = \psi(X) = \mathcal{I}_\psi(X) = \mathcal{I}_\psi(\mathbf{A})$.

P.1.1.1.2 $\mathbf{A} = Y \neq X$: then $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\varphi([\mathbf{B}/X](Y)) = \mathcal{I}_\varphi(Y) = \varphi(Y) = \psi(Y) = \mathcal{I}_\psi(Y) = \mathcal{I}_\psi(\mathbf{A})$.

P.1.1.1.3 $\mathbf{A}$ is a constant: analogous to the preceding case ($Y \neq X$)

P.1.1.2 This completes the base case (depth = 0). □

P.1.2 depth> 0: then $\mathbf{A} = f(\mathbf{A}_1, \ldots, \mathbf{A}_n)$ and we have

$$\begin{aligned}\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) &= \mathcal{I}(f)(\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}_1)), \ldots, \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}_n))) \\ &= \mathcal{I}(f)(\mathcal{I}_\psi(\mathbf{A}_1), \ldots, \mathcal{I}_\psi(\mathbf{A}_n)) \\ &= \mathcal{I}_\psi(\mathbf{A}).\end{aligned}$$

by inductive hypothesis

P.1.2.2 This completes the inductive case, and we have proven the assertion □

□

SOME RIGHTS RESERVED ©: Michael Kohlhase

Substitution Value Lemma for Propositions

▷ **Lemma 22** *Let* $\mathbf{B} \in wff_\iota(\Sigma_\iota)$ *be substitutable for* X *in* $\mathbf{A} \in wff_o(\Sigma)$*, then* $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\psi(\mathbf{A})$*, where* $\psi = \varphi, [\mathcal{I}_\varphi(\mathbf{B})/X]$.

▷ Proof: by induction on the number n of connectives and quantifiers in $\mathbf{A}$

P.1.1 $n = 0$: then $\mathbf{A}$ is an atomic proposition, and we can argue like in the inductive case of the substitution value lemma for terms.

P.1.2 $n > 0$ and $\mathbf{A} = \forall X.\mathbf{C}$: then $\mathcal{I}_\psi(\mathbf{A}) = \mathcal{I}_\psi(\forall X.\mathbf{C}) = \mathsf{T}$, iff $\mathcal{I}_{\psi,[a/X]}(\mathbf{C}) = \mathcal{I}_{\varphi,[a/X]}(\mathbf{C}) = \mathsf{T}$, for all $a \in \mathcal{D}_\iota$, which is the case, iff $\mathcal{I}_\varphi(\forall X.\mathbf{C}) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathsf{T}$.

P.1.3 $n > 0$ and $\mathbf{A} = \forall Y.\mathbf{C}$ where $X \neq Y$: then $\mathcal{I}_\psi(\mathbf{A}) = \mathcal{I}_\psi(\forall Y.\mathbf{C}) = \mathsf{T}$, iff $\mathcal{I}_{\psi,[a/Y]}(\mathbf{C}) = \mathcal{I}_{\varphi,[a/Y]}([\mathbf{B}/X](\mathbf{C})) = \mathsf{T}$, by inductive hypothesis. So $\mathcal{I}_\psi(\mathbf{A}) = \mathcal{I}_\varphi(\forall Y.[\mathbf{B}/X](\mathbf{C})) = \mathcal{I}_\varphi([\mathbf{B}/X](\forall Y.\mathbf{C})) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}))$

P.1.4 $n > 0$ and $\mathbf{A} = \neg\mathbf{B}$ or $\mathbf{A} = \mathbf{C} \circ \mathbf{D}$:

P.1.4.1 Here we argue like in the inductive case of the term lemma □

□

SOME RIGHTS RESERVED

5.3 Alpha-Renaming for First-Order Logic

Armed with the substitution value lemma we can now prove one of the main representational facts for first-order logic: the names of bound variables do not matter; they can be renamed at liberty without changing the meaning of a formula.

Alphabetic Renaming

▷ **Lemma 23** *Bound variables can be renamed: If* Y *is substitutable for* X *in* $\mathbf{A}$*, then* $\mathcal{I}_\varphi(\forall X.\mathbf{A}) = \mathcal{I}_\varphi(\forall Y.[Y/X](\mathbf{A}))$

▷ Proof: by the definitions:

P.1 $\mathcal{I}_\varphi(\forall X.\mathbf{A}) = \mathsf{T}$, iff

P.2 $\mathcal{I}_{\varphi,[a/X]}(\mathbf{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$, iff

P.3 $\mathcal{I}_{\varphi,[a/Y]}([Y/X](\mathbf{A})) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$, iff (by substitution value lemma)

P.4 $\mathcal{I}_\varphi(\forall Y.[Y/X](\mathbf{A})) = \mathsf{T}$. □

▷ **Definition 24** We call two formulae $\mathbf{A}$ and $\mathbf{B}$ alphabetical variants (or α-equal; write $\mathbf{A} =_\alpha \mathbf{B}$), iff $\mathbf{A} = \forall X.\mathbf{C}$ and $\mathbf{B} = \forall Y.[Y/X](\mathbf{C})$ for some variables X and Y.

SOME RIGHTS RESERVED

We have seen that naive substitutions can lead to variable capture. As a consequence, we always have to presuppose that all instantiations respect a substitutability condition, which is quite tedious. We will now come up with an improved definition of substitution application for first-order logic that does not have this problem.

Avoiding Variable Capture by Built-in α-renaming

- ▷ Idea: Given alphabetic renaming, we will consider alphabetical variants as identical
- ▷ So: Bound variable names in formulae are just a representational device (we rename bound variables wherever necessary)
- ▷ Formally: Take $cwff_o(\Sigma_\iota)$ (new) to be the quotient set of $cwff_o(\Sigma_\iota)$ (old) modulo $=_\alpha$. (formulae as syntactic representatives of equivalence classes)
- ▷ **Definition 25 (Capture-Avoiding Substitution Application)** Let σ be a substitution, $\mathbf{A}$ a formula, and $\mathbf{A}'$ an alphabetical variant of $\mathbf{A}$, such that $\mathbf{intro}(\sigma) \cap \mathbf{BVar}(\mathbf{A}) = \emptyset$. Then $[\mathbf{A}]_{=_\alpha} = [\mathbf{A}']_{=_\alpha}$ and we can define $\sigma([\mathbf{A}]_{=_\alpha}) := [\sigma(\mathbf{A}')]_{=_\alpha}$.
- ▷ **Notation 26** After we have understood the quotient construction, we will neglect making it explicit and write formulae and substitutions with the understanding that they act on quotients.

SOME RIGHTS RESERVED ©: Michael Kohlhase 27 JACOBS UNIVERSITY

6 Inference in First-Order Logic

In this section we will introduce inference systems (calculi) for first-order logic and study their properties, in particular soundness and completeness.

6.1 Recap: General Properties of Logics and Calculi

To prepare the ground for the particular developments coming up, let us spend some time on recapitulating the basic concerns of logical systems.

The notion of a logical system is at the basis of the field of logic. In its most abstract form, a logical system consists of a formal language, a class of models, and a satisfaction relation between models and expressions of the formal lanugage. The satisfaction relation tells us when an expression is deemed true in this model.

Logical Systems

▷ **Definition 27** A logical system is a triple $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$, where $\mathcal{L}$ is a formal language, $\mathcal{K}$ is a set and $\models \subseteq \mathcal{K} \times \mathcal{L}$. Members of $\mathcal{L}$ are called formulae of $\mathcal{S}$, members of $\mathcal{K}$ models for $\mathcal{S}$, and $\models$ the satisfaction relation.

▷ **Definition 28** Let $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, $\mathcal{M} \in \mathcal{K}$ be a model and $\mathbf{A} \in \mathcal{L}$ a formula, then we call $\mathbf{A}$

- ▷ satisfied by $\mathcal{M}$, iff $\mathcal{M} \models \mathbf{A}$
- ▷ falsified by $\mathcal{M}$, iff $\mathcal{M} \not\models \mathbf{A}$
- ▷ satisfiable in $\mathcal{K}$, iff $\mathcal{M} \models \mathbf{A}$ for some model $\mathcal{M} \in \mathcal{K}$.
- ▷ valid in $\mathcal{K}$ (write $\models \mathcal{M}$), iff $\mathcal{M} \models \mathbf{A}$ for all models $\mathcal{M} \in \mathcal{K}$
- ▷ falsifiable in $\mathcal{K}$, iff $\mathcal{M} \not\models \mathbf{A}$ for some $\mathcal{M} \in \mathcal{K}$.
- ▷ unsatisfiable in $\mathcal{K}$, iff $\mathcal{M} \not\models \mathbf{A}$ for all $\mathcal{M} \in \mathcal{K}$.

▷ **Definition 29** Let $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, then we define the entailment relation $\models \subseteq \mathcal{L} \times \mathcal{L}$. We say that $\mathbf{A}$ entails $\mathbf{B}$ (written $\mathbf{A} \models \mathbf{B}$), iff we have $\mathcal{M} \models \mathbf{B}$ for all models $\mathcal{M} \in \mathcal{K}$ with $\mathcal{M} \models \mathbf{A}$.

▷ **Observation 30** $\mathbf{A} \models \mathbf{B}$ *and* $\mathcal{M} \models \mathbf{A}$ *imply* $\mathcal{M} \models \mathbf{B}$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 28 JACOBS UNIVERSITY

Example 31 (First-Order Logic as a Logical System) Let $\mathcal{L} := \mathit{wff}_o(\Sigma)$, $\mathcal{K}$ be the class of first-order models, and $\mathcal{M} \models \mathbf{A} :\Leftrightarrow \mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$, then $\langle \mathcal{L}, \mathcal{K}, \models \rangle$ is a logical system in the sense of Definition 27.

Note that central notions like the entailment relation (which is central for understanding reasoning processes) can be defined independently of the concrete compositional setup we have used for first-order logic, and only need the general assumptions about logical systems.

Let us now turn to the syntactical counterpart of the entailment relation: derivability in a calculus. Again, we take care to define the concepts at the general level of logical systems.

The intuition of a calculus is that it provides a set of syntactic rules that allow to reason by considering the form of propositions alone. Such rules are called inference rules, and they can be strung together to derivations — which can alternatively be viewed either as sequences of formulae where all formulae are justified by prior formulae or as trees of inference rule applications. But we can also define a calculus in the more general setting of logical systems as an arbitrary relation on formulae with some general properties. That allows us to abstract away from the homomorphic setup of logics and calculi and concentrate on the basics.

Derivation Systems and Inference Rules

▷ **Definition 32** Let $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, then we call a relation $\vdash \subseteq \mathcal{P}(\mathcal{L}) \times \mathcal{L}$ a derivation system for $\mathcal{S}$, if it

- ▷ is proof-reflexive, i.e. $\mathcal{H} \vdash \mathbf{A}$, if $\mathbf{A} \in \mathcal{H}$;
- ▷ is proof-transitive, i.e. if $\mathcal{H} \vdash \mathbf{A}$ and $\mathcal{H}' \cup \{\mathbf{A}\} \vdash \mathbf{B}$, then $\mathcal{H} \cup \mathcal{H}' \vdash \mathbf{B}$;
- ▷ admits weakening, i.e. $\mathcal{H} \vdash \mathbf{A}$ and $\mathcal{H} \subseteq \mathcal{H}'$ imply $\mathcal{H}' \vdash \mathbf{A}$.

▷ **Definition 33** Let $\mathcal{L}$ be a formal language, then an inference rule over $\mathcal{L}$

$$\frac{\mathbf{A}_1 \quad \cdots \quad \mathbf{A}_n}{\mathbf{C}}\mathcal{N}$$

where $\mathbf{A}_1, \ldots, \mathbf{A}_n$ and $\mathbf{C}$ are formula schemata for $\mathcal{L}$ and $\mathcal{N}$ is a name.
The $\mathbf{A}_i$ are called assumptions, and $\mathbf{C}$ is called conclusion.

▷ **Definition 34** An inference rule without assumptions is called an axiom (schema).

▷ **Definition 35** Let $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, then we call a set $\mathcal{C}$ of inference rules over $\mathcal{L}$ a calculus for $\mathcal{S}$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 29 JACOBS UNIVERSITY

Derivations and Proofs

▷ **Definition 36** A derivation of a formula $\mathbf{C}$ from a set $\mathcal{H}$ of hypotheses (write $\mathcal{H} \vdash \mathbf{C}$) is a sequence $\mathbf{A}_1, \ldots, \mathbf{A}_m$ of formulae, such that

- ▷ $\mathbf{A}_m = \mathbf{C}$ (derivation culminates in $\mathbf{C}$)
- ▷ for all $(1 \leq i \leq m)$, either $\mathbf{A}_i \in \mathcal{H}$ (hypothesis)
  or there is an inference rule $\dfrac{\mathbf{A}_{l_1} \quad \cdots \quad \mathbf{A}_{l_k}}{\mathbf{A}_i}$, where $l_j < i$ for all $j \leq k$.

▷ **Example 37** In the propositional calculus of natural deduction we have $\mathbf{A} \vdash \mathbf{B} \Rightarrow \mathbf{A}$: the sequence is $\mathbf{A} \Rightarrow \mathbf{B} \Rightarrow \mathbf{A}, \mathbf{A}, \mathbf{B} \Rightarrow \mathbf{A}$

$$\dfrac{\dfrac{}{\mathbf{A} \Rightarrow \mathbf{B} \Rightarrow \mathbf{A}}\mathsf{Ax} \qquad \mathbf{A}}{\mathbf{B} \Rightarrow \mathbf{A}} \Rightarrow E$$

▷ **Observation 38** *Let $\mathcal{S} := \langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, then the $\mathcal{C}$ derivation relation defined in Definition 36 is a derivation system in the sense of Definition 32*

▷ **Definition 39** A derivation $\emptyset \vdash_{\mathcal{C}} \mathbf{A}$ is called a proof of $\mathbf{A}$ and if one exists ($\vdash_{\mathcal{C}} \mathbf{A}$) then $\mathbf{A}$ is called a $\mathcal{C}$-theorem.

▷ **Definition 40** an inference rule $\mathcal{I}$ is called admissible in $\mathcal{C}$, if the extension of $\mathcal{C}$ by $\mathcal{I}$ does not yield new theorems.

SOME RIGHTS RESERVED ©: Michael Kohlhase 30 JACOBS UNIVERSITY

With formula schemata we mean representations of sets of formulae. In our example above, we used uppercase boldface letters as (meta)-variables for formulae. For instance, the the "modus ponens" inference rule stands for[1] EdNote:1

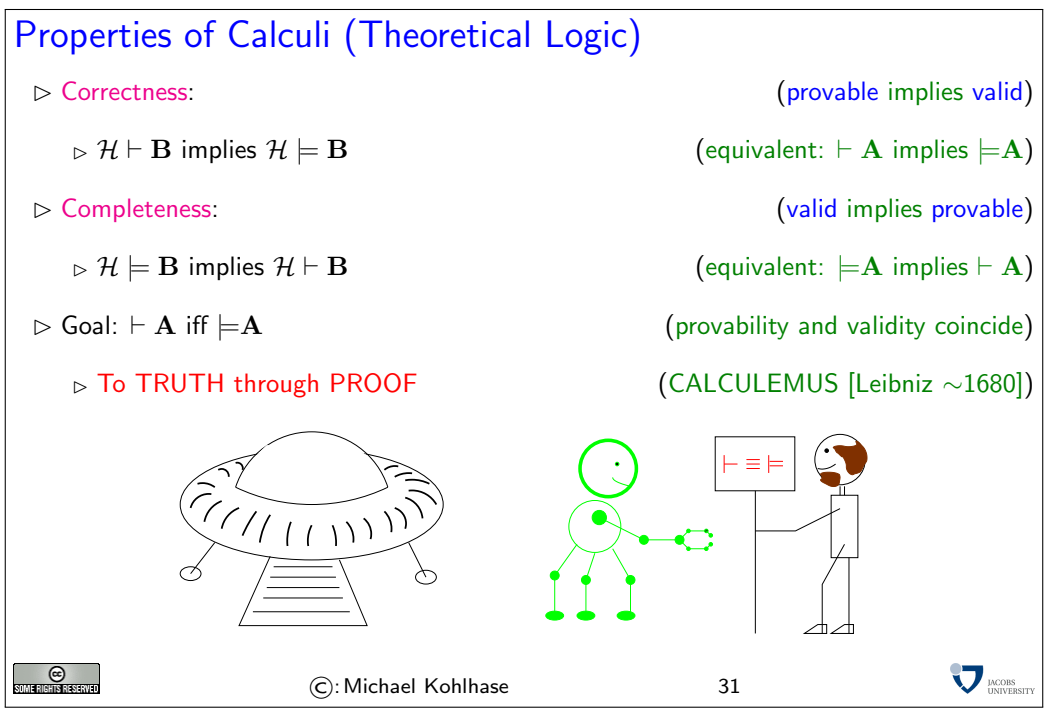
[1] EdNote: continue

As an axiom does not have assumptions, it can be added to a proof at any time. This is just what we did with the axioms in our example proof.

In general formulae can be used to represent facts about the world as propositions; they have a semantics that is a mapping of formulae into the real world (propositions are mapped to truth values.) We have seen two relations on formulae: the entailment relation and the deduction relation. The first one is defined purely in terms of the semantics, the second one is given by a calculus, i.e. purely syntactically. Is there any relation between these relations?

Ideally, both relations would be the same, then the calculus would allow us to infer all facts that can be represented in the given formal language and that are true in the real world, and only those. In other words, our representation and inference is faithful to the world.

A consequence of this is that we can rely on purely syntactical means to make predictions about the world. Computers rely on formal representations of the world; if we want to solve a problem on our computer, we first represent it in the computer (as data structures, which can be seen as a formal language) and do syntactic manipulations on these structures (a form of calculus). Now, if the provability relation induced by the calculus and the validity relation coincide (this will be quite difficult to establish in general), then the solutions of the program will be correct, and we will find all possible ones.



6.2 First-Order Calculi

In this section we will introduce two reasoning calculi for first-order logic, both were invented by Gerhard Gentzen in the 1930's and are very much related. The "natural deduction" calculus was created in order to model the natural mode of reasoning e.g. in everyday mathematical practice. This calculus was intended as a counter-approach to the well-known Hilbert-style calculi, which were mainly used as theoretical devices for studying reasoning in principle, not for modeling particular reasoning styles.

The "sequent calculus" was a rationalized version and extension of the natural deduction calculus that makes certain meta-proofs simpler to push through[2]. EdNote:2

[2] EdNote: say something about cut elimination/analytical calculi somewhere

Both calculi have a similar structure, which is motivated by the human-orientation: rather than using a minimal set of inference rules, they provide two inference rules for every connective and quantifier, one "introduction rule" (an inference rule that derives a formula with that symbol at the head) and one "elimination rule" (an inference rule that acts on a formula with this head and derives a set of subformulae).

This allows us to introduce the calculi in two stages, first for the propositional connectives and then extend this to a calculus for first-order logic by adding rules for the quantifiers.

We will now introduce the "natural deduction" calculus for propositional logic. The calculus was created in order to model the natural mode of reasoning e.g. in everyday mathematical practice. This calculus was intended as a counter-approach to the well-known Hilbert style calculi, which were mainly used as theoretical devices for studying reasoning in principle, not for modeling particular reasoning styles.

Rather than using a minimal set of inference rules, the natural deduction calculus provides two/three inference rules for every connective and quantifier, one "introduction rule" (an inference rule that derives a formula with that symbol at the head) and one "elimination rule" (an inference rule that acts on a formula with this head and derives a set of subformulae).

Calculi: Natural Deduction (ND^0) [Gentzen'30]

▷ Idea: ND^0 tries to mimic human theorem proving behavior (non- minimal)

▷ **Definition 41** The ND^0 calculus has rules for the introduction and elimination of connectives

Introduction	Elimination		Axiom
$\dfrac{\mathbf{A}\quad\mathbf{B}}{\mathbf{A}\wedge\mathbf{B}}\wedge I$	$\dfrac{\mathbf{A}\wedge\mathbf{B}}{\mathbf{A}}\wedge E_l$	$\dfrac{\mathbf{A}\wedge\mathbf{B}}{\mathbf{B}}\wedge E_r$	$\dfrac{}{\mathbf{A}\vee\neg\mathbf{A}}\mathsf{TND}$
$\dfrac{\dfrac{[\mathbf{A}]^1}{\mathbf{B}}}{\mathbf{A}\Rightarrow\mathbf{B}}\Rightarrow I^1$	$\dfrac{\mathbf{A}\Rightarrow\mathbf{B}\quad\mathbf{A}}{\mathbf{B}}\Rightarrow E$		

▷ TND is used only in classical logic (otherwise constructive/intuitionistic)

SOME RIGHTS RESERVED ©: Michael Kohlhase 32 JACOBS UNIVERSITY

The most characteristic rule in the natural deduction calculus is the $\Rightarrow I$ rule. It corresponds to the mathematical way of proving an implication $\mathbf{A}\Rightarrow\mathbf{B}$: We assume that $\mathbf{A}$ is true and show $\mathbf{B}$ from this assumption. When we can do this we discharge (get rid of) the assumption and conclude $\mathbf{A}\Rightarrow\mathbf{B}$. This mode of reasoning is called hypothetical reasoning. Note that the local hypothesis is discharged by the rule $\Rightarrow I$, i.e. it cannot be used in any other part of the proof. As the $\Rightarrow I$ rules may be nested, we decorate both the rule and the corresponding assumption with a marker (here the number 1).

Let us now consider an example of hypothetical reasoning in action.

Natural Deduction: Examples

▷ Inference with local hypotheses

$$\dfrac{\dfrac{\dfrac{[\mathbf{A}\wedge\mathbf{B}]^1}{\mathbf{B}}\wedge E_r \qquad \dfrac{[\mathbf{A}\wedge\mathbf{B}]^1}{\mathbf{A}}\wedge E_l}{\mathbf{B}\wedge\mathbf{A}}\wedge I}{\mathbf{A}\wedge\mathbf{B}\Rightarrow\mathbf{B}\wedge\mathbf{A}}\Rightarrow I^1$$

$$\dfrac{\dfrac{\begin{array}{c}[A]^1\\[B]^2\end{array}}{B\Rightarrow A}\Rightarrow I^2}{A\Rightarrow B\Rightarrow A}\Rightarrow I^1$$

SOME RIGHTS RESERVED

Another characteristic of the natural deduction calculus is that it has inference rules (introduction and elimination rules) for all connectives. So we extend the set of rules from Definition 41 for disjunction, negation and falsity.

More Rules for Natural Deduction

▷ **Definition 42** ND^0 has the following additional rules for the remaining connectives.

$$\dfrac{\mathbf{A}}{\mathbf{A}\vee\mathbf{B}}\vee I_l \qquad \dfrac{\mathbf{B}}{\mathbf{A}\vee\mathbf{B}}\vee I_r \qquad \dfrac{\mathbf{A}\vee\mathbf{B} \quad \begin{array}{c}[\mathbf{A}]^1\\ \vdots\\ \mathbf{C}\end{array} \quad \begin{array}{c}[\mathbf{B}]^1\\ \vdots\\ \mathbf{C}\end{array}}{\mathbf{C}}\vee E^1$$

$$\dfrac{\begin{array}{c}[\mathbf{A}]^1\\ \vdots\\ F\end{array}}{\neg\mathbf{A}}\neg I^1 \qquad \dfrac{\neg\neg\mathbf{A}}{\mathbf{A}}\neg E$$

$$\dfrac{\neg\mathbf{A} \quad \mathbf{A}}{F}FI \qquad \dfrac{F}{\mathbf{A}}FE$$

SOME RIGHTS RESERVED

Natural Deduction in Sequent Calculus Formulation

▷ Idea: Explicit representation of hypotheses (lift calculus to judgments)

▷ **Definition 43** A judgment is a meta-statement about the provability of propositions

▷ **Definition 44** A sequent is a judgment of the form $\mathcal{H} \vdash \mathbf{A}$ about the provability of the formula $\mathbf{A}$ from the set $\mathcal{H}$ of hypotheses.

