« Mathematical foundations: (2) Classical first-order logic »

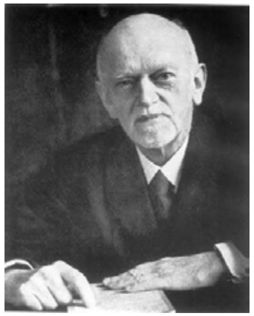
Patrick Cousot
Jerome C. Hunsaker Visiting Professor
Massachusetts Institute of Technology
Department of Aeronautics and Astronautics

cousot@mit.edu
www.mit.edu/~cousot

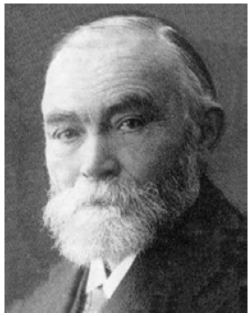
Course 16.399: “Abstract interpretation”

http://web.mit.edu/afs/athena.mit.edu/course/16/16.399/www/

George Boole



David Hilbert



Gottlob Frege

Reference

[1] Jean van Heijenoort, editor. “From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931”. Harvard University Press, 1967.

Formal logics

A formal logic consists of:

– a formal or informal language (formula expressing facts)

– a model-theoretic semantics (to define the meaning of the language, that is which facts are valid)

– a deductive system (made of axioms and inference rules to formaly derive theorems, that is facts that are provable)

Questions about formal logics

The main questions about a formal logic are:

– The soundness of the deductive system: no provable formula is invalid

– The completeness of the deductive system: all valid formulæ are provable

Propositional classical logic

Syntax of the classical propositional logic

Classical propositional logic

- $X \in \mathcal{V}$ are variables denoting unknown true or false facts
- The set of formulæ $\phi \in \mathcal{F}$ of the propositional logic are defined by the following grammar:

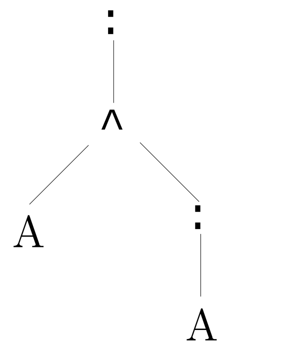
$$\begin{array}{rcl} \phi & ::= & X \\ & \mid & (\phi_1 \wedge \phi_2) \\ & \mid & (\neg \phi) \end{array}$$

- The relation “is a subformula of” is well founded, whence can be used for structural definitions and proofs

Example of formulæ

- A is a variable whence a formula
- $(\neg A)$ is a formula since A is a formula
- $(A \wedge (\neg A))$ is a formula since A and and $(\neg A)$ are formulae
- $(\neg(A \wedge (\neg A)))$ is a formula since $(A \wedge (\neg A))$ is a formula

The derivation tree of the formula is:



Abstract syntax

- In practice we avoid parentheses thanks to priorities:
  - $\neg$ has highest priority (evaluated first)
  - $\wedge$ has lowest priority (evaluated second)
  - $\wedge$ is left associative (evaluation from left to right)

  For example, $\neg A \wedge \neg B \wedge C$ stands for $(\neg A) \wedge (\neg B) \wedge C$ which stands for $((\neg A) \wedge (\neg B)) \wedge C$
- The derivation tree is given by the following abstract grammar:

$$\begin{array}{rcl} \phi & ::= & X \\ & \mid & \phi_1 \wedge \phi_2 \\ & \mid & \neg\phi \end{array}$$

Propositional identities

Abbreviations (de Morgan laws)

$$\begin{array}{rcl}
\phi_1 \vee \phi_2 & \stackrel{\text{def}}{=} & \neg(\neg\phi_1 \wedge \neg\phi_2) \\
\phi_1 \Longrightarrow \phi_2 & \stackrel{\text{def}}{=} & \neg\phi_1 \vee \phi_2 \\
\phi_1 \Longleftarrow \phi_2 & \stackrel{\text{def}}{=} & \phi_2 \Longrightarrow \phi_1 \\
\phi_1 \iff \phi_2 & \stackrel{\text{def}}{=} & (\phi_1 \Longrightarrow \phi_2) \wedge (\phi_1 \Longleftarrow \phi_2) \\
\phi_1 \veebar \phi_2 & \stackrel{\text{def}}{=} & (\phi_1 \vee \phi_2) \wedge \neg(\phi_1 \wedge \phi_2)
\end{array}$$

Free variables of proopositional formulae

The set $\mathrm{FV}(\phi)$ of free variables appearing in a formula ϕ is defined by structural induction as follows:

$$\begin{array}{rcl}
\mathrm{FV}(X) & \stackrel{\text{def}}{=} & \{X\} \\
\mathrm{FV}(\neg\phi) & \stackrel{\text{def}}{=} & \mathrm{FV}(\phi) \\
\mathrm{FV}(\phi_1 \wedge \phi_2) & \stackrel{\text{def}}{=} & \mathrm{FV}(\phi_1) \cup \mathrm{FV}(\phi_2)
\end{array}$$