▷ Idea: Reformulate ND rules so that they act on sequents

▷ **Example 45**

$$\dfrac{\dfrac{\dfrac{\mathbf{A}\wedge\mathbf{B}\vdash\mathbf{A}\wedge\mathbf{B}}{\mathbf{A}\wedge\mathbf{B}\vdash\mathbf{B}}\wedge E_r \quad \dfrac{\mathbf{A}\wedge\mathbf{B}\vdash\mathbf{A}\wedge\mathbf{B}}{\mathbf{A}\wedge\mathbf{B}\vdash\mathbf{A}}\wedge E_l}{\mathbf{A}\wedge\mathbf{B}\vdash\mathbf{B}\wedge\mathbf{A}}\wedge I}{\emptyset\vdash\mathbf{A}\wedge\mathbf{B}\Rightarrow\mathbf{B}\wedge\mathbf{A}}\Rightarrow I$$

▷ Note: Even though the antecedent of a sequent is written like a sequence, it is actually a set. In particular, we can permute and duplicate members at will.

SOME RIGHTS RESERVED ©: Michael Kohlhase 35 JACOBS UNIVERSITY

Sequent-Style Rules for Natural Deduction

▷ **Definition 46** The following inference rules make up the sequent calculus

$$\dfrac{}{\Gamma,\mathbf{A}\vdash\mathbf{A}}\mathsf{Ax} \qquad \dfrac{\Gamma\vdash\mathbf{B}}{\Gamma,\mathbf{A}\vdash\mathbf{B}}\mathsf{weaken} \qquad \dfrac{}{\Gamma\vdash\mathbf{A}\vee\neg\mathbf{A}}\mathsf{TND}$$

$$\dfrac{\Gamma\vdash\mathbf{A}\quad\Gamma\vdash\mathbf{B}}{\Gamma\vdash\mathbf{A}\wedge\mathbf{B}}\wedge I \qquad \dfrac{\Gamma\vdash\mathbf{A}\wedge\mathbf{B}}{\Gamma\vdash\mathbf{A}}\wedge E_l \qquad \dfrac{\Gamma\vdash\mathbf{A}\wedge\mathbf{B}}{\Gamma\vdash\mathbf{B}}\wedge E_r$$

$$\dfrac{\Gamma\vdash\mathbf{A}}{\Gamma\vdash\mathbf{A}\vee\mathbf{B}}\vee I_l \qquad \dfrac{\Gamma\vdash\mathbf{B}}{\Gamma\vdash\mathbf{A}\vee\mathbf{B}}\vee I_r \qquad \dfrac{\Gamma\vdash\mathbf{A}\vee\mathbf{B}\quad\Gamma,\mathbf{A}\vdash\mathbf{C}\quad\Gamma,\mathbf{B}\vdash\mathbf{C}}{\Gamma\vdash\mathbf{C}}\vee E$$

$$\dfrac{\Gamma,\mathbf{A}\vdash\mathbf{B}}{\Gamma\vdash\mathbf{A}\Rightarrow\mathbf{B}}\Rightarrow I \qquad \dfrac{\Gamma\vdash\mathbf{A}\Rightarrow\mathbf{B}\quad\Gamma\vdash\mathbf{A}}{\Gamma\vdash\mathbf{B}}\Rightarrow E$$

$$\dfrac{\Gamma,\mathbf{A}\vdash F}{\Gamma\vdash\neg\mathbf{A}}\neg I \qquad \dfrac{\Gamma\vdash\neg\neg\mathbf{A}}{\mathbf{A}}\neg E$$

$$\dfrac{\Gamma\vdash\neg\mathbf{A}\quad\Gamma\vdash\mathbf{A}}{\Gamma\vdash F}FI \qquad \dfrac{\Gamma\vdash F}{\Gamma\vdash\mathbf{A}}FE$$

SOME RIGHTS RESERVED ©: Michael Kohlhase 36 JACOBS UNIVERSITY

First-Order Natural Deduction

▷ Rules for propositional connectives just as always

▷ **Definition 47 (New Quantifier Rules)** The $\mathcal{ND}$ extends ND^0 by the following four rules

$$\frac{\mathbf{A}}{\forall X.\mathbf{A}}\forall I^* \qquad \frac{\forall X.\mathbf{A}}{[\mathbf{B}/X](\mathbf{A})}\forall E$$

$$\frac{[\mathbf{B}/X](\mathbf{A})}{\exists X.\mathbf{A}}\exists I \qquad \frac{\exists X.\mathbf{A} \quad \begin{array}{c}[[c/X](\mathbf{A})]^1\\ \vdots \\ \mathbf{C}\end{array}}{\mathbf{C}}\exists E^1$$

* means that $\mathbf{A}$ does not depend on any hypothesis in which X is free.

SOME RIGHTS RESERVED ©: Michael Kohlhase 37 JACOBS UNIVERSITY

The intuition behind the rule $\forall I$ is that a formula $\mathbf{A}$ with a (free) variable X can be generalized to $\forall X.\mathbf{A}$, if X stands for an arbitrary object, i.e. there are no restricting assumptions about X. The $\forall E$ rule is just a substitution rule that allows to instantiate arbitrary terms $\mathbf{B}$ for X in $\mathbf{A}$. The $\exists I$ rule says if we have a witness $\mathbf{B}$ for X in $\mathbf{A}$ (i.e. a concrete term $\mathbf{B}$ that makes $\mathbf{A}$ true), then we can existentially close $\mathbf{A}$. The $\exists E$ rule corresponds to the common mathematical practice, where we give objects we know exist a new name c and continue the proof by reasoning about this concrete object c. Anything we can prove from the assumption $[c/X](\mathbf{A})$ we can prove outright if $\exists X.\mathbf{A}$ is known.

This is the classical formulation of the calculus of natural deduction. To prepare the things we want to do later (and to get around the somewhat un-licensed extension by hypothetical reasoning in the calculus), we will reformulate the calculus by lifting it to the "judgements level". Instead of postulating rules that make statements about the validity of propositions, we postulate rules that make state about derivability. This move allows us to make the respective local hypotheses in ND derivations into syntactic parts of the objects (we call them "sequents") manipulated by the inference rules.

First-Order Natural Deduction in Sequent Formulation

▷ Rules for propositional connectives just as always

▷ **Definition 48 (New Quantifier Rules)**

$$\frac{\Gamma \vdash \mathbf{A} \quad X \notin \mathbf{free}(\Gamma)}{\Gamma \vdash \forall X.\mathbf{A}}\forall I \qquad \frac{\Gamma \vdash \forall X.\mathbf{A}}{\Gamma \vdash [\mathbf{B}/X](\mathbf{A})}\forall E$$

$$\frac{\Gamma \vdash [\mathbf{B}/X](\mathbf{A})}{\Gamma \vdash \exists X.\mathbf{A}}\exists I \qquad \frac{\Gamma \vdash \exists X.\mathbf{A} \quad \Gamma, [c/X](\mathbf{A}) \vdash \mathbf{C} \quad c \in \Sigma_0^{sk} \text{ new}}{\Gamma \vdash \mathbf{C}}\exists E$$

SOME RIGHTS RESERVED ©: Michael Kohlhase 38 JACOBS UNIVERSITY

6.3 Abstract Consistency and Model Existence

We will now come to an important tool in the theoretical study of reasoning calculi: the "abstract consistency"/"model existence" method. This method for analyzing calculi was developed by

Jaako Hintikka, Raymond Smullyann, and Peter Andrews in 1950-1970 as an encapsulation of similar constructions that were used in completeness arguments in the decades before.

The basic intuition for this method is the following: typically, a logical system $\mathcal{S} = \langle \mathcal{L}, \mathcal{K} \rangle$ has multiple calculi, human-oriented ones like the natural deduction calculi and machine-oriented ones like the automated theorem proving calculi. All of these need to be analyzed for completeness (as a basic quality assurance measure).

A completeness proof for a calculus $\mathcal{C}$ for $\mathcal{S}$ typically comes in two parts: one analyzes $\mathcal{C}$-consistency (sets that cannot be refuted in $\mathcal{C}$), and the other construct $\mathcal{K}$-models for $\mathcal{C}$-consistent sets.

In this situtation the "abstract consistency"/"model existence" method encapsulates the model construction process into a meta-theorem: the "model existence" theorem. This provides a set of syntactic ("abstract consistency") conditions for calculi that are sufficient to construct models.

With the model existence theorem it suffices to show that $\mathcal{C}$-consistency is an abstract consistency property (a purely syntactic task that can be done by a $\mathcal{C}$-proof transformation argument) to obtain a completeness result for $\mathcal{C}$.

Model Existence (Overview)

- ▷ Definition: Abstract consistency
- ▷ Definition: Hintikka set (maximally abstract consistent)
- ▷ Theorem: Hintikka sets are satisfiable
- ▷ Theorem: If Φ is abstract consistent, then Φ can be extended to a Hintikka set.
- ▷ Corollary: If Φ is abstract consistent, then Φ is satisfiable
- ▷ Application: Let $\mathcal{C}$ be a calculus, if Φ is $\mathcal{C}$-consistent, then Φ is abstract consistent.
- ▷ Corollary: $\mathcal{C}$ is complete.

SOME RIGHTS RESERVED ©: Michael Kohlhase 39 JACOBS UNIVERSITY

The proof of the model existence theorem goes via the notion of a Hintikka set, a set of formulae with very strong syntactic closure properties, which allow to read off models. Jaako Hintikka's original idea for completeness proofs was that for every complete calculus $\mathcal{C}$ and every $\mathcal{C}$-consistent set one can induce a Hintikka set, from which a model can be constructed. This can be considered as a first model existence theorem. However, the process of obtaining a Hintikka set for a set $\mathcal{C}$-consistent set Φ of sentences usually involves complicated calculus-dependent constructions.

In this situation, Raymond Smullyann was able to formulate the sufficient conditions for the existence of Hintikka sets in the form of "abstract consistency properties" by isolating the calculus-independent parts of the Hintikka set construction. His technique allows to reformulate Hintikka sets as maximal elements of abstract consistency classes and interpret the Hintikka set construction as a maximizing limit process.

To carry out the "model-existence"/"abstract consistency" method, we will first have to look at the notion of consistency.

Consistency and refutability are very important notions when studying the completeness for calculi; they form syntactic counterparts of satisfiability.

Consistency

- ▷ Let $\mathcal{C}$ be a calculus
- ▷ **Definition 49** Φ is called $\mathcal{C}$-refutable, if there is a formula $\mathbf{B}$, such that $\Phi \vdash_{\mathcal{C}} \mathbf{B}$ and $\Phi \vdash_{\mathcal{C}} \neg\mathbf{B}$.
- ▷ **Definition 50** We call a pair $\mathbf{A}$ and $\neg\mathbf{A}$ a contradiction.
- ▷ So a set Φ is $\mathcal{C}$-refutable, if $\mathcal{C}$ can derive a contradiction from it.
- ▷ **Definition 51** Φ is called $\mathcal{C}$-consistent, iff there is a formula $\mathbf{B}$, that is not derivable from Φ in $\mathcal{C}$.
- ▷ **Definition 52** We call a calculus $\mathcal{C}$ reasonable, iff implication elimination and conjunction introduction are admissible in $\mathcal{C}$ and $\mathbf{A} \wedge \neg\mathbf{A} \Rightarrow \mathbf{B}$ is a $\mathcal{C}$-theorem.
- ▷ **Theorem 53** *$\mathcal{C}$-inconsistency and $\mathcal{C}$-refutability coincide for reasonable calculi*

SOME RIGHTS RESERVED ©: Michael Kohlhase 40 JACOBS UNIVERSITY

It is very important to distinguish the syntactic $\mathcal{C}$-refutability and $\mathcal{C}$-consistency from satisfiability, which is a property of formulae that is at the heart of semantics. Note that the former specify the calculus (a syntactic device) while the latter does not. In fact we should actually say $\mathcal{S}$-satisfiability, where $\mathcal{S} = \langle \mathcal{L}, \mathcal{K}, \models \rangle$ is the current logical system.

Even the word "contradiction" has a syntactical flavor to it, it translates to "saying against each other" from its latin root.

The notion of an "abstract consistency class" provides the a calculus-independent notion of "consistency": A set Φ of sentences is considered "consistent in an abstract sense", iff it is a member of an abstract consistency class ∇.

Abstract Consistency

- ▷ **Definition 54** Let ∇ be a family of sets. We call ∇ closed under subsets, iff for each $\Phi \in \nabla$, all subsets $\Psi \subseteq \Phi$ are elements of ∇.
- ▷ **Notation 55** We will use $\Phi * \mathbf{A}$ for $\Phi \cup \{\mathbf{A}\}$.
- ▷ **Definition 56** A family $\nabla \subseteq \mathit{wff}_o(\Sigma)$ of sets of formulae is called a (first-order) abstract consistency class , iff it is closed under subsets, and for each $\Phi \in \nabla$
  - ∇_c) $\mathbf{A} \notin \Phi$ or $\neg\mathbf{A} \notin \Phi$ for atomic $\mathbf{A} \in \mathit{wff}_o(\Sigma)$.
  - $\nabla_\neg$) $\neg\neg\mathbf{A} \in \Phi$ implies $\Phi * \mathbf{A} \in \nabla$
  - $\nabla_\wedge$) $(\mathbf{A} \wedge \mathbf{B}) \in \Phi$ implies $(\Phi \cup \{\mathbf{A}, \mathbf{B}\}) \in \nabla$
  - $\nabla_\vee$) $\neg(\mathbf{A} \wedge \mathbf{B}) \in \Phi$ implies $\Phi * \neg\mathbf{A} \in \nabla$ or $\Phi * \neg\mathbf{B} \in \nabla$
  - $\nabla_\forall$) If $(\forall X.\mathbf{A}) \in \Phi$, then $\Phi * [\mathbf{B}/X](\mathbf{A}) \in \nabla$ for each closed term $\mathbf{B}$.
  - $\nabla_\exists$) If $\neg(\forall X.\mathbf{A}) \in \Phi$ and c is an individual constant that does not occur in Φ, then $\Phi * \neg[c/X](\mathbf{A}) \in \nabla$

SOME RIGHTS RESERVED ©: Michael Kohlhase 41 JACOBS UNIVERSITY

The conditions are very natural: Take for instance ∇_c, it would be foolish to call a set Φ of sentences "consistent under a complete calculus", if it contains an elementary contradiction. The next condition $\nabla_\neg$ says that if a set Φ that contains a sentence $\neg\neg\mathbf{A}$ is "consistent", then we

should be able to extend it by **A** without losing this property; in other words, a complete calculus should be able to recognize **A** and $\neg\neg\mathbf{A}$ to be equivalent.

We will carry out the proof here, since it gives us practice in dealing with the abstract consistency properties.

We now come to a very technical condition that will allow us to carry out a limit construction in the Hintikka set extension argument later.

Compact Collections

▷ **Definition 57** We call a collection ∇ of sets compact, iff for any set Φ we have $\Phi \in \nabla$, iff $\Psi \in \nabla$ for every finite subset Ψ of Φ.

▷ **Lemma 58** *If ∇ is compact, then ∇ is closed under subsets.*

▷ Proof:

P.1 Suppose $S \subseteq T$ and $T \in \nabla$.

P.2 Every finite subset A of S is a finite subset of T.

P.3 As ∇ is compact, we know that $A \in \nabla$.

P.4 Thus $S \in \nabla$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 42 JACOBS UNIVERSITY

The main result here is that abstract consistency classes can be extended to compact ones. The proof is quite tedious, but relatively straightforward. It allows us to assume that all abstract consistency classes are compact in the first place (otherwise we pass to the compact extension).

Compact Abstract Consistency Classes

▷ **Lemma 59** *Any abstract consistency class can be extended to a compact one.*

▷ Proof:

P.1 We choose $\nabla' := \{\Phi \subseteq \mathit{wff}_o(\mathcal{V}_o) \mid$ every finite subset of Φ is in $\nabla\}$.

P.2 Now suppose that $\Phi \in \nabla$. ∇ is closed under subsets, so every finite subset of Φ is in ∇ and thus $\Phi \in \nabla'$. Hence $\nabla \subseteq \nabla'$.

P.3 Next let us show that each ∇' is compact.

P.4.1 Suppose $\Phi \in \nabla'$ and Ψ is an arbitrary finite subset of Φ.

P.4.2 By definition of ∇' all finite subsets of Φ are in ∇ and therefore $\Psi \in \nabla'$.

P.4.3 Thus all finite subsets of Φ are in ∇' whenever Φ is in ∇'.

P.4.4 On the other hand, suppose all finite subsets of Φ are in ∇'.

P.4.5 Then by the definition of ∇' the finite subsets of Φ are also in ∇, so $\Phi \in \nabla'$. Thus ∇' is compact. □

P.4 Note that ∇' is closed under subsets by the Lemma above.

P.5.1 To show ∇_c, let $\Phi \in \nabla'$ and suppose there is an atom $\mathbf{A}$, such that $\{\mathbf{A}, \neg\mathbf{A}\} \subseteq \Phi$. Then $\{\mathbf{A}, \neg\mathbf{A}\} \in \nabla$ contradicting ∇_c.

P.5.2.1 Let Ψ be any finite subset of $\Phi * \mathbf{A}$, and $\Theta := (\Psi \backslash \{\mathbf{A}\}) * \neg\neg\mathbf{A}$.

P.5.2.2 Θ is a finite subset of Φ, so $\Theta \in \nabla$.

P.5.2.3 Since ∇ is an abstract consistency class and $\neg\neg\mathbf{A} \in \Theta$, we get $\Theta * \mathbf{A} \in \nabla$ by $\nabla_\neg$.

P.5.2.4 We know that $\Psi \subseteq \Theta * \mathbf{A}$ and ∇ is closed under subsets, so $\Psi \in \nabla$.

P.5.2.5 Thus every finite subset Ψ of $\Phi * \mathbf{A}$ is in ∇ and therefore by definition $\Phi * \mathbf{A} \in \nabla'$. □

P.5.2 the other cases are analogous to $\nabla_\neg$. □

□

SOME RIGHTS RESERVED ©: Michael Kohlhase 43 JACOBS UNIVERSITY

Hintikka sets are sets of sentences with very strong analytic closure conditions. These are motivated as maximally consistent sets i.e. sets that already contain everything that can be consistently added to them.

∇-Hintikka Set

▷ **Definition 60** Let ∇ be an abstract consistency class, then we call a set $\mathcal{H} \in \nabla$ a ∇-Hintikka Set, iff $\mathcal{H}$ is maximal in ∇, i.e. for all $\mathbf{A}$ with $\mathcal{H} * \mathbf{A} \in \nabla$ we already have $\mathbf{A} \in \mathcal{H}$.

▷ **Theorem 61 (Hintikka Properties)** *Let ∇ be an abstract consistency class and $\mathcal{H}$ be a ∇-Hintikka set, then*

$\mathcal{H}_c$) *For all $\mathbf{A} \in wff_o(\Sigma)$ we have $\mathbf{A} \notin \mathcal{H}$ or $\neg\mathbf{A} \notin \mathcal{H}$.*

$\mathcal{H}_\neg$) *If $\neg\neg\mathbf{A} \in \mathcal{H}$ then $\mathbf{A} \in \mathcal{H}$.*

$\mathcal{H}_\wedge$) *If $(\mathbf{A} \wedge \mathbf{B}) \in \mathcal{H}$ then $\mathbf{A}, \mathbf{B} \in \mathcal{H}$.*

$\mathcal{H}_\vee$) *If $\neg(\mathbf{A} \wedge \mathbf{B}) \in \mathcal{H}$ then $\neg\mathbf{A} \in \mathcal{H}$ or $\neg\mathbf{B} \in \mathcal{H}$.*

$\mathcal{H}_\forall$) *If $(\forall X.\mathbf{A}) \in \mathcal{H}$, then $[\mathbf{B}/X](\mathbf{A}) \in \mathcal{H}$ for each closed term $\mathbf{B}$.*

$\mathcal{H}_\exists$) *If $\neg(\forall X.\mathbf{A}) \in \mathcal{H}$ then $\neg[\mathbf{B}/X](\mathbf{A}) \in \mathcal{H}$ for some term closed term $\mathbf{B}$.*

▷ Proof:

P.1 We prove the properties in turn

P.1.1 $\mathcal{H}_c$: by induction on the structure of $\mathbf{A}$

P.1.1.1.1 $\mathbf{A} \in \mathcal{V}_o$: Then $\mathbf{A} \notin \mathcal{H}$ or $\neg\mathbf{A} \notin \mathcal{H}$ by ∇_c.

P.1.1.1.2 $\mathbf{A} = \neg\mathbf{B}$:

P.1.1.1.2.1 Let us assume that $\neg\mathbf{B} \in \mathcal{H}$ and $\neg\neg\mathbf{B} \in \mathcal{H}$,

P.1.1.1.2.2 then $\mathcal{H} * \mathbf{B} \in \nabla$ by $\nabla_\neg$, and therefore $\mathbf{B} \in \mathcal{H}$ by maximality.

P.1.1.1.2.3 So $\{\mathbf{B}, \neg\mathbf{B}\} \subseteq \mathcal{H}$, which contradicts the inductive hypothesis. □

P.1.1.1.3 $\mathbf{A} = \mathbf{B} \vee \mathbf{C}$: similar to the previous case: □

□

P.1.2 We prove $\mathcal{H}_\neg$ by maximality of $\mathcal{H}$ in ∇.:

P.1.2.1 If $\neg\neg\mathbf{A} \in \mathcal{H}$, then $\mathcal{H} * \mathbf{A} \in \nabla$ by $\nabla_\neg$.

P.1.2.2 The maximality of $\mathcal{H}$ now gives us that $\mathbf{A} \in \mathcal{H}$. □

P.1.3 other $\mathcal{H}_*$ are similar:

SOME RIGHTS RESERVED ©: Michael Kohlhase 44 JACOBS UNIVERSITY

The following theorem is one of the main results in the "abstract consistency"/"model existence" method. For any abstract consistent set Φ it allows us to construct a Hintikka set $\mathcal{H}$ with $\Phi \in \mathcal{H}$.

Extension Theorem

▷ **Theorem 62** *If ∇ is an abstract consistency class and $\Phi \in \nabla$ finite, then there is a ∇-Hintikka set $\mathcal{H}$ with $\Phi \subseteq \mathcal{H}$.*

▷ Proof: Wlog. assume that ∇ compact (else use compact extension)

P.1 Choose an enumeration $\mathbf{A}^1, \mathbf{A}^2, \ldots$ of $cwff_o(\Sigma_\iota)$ and $c^1, c^2, \ldots$ of Σ_0^{sk}.

P.2 and construct a sequence of sets H^i with $H^0 := \Phi$ and

$$H^{n+1} := \begin{cases} H^n & \text{if } H^n * \mathbf{A}^n \notin \nabla \\ H^n \cup \{\mathbf{A}^n, \neg[c^n/X](\mathbf{B})\} & \text{if } H^n * \mathbf{A}^n \in \nabla \text{ and } \mathbf{A}^n = \neg(\forall X.\mathbf{B}) \\ H^n * \mathbf{A}^n & \text{else} \end{cases}$$

P.3 Note that all $H^i \in \nabla$, choose $\mathcal{H} := \bigcup_{i \in \mathbb{N}} H^i$

P.4 $\Psi \subseteq \mathcal{H}$ finite implies there is a $j \in \mathbb{N}$ such that $\Psi \subseteq H^j$,

P.5 so $\Psi \in \nabla$ as ∇ closed under subsets and $\mathcal{H} \in \nabla$ as ∇ is compact.

P.6 Let $\mathcal{H} * \mathbf{B} \in \nabla$, then there is a $j \in \mathbb{N}$ with $\mathbf{B} = \mathbf{A}^j$, so that $\mathbf{B} \in H^{j+1}$ and $H^{j+1} \subseteq \mathcal{H}$

P.7 Thus $\mathcal{H}$ is ∇-maximal □

SOME RIGHTS RESERVED ©: Michael Kohlhase 45 JACOBS UNIVERSITY

Note that the construction in the proof above is non-trivial in two respects. First, the limit construction for $\mathcal{H}$ is not executed in our original abstract consistency class ∇, but in a suitably extended one to make it compact — the original would not have contained $\mathcal{H}$ in general. Second, the set $\mathcal{H}$ is not unique for Φ, but depends on the choice of the enumeration of $cwff_o(\Sigma_\iota)$. If we pick a different enumeration, we will end up with a different $\mathcal{H}$. Say if $\mathbf{A}$ and $\neg\mathbf{A}$ are both ∇-consistent[3] with Φ, then depending on which one is first in the enumeration $\mathcal{H}$, will contain that one; with all the consequences for subsequent choices in the construction process. EdNote:3

Valuation

▷ **Definition 63** A function $\nu\colon cwff_o(\Sigma_\iota) \to \mathcal{D}_o$ is called a (first-order) valuation, iff

- ▷ $\nu(\neg\mathbf{A}) = \mathsf{T}$, iff $\nu(\mathbf{A}) = \mathsf{F}$
- ▷ $\nu(\mathbf{A} \wedge \mathbf{B}) = \mathsf{T}$, iff $\nu(\mathbf{A}) = \mathsf{T}$ and $\nu(\mathbf{B}) = \mathsf{T}$
- ▷ $\nu(\forall X.\mathbf{A}) = \mathsf{T}$, iff $\nu([\mathbf{B}/X](\mathbf{A})) = \mathsf{T}$ for all closed terms $\mathbf{B}$.

▷ **Lemma 64** *If $\varphi\colon \mathcal{V}_o \to \mathcal{D}_o$ is a variable assignment, then $\mathcal{I}_\varphi\colon cwff_o(\Sigma_\iota) \to \mathcal{D}_o$ is a valuation.*

▷ Proof Sketch: Immediate from the definitions

SOME RIGHTS RESERVED ©: Michael Kohlhase 46 JACOBS UNIVERSITY

Thus a valuation is a weaker notion of evaluation in first-order logic; the other direction is also true, even though the proof of this result is much more involved: The existence of a first-order valuation that makes a set of sentences true entails the existence of a model that satisfies it.

[3] EdNote: introduce this above

Valuation and Satisfiability

▷ **Lemma 65** *If $\nu\colon cwff_o(\Sigma_\iota) \to \mathcal{D}_o$ is a valuation and $\Phi \subseteq cwff_o(\Sigma_\iota)$ with $\nu(\Phi) = \{\mathsf{T}\}$, then Φ is satisfiable.*

▷ Proof: We construct a model for Φ.

P.1 Let $\mathcal{D}_\iota := cwff_\iota(\Sigma_\iota)$, and

- ▷ $\mathcal{I}(f)\colon {\mathcal{D}_\iota}^k \to \mathcal{D}_\iota; \langle \mathbf{A}_1, \ldots, \mathbf{A}_k\rangle \mapsto f(\mathbf{A}_1, \ldots, \mathbf{A}_k)$ for $f \in \Sigma^f$
- ▷ $\mathcal{I}(p)\colon {\mathcal{D}_\iota}^k \to \mathcal{D}_o; \langle \mathbf{A}_1, \ldots, \mathbf{A}_k\rangle \mapsto \nu(p(\mathbf{A}_1, \ldots, \mathbf{A}_n))$ for $p \in \Sigma^p$.

P.2 Then variable assignments into $\mathcal{D}_\iota$ are ground substitutions.

P.3 We show $\mathcal{I}_\varphi(\mathbf{A}) = \varphi(\mathbf{A})$ for $\mathbf{A} \in wff_\iota(\Sigma_\iota)$ by induction on $\mathbf{A}$

P.3.1 $\mathbf{A} = X$: then $\mathcal{I}_\varphi(\mathbf{A}) = \varphi(X)$ by definition.

P.3.2 $\mathbf{A} = f(\mathbf{A}_1, \ldots, \mathbf{A}_n)$: then $\mathcal{I}_\varphi(\mathbf{A}) = \mathcal{I}(f)(\mathcal{I}_\varphi(\mathbf{A}_1), \ldots, \mathcal{I}_\varphi(\mathbf{A}_n)) = \mathcal{I}(f)(\varphi(\mathbf{A}_1), \ldots, \varphi(\mathbf{A}_n)) = f(\varphi(\mathbf{A}_1), \ldots, \varphi(\mathbf{A}_n)) = \varphi(f(\mathbf{A}_1, \ldots, \mathbf{A}_n)) = \varphi(\mathbf{A})$

P.4 We show $\mathcal{I}_\varphi(\mathbf{A}) = \nu(\varphi(\mathbf{A}))$ for $\mathbf{A} \in wff_o(\Sigma)$ by induction on $\mathbf{A}$

P.4.1 $\mathbf{A} = p(\mathbf{A}_1, \ldots, \mathbf{A}_n)$: then $\mathcal{I}_\varphi(\mathbf{A}) = \mathcal{I}(p)(\mathcal{I}_\varphi(\mathbf{A}_1), \ldots, \mathcal{I}_\varphi(\mathbf{A}_n)) = \mathcal{I}(p)(\varphi(\mathbf{A}_1), \ldots, \varphi(\mathbf{A}_n)) = \nu(p(\varphi(\mathbf{A}_1), \ldots, \varphi(\mathbf{A}_n))) = \nu(\varphi(p(\mathbf{A}_1, \ldots, \mathbf{A}_n))) = \nu(\varphi(\mathbf{A}))$

P.4.2 $\mathbf{A} = \neg\mathbf{B}$: then $\mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$, iff $\mathcal{I}_\varphi(\mathbf{B}) = \nu(\varphi(\mathbf{B})) = \mathsf{F}$, iff $\nu(\varphi(\mathbf{A})) = \mathsf{T}$.

P.4.3 $\mathbf{A} = \mathbf{B} \wedge \mathbf{C}$: similar

P.4.4 $\mathbf{A} = \forall X.\mathbf{B}$: then $\mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$, iff $\mathcal{I}_\psi(\mathbf{B}) = \nu(\psi(\mathbf{B})) = \mathsf{T}$, for all $\mathbf{C} \in \mathcal{D}_\iota$, where $\psi = \varphi, [\mathbf{C}/X]$. This is the case, iff $\nu(\varphi(\mathbf{A})) = \mathsf{T}$.

P.5 Thus $\mathcal{I}_\varphi(\mathbf{A}) = \nu(\varphi(\mathbf{A})) = \nu(\mathbf{A}) = \mathsf{T}$ for all $\mathbf{A} \in \Phi$.

P.6 Hence $\mathcal{M} \models \mathbf{A}$ for $\mathcal{M} := \langle \mathcal{D}_\iota, \mathcal{I}\rangle$. □

SOME RIGHTS RESERVED

Now, we only have to put the pieces together to obtain the model existence theorem we are after.

Model Existence

▷ **Theorem 66 (Hintikka-Lemma)** *If ∇ is an abstract consistency class and $\mathcal{H}$ a ∇-Hintikka set, then $\mathcal{H}$ is satisfiable.*

▷ Proof:

P.1 we define $\nu(\mathbf{A}) := \mathsf{T}$, iff $\mathbf{A} \in \mathcal{H}$,

P.2 then ν is a valuation by the Hintikka set properties.

P.3 We have $\nu(\mathcal{H}) = \{\mathsf{T}\}$, so $\mathcal{H}$ is satisfiable. □

▷ **Theorem 67 (Model Existence)** *If ∇ is an abstract consistency class and $\Phi \in \nabla$, then Φ is satisfiable.*

▷ Proof:

P.1 There is a ∇-Hintikka set $\mathcal{H}$ with $\Phi \subseteq \mathcal{H}$ (Extension Theorem)

P.2 We know that $\mathcal{H}$ is satisfiable. (Hintikka-Lemma)

P.3 In particular, $\Phi \subseteq \mathcal{H}$ is satisfiable. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 48 JACOBS UNIVERSITY

6.4 A Completeness Proof for First-Order ND

With the model existence proof we have introduced in the last section, the completeness proof for first-order natural deduction is rather simple, we only have to check that ND-consistency is an abstract consistency property.

Consistency, Refutability and Abstract Consistency

▷ **Theorem 68 (Non-Refutability is an Abstract Consistency Property)** $\Gamma := \{\Phi \subseteq cwff_o(\Sigma_\iota) \mid \Phi \text{ not } \mathcal{ND}\text{−refutable}\}$ *is an abstract consistency class.*

▷ Proof: We check the properties of an ACC

P.1 If Φ is non-refutable, then any subset is as well, so Γ is closed under subsets.

P.2 We show the abstract consistency conditions ∇_* for $\Phi \in \Gamma$.

P.2.1 ∇_c:

P.2.1.1 We have to show that $\mathbf{A} \notin \Phi$ or $\neg\mathbf{A} \notin \Phi$ for atomic $\mathbf{A} \in wff_o(\Sigma)$.

P.2.1.2 Equivalently, we show the contrapositive: If $\{\mathbf{A}, \neg\mathbf{A}\} \subseteq \Phi$, then $\Phi \notin \Gamma$.

P.2.1.3 So let $\{\mathbf{A}, \neg\mathbf{A}\} \subseteq \Phi$, then Φ is $\mathcal{ND}$-refutable by construction.

P.2.1.4 So $\Phi \notin \Gamma$. □

P.2.2 $\nabla_\neg$: We show the contrapositive again:

P.2.2.1 Let $\neg\neg\mathbf{A} \in \Phi$ and $\Phi * \mathbf{A} \notin \Gamma$

P.2.2.2 Then we have a refutation $\mathcal{D}\colon \Phi * \mathbf{A} \vdash_{\mathcal{ND}} F$

P.2.2.3 By prepending an application of $\neg E$ for $\neg\neg\mathbf{A}$ to $\mathcal{D}$, we obtain a refutation $\mathcal{D}'\colon \Phi \vdash_{\mathcal{ND}} F$.

P.2.2.4 Thus $\Phi \notin \Gamma$. □

P.2.3 other ∇_* similar:

SOME RIGHTS RESERVED

Henkin's Theorem

▷ **Corollary 69 (Henkin's Theorem)** *Every $\mathcal{ND}$-consistent set of sentences has a model.*

▷ Proof:

P.1 Let Φ be a $\mathcal{ND}$-consistent set of sentences.

P.2 The class of sets of $\mathcal{ND}$-consistent propositions constitute an abstract consistency class

P.3 Thus the model existence theorem guarantees a countable model for Φ. □

▷ **Corollary 70 (Löwenheim&Skolem Theorem)** *Satisfiable set Φ of first-order sentences has a countable model.*

▷ Proof Sketch: The model we constructed is countable, since the set of ground terms is.

SOME RIGHTS RESERVED

Completeness and Compactness

▷ **Theorem 71 (Completeness Theorem for $\mathcal{ND}$)** *If $\Phi \models \mathbf{A}$, then $\Phi \vdash_{\mathcal{ND}} \mathbf{A}$.*

▷ Proof:

P.1 If $\mathbf{A}$ is valid in all models of Φ, then $\Phi * \neg\mathbf{A}$ has no model

P.2 Thus $\Phi * \neg\mathbf{A}$ is inconsistent by Henkins Theorem.

P.3 So $\Phi \vdash_{\mathcal{ND}} \neg\neg\mathbf{A}$ by $\neg I$

P.4 So $\Phi \vdash_{\mathcal{ND}} \mathbf{A}$ by $\neg E$. □

▷ **Theorem 72 (Compactness Theorem for first-order logic)** *If $\Phi \models \mathbf{A}$, then there is already a finite set $\Psi \subseteq \Phi$ with $\Psi \models \mathbf{A}$.*

▷ Proof: This is a direct consequence of the completeness theorem

P.1 We have $\Phi \models \mathbf{A}$, iff $\Phi \vdash_{\mathcal{ND}} \mathbf{A}$.

P.2 As a proof is a finite object, only a finite subset $\Psi \subseteq \Phi$ can appear as leaves in the proof. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 51 JACOBS UNIVERSITY

6.5 Limits of First-Order Logic

We will now come to the limits of first-order Logic.

Gödel's Incompleteness Theorem

▷ **Theorem 73** *No logical system that can Peano-Arithmetic $(\mathbb{N}, s, 0, +, *)$ admits complete calculi.*

▷ Proof: (Sketch)

P.1 Let $\mathcal{L} := \langle \mathcal{S}, \mathcal{C} \rangle$ be such a system. We show that there is a valid $\mathcal{S}$-sentence $\mathbf{A}_{\mathcal{C}}$, that is no $\mathcal{C}$-theorem.

P.2 Encode the syntax of $\mathcal{S}$ and the $\mathcal{C}$ in Peano-arithmetic

P.3 We can now talk about $\mathcal{S}$ and $\mathcal{C}$ in $\mathcal{S}$ itself.

P.4 E.g. there is a $\mathcal{S}$-sentence $\mathbf{B}$ with the meaning: $\mathbf{A}$ is a $\mathcal{C}$-theorem.

P.5 Choose $\mathbf{A}_{\mathcal{C}}$ as "$\mathbf{A}_{\mathcal{C}}$ is no $\mathcal{C}$-theorem" (cf. Russell's set)

P.6 Obviously: $\mathbf{A}_{\mathcal{C}}$ ist valid in all standard models.

P.7 So $\mathcal{C}$ is either not correct or cannot derive $\mathbf{A}_{\mathcal{C}}$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 52 JACOBS UNIVERSITY

7 First-Order Inference with Tableaux

7.1 First-Order Tableaux

Test Calculi: Tableaux and Model Generation

▷ Idea: instead of showing $\emptyset \vdash Th$, show $\neg Th \vdash trouble$ (use $\bot$ for trouble)

▷ **Example 74** Tableau Calculi try to construct models.

Tableau Refutation (Validity)	Model generation (Satisfiability)
$\models P \wedge Q \Rightarrow Q \wedge P$	$\models P \wedge (Q \vee \neg R) \wedge \neg Q$
$P \wedge Q \Rightarrow Q \wedge P^{\mathsf{F}}$ $P \wedge Q^{\mathsf{T}}$ $Q \wedge P^{\mathsf{F}}$ P^{T} Q^{T} $P^{\mathsf{F}} \mid Q^{\mathsf{F}}$ $\bot \mid \bot$	$P \wedge (Q \vee \neg R) \wedge \neg Q^{\mathsf{T}}$ $P \wedge (Q \vee \neg R)^{\mathsf{T}}$ $\neg Q^{\mathsf{T}}$ Q^{F} P^{T} $Q \vee \neg R^{\mathsf{T}}$ $Q^{\mathsf{T}} \mid \neg R^{\mathsf{T}}$ $\bot \mid R^{\mathsf{F}}$
No Model	Herbrand Model $\{P^{\mathsf{T}}, Q^{\mathsf{F}}, R^{\mathsf{F}}\}$ $\varphi := \{P \mapsto \mathsf{T}, Q \mapsto \mathsf{F}, R \mapsto \mathsf{F}\}$

Algorithm: Fully expand all possible tableaux, (no rule can be applied)

▷ ▷ Satisfiable, iff there are open branches (correspond to models)

SOME RIGHTS RESERVED ©: Michael Kohlhase 53 JACOBS UNIVERSITY

Tableau calculi develop a formula in a tree-shaped arrangement that represents a case analysis on when a formula can be made true (or false). Therefore the formulae are decorated with exponents that hold the intended truth value.

On the left we have a refutation tableau that analyzes a negated formula (it is decorated with the intended truth value F). Both branches contain an elementary contradiction $\bot$.

On the right we have a model generation tableau, which analyzes a positive formula (it is decorated with the intended truth value T. This tableau uses the same rules as the refutation tableau, but makes a case analysis of when this formula can be satisfied. In this case we have a closed branch and an open one, which corresponds a model).

Now that we have seen the examples, we can write down the tableau rules formally.

Analytical Tableaux (Formal Treatment of $\mathcal{T}_0$)

- ▷ formula is analyzed in a tree to determine satisfiability
- ▷ branches correspond to valuations (models)
- ▷ one per connective

$$\frac{\mathbf{A}\wedge\mathbf{B}^{\mathsf{T}}}{\begin{array}{c}\mathbf{A}^{\mathsf{T}}\\ \mathbf{B}^{\mathsf{T}}\end{array}}\mathcal{T}_0\wedge \qquad \frac{\mathbf{A}\wedge\mathbf{B}^{\mathsf{F}}}{\mathbf{A}^{\mathsf{F}}\,\Big|\,\mathbf{B}^{\mathsf{F}}}\mathcal{T}_0\vee \qquad \frac{\neg\mathbf{A}^{\mathsf{T}}}{\mathbf{A}^{\mathsf{F}}}\mathcal{T}_0\neg^{\mathsf{T}} \qquad \frac{\neg\mathbf{A}^{\mathsf{F}}}{\mathbf{A}^{\mathsf{T}}}\mathcal{T}_0\neg^{\mathsf{F}} \qquad \frac{\begin{array}{c}\mathbf{A}^{\alpha}\\ \mathbf{A}^{\beta}\end{array}\quad \alpha\neq\beta}{\bot}\mathcal{T}_0\mathsf{cut}$$

- ▷ Use rules exhaustively as long as they contribute new material
- ▷ **Definition 75** Call a tableau saturated, iff no rule applies, and a branch closed, iff it ends in $\bot$, else open. (open branches in saturated tableaux yield models)
- ▷ **Definition 76 ($\mathcal{T}_0$-Theorem/Derivability)** $\mathbf{A}$ is a $\mathcal{T}_0$-theorem ($\vdash_{\mathcal{T}_0}\mathbf{A}$), iff there is a closed tableau with $\mathbf{A}^{\mathsf{F}}$ at the root.

  $\Phi\subseteq wff_o(\mathcal{V}_o)$ derives $\mathbf{A}$ in $\mathcal{T}_0$ ($\Phi\vdash_{\mathcal{T}_0}\mathbf{A}$), iff there is a closed tableau starting with $\mathbf{A}^{\mathsf{F}}$ and Φ^{T}.

SOME RIGHTS RESERVED ©: Michael Kohlhase 54 JACOBS UNIVERSITY

These inference rules act on tableaux have to be read as follows: if the formulae over the line appear in a tableau branch, then the branch can be extended by the formulae or branches below the line. There are two rules for each primary connective, and a branch closing rule that adds the special symbol $\bot$ (for unsatisfiability) to a branch.

We use the tableau rules with the convention that they are only applied, if they contribute new material to the branch. This ensures termination of the tableau procedure for propositional logic (every rule eliminates one primary connective).

Definition 77 We will call a closed tableau with the signed formula $\mathbf{A}^{\alpha}$ at the root a tableau refutation for $\mathcal{A}^{\alpha}$.

The saturated tableau represents a full case analysis of what is necessary to give $\mathbf{A}$ the truth value α; since all branches are closed (contain contradictions) this is impossible.

Definition 78 We will call a tableau refutation for $\mathbf{A}^{\mathsf{F}}$ a tableau proof for $\mathbf{A}$, since it refutes the possibility of finding a model where $\mathbf{A}$ evaluates to F. Thus $\mathbf{A}$ must evaluate to T in all models, which is just our definition of validity.

Thus the tableau procedure can be used as a calculus for propositional logic. In contrast to the calculus in section ?? it does not prove a theorem $\mathbf{A}$ by deriving it from a set of axioms, but it proves it by refuting its negation. Such calculi are called negative or test calculi. Generally negative calculi have computational advanages over positive ones, since they have a built-in sense of direction.

We have rules for all the necessary connectives (we restrict ourselves to $\wedge$ and $\neg$, since the others can be expressed in terms of these two via the propositional identities above. For instance, we can write $\mathbf{A}\vee\mathbf{B}$ as $\neg(\neg\mathbf{A}\wedge\neg\mathbf{B})$, and $\mathbf{A}\Rightarrow\mathbf{B}$ as $\neg\mathbf{A}\vee\mathbf{B}$,....)

We will now extend the propositional tableau techiques to first-order logic. We only have to add two new rules for the universal quantifiers (in positive and negative polarity).

First-Order Standard Tableaux ($\mathcal{T}_1$)

▷ Refutation calculus based on trees of labeled formulae

▷ Tableau-Rules: propositional tableau rules plus

$$\frac{\forall X.\mathbf{A}^{\mathsf{T}} \quad \mathbf{C} \in cwff_\iota(\Sigma_\iota)}{[\mathbf{C}/X](\mathbf{A})^{\mathsf{T}}}\mathcal{T}_1{:}\forall \qquad \frac{\forall X.\mathbf{A}^{\mathsf{F}} \quad c \in (\Sigma_0^{sk}\backslash\mathcal{H})}{[c/X](\mathbf{A})^{\mathsf{F}}}\mathcal{T}_1{:}\exists$$

SOME RIGHTS RESERVED ©: Michael Kohlhase 55 JACOBS UNIVERSITY

The rule $\mathcal{T}_1{:}\forall$ rule operationalizes the intuition that a universally quantified formula is true, iff all of the instances of the scope are. To understand the $\mathcal{T}_1{:}\exists$ rule, we have to keep in mind that $\exists X.\mathbf{A}$ abbreviates $\neg(\forall X.\neg\mathbf{A})$, so that we have to read $\forall X.\mathbf{A}^{\mathsf{F}}$ existentially — i.e. as $\exists X.\neg\mathbf{A}^{\mathsf{T}}$, stating that there is an object with property $\neg\mathbf{A}$. In this situation, we can simply give this object a name: c, which we take from our (infinite) set of witness constants Σ_0^{sk}, which we have given ourselves expressly for this purpose when we defined first-order syntax. In other words $[c/X](\neg\mathbf{A})^{\mathsf{T}} = [c/X](\mathbf{A})^{\mathsf{F}}$ holds, and this is just the conclusion of the $\mathcal{T}_1{:}\exists$ rule.

Note that the $\mathcal{T}_1{:}\forall$ rule is computationally extremely inefficient: we have to guess an (i.e. in a search setting to systematically consider all) instance $\mathbf{C} \in wff_\iota(\Sigma_\iota)$ for X. This makes the rule infinitely branching.

7.2 Free Variable Tableaux

In the next calculus we will try to remedy the computational inefficiency of the $\mathcal{T}_1{:}\forall$ rule. We do this by delaying the choice in the universal rule.

Free variable Tableaux ($\mathcal{T}_1^f$)

▷ Refutation calculus based on trees of labeled formulae

▷ Tableau rules

$$\frac{\forall X.\mathbf{A}^{\mathsf{T}} \quad Y \text{ new}}{[Y/X](\mathbf{A})^{\mathsf{T}}}\mathcal{T}_1^f{:}\forall \qquad \frac{\forall X.\mathbf{A}^{\mathsf{F}} \quad \mathbf{free}(\forall X.\mathbf{A}) = \{X^1,\ldots,X^k\} \quad f \in \Sigma_k^{sk}}{[f(X^1,\ldots,X^k)/X](\mathbf{A})^{\mathsf{F}}}\mathcal{T}_1^f{:}\exists$$

▷ Generalized cut rule $\mathcal{T}_1^f{:}\bot$ instantiates the whole tableau by σ.

$$\frac{\begin{array}{c}\mathbf{A}^\alpha \\ \mathbf{B}^\beta\end{array} \quad \alpha \neq \beta \quad \sigma(\mathbf{A}) = \sigma(\mathbf{B})}{\bot}\mathcal{T}_1^f{:}\bot$$

Advantage: no guessing necessary in $\mathcal{T}_1^f{:}\forall$-rule

⊳⊳ New: find suitable substitution (most general unifier)

SOME RIGHTS RESERVED ©: Michael Kohlhase 56 JACOBS UNIVERSITY

Metavariables: Instead of guessing a concrete instance for the universally quantified variable as in the $\mathcal{T}_1{:}\forall$ rule, $\mathcal{T}_1^f{:}\forall$ instantiates it with a new meta-variable Y, which will be instantiated by need in the course of the derivation.

Skolem terms as witnesses: The introduction of meta-variables makes is necessary to extend the treatment of witnesses in the existential rule. Intuitively, we cannot simply invent a new name,

since the meaning of the body $\mathbf{A}$ may contain meta-variables introduced by the $\mathcal{T}_1^f{:}\forall$ rule. As we do not know their values yet, the witness for the existential statement in the antecedent of the $\mathcal{T}_1^f{:}\exists$ rule needs to depend on that. So witness it using a witness term, concretely by applying a Skolem function to the meta-variables in $\mathbf{A}$.

Instantiating Metavariables: Finally, the $\mathcal{T}_1^f{:}\bot$ rule completes the treatment of meta-variables, it allows to instantiate the whole tableau in a way that the current branch closes. This leaves us with the problem of finding substitutions that make two terms equal.

7.3 First-Order Unification

We will now look into the problem of finding a substitution σ that make two terms equal (we say it unifies them) in more detail. The presentation of the unification algorithm we give here "transformation-based" this has been a very influential way to treat certain algorithms in theoretical computer science.

A transformation-based view of algorithms: The "transformation-based" view of algorithms divides two concerns in presenting and reasoning about algorithms according to Kowalski's slogan[4] EdNote:4

$$\text{computation} = \text{logic} + \text{control}$$

The computational paradigm highlighted by this quote is that (many) algorithms can be thought of as manipulating representations of the problem at hand and transforming them into a form that makes it simple to read off solutions. Given this, we can simplify thinking and reasoning about such algorithms by separating out their "logical" part, which deals with is concerned with how the problem representations can be manipulated in principle from the "control" part, which is concerned with questions about when to apply which transformations.

It turns out that many questions about the algorithms can already be answered on the "logic" level, and that the "logical" analysis of the algorithm can already give strong hints as to how to optimize control.

In fact we will only concern ourselves with the "logical" analysis of unification here.

The first step towards a theory of unification is to take a closer look at the problem itself. A first set of examples show that we have multiple solutions to the problem of finding substitutions that make two terms equal. But we also see that these are related in a systematic way.

Unification (Definitions)

- ▷ Problem: For given terms $\mathbf{A}$ and $\mathbf{B}$ find a substitution σ, such that $\sigma(\mathbf{A}) = \sigma(\mathbf{B})$.
- ▷ **Notation 79** term pairs $\mathbf{A} =^? \mathbf{B}$ e.g. $f(X) =^? f(g(Y))$
- ▷ Solutions (e.g. $[g(a)/X], [a/Y]$, $[g(g(a))/X], [g(a)/Y]$, or $[g(Z)/X], [Z/Y]$) are called unifiers, $\mathbf{U}(\mathbf{A} =^? \mathbf{B}) := \{\sigma \mid \sigma(\mathbf{A}) = \sigma(\mathbf{B})\}$
- ▷ Idea: find representatives in $\mathbf{U}(\mathbf{A} =^? \mathbf{B})$, that generate the set of solutions
- ▷ **Definition 80** Let σ and θ be substitutions and $W \subseteq \mathcal{V}_\iota$, we say that a substitution σ is more general than θ (on W write $\sigma \leq \theta[W]$), iff there is a substitution ρ, such that $\theta = \rho \circ \sigma[W]$, where $\sigma = \rho[W]$, iff $\sigma(X) = \rho(X)$ for all $X \in W$.
- ▷ **Definition 81** σ is called a most general unifier of $\mathbf{A}$ and $\mathbf{B}$, iff it is minimal in $\mathbf{U}(\mathbf{A} =^? \mathbf{B})$ wrt. $\leq [\mathbf{free}(\mathbf{A}) \cup \mathbf{free}(\mathbf{B})]$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 57 JACOBS UNIVERSITY

[4] EdNote: find the reference, and see what he really said

The idea behind a most general unifier is that all other unifiers can be obtained from it by (further) instantiation. In an automated theorem proving setting, this means that using most general unifiers is the least committed choice — any other choice of unifiers (that would be necessary for completeness) can later be obtained by other substitutions.

Note that there is a subtlety in the definition of the ordering on substitutions: we only compare on a subset of the variables. The reason for this is that we have defined substitutions to be total on (the infinite set of) variables for flexibility, but in the applications (see the definition of a most general unifiers), we are only interested in a subset of variables: the ones that occur in the initial problem formulation. Intuitively, we do not care what the unifiers do off that set. If we did not have the restriction to the set W of variables, the ordering relation on substitutions would become much too fine-grained to be useful (i.e. to guarantee unique most general unifiers in our case).

Now that we have defined the problem, we can turn to the unification algorithm itself. We will define it in a way that is very similar to logic programming: we first define a calculus that generates "solved forms" (formulae from which we can read off the solution) and reason about control later. In this case we will reason that control does not matter.

Unification (Equational Systems)

- ▷ Idea: Unification is equation solving.
- ▷ **Definition 82** We call a formula $\mathbf{A}^1 =^? \mathbf{B}^1 \wedge \ldots \wedge \mathbf{A}^n =^? \mathbf{B}^n$ an equational system.

  2
- ▷ We consider equational systems as sets of equations ($\wedge$ is ACI), and equations as two-element multisets ($=^?$ is C).
- ▷ **Definition 83** We say that $X^1 =^? \mathbf{B}^1 \wedge \ldots \wedge X^n =^? \mathbf{B}^n$ is a solved form, iff the X^i are distinct and $X^i \notin \mathbf{free}(\mathbf{B}^j)$.
- ▷ **Lemma 84** *If $\mathcal{E} = X^1 =^? \mathbf{B}^1 \wedge \ldots \wedge X^n =^? \mathbf{B}^n$ is a solved form, then $\mathcal{E}$ has the unique most general unifier $\sigma_{\mathcal{E}} := [\mathbf{B}^1/X^1], \ldots, [\mathbf{B}^n/X^n]$.*
- ▷ Proof:

  P.1 Let $\theta \in \mathbf{U}(\mathcal{E})$, then $\theta(X^i) = \theta(\mathbf{B}^i) = \theta \circ \sigma_{\mathcal{E}}(X^i)$

  P.2 and thus $\theta = \theta \circ \sigma_{\mathcal{E}}[\mathbf{supp}(\sigma)]$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 58 JACOBS UNIVERSITY

In principle, unification problems are sets of equations, which we write as conjunctions, since all of them have to be solved for finding a unifier. Note that it is not a problem for the "logical view" that the representation as conjunctions induces an order, since we know that conjunction is associative, commutative and idempotent, i.e. that conjuncts do not have an intrinsic order or multiplicity, if we consider two equational problems as equal, if they are equivalent as propositional formulae. In the same way, we will abstract from the order in equations, since we know that the equality relation is symmetric. Of course we would have to deal with this somehow in the implementation (typically, we would implement equational problems as lists of pairs), but that belongs into the "control" aspect of the algorithm, which we are abstracting from at the moment.

It is essential to our "logical" analysis of the unification algorithm that we arrive at equational problems whose unifiers we can read off easily. Solved forms serve that need perfectly as the Lemma[5] shows.[6] EdNote:5 EdNote:6

[5] EdNote: reference

[6] EdNote: say something about the occurs-in-check,...

Given the idea that unification problems can be expressed as formulae, we can express the algorithm in three simple rules that transform unification problems into solved forms (or unsolvable ones).

Unification Algorithm

▷ **Definition 85** Inference system $\mathcal{U}$

$$\frac{\mathcal{E} \wedge f(\mathbf{A}^1, \ldots, \mathbf{A}^n) =^? f(\mathbf{B}^1, \ldots, \mathbf{B}^n)}{\mathcal{E} \wedge \mathbf{A}^1 =^? \mathbf{B}^1 \wedge \ldots \wedge \mathbf{A}^n =^? \mathbf{B}^n}\mathcal{U}\mathsf{dec} \qquad \frac{\mathcal{E} \wedge \mathbf{A} =^? \mathbf{A}}{\mathcal{E}}\mathcal{U}\mathsf{triv}$$

$$\frac{\mathcal{E} \wedge X =^? \mathbf{A} \quad X \notin \mathbf{free}(\mathbf{A}) \quad X \in \mathbf{free}(\mathcal{E})}{[\mathbf{A}/X](\mathcal{E}) \wedge X =^? \mathbf{A}}\mathcal{U}\mathsf{elim}$$

▷ **Lemma 86** *$\mathcal{U}$ is correct: $\mathcal{E} \vdash_{\mathcal{U}} \mathcal{F}$ implies $\mathbf{U}(\mathcal{F}) \subseteq \mathbf{U}(\mathcal{E})$*

▷ **Lemma 87** *$\mathcal{U}$ is complete: $\mathcal{E} \vdash_{\mathcal{U}} \mathcal{F}$ implies $\mathbf{U}(\mathcal{E}) \subseteq \mathbf{U}(\mathcal{F})$*

▷ **Lemma 88** *$\mathcal{U}$ is confluent: the order of derivations does not matter*

▷ **Corollary 89** *First-Order Unification is unitary: unique most general unifiers* Proof Sketch: the inference system $\mathcal{U}$ is trivially branching

SOME RIGHTS RESERVED ©: Michael Kohlhase 59 JACOBS UNIVERSITY

Unification Examples

Example 90 Two similar unification problems:

$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{f(g(x,x),h(a)) =^? f(g(a,z),h(z))}{g(x,x) =^? g(a,z) \wedge h(a) =^? h(z)}\mathcal{U}\mathsf{dec}}{x =^? a \wedge x =^? z \wedge h(a) =^? h(z)}\mathcal{U}\mathsf{dec}}{x =^? a \wedge x =^? z \wedge a =^? z}\mathcal{U}\mathsf{elim}}{x =^? a \wedge a =^? z \wedge a =^? z}\mathcal{U}\mathsf{elim}}{x =^? a \wedge z =^? a \wedge a =^? a}\mathcal{U}\mathsf{triv}}{x =^? a \wedge z =^? a}$	$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{f(g(x,x),h(a)) =^? f(g(b,z),h(z))}{g(x,x) =^? g(b,z) \wedge h(a) =^? h(z)}\mathcal{U}\mathsf{dec}}{x =^? b \wedge x =^? z \wedge h(a) =^? h(z)}\mathcal{U}\mathsf{dec}}{x =^? b \wedge x =^? z \wedge a =^? z}\mathcal{U}\mathsf{elim}}{x =^? b \wedge b =^? z \wedge a =^? z}\mathcal{U}\mathsf{elim}}{x =^? a \wedge z =^? a \wedge a =^? b}$
MGU: $[a/x], [a/z]$	not unifiable

SOME RIGHTS RESERVED ©: Michael Kohlhase 60 JACOBS UNIVERSITY

Unification (Termination)

▷ **Definition 91** Let S and T be multisets and $\prec$ a partial ordering on $S \cup T$. Then we define $(S \prec^m T)$, iff $S = C \uplus T'$ and $T = C \uplus \{t\}$, where $s \prec t$ for all $s \in S'$. We call $\prec^m$ the multiset ordering induced by $\prec$.

▷ **Lemma 92** *If $\prec$ is total/terminating on S, then $\prec^m$ is total/terminating on $\mathcal{P}(S)$.*

▷ **Lemma 93** *$\mathcal{U}$ is terminating* *(any $\mathcal{U}$-derivation is finite)*

▷ Proof:

P.1 Let $\mu(\mathcal{E}) := \langle m, \mathcal{N}, n \rangle$, where

- ▷ m is the number of unsolved variables in $\mathcal{E}$
- ▷ $\mathcal{N}$ is the multi-set of term depths in $\mathcal{E}$
- ▷ n the number of term pairs in $\mathcal{E}$

P.2 The lexicographic order $\prec$ on triples $\mu(\mathcal{E})$ is decreased by all inference rules. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 61 JACOBS UNIVERSITY

Unification (decidable)

▷ **Definition 94** We call an equational problem $\mathcal{E}$ $\mathcal{U}$-reducible, iff there is a $\mathcal{U}$-step $\mathcal{E} \vdash_{\mathcal{U}} \mathcal{F}$ from $\mathcal{E}$.

▷ **Lemma 95** *If $\mathcal{E}$ is unifiable but not solved, then it is $\mathcal{U}$-reducible*

▷ Proof:

P.1 There is an unsolved pair $\mathbf{A} =^? \mathbf{B}$ in $\mathcal{E} = \mathcal{E}' \wedge \mathbf{A} =^? \mathbf{B}$.

P.2 we have two cases

P.2.1 $\mathbf{A}, \mathbf{B} \notin \mathcal{V}_\iota$:

P.2.1.1 then $\mathbf{A} = f(\mathbf{A}^1 \ldots \mathbf{A}^n)$ and $\mathbf{B} = f(\mathbf{B}^1 \ldots \mathbf{B}^n)$, and thus $\mathcal{U}$dec is applicable □

P.2.2 $\mathbf{A} = X \in \mathbf{free}(\mathcal{E})$:

P.2.2.1 then $\mathcal{U}$elim (if $\mathbf{B} \neq X$) or $\mathcal{U}$triv (if $\mathbf{B} = X$) is applicable. □

□

▷ **Corollary 96** *Unification is decidable in PL^1.*

▷ Proof Idea: $\mathcal{U}$-irreducible set of equations are either solved or unsolvable. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 62 JACOBS UNIVERSITY

Now that we understand basic unification theory, we can come to the meta-theoretical properties of the tableau calculus, which we now discuss to make the understanding of first-order inference complete.

7.4 Soundness and Completeness of First-Order Tableaux

For the soundness result, we recap the definition of soundness for test calculi from the propositional case.

Soundness (Tableau)

▷ Idea: A test calculus is sound, iff it preserves satisfiability and the goal formulae are unsatisfiable.

▷ **Definition 97** A labeled formula $\mathbf{A}^\alpha$ is valid under φ, iff $\mathcal{I}_\varphi(\mathbf{A}) = \alpha$.

▷ **Definition 98** A tableau $\mathcal{T}$ is satisfiable, iff there is a satisfiable branch $\mathcal{P}$ in $\mathcal{T}$, i.e. if the set of formulae in $\mathcal{P}$ is satisfiable.

▷ **Lemma 99** *Tableau rules transform satisfiable tableaux into satisfiable ones.*

▷ **Theorem 100 (Soundness)** *A set Φ of propositional formulae is valid, if there is a closed tableau $\mathcal{T}$ for Φ^F.*

▷ Proof: by contradiction: Suppose Φ is not valid.

P.1 then the initial tableau is satisfiable (Φ^F satisfiable)

P.2 $\mathcal{T}$ satisfiable, by our Lemma.

P.3 there is a satisfiable branch (by definition)

P.4 but all branches are closed ($\mathcal{T}$ closed)

□

SOME RIGHTS RESERVED ©: Michael Kohlhase JACOBS UNIVERSITY

Thus we only have to prove Lemma 99, this is relatively easy to do. For instance for the first rule: if we have a tableau that contains $\mathbf{A} \wedge \mathbf{B}^\mathsf{T}$ and is satisfiable, then it must have a satisfiable branch. If $\mathbf{A} \wedge \mathbf{B}^\mathsf{T}$ is not on this branch, the tableau extension will not change satisfiability, so we can assue that it is on the satisfiable branch and thus $\mathcal{I}_\varphi(\mathbf{A} \wedge \mathbf{B}) = \mathsf{T}$ for some variable assignment φ. Thus $\mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$ and $\mathcal{I}_\varphi(\mathbf{B}) = \mathsf{T}$, so after the extension (which adds the formulae $\mathbf{A}^\mathsf{T}$ and $\mathbf{B}^\mathsf{T}$ to the branch), the branch is still satisfiable. The cases for the other rules are similar.

The soundness of the first-order free-variable tableaux calculus can be established a simple induction over the size of the tableau.

Soundness of $\mathcal{T}_1^f$

▷ **Lemma 101** *Tableau rules transform satisfiable tableaux into satisfiable ones.*

▷ Proof: we examine the tableau rules in turn

P.1.1 propositional rules: as in propositional tableaux

P.1.2 $\mathcal{T}_1^f{:}\exists$: by Lemma 103

P.1.3 $\mathcal{T}_1^f{:}\bot$: by Lemma 22 (substitution value lemma)

P.1.4 $\mathcal{T}_1^f{:}\forall$:

P.1.4.1 $\mathcal{I}_\varphi(\forall X.\mathbf{A}) = \mathsf{T}$, iff $\mathcal{I}_\psi(\mathbf{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$

P.1.4.2 so in particular for some $a \in \mathcal{D}_\iota \neq \emptyset$. □

□

▷ **Corollary 102** $\mathcal{T}_1^f$ *is correct.*

SOME RIGHTS RESERVED ©: Michael Kohlhase JACOBS UNIVERSITY

The only interesting steps are the cut rule, which can be directly handled by the substitution value lemma, and the rule for the existential quantifier, which we do in a separate lemma.

Soundness of $\mathcal{T}_1^f{:}\exists$

▷ **Lemma 103** $\mathcal{T}_1^f{:}\exists$ *transforms satisfiable tableaux into satisfiable ones.*

▷ Proof: Let $\mathcal{T}'$ be obtained by applying $\mathcal{T}_1^f{:}\exists$ to $\forall X.\mathbf{A}^{\mathsf{F}}$ in $\mathcal{T}$, extending it with $[f(X^1,\ldots,X^n)/X](\mathbf{A})^{\mathsf{F}}$, where $W := \mathbf{free}(\forall X.\mathbf{A}) = \{X^1,\ldots,X^k\}$

P.1 Let $\mathcal{T}$ be satisfiable in $\mathcal{M} := \langle\mathcal{D},\mathcal{I}\rangle$, then $\mathcal{I}_\varphi(\forall X.\mathbf{A}) = \mathsf{F}$.

P.2 We need to find a model $\mathcal{M}'$ that satisfies $\mathcal{T}'$ (find interpretation for f)

P.3 By definition $\mathcal{I}_{\varphi,[a/X]}(\mathbf{A}) = \mathsf{F}$ for some $a \in \mathcal{D}$ (depends on $\varphi\big|_W$)

P.4 Let $g\colon \mathcal{D}^k \to \mathcal{D}$ be defined by $g(a_1,\ldots,a_k) := a$, if $\varphi(X^i) = a_i$

P.5 choose $\mathcal{M}' = \langle\mathcal{D},\mathcal{I}'\rangle$ with $\mathcal{I}' := \mathcal{I},[g/f]$, then by subst. value lemma

$$\mathcal{I}'_\varphi([f(X^1,\ldots,X^k)/X](\mathbf{A})) = \mathcal{I}'_{\varphi,[\mathcal{I}'_\varphi(f(X^1,\ldots,X^k))/X]}(\mathbf{A}) = \mathcal{I}'_{\varphi,[a/X]}(\mathbf{A}) = \mathsf{F}$$

P.6 So $[f(X^1,\ldots,X^k)/X](\mathbf{A})^{\mathsf{F}}$ satisfiable in $\mathcal{M}'$ □

SOME RIGHTS RESERVED ©: Michael Kohlhase 65 JACOBS UNIVERSITY

This proof is paradigmatic for soundness proofs for calculi with Skolemization. We use the axiom of choice at the meta-level to choose a meaning for the Skolem function symbol.

Armed with the Model Existence Theorem for first-order logic (Theorem 67), the completeness of first-order tableaux is similarly straightforward. We just have to show that the collection of tableau-irrefutable sentences is an abstract consistency class, which is a simple proof-transformation exercise in all but the universal quantifier case, which we postpone to its own Lemma.

Completeness of $(\mathcal{T}_1^f)$

▷ **Theorem 104** $\mathcal{T}_1^f$ *is refutation complete.*

▷ Proof: We show that $\nabla := \{\Phi \mid \Phi^{\mathsf{T}} \text{ has no closed Tableau}\}$ is an abstract consistency class

P.1 (∇_c, $\nabla_\neg$, $\nabla_\vee$, and $\nabla_\wedge$) as for propositional case.

P.2 ($\nabla_\forall$) by the lifting lemma below

P.3 ($\nabla_\exists$) Let $\mathcal{T}$ be a closed tableau for $\neg(\forall X.\mathbf{A}) \in \Phi$ and $\Phi^{\mathsf{T}} * [c/X](\mathbf{A})^{\mathsf{F}} \in \nabla$.

$$\begin{array}{c} \Psi^{\mathsf{T}} \\ \forall X.\mathbf{A}^{\mathsf{F}} \\ [c/X](\mathbf{A})^{\mathsf{F}} \\ Rest \end{array} \qquad \begin{array}{c} \Psi^{\mathsf{T}} \\ \forall X.\mathbf{A}^{\mathsf{F}} \\ [f(X^1,\ldots,X^k)/X](\mathbf{A})^{\mathsf{F}} \\ [f(X^1,\ldots,X^k)/c](Rest) \end{array}$$

□

SOME RIGHTS RESERVED ©: Michael Kohlhase 66 JACOBS UNIVERSITY

So we only have to treat the case for the universal quantifier. This is what we usually call a "lifting argument", since we have to transform ("lift") a proof for a formula $\theta(\mathbf{A})$ to one for $\mathbf{A}$. In the

case of tableaux we do that by an induction on the tableau refutation for $\theta(\mathbf{A})$ which creates a tableau-isomorphism to a tableau refutation for $\mathbf{A}$.

Tableau-Lifting

▷ **Theorem 105** *If $\mathcal{T}_\theta$ is a closed tableau for a st $\theta(\Phi)$ of formulae, then there is a closed tableau $\mathcal{T}$ for Φ.*

▷ Proof: by induction over the structure of $\mathcal{T}_\theta$ we build an isomorphic tableau $\mathcal{T}$, and a tableau-isomorphism $\omega\colon \mathcal{T} \to \mathcal{T}_\theta$, such that $\omega(\mathbf{A}) = \theta(\mathbf{A})$.

P.1 only the tableau-substitution rule is interesting.

P.2 Let $\theta(\mathbf{A}^i)^\mathsf{T}$ and $\theta(\mathbf{B}^i)^\mathsf{F}$ cut formulae in the branch Θ^i_θ of $\mathcal{T}_\theta$

P.3 there is a joint unifier σ of $\theta(\mathbf{A}^1) =^? \theta(\mathbf{B}^1) \wedge \ldots \wedge \theta(\mathbf{A}^n) =^? \theta(\mathbf{B}^n)$

P.4 thus $\sigma \circ \theta$ is a unifier of $\mathbf{A}$ and $\mathbf{B}$

P.5 hence there is a most general unifier ρ of $\mathbf{A}^1 =^? \mathbf{B}^1 \wedge \ldots \wedge \mathbf{A}^n =^? \mathbf{B}^n$

P.6 so Θ is closed □

SOME RIGHTS RESERVED ©: Michael Kohlhase 67 JACOBS UNIVERSITY

Again, the "lifting lemma for tableaux" is paradigmatic for lifting lemmata for other refutation calculi.

8 Higher-Order Logic and λ-Calculus

In this section we set the stage for a deeper discussions of the logical foundations of mathematics by introducing a particular higher-order logic, which gets around the limitations of first-order logic — the restriction of quantification to individuals. This raises a couple of questions (paradoxes, comprehension, completeness) that have been very influential in the development of the logical systems we know today.

Therefore we use the discussion of higher-order logic as an introduction and motivation for the λ-calculus, which answers most of these questions in a term-level, computation-friendly system.

The formal development of the simply typed λ-calculus and the establishment of its (meta-logical) properties will be the body of work in this section. Once we have that we can reconstruct a clean version of higher-order logic by adding special provisions for propositions.

Higher-Order Predicate Logic

8.1 Higher-Order Predicate Logic

The main motivation for higher-order logic is to allow quantification over classes of objects that are not individuals — because we want to use them as functions or predicates, i.e. apply them to arguments in other parts of the formula.

Higher-Order Predicate Logic (PLΩ)

- ▷ Quantification over functions and Predicates: $\forall P.\exists F.P(a) \vee \neg P(F(a))$
- ▷ Comprehension: (Existence of Functions)
  $\exists F.\forall X.FX = \mathbf{A}$ e.g. $f(x) = 3x^2 + 5x - 7$
- ▷ Extensionality: (Equality of functions and truth values)
  $\forall F.\forall G.(\forall X.FX = GX) \Rightarrow F = G$
  $\forall P.\forall Q.(P \Leftrightarrow Q) \Leftrightarrow P = Q$
- ▷ Leibniz Equality: (Indiscernability)
  $\mathbf{A} = \mathbf{B}$ for $\forall P.P\mathbf{A} \Rightarrow P\mathbf{B}$

SOME RIGHTS RESERVED ©: Michael Kohlhase 68 JACOBS UNIVERSITY

Indeed, if we just remove the restriction on quantification we can write down many things that are essential on everyday mathematics, but cannot be written down in first-order logic. But the naive logic we have created (BTW, this is essentially the logic of Frege [Fre79]) is much too expressive, it allows us to write down completely meaningless things as witnessed by Russell's paradox.

Problems with PLΩ

- ▷ Problem: Russell's Antinomy: $\forall Q.\mathcal{M}(Q) \Leftrightarrow \neg Q(Q)$
  - ▷ the set $\mathcal{M}$ of all sets that do not contain themselves
  - ▷ Question: Is $\mathcal{M} \in \mathcal{M}$? Answer: $\mathcal{M} \in \mathcal{M}$ iff $\mathcal{M} \notin \mathcal{M}$.
- ▷ What has happened? the predicate Q has been applied to itself
- ▷ Solution for this course: Forbid self-applications by types!!
  - ▷ ι, o (type of individuals, truth values), $\alpha \to \beta$ (function type)
  - ▷ right associative bracketing: $\alpha \to \beta \to \gamma$ abbreviates $\alpha \to (\beta \to \gamma)$
  - ▷ vector notation: $\overline{\alpha_n} \to \beta$ abbreviates $\alpha_1 \to \ldots \to \alpha_n \to \beta$
- ▷ Well-typed formulae (prohibits paradoxes like $\forall Q.\mathcal{M}(Q) \Leftrightarrow \neg Q(Q)$)
- ▷ Other solution: Give it a non-standard semantics (Domain-Theory [Scott])

SOME RIGHTS RESERVED ©: Michael Kohlhase 69 JACOBS UNIVERSITY

The solution to this problem turns out to be relatively simple with the benefit of hindsight: we just introduce a syntactic device that prevents us from writing down paradoxical formulae. This idea was first introduced by Russell and Whitehead in their Principia Mathematica [WR10].

Their system of "ramified types" was later radically simplified by Alonzo Church to the form we use here in [Chu40]. One of the simplifications is the restriction to unary functions that is made possible by the fact that we can re-interpret binary functions as unary ones using a technique called "Currying" after the Logician Haskell Brooks Curry ($*$(1900), †(1982)). Of course we can extend this to higher arities as well. So in theory we can consider n-ary functions as syntactic sugar for suitable higher-order functions. The vector notation for types defined above supports this intuition.

Types

- ▷ Types are semantic annotations for terms that prevent antinomies
- ▷ **Definition 106** Given a set $\mathcal{BT}$ of base types, construct function types: $\alpha \to \beta$ is the type of functions with domain type α and range type β. We call the closure $\mathcal{T}$ of $\mathcal{BT}$ under function types the set of types over $\mathcal{BT}$.
- ▷ **Definition 107** We will use ι for the type of individuals and o for the type of truth values.
- ▷ The type constructor is used as a right-associative operator, i.e. we use $\alpha \to \beta \to \gamma$ as an abbreviation for $\alpha \to (\beta \to \gamma)$
- ▷ We will use a kind of vector notation for function types, abbreviating $\alpha_1 \to \ldots \to \alpha_n \to \beta$ with $\overline{\alpha_n} \to \beta$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 70 JACOBS UNIVERSITY

Armed with a system of types, we can now define a typed higher-order logic, by insisting that all formulae of this logic be well-typed. One advantage of typed logics is that the natural classes of objects that have otherwise to be syntactically kept apart in the definition of the logic (e.g. the term and proposition levels in first-order logic), can now be distinguished by their type, leading to a much simpler exposition of the logic. Another advantage is that concepts like connectives that were at the language level e.g. in PL^0, can be formalized as constants in the signature, which again makes the exposition of the logic more flexible and regular. We only have to treat the quantifiers at the language level (for the moment).

Well-Typed Formulae (PLΩ)

- ▷ signature $\Sigma = \bigcup_{\alpha \in \mathcal{T}} \Sigma_\alpha$ with
- ▷ connectives: $\neg \in \Sigma_{o \to o}$ $\{\vee, \wedge, \Rightarrow, \Leftrightarrow \ldots\} \subseteq \Sigma_{o \to o \to o}$
- ▷ variables $\mathcal{V}_\mathcal{T} = \bigcup_{\alpha \in \mathcal{T}} \mathcal{V}_\alpha$, such that every $\mathcal{V}_\alpha$ countably infinite.
- ▷ well-typed formulae $wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ of type α
  - ▷ $\mathcal{V}_\alpha \cup \Sigma_\alpha \subseteq wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$
  - ▷ If $\mathbf{C} \in wff_{(\alpha \to \beta)}(\Sigma, \mathcal{V}_\mathcal{T})$ and $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$, then $(\mathbf{CA}) \in wff_\beta(\Sigma, \mathcal{V}_\mathcal{T})$
  - ▷ If $\mathbf{A} \in wff_o(\Sigma, \mathcal{V}_\mathcal{T})$, then $(\forall X_\alpha.\mathbf{A}) \in wff_o(\Sigma, \mathcal{V}_\mathcal{T})$
- ▷ first-order terms have type ι, formulae (propositions) the type o.
- ▷ there is no type annotation such that $\forall Q.\mathcal{M}(Q) \Leftrightarrow \neg Q(Q)$ is well-typed. Q needs type α as well as $\alpha \to o$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 71 JACOBS UNIVERSITY

The semantics is similarly regular: We have universes for every type, and all functions are "typed functions", i.e. they respect the types of objects. Other than that, the setup is very similar to what we already know.

Standard Semantics

▷ **Definition 108** The universe of discourse (carrier)

- ▷ arbitrary, non-empty set of individuals $\mathcal{D}_\iota$
- ▷ fixed set of truth values $\mathcal{D}_o = \{\mathsf{T}, \mathsf{F}\}$
- ▷ function universes $\mathcal{D}_{\alpha\to\beta} = \mathcal{F}(\mathcal{D}_\alpha; \mathcal{D}_\beta)$

▷ interpretation of constants: typed mapping $\mathcal{I}\colon \Sigma \to \mathcal{D}$ (i.e. $\mathcal{I}(\Sigma_\alpha) \subseteq \mathcal{D}_\alpha$)

▷ variable assignment: typed mapping $\varphi\colon \mathcal{V}_\mathcal{T} \to \mathcal{D}$

▷ **Definition 109** value function: typed mapping $\mathcal{I}_\varphi\colon wff_\mathcal{T}(\Sigma, \mathcal{V}_\mathcal{T}) \to \mathcal{D}$

- ▷ $\mathcal{I}_\varphi\big|_{\mathcal{V}_\mathcal{T}} = \varphi \qquad \mathcal{I}_\varphi\big|_{\Sigma_\mathcal{T}} = \mathcal{I}$
- ▷ $\mathcal{I}_\varphi(\mathbf{AB}) = \mathcal{I}_\varphi(\mathbf{A})(\mathcal{I}_\varphi(\mathbf{B}))$
- ▷ $\mathcal{I}_\varphi(\forall X_\alpha.\mathbf{A}) = \mathsf{T}$, iff $\mathcal{I}_{\varphi,[\mathsf{a}/X]}(\mathbf{A}) = \mathsf{T}$ for all $\mathsf{a} \in \mathcal{D}_\alpha$.

▷ $\mathbf{A}_o$ valid under φ, iff $\mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 72 JACOBS UNIVERSITY

We now go through a couple of examples of what we can express in PLΩ, and that works out very straightforwardly. For instance, we can express equality in PLΩ by Leibniz equality, and it has the right meaning.

Equality

▷ "Leibniz equality" (Indiscernability) $\mathbf{Q}^\alpha\mathbf{A}_\alpha\mathbf{B}_\alpha = \forall P_{\alpha\to o}.P\mathbf{A} \Leftrightarrow P\mathbf{B}$

▷ not that $\forall P_{\alpha\to o}.P\mathbf{A} \Rightarrow P\mathbf{B}$(get the other direction by instantiating P with Q, where $QX \Leftrightarrow \neg PX$)

▷ **Theorem 110** *If $\mathcal{M} = \langle\mathcal{D}, \mathcal{I}\rangle$ is a standard model, then $\mathcal{I}_\varphi(\mathbf{Q}^\alpha)$ is the identity relation on $\mathcal{D}_\alpha$.*

▷ **Notation 111** We write $\mathbf{A} = \mathbf{B}$ for $\mathbf{QAB}$ ($\mathbf{A}$ and $\mathbf{B}$ are equal, iff there is no property P that can tell them apart.)

▷ Proof:

P.1 $\mathcal{I}_\varphi(\mathbf{QAB}) = \mathcal{I}_\varphi(\forall P.P\mathbf{A} \Rightarrow P\mathbf{B}) = \mathsf{T}$, iff $\mathcal{I}_{\varphi,[r/P]}(PX \Rightarrow PY) = \mathsf{T}$ for all $r \in \mathcal{D}_{\alpha\to o}$.

P.2 For $\mathbf{A} = \mathbf{B}$ we have $\mathcal{I}_{\varphi,[r/P]}(P\mathbf{A}) = r(\mathcal{I}_\varphi(\mathbf{A})) = \mathsf{F}$ or $\mathcal{I}_{\varphi,[r/P]}(P\mathbf{A}) = r(\mathcal{I}_\varphi(\mathbf{A})) = \mathsf{T}$.

P.3 Thus $\mathcal{I}_\varphi(\mathbf{QAB}) = \mathsf{T}$.

P.4 Let $\mathcal{I}_\varphi(\mathbf{A}) \neq \mathcal{I}_\varphi(\mathbf{B})$ and $r = \{\mathcal{I}_\varphi(\mathbf{A})\}$

P.5 so $r(\mathcal{I}_\varphi(\mathbf{A})) = \mathsf{T}$ and $r(\mathcal{I}_\varphi(\mathbf{B})) = \mathsf{F}$

P.6 $\mathcal{I}_\varphi(\mathbf{QAB}) = \mathsf{F}$, as $\mathcal{I}_{\varphi,[r/P]}(P\mathbf{A} \Rightarrow P\mathbf{B}) = \mathsf{F}$, since $\mathcal{I}_{\varphi,[r/P]}(P\mathbf{A}) = r(\mathcal{I}_\varphi(\mathbf{A})) = \mathsf{T}$ and $\mathcal{I}_{\varphi,[r/P]}(P\mathbf{B}) = r(\mathcal{I}_\varphi(\mathbf{B})) = \mathsf{F}$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 73 JACOBS UNIVERSITY

Another example are the Peano Axioms for the natural numbers, though we omit the proofs of adequacy of the axiomatization here.

Example: Peano Axioms for the Natural Numbers

- ▷ $\Sigma = \{[\mathbb{N}\colon \iota \to o], [0\colon \iota], [s\colon \iota \to \iota]\}$
- ▷ $\mathbb{N}0$ (0 is a natural number)
- ▷ $\forall X_\iota.\mathbb{N}X \Rightarrow \mathbb{N}(sX)$ (the successor of a natural number is natural)
- ▷ $\neg(\exists X_\iota.\mathbb{N}X \wedge sX = 0)$ (0 has no predecessor)
- ▷ $\forall X_\iota.\forall Y_\iota.(sX = sY) \Rightarrow X = Y$ (the successor function is injective)
- ▷ $\forall P_{\iota\to o}.P0 \Rightarrow (\forall X_\iota.\mathbb{N}X \Rightarrow PX \Rightarrow P(sX)) \Rightarrow (\forall Y_\iota.\mathbb{N}Y \Rightarrow P(Y))$
  induction axiom: all properties P, that hold of 0, and with every n for its successor $s(n)$, hold on all $\mathbb{N}$

SOME RIGHTS RESERVED ©: Michael Kohlhase 74 JACOBS UNIVERSITY

Finally, we show the expressivity of PLΩ by formalizing a version of Cantor's theorem.

Expressive Formalism for Mathematics

- ▷ **Example 112 (Cantor's Theorem)** The cardinality of a set is smaller than that of its power set.
  - ▷ smaller-card$(M, N) := \neg(\exists F.\text{surjective}(F, M, N))$
  - ▷ surjective$(F, M, N) := \forall X \in M.\exists Y \in N.FY = X$

  Simplified Formalization: $\neg\exists F_{\iota\to\iota\to\iota}.\forall G_{\iota\to\iota}.\exists J_\iota.FJ = G$
- ▷ Standard-Benchmark for higher-order theorem provers
- ▷ can be proven by TPS and LEO (see below)

SOME RIGHTS RESERVED ©: Michael Kohlhase 75 JACOBS UNIVERSITY

The next concern is to find a calculus for PLΩ.

We start out with the simplest one we can imagine, a Hilbert-style calculus that has been adapted to higher-order logic by letting the inference rules range over PLΩ formulae and insisting that substitutions are well-typed.

Hilbert-Calculus

- ▷ **Definition 113 ($\mathcal{H}_\Omega$ Axioms)** ▷ $\forall P_o, Q_o.P \Rightarrow Q \Rightarrow P$
  - ▷ $\forall P_o, Q_o, R_o.(P \Rightarrow Q \Rightarrow R) \Rightarrow (P \Rightarrow Q) \Rightarrow P \Rightarrow R$
  - ▷ $\forall P_o, Q_o.(\neg P \Rightarrow \neg Q) \Rightarrow P \Rightarrow Q$
- ▷ **Definition 114 ($\mathcal{H}_\Omega$ Inference rules)**

$$\frac{\mathbf{A}_o \Rightarrow \mathbf{B}_o \quad \mathbf{A}}{\mathbf{B}} \qquad \frac{\forall X_\alpha.\mathbf{A}}{[\mathbf{B}/X_\alpha](\mathbf{A})} \qquad \frac{\mathbf{A}}{\forall X_\alpha.\mathbf{A}} \qquad \frac{X \notin \mathbf{free}(\mathbf{A}) \quad \forall X_\alpha.\mathbf{A} \wedge \mathbf{B}}{\mathbf{A} \wedge (\forall X_\alpha.\mathbf{B})}$$

- ▷ **Theorem 115** *Sound, wrt. standard semantics*
- ▷ Also Complete?

SOME RIGHTS RESERVED ©: Michael Kohlhase 76 JACOBS UNIVERSITY

Not surprisingly, $\mathcal{H}_\Omega$ is sound, but it shows big problems with completeness. For instance, if we turn to a proof of Cantor's theorem via the well-known diagonal sequence argument, we will have to construct the diagonal sequence as a function of type $\iota \to \iota$, but up to now, we cannot in $\mathcal{H}_\Omega$. Unlike mathematical practice, which silently assumes that all functions we can write down in closed form exists, in logic, we have to have an axiom that guarantees (the existence of) such a function: the comprehension axioms.

Hilbert-Calculus $\mathcal{H}_\Omega$ (continued)

- ▷ valid sentences that are not $\mathcal{H}_\Omega$-theorems:
  - ▷ Cantor's Theorem:
    $\neg(\exists F_{\iota\to\iota\to\iota}.\forall G_{\iota\to\iota}.(\forall K_\iota.(\mathbb{N}K) \Rightarrow (\mathbb{N}(GK))) \Rightarrow (\exists J_\iota.(\mathbb{N}J) \wedge FJ = G))$
    (There is no surjective mapping from $\mathbb{N}$ into the set $\mathcal{F}(\mathbb{N};,)\mathbb{N}$ of natural number sequences)
  - ▷ proof attempt fails at the subgoal $\exists G_{\iota\to\iota}.\forall X_\iota.GX = s(fXX)$
- ▷ Comprehension $\exists F_{\alpha\to\beta}.\forall X_\alpha.FX = \mathbf{A}_\beta$ (for every variable X_α and every term $\mathbf{A} \in wff_\beta(\Sigma, \mathcal{V}_\mathcal{T})$)
- ▷ extensionality
  $\mathbf{Ext}^{\alpha\beta}$ $\forall F_{\alpha\to\beta}.\forall G_{\alpha\to\beta}.(\forall X_\alpha.FX = GX) \Rightarrow F = G$
  $\mathbf{Ext}^{\mathbf{o}}$ $\forall F_o.\forall G_o.(F \Leftrightarrow G) \Leftrightarrow F = G$
- ▷ correct! complete? cannot be!! [Göd31]

SOME RIGHTS RESERVED ©: Michael Kohlhase 77 JACOBS UNIVERSITY

Actually it turns out that we need more axioms to prove elementary facts about mathematics: the extensionality axioms. But even with those, the calculus cannot be complete, even though empirically it proves all mathematical facts we are interested in.

Way Out: Henkin-Semantics

- ▷ Gödel's incompleteness theorem only holds for standard semantics
- ▷ find generalization that admits complete calculi:
- ▷ Idea: generalize so that the carrier only contains those functions that are requested by the comprehension axioms.
- ▷ **Theorem 116 (Henkin 1950)** *$\mathcal{H}_\Omega$ is complete wrt. this semantics.*
- ▷ Proof: Idea
  more models $\rightsquigarrow$ less valid sentences (these are $\mathcal{H}_\Omega$-theorems)
  □
- ▷ Henkin-models induce sensible measure of completeness for higher-order logic.

SOME RIGHTS RESERVED ©: Michael Kohlhase 78 JACOBS UNIVERSITY

Actually, there is another problem with PLΩ: The comprehension axioms are computationally very problematic. First, we observe that they are equality axioms, and thus are needed to show that two objects of PLΩ are equal. Second we observe that there are countably infinitely many of them (they are parametric in the term $\mathbf{A}$, the type α and the variable name), which makes dealing

with them difficult in practice. Finally, axioms with both existential and universal quantifiers are always difficul to reason with.

Therefore we would like to have a formulation of higher-order logic without comprehension axioms. In the next slide we take a close look at the comprehension axioms and transform them into a form without quantifiers, which will turn out useful.

From Comprehension to β-Conversion

- ▷ $\exists F_{\alpha\to\beta}.\forall X_\alpha.FX = \mathbf{A}_\beta$ for arbitrary variable X_α and term $\mathbf{A} \in \mathit{wff}_\beta(\Sigma, \mathcal{V}_\mathcal{T})$ (for each term $\mathbf{A}$ and each variable X there is a function $f \in \mathcal{D}_{\alpha\to\beta}$, with $f(\varphi(X)) = \mathcal{I}_\varphi(\mathbf{A})$)
  - ▷ schematic in α, β, X_α and $\mathbf{A}_\beta$, very inconvenient for deduction
- ▷ Transformation in $\mathcal{H}_\Omega$
  - ▷ $\exists F_{\alpha\to\beta}.\forall X_\alpha.FX = \mathbf{A}_\beta$
  - ▷ $\forall X_\alpha.(\lambda X_\alpha.\mathbf{A})X = \mathbf{A}_\beta$ ($\exists E$)
    Call the function F whose existence is guaranteed "$(\lambda X_\alpha.\mathbf{A})$"
  - ▷ $(\lambda X_\alpha.\mathbf{A})\mathbf{B} = [\mathbf{B}/X]\mathbf{A}_\beta$ ($\forall E$), in particular for $\mathbf{B} \in \mathit{wff}_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$.
- ▷ **Definition 117** Axiom of β-equality: $(\lambda X_\alpha.\mathbf{A})\mathbf{B} = [\mathbf{B}/X](\mathbf{A}_\beta)$
- ▷ new formulae (λ-calculus [Church 1940])

SOME RIGHTS RESERVED ©: Michael Kohlhase 79 JACOBS UNIVERSITY

In a similar way we can treat (functional) extensionality.

From Extensionality to η-Conversion

- ▷ **Definition 118** Extensionality Axiom: $\forall F_{\alpha\to\beta}.\forall G_{\alpha\to\beta}.(\forall X_\alpha.FX = GX) \Rightarrow F = G$
- ▷ Idea: Maybe we can get by with a simplified equality schema here as well.
- ▷ **Definition 119** We say that $\mathbf{A}$ and $\lambda X_\alpha.\mathbf{A}X$ are η-equal, (write $\mathbf{A}_{\alpha\to\beta} =_\eta \lambda X_\alpha.\mathbf{A}X$, if), iff $X \notin \mathbf{free}(\mathbf{A})$.
- ▷ **Theorem 120** *η-equality and Extensionality are equivalent*
- ▷ Proof: We show that η-equality is special case of extensionality; the converse entailment is trivial

  P.1 Let $\forall X_\alpha.\mathbf{A}X = \mathbf{B}X$, thus $\mathbf{A}X = \mathbf{B}X$ with $\forall E$

  P.2 $\lambda X_\alpha.\mathbf{A}X = \lambda X_\alpha.\mathbf{B}X$, therefore $\mathbf{A} = \mathbf{B}$ with η

  P.3 Hence $\forall F_o.\forall G_o.(F \Leftrightarrow G) \Leftrightarrow F = G$ □
- ▷ Axiom of truth values: $\forall F_o.\forall G_o.(F \Leftrightarrow G) \Leftrightarrow F = G$ unsolved.

SOME RIGHTS RESERVED ©: Michael Kohlhase 80 JACOBS UNIVERSITY

The price to pay is that we need to pay for getting rid of the comprehension and extensionality axioms is that we need a logic that systematically includes the λ-generated names we used in the transformation as (generic) witnesses for the existential quantifier. Alonzo Church did just that with his "simply typed λ-calculus" which we will introduce next.

8.2 Simply Typed λ-Calculus

In this section we will present a logic that can deal with functions – the simply typed λ-calculus. It is a typed logic, so everything we write down is typed (even if we do not always write the types down).

Simply typed λ-Calculus (Syntax)

- ▷ Signature $\Sigma = \bigcup_{\alpha\in\mathcal{T}} \Sigma_\alpha$
- ▷ $\mathcal{V}_\mathcal{T} = \bigcup_{\alpha\in\mathcal{T}} \mathcal{V}_\alpha$, such that $\mathcal{V}_\alpha$ are countably infinite
- ▷ **Definition 121** We call the set $wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ defined by the rules
  - ▷ $\mathcal{V}_\alpha \cup \Sigma_\alpha \subseteq wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$
  - ▷ If $\mathbf{C} \in wff_{(\alpha\to\beta)}(\Sigma, \mathcal{V}_\mathcal{T})$ and $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$, then $(\mathbf{CA}) \in wff_\beta(\Sigma, \mathcal{V}_\mathcal{T})$
  - ▷ If $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$, then $(\lambda X_\beta.\mathbf{A}) \in wff_{(\beta\to\alpha)}(\Sigma, \mathcal{V}_\mathcal{T})$

  the set of well-typed formulae of type α over the signature Σ and use $wff_\mathcal{T}(\Sigma, \mathcal{V}_\mathcal{T}) := \bigcup_{\alpha\in\mathcal{T}} wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ for the set of all well-typed formulae.
- ▷ **Definition 122** We will call all occurrences of the variable X in $\mathbf{A}$ bound in $\lambda X.\mathbf{A}$. Variables that are not bound in $\mathbf{B}$ are called free in $\mathbf{B}$.
- ▷ Substitutions are well-typed, i.e. $\sigma(X_\alpha) \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ and capture-avoiding.
- ▷ **Definition 123 (Simply Typed λ-Calculus)** The simply typed λ-calculus $\Lambda^\to$ over a signature Σ has the formulae $wff_\mathcal{T}(\Sigma, \mathcal{V}_\mathcal{T})$ (they are called λ-terms) and the following equalities:
  - ▷ α conversion: $\lambda X.\mathbf{A} =_\alpha \lambda Y.[Y/X](\mathbf{A})$
  - ▷ β conversion: $(\lambda X.\mathbf{A})\mathbf{B} =_\beta [\mathbf{B}/X](\mathbf{A})$
  - ▷ η conversion: $\lambda X.\mathbf{A}X =_\eta \mathbf{A}$

SOME RIGHTS RESERVED ©: Michael Kohlhase 81 JACOBS UNIVERSITY

The intuitions about functional structure of λ-terms and about free and bound variables are encoded into three transformation rules $\Lambda^\to$: The first rule (α-conversion) just says that we can rename bound variables as we like. β-conversion codifies the intuition behind function application by replacing bound variables with argument. The equality relation induced by the η-reduction is a special case of the extensionality principle for functions ($f = g$ iff $f(a) = g(a)$ for all possible arguments a): If we apply both sides of the transformation to the same argument – say $\mathbf{B}$ and then we arrive at the right hand side, since $\lambda X_\alpha.\mathbf{A}X\mathbf{B} =_\beta \mathbf{AB}$.

We will use a set of bracket elision rules that make the syntax of $\Lambda^\to$ more palatable. This makes $\Lambda^\to$ expressions look much more like regular mathematical notation, but hides the internal structure. Readers should make sure that they can always reconstruct the brackets to make sense of the syntactic notions below.

Simply typed λ-Calculus (Notations)

▷ **Notation 124 (Application is left-associative)** We abbreviate $(((\mathbf{F}\mathbf{A}^1)\mathbf{A}^2)\ldots)\mathbf{A}^n$ with $\mathbf{F}\mathbf{A}^1\ldots\mathbf{A}^n$ eliding the brackets and further with $\mathbf{F}\overline{\mathbf{A}^n}$ in a kind of vector notation.

▷ A . stands for a left bracket whose partner is as far right as is consistent with existing brackets; i.e. $\mathbf{A}.(\mathbf{B}\mathbf{C})$ abbreviates $\mathbf{A}(\mathbf{B}\mathbf{C})$.

▷ **Notation 125 (Abstraction is right-associative)** We abbreviate $\lambda X^1.\lambda X^2.\cdots\lambda X^n.\mathbf{A}\cdots$ with $\lambda X^1\ldots X^n.\mathbf{A}$ eliding brackets, and further to $\lambda\overline{X^n}.\mathbf{A}$ in a kind of vector notation.

▷ **Notation 126 (Outer brackets)** Finally, we allow ourselves to elide outer brackets where they can be inferred.

SOME RIGHTS RESERVED ©: Michael Kohlhase 82 JACOBS UNIVERSITY

Intuitively, $\lambda X.\mathbf{A}$ is the function f, such that $f(\mathbf{B})$ will yield $\mathbf{A}$, where all occurrences of the formal parameter X are replaced by $\mathbf{B}$.[7] EdNote:7

In this presentation of the simply typed λ-calculus we build-in α-equality and use capture-avoiding substitutions directly. A clean introduction would followed the steps in Section 5 by introducing substitutions with a substitutability condition like the one in Definition 20, then establishing the soundness of α conversion, and only then postulating defining capture-avoiding substitution application as in Definition 25. The development for $\Lambda^{\rightarrow}$ is directly parallel to the one for PL^1, so we leave it as an exercise to the reader and turn to the computational properties of the λ-calculus.

Computationally, the λ-calculus obtains much of its power from the fact that two of its three equalities can be oriented into a reduction system. Intuitively, we only use the equalities in one direction, i.e. in one that makes the terms "simpler". If this terminates (and is confluent), then we can establish equality of two λ-terms by reducing them to normal forms and comparing them structurally. This gives us a decision procedure for equality. Indeed, we have these properties in $\Lambda^{\rightarrow}$ as we will see below.

$\alpha\beta\eta$-Equality (Overview)

▷ reduction with $\begin{cases} \beta : & (\lambda X.\mathbf{A})\mathbf{B} \rightarrow_\beta [\mathbf{B}/X](\mathbf{A}) \\ \eta : & \lambda X.\mathbf{A}X \rightarrow_\eta \mathbf{A} \end{cases}$ under $=_\alpha$: $\begin{array}{c} \lambda X.\mathbf{A} \\ =_\alpha \\ \lambda Y.[Y/X](\mathbf{A}) \end{array}$

▷ **Theorem 127** *$\beta\eta$-reduction is well-typed, terminating and confluent in the presence of $=_\alpha$-conversion.*

▷ **Definition 128 (Normal Form)** We call a λ-term $\mathbf{A}$ a normal form (in a reduction system $\mathcal{E}$), iff no rule (from $\mathcal{E}$) can be applied to $\mathbf{A}$.

▷ **Corollary 129** *$\beta\eta$-reduction yields unique normal forms (up to α-equivalence).*

SOME RIGHTS RESERVED ©: Michael Kohlhase 83 JACOBS UNIVERSITY

We will now introduce some terminology to be able to talk about λ-terms and their parts.

[7] EdNote: rationalize the semantic macros for syntax!

Syntactic Parts of λ-Terms

▷ **Definition 130 (Parts of λ-Terms)** We can always write a λ-term in the form $\mathbf{T} = \lambda X^1 \ldots X^k.\mathbf{H}\mathbf{A}^1 \ldots \mathbf{A}^n$, where $\mathbf{H}$ is not an application. We call

- ▷ $\mathbf{H}$ the syntactic head of $\mathbf{T}$
- ▷ $h\mathbf{A}^1 \ldots \mathbf{A}^n$ the matrix of $\mathbf{T}$, and
- ▷ $\lambda X^1 \ldots X^k.$ the binder of $\mathbf{T}$

▷ **Definition 131** Head Reduction always has a unique β redex

$$\lambda\overline{X^n}.(\lambda Y.\mathbf{A})\mathbf{B}^1 \ldots \mathbf{B}^n \rightarrow^h_\beta \lambda\overline{X^n}.[\mathbf{B}^1/Y](\mathbf{A})\mathbf{B}^2 \ldots \mathbf{B}^n$$

▷ **Theorem 132** *The syntactic heads of β-normal forms are constant or variables.*

▷ **Definition 133** Let $\mathbf{A}$ be a λ-term, then the syntactic head of the β-normal form of $\mathbf{A}$ is called the head symbol of $\mathbf{A}$. We call a λ-term a j-projection, iff its head is the jth bound variable.

▷ **Definition 134** We call a λ-term a η-long form, iff its matrix has base type.

▷ **Definition 135** η-Expansion: $\eta\big[\lambda X^1 \ldots X^n.\mathbf{A}\big] := \lambda X^1 \ldots X^n Y^1 \ldots Y^m.\mathbf{A}Y^1 \ldots Y^m$

▷ **Definition 136** Long $\beta\eta$-normal form , iff it is β-normal and η-long.

SOME RIGHTS RESERVED ©: Michael Kohlhase 84 JACOBS UNIVERSITY

η long forms are structurally convenient since for them, the structure of the term is isomorphic to the structure of its type (argument types correspond to binders): if we have a term $\mathbf{A}$ of type $\overline{\alpha_n} \rightarrow \beta$ in η-long form, where $\beta \in \mathcal{BT}$, then $\mathbf{A}$ must be of the form $\lambda\overline{X^n_\alpha}.\mathbf{B}$, where $\mathbf{B}$ has type β. Furthermore, the set of η-long forms is closed under β-equality, which allows us to treat the two equality theories of $\Lambda^\rightarrow$ separately and thus reduce argumentational complexity.

8.3 Computational Properties of λ-Calculus

As we have seen above, the main contribution of the λ-calculus is that it casts the comprehension and (functional) extensionality axioms in a way that is more amenable to automation in reasoning systems, since they can be oriented into a confluent and terminating reduction system. In this subsection we prove the respective properties. We start out with termination, since we will need it later in the proof of confluence.

8.3.1 Termination of β-reduction

We will use the termination of β reduction to present a very powerful proof method, called the "logical relations method", which is one of the basic proof methods in the repertoire of a proof theorist, since it can be extended to many situations, where other proof methods have no chance of succeeding.

Before we start into the termination proof, we convince ourselves that a straightforward induction over the structure of expressions will not work, and we need something more powerful.

Termination of β-Reduction

- ▷ only holds for the typed case
  $(\lambda X.XX)(\lambda X.XX) \rightarrow_\beta (\lambda X.XX)(\lambda X.XX)$
- ▷ **Theorem 137 (Typed β-Reduction terminates)** *For all* $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$*, the chain of reductions from* $\mathbf{A}$ *is finite.*
- ▷ proof attempts:
  - ▷ Induction on the structure $\mathbf{A}$ must fail, since this would also work for the untyped case.
  - ▷ Induction on the type of $\mathbf{A}$ must fail, since β-reduction conserves types.
- ▷ combined induction on both: Logical Relations [Tait 1967]

SOME RIGHTS RESERVED ©: Michael Kohlhase 85 JACOBS UNIVERSITY

The overall shape of the proof is that we reason about two relations: $\mathcal{SR}$ and $\mathcal{LR}$ between λ-terms and their types. The first is the one that we are interested in, $\mathcal{LR}(\mathbf{A}, \alpha)$ essentially states the property that $\beta\eta$ reduction terminates at $\mathbf{A}$. Whenever the proof needs to argue by induction on types it uses the "logical relation" $\mathcal{LR}$, which is more "semantic" in flavor. It coincides with $\mathcal{SR}$ on base types, but is defined via a functionality property".

Relations $\mathcal{SR}$ and $\mathcal{LR}$

- ▷ **Definition 138** $\mathbf{A}$ is called strongly reducing at type α (write $\mathcal{SR}(\mathbf{A}, \alpha)$), iff each chain β-reductions from $\mathbf{A}$ terminates.
- ▷ We define a logical relation $\mathcal{LR}$ inductively on the structure of the type
  - ▷ α base type: $\mathcal{LR}(\mathbf{A}, \alpha)$, iff $\mathcal{SR}(\mathbf{A}, \alpha)$
  - ▷ $\mathcal{LR}(\mathbf{C}, \alpha \rightarrow \beta)$, iff $\mathcal{LR}(\mathbf{CA}, \beta)$ for all $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ with $\mathcal{LR}(\mathbf{A}, \alpha)$.
- ▷ Proof: Termination Proof

  P.1 $\mathcal{LR} \subseteq \mathcal{SR}$ (Lemma 140 b))

  P.2 $\mathbf{A} \in wff_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ implies $\mathcal{LR}(\mathbf{A}, \alpha)$ (Theorem 142 with $\sigma = \emptyset$)

  P.3 also $\mathcal{SR}(\mathbf{A}, \alpha)$ □
- ▷ **Lemma 139 ($\mathcal{SR}$ is closed under subterms)** *If* $\mathcal{SR}(\mathbf{A}, \alpha)$ *and* $\mathbf{B}_\beta$ *is a subterm of* $\mathbf{A}$*, then* $\mathcal{SR}(\mathbf{B}, \beta)$.
- ▷ Proof Idea: Every infinite β-reduction from $\mathbf{B}$ would be one from $\mathbf{A}$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 86 JACOBS UNIVERSITY

The termination proof proceeds in two steps, the first one shows that $\mathcal{LR}$ is a sub-relation of $\mathcal{SR}$, and the second that $\mathcal{LR}$ is total on λ-terms. Togther the give the termination result.

The next result proves two important technical side results for the termination proofs in a joint induction over the structure of the types involved. The name "rollercoaster lemma" alludes to the fact that the argument starts with base type, where things are simple, and iterates through the two parts each leveraging the proof of the other to higher and higher types.

$\mathcal{LR} \subseteq \mathcal{SR}$ (Rollercoaster Lemma)

▷ **Lemma 140 (Rollercoaster Lemma)**

a) If h is a constant or variable of type $\overline{\alpha_n} \to \alpha$ and $\mathcal{SR}(\mathbf{A}^i, \alpha^i)$, then $\mathcal{LR}(h\overline{\mathbf{A}^n}, \alpha)$.

b) $\mathcal{LR}(\mathbf{A}, \alpha)$ implies $\mathcal{SR}(\mathbf{A}, \alpha)$.

▷ Proof: we prove both assertions by simultaneous induction on α

P.1.1 α base type:

P.1.1.1.1 a): $h\overline{\mathbf{A}^n}$ is strongly reducing, since the $\mathbf{A}^i$ are (brackets!)

P.1.1.1.1.2 so $\mathcal{LR}(h\overline{\mathbf{A}^n}, \alpha)$ as α is a base type ($\mathcal{SR} = \mathcal{LR}$) □

P.1.1.1.2 b): by definition

P.1.2 $\alpha = \beta \to \gamma$:

P.1.2.1.1 a): Let $\mathcal{LR}(\mathbf{B}, \beta)$.

P.1.2.1.1.2 by IH b) we have $\mathcal{SR}(\mathbf{B}, \beta)$, and $\mathcal{LR}((h\overline{\mathbf{A}^n})\mathbf{B}, \gamma)$ by IH a)

P.1.2.1.1.3 so $\mathcal{LR}(h\overline{\mathbf{A}^n}, \beta)$ by definition. □

P.1.2.1.2 b): Let $\mathcal{LR}(\mathbf{A}, \alpha)$ and $X_\beta \notin \mathbf{free}(\mathbf{A})$.

P.1.2.1.2.2 $\mathcal{LR}(X, \beta)$ by IH a) with $n = 0$, thus $\mathcal{LR}(\mathbf{A}X, \gamma)$ by definition.

P.1.2.1.2.3 By IH b) we have $\mathcal{SR}(\mathbf{A}X, \gamma)$ and by Lemma 139 $\mathcal{SR}(\mathbf{A}, \alpha)$. □

□

□

SOME RIGHTS RESERVED

The part of the rollercoaster lemma we are really interested in is part b). But part a) will become very important for the case where $n = 0$; here it states that constants and variables are $\mathcal{LR}$.

$\mathbf{A} \in \mathit{wff}_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$ implies $\mathcal{LR}(\mathbf{A}, \alpha)$

▷ **Definition 141** We write $\mathcal{LR}(\sigma)$ if $\mathcal{LR}(\sigma(X_\alpha), \alpha)$ for all $X \in \mathbf{supp}(\sigma)$.

▷ **Theorem 142** *If $\mathbf{A} \in \mathit{wff}_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$, then $\mathcal{LR}(\sigma(\mathbf{A}), \alpha)$ for any substitution σ with $\mathcal{LR}(\sigma)$.*

▷ Proof: by induction on the structure of $\mathbf{A}$

P.1.1 $\mathbf{A} = X_\alpha \in \mathbf{supp}(\sigma)$: then $\mathcal{LR}(\sigma(\mathbf{A}), \alpha)$ by assumption

P.1.2 $\mathbf{A} = X \notin \mathbf{supp}(\sigma)$: then $\sigma(\mathbf{A}) = \mathbf{A}$ and $\mathcal{LR}(\mathbf{A}, \alpha)$ by Lemma 140 with $n = 0$.

P.1.3 $\mathbf{A} \in \Sigma$: then $\sigma(\mathbf{A}) = \mathbf{A}$ as above

P.1.4 $\mathbf{A} = \mathbf{BC}$: by IH $\mathcal{LR}(\sigma(\mathbf{B}), \gamma \to \alpha)$ and $\mathcal{LR}(\sigma(\mathbf{C}), \gamma)$

P.1.4.2 so $\mathcal{LR}(\sigma(\mathbf{B})\sigma(\mathbf{C}), \alpha)$ by definition of $\mathcal{LR}$. □

P.1.5 $\mathbf{A} = \lambda X_\beta.\mathbf{C}_\gamma$: Let $\mathcal{LR}(\mathbf{B}, \beta)$ and $\theta := \sigma, [\mathbf{B}/X]$, then θ meets the conditions of the IH.

P.1.5.2 Moreover $\sigma(\lambda X_\beta.\mathbf{C}_\gamma)\mathbf{B} \to_\beta \sigma, [\mathbf{B}/X](\mathbf{C}) = \theta(\mathbf{C})$.

P.1.5.3 Now, $\mathcal{LR}(\theta(\mathbf{C}), \gamma)$ by IH and thus $\mathcal{LR}(\sigma(\mathbf{A})\mathbf{B}, \gamma)$ by Lemma 144.

P.1.5.4 So $\mathcal{LR}(\sigma(\mathbf{A}), \alpha)$ by definition of $\mathcal{LR}$. □

□

SOME RIGHTS RESERVED

β-Expansion Lemma

▷ **Lemma 143** *If* $\mathcal{LR}([\mathbf{B}/X](\mathbf{A}),\alpha)$ *and* $\mathcal{LR}(\mathbf{B},\beta)$ *for* $X_\beta \notin \mathbf{free}(\mathbf{B})$, *then* $\mathcal{LR}((\lambda X_\alpha.\mathbf{A})\mathbf{B},\alpha)$.

▷ Proof:

P.1 Let $\alpha = \overline{\gamma_i} \to \delta$ where δ base type and $\mathcal{LR}(\mathbf{C}^i,\gamma^i)$

P.2 It is sufficient to show that $\mathcal{SR}(((\lambda X.\mathbf{A})\mathbf{B})\overline{\mathbf{C}},\delta)$, as δ base type

P.3 We have $\mathcal{LR}([\mathbf{B}/X](\mathbf{A})\overline{\mathbf{C}},\delta)$ by hypothesis and definition of $\mathcal{LR}$.

P.4 thus $\mathcal{SR}([\mathbf{B}/X](\mathbf{A})\overline{\mathbf{C}},\delta)$, as δ base type.

P.5 in particular $\mathcal{SR}([\mathbf{B}/X](\mathbf{A}),\alpha)$ and $\mathcal{SR}(\mathbf{C}^i,\gamma^i)$ (subterms)

P.6 $\mathcal{SR}(\mathbf{B},\beta)$ by hypothesis and Lemma 140

P.7 So an infinite reduction from $((\lambda X.\mathbf{A})\mathbf{B})\overline{\mathbf{C}}$ cannot solely consist of redexes from $[\mathbf{B}/X](\mathbf{A})$ and the $\mathbf{C}^i$.

P.8 so an infinite reduction from $((\lambda X.\mathbf{A})\mathbf{B})\overline{\mathbf{C}}$ must have the form

$$\begin{array}{rcl}((\lambda X.\mathbf{A})\mathbf{B})\overline{\mathbf{C}} & \to_\beta^* & ((\lambda X.\mathbf{A}')\mathbf{B}')\overline{\mathbf{C}'} \\ & \to_\beta^1 & [\mathbf{B}'/X](\mathbf{A}')\overline{\mathbf{C}'} \\ & \to_\beta^* & \dots\end{array}$$

where $\mathbf{A} \to_\beta^* \mathbf{A}'$, $\mathbf{B} \to_\beta^* \mathbf{B}'$ and $\mathbf{C}^i \to_\beta^* \mathbf{C}^{i'}$

P.9 so we have $[\mathbf{B}/X](\mathbf{A}) \to_\beta^* [\mathbf{B}'/X](\mathbf{A}')$

P.10 so we have the infinite reduction

$$\begin{array}{rcl}[\mathbf{B}/X](\mathbf{A})\overline{\mathbf{C}} & \to_\beta^* & [\mathbf{B}'/X](\mathbf{A}')\overline{\mathbf{C}'} \\ & \to_\beta^* & \dots\end{array}$$

which contradicts our assumption □

▷ **Lemma 144 ($\mathcal{LR}$ is closed under β-expansion)**
If $\mathbf{C} \to_\beta \mathbf{D}$ *and* $\mathcal{LR}(\mathbf{D},\alpha)$, *so is* $\mathcal{LR}(\mathbf{C},\alpha)$.

SOME RIGHTS RESERVED ©: Michael Kohlhase 89 JACOBS UNIVERSITY

This was the last result we needed to complete the proof of termiation of β-reduction.

Remark: If we are only interested in the termination of head reductions, we can get by with a much simpler version of this lemma, that basically relies on the uniqueness of head β reduction.

Lemma 145 ($\mathcal{LR}$ is closed under head β-expansion) *If* $\mathbf{C} \to_\beta^h \mathbf{D}$ *and* $\mathcal{LR}(\mathbf{D},\alpha)$, *so is* $\mathcal{LR}(\mathbf{C},\alpha)$.

Proof: by induction over the structure of α

P.1.1 α base type:

P.1.1.1 we have $\mathcal{SR}(\mathbf{D},\alpha)$ by definition

P.1.1.2 so $\mathcal{SR}(\mathbf{C},\alpha)$, since head reduction is unique

P.1.1.3 and thus $\mathcal{LR}(\mathbf{C},\alpha)$. □

P.1.2 $\alpha = \beta \to \gamma$:

P.1.2.1 Let $\mathcal{LR}(\mathbf{B}, \beta)$, by definition we have $\mathcal{LR}(\mathbf{DB}, \gamma)$.

P.1.2.2 but $\mathbf{CB} \rightarrow_\beta^h \mathbf{DB}$, so $\mathcal{LR}(\mathbf{CB}, \gamma)$ by IH

P.1.2.3 and $\mathcal{LR}(\mathbf{C}, \alpha)$ by definition. □

□

Note: This result only holds for weak reduction (any chain of β head reductions terminates) for strong reduction we need a stronger Lemma.

For the termination proof of head β-reduction we would just use the same proof as above, just for a variant of $\mathcal{SR}$, where $\mathcal{SR}\mathbf{A}\alpha$ that only requires that the head reduction sequence out of $\mathbf{A}$ terminates. Note that almost all of the proof except Lemma 139 (which holds by the same argument) is invariant under this change. Indeed Rick Statman uses this observation in [Sta85] to give a set of conditions when logical relations proofs work.

8.3.2 Confluence of $\beta\eta$ Conversion

η-Reduction ist terminating and confluent

▷ **Lemma 146** *η-Reduction ist terminating*

▷ Proof: by a simple counting argument □

▷ **Lemma 147** *η-Reduction ist confluent*

▷ Proof Idea: by diagram chase □

SOME RIGHTS RESERVED ©: Michael Kohlhase 90 JACOBS UNIVERSITY

Newman's Lemma

▷ **Definition 148** We call a rewrite relation $R \subseteq A^2$ confluent iff for every $a, b, c \in A$ with $a \rightarrow_R b$ $a \rightarrow_R c$ there is a $d \in A$ with $b \rightarrow_R d$ and $c \rightarrow_R d$

$$\begin{tikzcd} & a \arrow[dl, "*"'] \arrow[dr, "*"] & \\ b \arrow[dr, "*"'] & & c \arrow[dl, "*"] \\ & d & \end{tikzcd}$$

▷ **Definition 149** We call a rewrite relation $R \subseteq A^2$ weakly confluent iff for every $a, b, c \in A$ with $a \rightarrow_R b$ $a \rightarrow_R c$ there is a $d \in A$ with $b \rightarrow_R^* d$ and $c \rightarrow_R^* d$.

$$\begin{tikzcd} & a \arrow[dl] \arrow[dr] & \\ b \arrow[dr, "*"'] & & c \arrow[dl, "*"] \\ & d & \end{tikzcd}$$

▷ **Theorem 150 (Newman's Lemma)** *If a relation is terminating and weakly confluent, then it is also confluent.*

SOME RIGHTS RESERVED ©: Michael Kohlhase 91 JACOBS UNIVERSITY

β is confluent

▷ **Lemma 151** *β-Reduction ist weakly confluent*

▷ Proof Idea: by diagram chase □

▷ **Corollary 152** *β-Reduction ist confluent*

▷ Proof Idea: by Newman's Lemma □

SOME RIGHTS RESERVED ©: Michael Kohlhase 92 JACOBS UNIVERSITY

$\beta\eta$ is confluent

▷ **Lemma 153** *$\rightarrow_\beta^*$ and $\rightarrow_\eta^*$ commute.*

▷ Proof: diagram chase

□

SOME RIGHTS RESERVED ©: Michael Kohlhase 93 JACOBS UNIVERSITY

8.4 The Semantics of the Simply Typed λ-Calculus

The semantics of $\Lambda^\rightarrow$ is structured around the types. Like the models we discussed before, a model (we call them "algebras", since we do not have truth values in $\Lambda^\rightarrow$) is a pair $\langle\mathcal{D},\mathcal{I}\rangle$, where $\mathcal{D}$ is the universe of discourse and $\mathcal{I}$ is the interpretation of constants.

Semantics of $\Lambda^\rightarrow$

▷ **Definition 154** We call a collection $\mathcal{D}_\mathcal{T} := \{\mathcal{D}_\alpha \mid \alpha \in \mathcal{T}\}$ a typed collection (of sets) and a collection $f_\mathcal{T}\colon \mathcal{D}_\mathcal{T} \rightarrow \mathcal{E}_\mathcal{T}$, a typed function, iff $f_\alpha\colon \mathcal{D}_\alpha \rightarrow \mathcal{E}_\alpha$.

▷ **Definition 155** A typed collection $\mathcal{D}_\mathcal{T}$ is called a frame, iff $\mathcal{D}_{\alpha\rightarrow\beta} \subseteq \mathcal{D}_\alpha \rightarrow \mathcal{D}_\beta$

▷ **Definition 156** Given a frame $\mathcal{D}_\mathcal{T}$, and a typed function $\mathcal{I}\colon \Sigma \rightarrow \mathcal{D}$, then we call $\mathcal{I}_\varphi\colon wff_\mathcal{T}(\Sigma,\mathcal{V}_\mathcal{T}) \rightarrow \mathcal{D}$ the value function induced by $\mathcal{I}$, iff

- ▷ $\mathcal{I}_\varphi\big|_{\mathcal{V}_\mathcal{T}} = \varphi$, $\qquad \mathcal{I}_\varphi\big|_\Sigma = \mathcal{I}$
- ▷ $\mathcal{I}_\varphi(\mathbf{AB}) = \mathcal{I}_\varphi(\mathbf{A})(\mathcal{I}_\varphi(\mathbf{B}))$
- ▷ $\mathcal{I}_\varphi(\lambda X_\alpha.\mathbf{A})$ is that function $f \in \mathcal{D}_{\alpha\rightarrow\beta}$, such that $f(a) = \mathcal{I}_{\varphi,[a/X]}(\mathbf{A})$ for all $a \in \mathcal{D}_\alpha$

▷ **Definition 157** We call a frame $\langle\mathcal{D},\mathcal{I}\rangle$ comprehension-closed or a Σ-algbra, iff $\mathcal{I}_\varphi\colon wff_\mathcal{T}(\Sigma,\mathcal{V}_\mathcal{T}) \rightarrow \mathcal{D}$ is total. (every λ-term has a value)

SOME RIGHTS RESERVED ©: Michael Kohlhase 94 JACOBS UNIVERSITY

8.4.1 Soundness of the Simply Typed λ-Calculus

We will now show is that $\alpha\beta\eta$-reduction does not change the value of formulae, i.e. if $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$, then $\mathcal{I}_\varphi(\mathbf{A}) = \mathcal{I}_\varphi(\mathbf{B})$, for all $\mathcal{D}$ and φ. We say that the reductions are sound. As always, the main tool for proving soundess is a substitution value lemma. It works just as always and verifies that we the definitions are in our semantics plausible.

Substitution Value Lemma for λ-Terms

▷ **Lemma 158 (Substitution Value Lemma)** *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be terms, then* $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\psi(\mathbf{A})$, *where* $\psi = \varphi, [\mathcal{I}_\varphi(\mathbf{B})/X]$

▷ Proof: by induction on the depth of $\mathbf{A}$

P.1 we have five cases

P.1.1 $\mathbf{A} = X$: Then $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\varphi([\mathbf{B}/X](X)) = \mathcal{I}_\varphi(\mathbf{B}) = \psi(X) = \mathcal{I}_\psi(X) = \mathcal{I}_\psi(\mathbf{A})$.

P.1.2 $\mathbf{A} = Y \neq X$ and $Y \in \mathcal{V}_\mathcal{T}$: then $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\varphi([\mathbf{B}/X](Y)) = \mathcal{I}_\varphi(Y) = \varphi(Y) = \psi(Y) = \mathcal{I}_\psi(Y) = \mathcal{I}_\psi(\mathbf{A})$.

P.1.3 $\mathbf{A} \in \Sigma$: This is analogous to the last case.

P.1.4 $\mathbf{A} = \mathbf{CD}$: then $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A})) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{CD})) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{C})[\mathbf{B}/X](\mathbf{D})) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{C}))(\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{D}))) = \mathcal{I}_\psi(\mathbf{C})(\mathcal{I}_\psi(\mathbf{D})) = \mathcal{I}_\psi(\mathbf{CD}) = \mathcal{I}_\psi(\mathbf{A})$

P.1.5 $\mathbf{A} = \lambda Y_\alpha.\mathbf{C}$:

P.1.5.1 We can assume that $X \neq Y$ and $Y \notin \mathbf{free}(\mathbf{B})$

P.1.5.2 Thus for all $a \in \mathcal{D}_\alpha$ we have $\mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}))(a) = \mathcal{I}_\varphi([\mathbf{B}/X](\lambda Y.\mathbf{C}))(a) = \mathcal{I}_\varphi(\lambda Y.[\mathbf{B}/X](\mathbf{C}))(a) = \mathcal{I}_{\varphi,[a/Y]}([\mathbf{B}/X](\mathbf{C})) = \mathcal{I}_{\psi,[a/Y]}(\mathbf{C}) = \mathcal{I}_\psi(\lambda Y.\mathbf{C})(a) = \mathcal{I}_\psi(\mathbf{A})(a)$ □

□

SOME RIGHTS RESERVED

Soundness of $\alpha\beta\eta$-Equality

▷ **Theorem 159** *Let $\mathcal{A} := \langle \mathcal{D}, \mathcal{I} \rangle$ be a Σ-algebra and $Y \notin \mathbf{free}(\mathbf{A})$, then $\mathcal{I}_\varphi(\lambda X.\mathbf{A}) = \mathcal{I}_\varphi(\lambda Y.[Y/X]\mathbf{A})$ for all assignments φ.*

▷ Proof: by substitution value lemma

P.1

$$\begin{aligned}(\mathcal{I}_\varphi(\lambda Y.[Y/X]\mathbf{A})@\mathsf{a}) &= \mathcal{I}_{\varphi,[a/Y]}([Y/X](\mathbf{A})) \\ &= \mathcal{I}_{\varphi,[a/X]}(\mathbf{A}) \\ &= (\mathcal{I}_\varphi(\lambda X.\mathbf{A})@\mathsf{a})\end{aligned}$$

□

▷ **Theorem 160** *If $\mathcal{A} := \langle \mathcal{D}, \mathcal{I} \rangle$ is a Σ-algebra and X not bound in $\mathbf{A}$, then $\mathcal{I}_\varphi((\lambda X.\mathbf{A})\mathbf{B}) = \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}))$.*

▷ Proof: by substitution value lemma

P.1

$$\begin{aligned}\mathcal{I}_\varphi((\lambda X.\mathbf{A})\mathbf{B}) &= (\mathcal{I}_\varphi(\lambda X.\mathbf{A})@\mathcal{I}_\varphi(\mathbf{B})) \\ &= \mathcal{I}_{\varphi,[\mathcal{I}_\varphi(\mathbf{B})/X]}(\mathbf{A}) \\ &= \mathcal{I}_\varphi([\mathbf{B}/X](\mathbf{A}))\end{aligned}$$

□

SOME RIGHTS RESERVED ©: Michael Kohlhase 96 JACOBS UNIVERSITY

Soundness of $\alpha\beta\eta$ (continued)

▷ **Theorem 161** *If $X \notin \mathbf{free}(\mathbf{A})$, then $\mathcal{I}_\varphi(\lambda X.\mathbf{A}X) = \mathcal{I}_\varphi(\mathbf{A})$ for all φ.*

▷ Proof:

$$\begin{aligned}(\mathcal{I}_\varphi(\lambda X.\mathbf{A}X)@\mathsf{a}) &= \mathcal{I}_{\varphi,[\mathsf{a}/X]}(\mathbf{A}X) \\ &= (\mathcal{I}_\varphi(\mathbf{A})@\mathcal{I}_{\varphi,[\mathsf{a}/X]}(X)) \\ &= (\mathcal{I}_\varphi(\mathbf{A})@\mathsf{a})\end{aligned}$$

as $X \notin \mathbf{free}(\mathbf{A})$. □

▷ **Theorem 162** *$\alpha\beta\eta$-equality is sound wrt. Σ-algebras. (if $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$, then $\mathcal{I}_\varphi(\mathbf{A}) = \mathcal{I}_\varphi(\mathbf{B})$ for all assignments φ)*

SOME RIGHTS RESERVED ©: Michael Kohlhase 97 JACOBS UNIVERSITY

8.4.2 Completeness of $\alpha\beta\eta$-Equality

We will now show is that $\alpha\beta\eta$-equality is complete for the semantics we defined, i.e. that whenever $\mathcal{I}_\varphi(\mathbf{A}) = \mathcal{I}_\varphi(\mathbf{B})$ for all variable assignments φ, then $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$. We will prove this by a model existence argument: we will construct a model $\mathcal{M} := \langle \mathcal{D}, \mathcal{I} \rangle$ such that if $\mathbf{A} \neq_{\alpha\beta\eta} \mathbf{B}$ then $\mathcal{I}_\varphi(\mathbf{A}) \neq \mathcal{I}_\varphi(\mathbf{B})$ for some φ.

As in other completeness proofs, the model we will construct is a "ground term model", i.e. a model where the carrier (the frame in our case) consists of ground terms. But in the λ-calculus, we have to do more work, as we have a non-trivial built-in equality theory; we will construct the "ground term model" from sets of normal forms. So we first fix some notations for them.

Normal Forms in the simply typed λ-calculus

▷ **Definition 163** We call a term $\mathbf{A} \in \mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$ a β normal form iff there is no $\mathbf{B} \in \mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$ with $\mathbf{A} \rightarrow_{\beta} \mathbf{B}$.
We call $\mathbf{N}$ a β normal form of $\mathbf{A}$, iff $\mathbf{N}$ is a β-normal form and $\mathbf{A} \rightarrow_{\beta} \mathcal{N}$.
We denote the set of β-normal forms with $\mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})\downarrow_{\beta\eta}$.

▷ We have just proved that $\beta\eta$-reduction is terminating and confluent, so we have

▷ **Corollary 164 (Normal Forms)** *Every $\mathbf{A} \in \mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$ has a unique β normal form ($\beta\eta$, long $\beta\eta$ normal form), which we denote by $\mathbf{A}\downarrow_{\beta}$ ($\mathbf{A}\downarrow_{\beta\eta}$ $\mathbf{A}\downarrow^{l}_{\beta\eta}$)*

SOME RIGHTS RESERVED ©: Michael Kohlhase 98 JACOBS UNIVERSITY

The term frames will be a quotient spaces over the equality relations of the λ-calculus, so we introduce this construction generally.

Frames and Quotients

▷ **Definition 165** Let $\mathcal{D}$ be a frame and $\sim$ a typed equivalence relation on $\mathcal{D}$, then we call $\sim$ a congruence on $\mathcal{D}$, iff $f \sim f'$ and $g \sim g'$ imply $f(g) \sim f'(g')$.

▷ **Definition 166** We call a congruence $\sim$ functional, iff for all $f, g \in \mathcal{D}_{\alpha\rightarrow\beta}$ the fact that $f(a) \sim g(a)$ holds for all $a \in \mathcal{D}_{\alpha}$ implies that $f \sim g$.

▷ **Example 167** $=_{\beta}$ ($=_{\beta\eta}$) is a (functional) congruence on $\mathit{cwff}_{\mathcal{T}}(\Sigma)$ by definition.

▷ **Theorem 168** *Let $\mathcal{D}$ be a Σ-frame and $\sim$ a functional congruence on $\mathcal{D}$, then the quotient space $\mathcal{D}/_{\sim}$ is a Σ-frame.*

▷ Proof:

P.1 $\mathcal{D}/_{\sim} = \{[f]_{\sim} \mid f \in \mathcal{D}\}$, define $[f]_{\sim}([a]_{\sim}) := [f(a)]_{\sim}$.

P.2 This only depends on equivalence classes: Let $f' \in [f]_{\sim}$ and $a' \in [a]_{\sim}$.

P.3 Then $[f(a)]_{\sim} = [f'(a)]_{\sim} = [f'(a')]_{\sim} = [f(a')]_{\sim}$

P.4 To see that we have $[f]_{\sim} = [g]_{\sim}$, iff $f \sim g$, iff $f(a) = g(a)$ since $\sim$ is functional.

P.5 This is the case iff $[f(a)]_{\sim} = [g(a)]_{\sim}$, iff $[f]_{\sim}([a]_{\sim}) = [g]_{\sim}([a]_{\sim})$ for all $a \in \mathcal{D}_{\alpha}$ and thus for all $[a]_{\sim} \in \mathcal{D}/_{\sim}$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 99 JACOBS UNIVERSITY

To apply this result, we have to establish that $\beta\eta$-equality is a functional congruence.

$\beta\eta$-Equivalence as a Functional Congruence

▷ **Definition 169** We call an injective substitution $\sigma\colon \mathbf{free}(\mathbf{C}) \to \Sigma$ a grounding substitution for $\mathbf{C} \in \mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$, iff no $\sigma(X)$ occurs in $\mathbf{C}$.

▷ Observation: They always exist, since all Σ_α are infinite and $\mathbf{free}(\mathbf{C})$ is finite.

▷ **Lemma 170** *$\beta\eta$-equality is a functional congruence on $\mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$.*

▷ Proof: Let $\mathbf{AC} =_{\beta\eta} \mathbf{BC}$ for all $\mathbf{C}$ and $X \in (\mathcal{V}_\gamma \backslash (\mathbf{free}(\mathbf{A}) \cup \mathbf{free}(\mathbf{B})))$.

P.1 then $\mathbf{A}X =_{\beta\eta} \mathbf{B}X$ and $\lambda X.\mathbf{A}X =_{\beta\eta} \lambda X.\mathbf{B}X$.

P.2 By definition we have $\mathbf{A} =_\eta \lambda X_\alpha.\mathbf{A}X =_{\beta\eta} \lambda X_\alpha.\mathbf{B}X =_\eta \mathbf{B}$. □

▷ **Theorem 171** *$\beta\eta$-equality is a functional congruence on $\mathit{cwff}_{\mathcal{T}}(\Sigma)$.*

▷ Proof: We use Lemma 170

P.1 Let $\mathbf{A}, \mathbf{B} \in \mathit{cwff}_{(\alpha\to\beta)}(\Sigma)$, such that $\mathbf{A} \neq_{\beta\eta} \mathbf{B}$.

P.2 Since $\beta\eta$ is functional on $\mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$, there must be a $\mathbf{C}$ with $\mathbf{AC} \neq_{\beta\eta} \mathbf{BC}$.

P.3 Now let $\mathbf{C}' := \sigma(\mathbf{C})$, for a grounding substitution σ.

P.4 Any $\beta\eta$ conversion sequence for $\mathbf{AC}' \neq_{\beta\eta} \mathbf{BC}'$ induces one for $\mathbf{AC} \neq_{\beta\eta} \mathbf{BC}$.

P.5 Thus we have shown that $\mathbf{A} \neq_{\beta\eta} \mathbf{B}$ entails $\mathbf{AC}' \neq_{\beta\eta} \mathbf{BC}'$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase JACOBS UNIVERSITY

Note that: the result for $\mathit{cwff}_{\mathcal{T}}(\Sigma)$ is sharp. For instance, if $\Sigma = \{c_\iota\}$, then $\lambda X.X \neq_{\beta\eta} \lambda X.c$, but $(\lambda X.X)c =_{\beta\eta} c =_{\beta\eta} (\lambda X.c)c$, as $\{c\} = \mathit{cwff}_\iota(\Sigma)$ (it is a relatively simple exercise to extend this problem to more than one constant). The problem here is that we do not have a constant d_ι that would help distinguish the two functions. In $\mathit{wff}_{\mathcal{T}}(\Sigma, \mathcal{V}_{\mathcal{T}})$ we could always have used a variable.

This completes the preparation and we can define the notion of a term algebra, i.e. a Σ-algebra whose frame is made of $\beta\eta$-normal λ-terms.

A Herbrand Model for $\Lambda^\to$

▷ **Definition 172** We call $\mathcal{T}_{\beta\eta} := \langle \mathit{cwff}_{\mathcal{T}}(\Sigma)\big\downarrow_{\beta\eta}, \mathcal{I}^{\beta\eta}\rangle$ the Σ term algebra, if $\mathcal{I}^{\beta\eta} = \mathsf{Id}_\Sigma$.

▷ The name "term algebra" in the previous definition is justified by the following

▷ **Theorem 173** *$\mathcal{T}_{\beta\eta}$ is a Σ-algebra*

▷ Proof: We use the work we did above

P.1 Note that $\mathit{cwff}_{\mathcal{T}}(\Sigma)\big\downarrow_{\beta\eta} = \mathit{cwff}_{\mathcal{T}}(\Sigma)/_{=_{\beta\eta}}$ and thus a Σ-frame by Theorem 168 and Lemma 170.

P.2 So we only have to show that the value function $\mathcal{I}^{\beta\eta} = \mathsf{Id}_\Sigma$ is total.

P.3 Let φ be an assignment into $\mathit{cwff}_{\mathcal{T}}(\Sigma)\big\downarrow_{\beta\eta}$.

P.4 Note that $\sigma := \varphi\big|_{\mathbf{free}(\mathbf{A})}$ is a substitution, since $\mathbf{free}(\mathbf{A})$ is finite.

P.5 A simple induction on the structure of $\mathbf{A}$ shows that $\mathcal{I}^{\beta\eta}_\varphi(\mathbf{A}) = \sigma(\mathbf{A})\big\downarrow_{\beta\eta}$.

P.6 So the value function is total since substitution application is. □

SOME RIGHTS RESERVED ©: Michael Kohlhase JACOBS UNIVERSITY

And as always, once we have a term model, showing completeness is a rather simple exercise. That amounts to showing that the

We can see that $\alpha\beta\eta$-equality is complete for the class of Σ-algebras, i.e. if the equation $\mathbf{A} = \mathbf{B}$ is valid, then $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$. Thus $\alpha\beta\eta$ equivalence fully characterizes equality in the class of all Σ-algebras.

Completetness of $\alpha\beta\eta$-Equality

▷ **Theorem 174** $\mathbf{A} = \mathbf{B}$ *is valid in the class of* Σ*-algebras, iff* $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$.

▷ Proof: For $\mathbf{A}$, $\mathbf{B}$ closed this is a simple consequence of the fact that $\mathcal{T}_{\beta\eta}$ is a Σ-algebra.

P.1 If $\mathbf{A} = \mathbf{B}$ is valid in all Σ-algebras, it must be in $\mathcal{T}_{\beta\eta}$ and in particular $\mathbf{A}{\downarrow_{\beta\eta}} = \mathcal{I}^{\beta\eta}(\mathbf{A}) = \mathcal{I}^{\beta\eta}(\mathbf{B}) = \mathbf{B}{\downarrow_{\beta\eta}}$ and therefore $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$.

P.2 If the equation has free variables, then the argument is more subtle.

P.3 Let σ be a grounding substitution for $\mathbf{A}$ and $\mathbf{B}$ and φ the induced variable assignment.

P.4 Thus ${\mathcal{I}^{\beta\eta}}_{\varphi}(\mathbf{A}) = {\mathcal{I}^{\beta\eta}}_{\varphi}(\mathbf{B})$ is the $\beta\eta$-normal form of $\sigma(\mathbf{A})$ and $\sigma(\mathbf{B})$.

P.5 Since φ is a structure preserving homomorphism on well-formed formulae, $(\varphi)^{-1}({\mathcal{I}^{\beta\eta}}_{\varphi}(\mathbf{A}))$ is the is the $\beta\eta$-normal form of both $\mathbf{A}$ and $\mathbf{B}$ and thus $\mathbf{A} =_{\alpha\beta\eta} \mathbf{B}$. □

SOME RIGHTS RESERVED ©: Michael Kohlhase 102 JACOBS UNIVERSITY

8.5 Simple Type Theory (Higher-Order Logic based on the Simply Typed λ-Calculus)

In this subsection we will revisit the higher-order predicate logic introduced in Subsection 8.1 with the base given by the simply typed λ-calculus. It turns out that we can define a higher-order logic by just introducing a type of propositions in the λ-calculus and extending the signatures by logical constants (connectives and quantifiers).

Higher-Order Logic Revisited

▷ specialize type o to truth values $\mathcal{D}_o = \{\mathsf{T}, \mathsf{F}\}$

▷ $\mathbf{A}_o$ valid under φ, iff $\mathcal{I}_\varphi(\mathbf{A}) = \mathsf{T}$

▷ connectives in Σ: $\neg \in \Sigma_{o\to o}$ $\{\vee, \wedge, \Rightarrow, \Leftrightarrow, \ldots\} \subseteq \Sigma_{o\to o\to o}$ (with the intuitive $\mathcal{I}$-values)

▷ quantifiers: $\Pi^\alpha \in \Sigma_{(\alpha\to o)\to so}$ with $\mathcal{I}(\Pi^\alpha)(p) = \mathsf{T}$, iff $p(a) = \mathsf{T}$ for all $a \in \mathcal{D}_\alpha$.

▷ quantified formulae: $\forall X_\alpha.\mathbf{A}$ stands for $\Pi^\alpha(\lambda X_\alpha.\mathbf{A})$

▷ $\mathcal{I}_\varphi(\forall X_\alpha.\mathbf{A}) = \mathcal{I}(\Pi^\alpha)(\mathcal{I}_\varphi(\lambda X_\alpha.\mathbf{A})) = \mathsf{T}$, iff $\mathcal{I}_{\varphi,[a/X]}(\mathbf{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\alpha$

▷ looks like PLΩ (Call any such system HOL$^{\to}$)

SOME RIGHTS RESERVED ©: Michael Kohlhase 103 JACOBS UNIVERSITY

There is a more elegant way to treat quantifiers in HOL$^{\to}$. It builds on the realization that the λ-abstraction is the only variable binding operator we need, quantifiers are then modeled

as second-order logical constants. Note that we do not have to change the syntax of HOL$^{\rightarrow}$ to introduce quantifiers; only the "lexicon", i.e. the set of logical constants. Since Π^α and Σ^α are logical constants, we need to fix their semantics.

Higher-Order Abstract Syntax

▷ Idea: In HOL$^{\rightarrow}$, we already have variable binder: λ, use that to treat quantification.

▷ **Definition 175** We assume logical constants Π^α and Σ^α of type $\alpha \rightarrow o \rightarrow o$.

Regain quantifiers as abbreviations:

$$\forall X_\alpha.\mathbf{A} := \Pi^\alpha(\lambda X_\alpha.\mathbf{A}) \qquad \exists X_\alpha.\mathbf{A} := \Sigma^\alpha(\lambda X_\alpha.\mathbf{A})$$

▷ **Definition 176** We must fix the semantics of logical constants:

1. $\mathcal{I}(\Pi^\alpha)(p) = \mathsf{T}$, iff $p(a) = \mathsf{T}$ for all a $\in \mathcal{D}_\alpha$ (i.e. if p is the universal set)
2. $\mathcal{I}(\Sigma^\alpha)(p) = \mathsf{T}$, iff $p(a) = \mathsf{T}$ for some a $\in \mathcal{D}_\alpha$ (i.e. iff p is non-empty)

▷ With this, we re-obtain the semantics we have given for quantifiers above:

$$\mathcal{I}_\varphi(\forall X_\iota.\mathbf{A}) = \mathcal{I}_\varphi(\Pi^\iota(\lambda X_\iota.\mathbf{A})) = \mathcal{I}(\Pi^\iota)(\mathcal{I}_\varphi(\lambda X_\iota.\mathbf{A})) = \mathsf{T}$$

iff $\mathcal{I}_\varphi(\lambda X_\iota.\mathbf{A})(a) = \mathcal{I}_{[a/X],\varphi}(\mathbf{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\alpha$

SOME RIGHTS RESERVED ©: Michael Kohlhase 104 JACOBS UNIVERSITY

But there is another alternative of introducing higher-order logic due to Peter Andrews. Instead of using connectives and quantifiers as primitives and defining equality from them via the Leibniz indiscernability principle, we use equality as a primitive logical constant and define everything else from it.

Alternative: HOL$^{=}$

▷ only one logical constant $q^\alpha \in \Sigma_{\alpha\rightarrow\alpha\rightarrow o}$ with $\mathcal{I}(q^\alpha)(a,b) = \mathsf{T}$, iff $a = b$.

▷ Definitions (D) and Notations (N)

N	$\mathbf{A}_\alpha = \mathbf{B}_\alpha$	for	$q^\alpha \mathbf{A}_\alpha \mathbf{B}_\alpha$
D	T	for	$q^o = q^o$
D	F	for	$\lambda X_o.T = \lambda X_o.X_o$
D	Π^α	for	$q^{(\alpha\rightarrow o)}(\lambda X_\alpha.T)$
N	$\forall X_\alpha.\mathbf{A}$	for	$\Pi^\alpha(\lambda X_\alpha.\mathbf{A})$
D	$\wedge$	for	$\lambda X_o.\lambda Y_o.\lambda G_{o\rightarrow o\rightarrow o}.GTT = \lambda G_{o\rightarrow o\rightarrow o}.GXY$
N	$\mathbf{A} \wedge \mathbf{B}$	for	$\wedge \mathbf{A}_o\ \mathbf{B}_o$
D	$\Rightarrow$	for	$\lambda X_o.\lambda Y_o.X = .X \wedge Y$
N	$\mathbf{A} \Rightarrow \mathbf{B}$	for	$\Rightarrow \mathbf{A}_o\ \mathbf{B}_o$
D	$\neg$	for	$q^o F$
D	$\vee$	for	$\lambda X_o.\lambda Y_o.\neg(\neg X \wedge \neg Y)$
N	$\mathbf{A} \vee \mathbf{B}$	for	$\vee \mathbf{A}_o\ \mathbf{B}_o$
D	$\exists X_\alpha.\mathbf{A}_o$	for	$\neg(\forall X_\alpha.\neg\mathbf{A})$
N	$\mathbf{A}_\alpha \neq \mathbf{B}_\alpha$	for	$\neg(q^\alpha \mathbf{A}_\alpha\ \mathbf{B}_\alpha)$

▷ yield the intuitive meanings for connectives and quantifiers.

SOME RIGHTS RESERVED ©: Michael Kohlhase 105 JACOBS UNIVERSITY

In a way, this development of higher-order logic is more foundational, especially in the context of Henkin semantics. There, Theorem 110 does not hold (see [And72] for details). Indeed the proof

of Theorem 110 needs the existence of "singleton sets", which can be shown to be equivalent to the existence of the identity relation. In other words, Leibniz equality only denotes the equality relation, if we have an equality relation in the models. However, the only way of enforcing this (remember that Henkin models only guarantee functions that can be explicitly written down as λ-terms) is to add a logical constant for equality to the signature.

We will conclude this section with a discussion on two additional "logical constants" (constants with a fixed meaning) that are needed to make any progress in mathematics. Just like above, adding them to the logic guarantees the existence of certain functions in Henkin models. The most important one is the description operator that allows us to make definite descriptions like "the largest prime number" or "the solution to the differential equation $f' = f$.

More Axioms for HOL$^{\rightarrow}$

- ▷ **Definition 177** unary conditional $\mathbf{w} \in \Sigma_{\alpha\rightarrow o\rightarrow\alpha}$
  $\mathbf{w}\mathbf{A}_o\ \mathbf{B}_\alpha$ means: "If $\mathbf{A}$, then $\mathbf{B}$"
- ▷ **Definition 178** binary conditional $\mathbf{if} \in \Sigma_{\alpha\rightarrow\alpha\rightarrow o\rightarrow\alpha}$
  $\mathbf{if}\mathbf{A}_o\ \mathbf{B}_\alpha\ \mathbf{C}_\alpha$ means: "if $\mathbf{A}$, then $\mathbf{B}$ else $\mathbf{C}$".
- ▷ **Definition 179** description operator $\iota \in \Sigma_{(\alpha\rightarrow o)\rightarrow\alpha}$
  if $\mathbf{P}$ is a singleton set, then $\iota\mathbf{P}_{(\alpha\rightarrow o)}$ is the element in $\mathbf{P}$,
- ▷ **Definition 180** choice operator $\gamma \in \Sigma_{(\alpha\rightarrow o)\rightarrow\alpha}$
  if $\mathbf{P}$ is non-empty, then $\gamma\mathbf{P}_{(\alpha\rightarrow o)}$ is an arbitrary element from $\mathbf{P}$
- ▷ Formalization
  $\forall X_\alpha.\mathbf{A} \Rightarrow (\mathbf{w}\mathbf{A}\ X) = X$
  $\forall X_\alpha, Y_\alpha, Z_\alpha.(\mathbf{A} \Rightarrow (\mathbf{if}\mathbf{A}\ X\ Y) = X) \wedge (\neg\mathbf{A} \Rightarrow (\mathbf{if}\mathbf{A}\ Z\ X) = X)$
  $\forall P_{\alpha\rightarrow o}.(\exists^1 X_\alpha.PX) \Rightarrow (\forall Y_\alpha.PY \Rightarrow (\iota P) = Y)$
  $\forall P_{\alpha\rightarrow o}.(\exists X_\alpha.PX) \Rightarrow (\forall Y_\alpha.PY \Rightarrow (\gamma P) = Y)$
- ▷ Ensure a much larger supply of functions in Henkin models.

SOME RIGHTS RESERVED ©: Michael Kohlhase 106 JACOBS UNIVERSITY

More on the Description Operator

- ▷ ι is a weak form of the choice operator (only works on singleton sets)
- ▷ Alternative Axiom of Descriptions: $\forall X_\alpha.\iota^\alpha(=X) = X$.
  - ▷ use that $\mathcal{I}_{[\mathsf{a}/X]}(=X) = \{\mathsf{a}\}$
  - ▷ we only need this for base types $\neq o$
  - ▷ Define $\iota^o := =(\lambda X_o.X)$ or $\iota^o := \lambda G_{o\rightarrow o}.GT$ or $\iota^o := =(=T)$
  - ▷ $\iota^{\alpha\rightarrow\beta} := \lambda H_{(\alpha\rightarrow\beta)\rightarrow o} X_\alpha.\iota^\beta \lambda Z_\beta.(\exists F_{\alpha\rightarrow\beta}.(HF) \wedge (FX) = Z)$

SOME RIGHTS RESERVED ©: Michael Kohlhase 107 JACOBS UNIVERSITY

8.6 The Curry-Howard Isomorphism

The Curry-Howard Iso. for minimal $\Rightarrow$-Logic

- ▷ make the structural similarity between λ-calculus and propositional ND explicit
  - ▷ $\rightarrow$ resembles $\Rightarrow$
  - ▷ types resemble formulae ("propositions as types")
  - ▷ λ-terms resemble proofs ("proof terms", "proofs as programs")
  - ▷ wff:app resembles $\Rightarrow E$, wff:abs resembles $\Rightarrow I$
- ▷ $\mathbf{A}$ provable, iff α non-empty (e.g. for the Hilbert-axioms)
  - ▷ $\lambda X_\alpha.\lambda Y_\beta.X_\alpha$ has type $\alpha \rightarrow \beta \rightarrow \alpha$
  - ▷ $\lambda X_{\alpha\rightarrow\beta\rightarrow\gamma}.\lambda Y_{\alpha\rightarrow\gamma}.\lambda Z_\gamma.XZ\ (YZ)$ has type $(\alpha \rightarrow \beta \rightarrow \gamma) \rightarrow (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \gamma$
- ▷ works well for $\rightarrow$ and $\Rightarrow$

SOME RIGHTS RESERVED

Types for Conjunctions

- ▷ new type constructor: $\times$ (product type $\alpha \times \beta$)
- ▷ new term constructors: $\langle\cdot,\cdot\rangle$, π_1 and π_2
- ▷ new type inference rules

$$\frac{\Gamma \vdash_\Sigma \mathbf{A}\colon \alpha \quad \Gamma \vdash_\Sigma \mathbf{B}\colon \beta}{\Gamma \vdash_\Sigma \langle \mathbf{A}, \mathbf{B}\rangle\colon \alpha \times \beta}\text{wff:pair} \qquad \frac{\Gamma \vdash_\Sigma \mathbf{A}\colon \alpha \times \beta}{\Gamma \vdash_\Sigma \pi_1(\mathbf{A})\colon \alpha}\text{wff:}\pi_1 \qquad \frac{\Gamma \vdash_\Sigma \mathbf{A}\colon \alpha \times \beta}{\Gamma \vdash_\Sigma \pi_2(\mathbf{A})\colon \beta}\text{wff:}\pi_2$$

- ▷ new reductions (gives canonical reduction system)

$$(\pi_1(\langle \mathbf{A}, \mathbf{B}\rangle) \rightarrow^1_\beta \mathbf{A}) \qquad (\pi_2(\langle \mathbf{A}, \mathbf{B}\rangle) \rightarrow^1_\beta \mathbf{B}) \qquad (\langle \pi_1(\mathbf{A}), \pi_2(\mathbf{A})\rangle \rightarrow^1_\eta \mathbf{A})$$

- ▷ Others: disjoint sum for disjunction, complement for negation,...

SOME RIGHTS RESERVED

Example (Conjunction)

$$\dfrac{\dfrac{\dfrac{[\mathbf{A}\wedge\mathbf{B}]^X}{\mathbf{B}}\wedge E_r \quad \dfrac{[\mathbf{A}\wedge\mathbf{B}]^X}{\mathbf{A}}\wedge E_l}{\mathbf{B}\wedge\mathbf{A}}\wedge I}{\mathbf{A}\wedge\mathbf{B}\Rightarrow\mathbf{B}\wedge\mathbf{A}}\Rightarrow I^X$$

corresponds to $\lambda X_{\alpha\times\beta}.\langle\pi_2(X),\pi_1(X)\rangle$

▷ Normalization

$$\dfrac{\dfrac{\begin{matrix}\mathcal{D}\\ \mathbf{A}\end{matrix} \quad \begin{matrix}\mathcal{E}\\ \mathbf{B}\end{matrix}}{\mathbf{A}\wedge\mathbf{B}}\wedge I}{\mathbf{A}}\wedge E_l \qquad \begin{matrix}\mathcal{D}\\ \mathbf{A}\end{matrix}$$

since $(\pi_1(\langle\mathbf{M},\mathbf{N}\rangle)\rightarrow^1_\beta\mathbf{M})$

▷ analogous for the other reductions

SOME RIGHTS RESERVED ©: Michael Kohlhase

The Curry-Howard Isomorphism by Example

▷ **Example 181 (Deriving the S-Axiom)**

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma\vdash\alpha\to\beta\to\gamma \quad \Gamma\vdash\alpha \quad \dfrac{\Gamma\vdash\alpha\to\beta \quad \Gamma\vdash\alpha}{\Gamma\vdash\beta}}{\Gamma\vdash\gamma}}{\alpha\to\beta\to\gamma\,,\ \alpha\to\beta\vdash\alpha\to\gamma}}{\alpha\to\beta\to\gamma\vdash(\alpha\to\beta)\to\alpha\to\gamma}}{\emptyset\vdash(\alpha\to\beta\to\gamma)\to(\alpha\to\beta)\to\alpha\to\gamma}$$

where $\Gamma = \alpha\to\beta\to\gamma\,,\ \alpha\to\beta\,,\ \alpha$

SOME RIGHTS RESERVED ©: Michael Kohlhase

The Curry-Howard Isomorphism by Example

▷ **Example 182 (Deriving the S-Axiom)**

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash_\Sigma X\colon \alpha \to \beta \to \gamma \quad \Gamma \vdash_\Sigma Z\colon \alpha \qquad \dfrac{\Gamma \vdash_\Sigma Y\colon \alpha \to \beta \quad \Gamma \vdash_\Sigma Z\colon \alpha}{\Gamma \vdash_\Sigma YZ\colon \beta}}{\Gamma \vdash_\Sigma XZ(YZ)\colon \gamma}}{[X\colon \alpha \to \beta \to \gamma], [Y\colon \alpha \to \beta] \vdash_\Sigma \lambda Z.XZ(YZ)\colon \alpha \to \gamma}}{[X\colon \alpha \to \beta \to \gamma] \vdash_\Sigma \lambda YZ.XZ(YZ)\colon (\alpha \to \beta) \to \alpha \to \gamma}}{\emptyset \vdash_\Sigma \lambda XYZ.XZ(YZ)\colon (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma}$$

where $\Gamma = [X\colon \alpha \to \beta \to \gamma], [Y\colon \alpha \to \beta], [Z\colon \alpha]$

SOME RIGHTS RESERVED

What next for the Curry Howard Isomorphism?

▷ Additional types for disjunction and negation

▷ Problem: What about quantifiers?

▷ Idea 1: Introduce type polymorphism ("Second-Order λ-Calculus")

- ▷ type-variables, type-quantification
- ▷ $\lambda X.X$ has the type $\forall \alpha.\alpha \to \alpha$

▷ Idea 2: make types dependent on terms ("Edinburgh Logical Framework")

- ▷ typ-constructors "type-families": $\vec{(7)}$ is the type of vectors of length 7
- ▷ $mat(3 \to 5) \to \vec{(3)} \to \vec{(5)}$ is the type of a matrix multiplication operator
- ▷ $pf(\mathbf{A})$ the type of proofs of a formula $\mathbf{A}$.

SOME RIGHTS RESERVED

Example: Proof Normalization

▷ $$\dfrac{\dfrac{\mathcal{D}}{\mathbf{A}} \quad \dfrac{\dfrac{[\mathbf{A}]^X}{\mathbf{B}}\,\mathcal{E}}{\mathbf{A}\Rightarrow\mathbf{B}}\Rightarrow I^X}{\mathbf{B}}\Rightarrow E$$

▷ $\mathcal{E}$ corresponds to $\mathbf{M}_\mu$ ($\mathbf{B}$ to β)

▷ $\mathcal{D}$ corresponds to $\mathbf{N}_\nu$ ($\mathbf{A}$ to α)

▷ the whole to $(\lambda X_\alpha.\mathbf{M}_\mu)\mathbf{N}_\nu$

▷ Normalize: $((\lambda X_\alpha.\mathbf{M}_\mu)\mathbf{N}_\nu) \to^1_\beta [\mathbf{N}/X](\mathbf{M})$ corresponds to $\dfrac{\dfrac{\mathcal{D}}{\mathbf{A}}}{\mathbf{B}}\,\mathcal{E}$

▷ **Theorem 183 (Cut Elimination)** *For each ND-proof of a formula* $\mathbf{A}$ *there is a* $\Rightarrow E$*-free proof of* $\mathbf{A}$.

SOME RIGHTS RESERVED

λ-calculus and functional Programming

▷ z.B. LISP (λ-calculus with lists $[a|b|c]$ and conditionals)

```
sort=(λ (L) (if L=[] then []else[min(L)|sort(del(min(L),L))]))
del=(λ (X,L) (if L=[] then ⊥
            elif X=first(L) then rest(L)
            else [first(L)|del(X,rest(L))]))
min=(λ (L) (if L=[]then ⊥
            elif L=[first(L)]then first(L)
            elif first(L)<min(rest(L))then first(L)
            else min(rest(L))))
```

▷ Program-Evaluation is β-reduction and list-reduction

```
del c {\red[a|b|c]}
        →¹β [first({\red[a|b|c]})|del(c,rest({\red[a|b|c]}))]→¹β [a|del(c,{\red[b|c]})]
        →¹β [a|first({\red[b|c]})|del(c,rest({\red[b|c]}))]→¹β [a|b|del(c,{\red[c]})]
        →¹β [a|b|rest({\red[c]})]→¹β [a|b|[]]→¹β [a|b]
```

SOME RIGHTS RESERVED

Proofs as Programs

▷ Remember: Proofs are λ-terms & λ-terms are programs

▷ Idea: Then proofs should be programs (well only constructive ones)

▷ **Example 184 (Sorting)** a theory of ordered lists:

- ▷ $\models perm(L, M)$, if M is a permutation of L
- ▷ $\models ord(L)$, if L ordered wrt. $<$
- ▷ $X < L$ if $X < Y$ for all $Y \in L$

Theorems:

- ▷ $\models min(L) < del(min(L), L)$
- ▷ $\models ord(L) \wedge x < L \Rightarrow ord([x|L])$
- ▷ $\models perm(L \Rightarrow M) \Rightarrow perm([x|L], [x|M])$

▷ **Theorem 185** $\forall L.\exists M.ord(L)perm(L, M)$

▷ Proof: by induction on the structure of the list L

P.1.1 If $L = []$: choose $M = []$

P.1.2 If $L \neq []$:

P.1.2.1 by IH there is a list W, such that $ord(W)perm(W, del(min(L), L))$

P.1.2.2 chose $M = [min(L)|W]$ □

□

Programm:

```
sort=($\lambda$ L (if L=[] then $\bot$ else[min(L)|sort(del(min(L),L))]))
```

▷

▷ Note: the correcness of this program is ensured by the proof

▷ Note: different proofs yield different programs

▷ Note: the programs extracted from automatically found proofs are not always efficient (Slowsort!)

SOME RIGHTS RESERVED

8.7 Simply Typed λ-Calculus as an Inference System

Simply Typed λ-Calculus as an Inference System

▷ Judgment $\Gamma \vdash_\Sigma \mathbf{A}\colon \alpha$ (formula $\mathbf{A}$ has type α given the type assumptions in Γ)

▷ $\mathbf{A} \in \mathit{wff}_\alpha(\Sigma, \mathcal{V}_\mathcal{T})$, iff $\Gamma \vdash_\Sigma \mathbf{A}\colon \alpha$ derivable in

$$\frac{c \in \Sigma_\alpha}{\Gamma \vdash_\Sigma c\colon \alpha}\text{wff:const} \qquad \frac{\Gamma \vdash_\Sigma \mathbf{A}\colon (\beta \to \alpha) \quad \Gamma \vdash_\Sigma \mathbf{B}\colon \beta}{\Gamma \vdash_\Sigma (\mathbf{AB})\colon \alpha}\text{wff:app}$$

$$\frac{\Gamma(X) = \alpha}{\Gamma \vdash_\Sigma X\colon \alpha}\text{wff:var} \qquad \frac{\Gamma, [X\colon \alpha] \vdash_\Sigma \mathbf{A}\colon \alpha}{\Gamma \vdash_\Sigma (\lambda X_\beta.\mathbf{A})\colon (\beta \to \alpha)}\text{wff:abs}$$

▷ Oops: this looks surprisingly like a natural deduction calculus. (⇒ Curry Howard Isomorphism)

SOME RIGHTS RESERVED

$\beta\eta$-Equality by Inference Rules: One-Step Reduction

▷ One-step Reduction ($+ \in \{\alpha, \beta, \eta\}$)

$$\frac{}{\vdash ((\lambda X.\mathbf{A})\mathbf{B}) \to^1_\beta [\mathbf{B}/X](\mathbf{A})}\text{wff}\beta\text{:top} \qquad \frac{X \notin \mathbf{free}(\mathbf{A})}{\vdash (\lambda X.\mathbf{A}X) \to^1_\beta \mathbf{A}}\text{wff}\eta\text{:top}$$

$$\frac{\vdash \mathbf{A} \to^1_+ \mathbf{B}}{\vdash (\mathbf{AC}) \to^1_+ (\mathbf{BC})}\text{tr:app}fn \qquad \frac{\vdash \mathbf{A} \to^1_+ \mathbf{B}}{\vdash (\mathbf{CA}) \to^1_+ (\mathbf{CB})}\text{tr:app}arg$$

$$\frac{\vdash \mathbf{A} \to^1_+ \mathbf{B}}{\vdash (\lambda X.\mathbf{A}) \to^1_+ (\lambda X.\mathbf{B})}\text{tr:abs}$$

SOME RIGHTS RESERVED

$\beta\eta$-Equality by Inference Rules: Multi-Step Reduction

▷ Multi-Step-Reduction ($+ \in \{\alpha, \beta, \eta\}$)

$$\frac{\vdash \mathbf{A} \to^1_+ \mathbf{B}}{\vdash \mathbf{A} \to^*_+ \mathbf{B}}\text{ms:start} \qquad \frac{}{\vdash \mathbf{A} \to^*_+ \mathbf{A}}\text{ms:ref}$$

$$\frac{\vdash \mathbf{A} \to^*_+ \mathbf{B} \quad \vdash \mathbf{B} \to^*_+ \mathbf{C}}{\vdash \mathbf{A} \to^*_+ \mathbf{C}}\text{ms:trans}$$

▷ Congruence Relation

$$\frac{\vdash \mathbf{A} \to^*_+ \mathbf{B}}{\vdash \mathbf{A} =_+ \mathbf{B}}\text{eq:start}$$

$$\frac{\vdash \mathbf{A} =_+ \mathbf{B}}{\vdash \mathbf{B} =_+ \mathbf{A}}\text{eq:sym} \qquad \frac{\vdash \mathbf{A} =_+ \mathbf{B} \quad \vdash \mathbf{B} =_+ \mathbf{C}}{\vdash \mathbf{A} =_+ \mathbf{C}}\text{eq:trans}$$

SOME RIGHTS RESERVED

References

[And72] Peter B. Andrews. General models and extensionality. *Journal of Symbolic Logic*, 37(2):395–397, 1972.

[Chu40] Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[Fre79] Gottlob Frege. Begriffsschrift: eine der arithmetischen nachgebildete formelsprache des reinen denkens, 1879.

[Göd31] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte der Mathematischen Physik*, 38:173–198, 1931. English Version in [**?**].

[Koh08] Michael Kohlhase. Using LaTeX as a semantic markup format. *Mathematics in Computer Science*, 2(2):279–304, 2008.

[Koh10] Michael Kohlhase. sTeX: Semantic markup in TeX/LaTeX. Technical report, Comprehensive TeX Archive Network (CTAN), 2010.

[Sta85] Rick Statman. Logical relations and the typed lambda calculus. *Information and Computation*, 65, 1985.

[WR10] Alfred North Whitehead and Bertrand Russell. *Principia Mathematica*, volume I. Cambridge University Press, Cambridge, UK, 2 edition, 1910.

Index

abstract
    consistency
        class, 26
admissible, 19
admits
    weakening, 19
α-equal, 16
alphabetical
    variants, 16
assignment
    variable, 12
assignment
    variable, 47
assumption, 19
Axiom
    Extensionality, 50
axiom, 19

base
    type, 46
β-equality
    Axiom of, 50
Axiom of
    β-equality, 50
β normal form, 62
β normal form of $\mathbf{A}$, 62
binary
    conditional, 66
binder, 53
bound, 51

calculus, 19
calculus
    sequent, 23
carrier, 47
$\mathcal{C}$-consistent, 26
choice
    operator, 66
closed, 13
closed under
    subset, 26
collection
    typed, 59
compact, 27
complete, 40
comprehension-closed, 59
conclusion, 19
conditional
    binary, 66
    unary, 66
confluent, 58
    weakly, 58
congruence, 62
connective, 11
connective, 64
connectives, 46
consistency
    abstract (class), 26
constant
    Skolem, 11
    function, 11
    predicate, 11
constants
    interpretation of, 47
contradiction, 26
correct, 40
$\mathcal{C}$-refutable, 26
Currying, 45

derivation, 19
derivation
    system, 19
derives, 36
description
    operator, 66
discharged, 21
domain
    type, 46

entailment
    relation, 18
entails, 18
Equality
    Leibniz, 45
equational
    system, 39
η-Expansion, 53
η-long
    form, 53
form
    η-long, 53
extension, 14
Extensionality
    Axiom, 50
Extensionality, 45
extensionality, 49

falsified by $\mathcal{M}$, 18
falsifiable, 18
first-order
    signature, 11
valuation, 30

form
    normal, 52
form
    solved, 39
formula
    well-typed, 46
formula
    well-typed, 51
formula, 18
formula
    quantified, 64
$\mathcal{U}$-reducible, 41
frame, 59
free
    variable, 13
free, 51
function
    typed, 59
    value, 59
function
    value, 13
function
    constant, 11
function
    type, 46
function
    universe, 47
    value, 47

general
    more, 38
ground, 13
grounding
    substitution, 63

Head
    Reduction, 53
head
    symbol, 53
    syntactic, 53
hypotheses, 19
hypothetical
    reasoning, 21

individual, 11
    variable, 11
individuals, 12
inference
    rule, 19
interpretation, 12
interpretation of
    constants, 47
introduced, 14

Judgment, 72

judgment, 23

Leibniz
    Equality, 45
logical
    relation, 54
logical
    system, 18
$\beta\eta$-normal
    Long (form), 53
Long
    $\beta\eta$-normal
        form, 53

matrix, 53
most general
    unifier, 38
unifier
    most general, 38
Model, 13
model, 18
more
    general, 38
multiset
    ordering, 41

∇-Hintikka Set, 29
negative, 36
normal
    form, 52

operator
    choice, 66
    description, 66
ordering
    multiset, 41

predicate
    constant, 11
product
    type, 67
projection, 53
proof, 19
proof
    tableau, 36
proof-reflexive, 19
proof-transitive, 19
proposition, 12
$\mathcal{T}_0$-theorem, 36

quantified
    formula, 64
quantifier, 64

range

type, 46
reasonable, 26
reasoning
hypothetical, 21
reducing
strongly, 54
Reduction
Head, 53
refutation
tableau, 36
relation
logical, 54
relation
entailment, 18
satisfaction, 18
rule
inference, 19

Σ-algbra, 59
satisfaction
relation, 18
satisfiable, 18
satisfied by $\mathcal{M}$, 18
saturated, 36
sentence, 13
sequent, 23
calculus, 23
individuals
set of, 47
set of
individuals, 47
set of
truth values, 47
truth values
set of, 47
signature, 46
signature
first-order, 11
Skolem
constant, 11
solved
form, 39
strongly
reducing, 54
subset
closed under, 26
substitutable, 15
substitution
grounding, 63
substitution, 14
support, 14
symbol
head, 53
syntactic
head, 53
system
derivation, 19
system
logical, 18
system
equational, 39

tableau
proof, 36
refutation, 36
term, 12
test calculi, 36
theorem, 19
truth
value, 12
truth
value, 11
type
product, 67
type, 46
base, 46
domain, 46
function, 46
range, 46
truth values
type of, 46
type of
truth values, 46
typed
collection, 59
function, 59
individuals
type of, 46
type of
individuals, 46

unary
conditional, 66
unifier, 38
unitary, 40
Universe, 12
universe, 12
universe
function, 47
unsatisfiable, 18

valid, 18
valid, 47
valid, 64
value
function, 59
value
function, 13

truth, 12
value
truth, 11
value
function, 47
variable
free, 13
variable, 46
variable
assignment, 12
variable
individual, 11
variable
assignment, 47
variants
alphabetical, 16

weakening
admits, 19
weakly
confluent, 58
well-typed
formula, 46
well-typed
formula, 51