Semantics of the propositional classical logic

Booleans

We define the booleans $\mathbb{B} \stackrel{\text{def}}{=} \{\mathtt{tt}, \mathtt{ff}\}$ and boolean operators by the following truth table:

$\overline{\&}$	$\mathtt{tt}$	$\mathtt{ff}$
$\mathtt{tt}$	$\mathtt{tt}$	$\mathtt{ff}$
$\mathtt{ff}$	$\mathtt{ff}$	$\mathtt{ff}$

$\overline{\neg}$	
$\mathtt{tt}$	$\mathtt{ff}$
$\mathtt{ff}$	$\mathtt{tt}$

Environment/Assignment

– An environment[1] $\rho \in \mathcal{V} \not\mapsto \mathbb{B}$ assigns boolean values $\rho(X)$ to free propositional variables X.

– An example of assignment is $\rho = \{X \to \mathtt{tt}, Y \to \mathtt{ff}\}$ such that $\rho(X) = \mathtt{tt}$, $\rho(Y) = \mathtt{ff}$ and the value for all other propositional variables $Z \in \mathcal{V} \setminus \{X, Y\}$ is undefined

[1] Also called *assignment* in logic.

Tarskian/model-theoretic semantics of the classical propositional logic

The semantics[2] $\mathcal{S} \in \mathcal{F} \mapsto (\mathcal{V} \mapsto \mathbb{B}) \mapsto \mathbb{B}$ of a propositional formula ϕ assign a meaning $\mathcal{S}[\![\phi]\!]\rho$ to the formula for any given environment ρ[3]:

$$\begin{aligned}
\mathcal{S}[\![X]\!]\rho &\stackrel{\text{def}}{=} \rho(X)\\
\mathcal{S}[\![\neg\phi]\!]\rho &\stackrel{\text{def}}{=} \overline{\neg}(\mathcal{S}[\![\phi]\!]\rho)\\
\mathcal{S}[\![\phi_1 \wedge \phi_2]\!]\rho &\stackrel{\text{def}}{=} \mathcal{S}[\![\phi_1]\!]\rho \mathbin{\overline{\&}} \mathcal{S}[\![\phi_2]\!]\rho
\end{aligned}$$

[2] Also called an *interpretation* in logic

[3] Hilbert used instead an arithmetic interpretation where 0 is true and 1 is false.

Models

ρ is a model of ϕ (or that ρ satisfies ϕ) if and only if:

$$\mathcal{S}[\![\phi]\!]\rho = \mathtt{tt}$$

which is written:

$$\rho \Vdash \phi$$

Entailment

– A set $\Gamma \in \wp(\mathcal{F})$ of formulae entails ϕ whenever:

$$\forall \rho : (\forall \phi' \in \Gamma : \rho \Vdash \phi') \Longrightarrow \rho \Vdash \phi$$

which is written:

$$\Gamma \Vdash \phi$$

Validity

– We say that ϕ is valid if and only if:

$$\forall \rho \in (\mathrm{V} \mapsto \mathbb{B}) : \mathcal{S}[\![\phi]\!]\rho = \mathtt{tt}$$

which is written:

$$\Vdash \phi$$

(i.e. ϕ is a tautaulogy, always true)

Examples of tautologies

$$P \Longrightarrow P$$
$$(\neg\neg P) \Longrightarrow P$$
$$P \Longrightarrow (\neg\neg P)$$
$$P \Longrightarrow (Q \Longrightarrow P)$$
$$P \Longrightarrow (Q \Longrightarrow Q)$$
$$(\neg P \Longrightarrow P) \Longrightarrow P$$
$$P \Longrightarrow (\neg P \Longrightarrow Q)$$
$$\neg P \Longrightarrow (P \Longrightarrow Q)$$
$$(\neg(P \Longrightarrow P)) \Longrightarrow Q$$
$$P \Longrightarrow (\neg(P \Longrightarrow \neg P))$$
$$(P \Longrightarrow \neg P) \Longrightarrow \neg P$$
$$(\neg(P \Longrightarrow Q)) \Longrightarrow P$$
$$(\neg(P \Longrightarrow Q)) \Longrightarrow (\neg\neg P)$$
$$(\neg(P \Longrightarrow Q)) \Longrightarrow \neg Q$$
$$(P \Longrightarrow \neg P) \Longrightarrow (P \Longrightarrow Q)$$
$$(P \Longrightarrow Q) \Longrightarrow (\neg Q \Longrightarrow \neg P)$$
$$(P \Longrightarrow \neg Q) \Longrightarrow (Q \Longrightarrow \neg P)$$
$$(\neg P \Longrightarrow \neg Q) \Longrightarrow (Q \Longrightarrow P)$$
$$(\neg P \Longrightarrow \neg Q) \Longrightarrow (\neg P \Longrightarrow Q) \Longrightarrow P)$$
$$(\neg(P \Longrightarrow Q)) \Longrightarrow (Q \Longrightarrow R)$$
$$(\neg(P \Longrightarrow Q)) \Longrightarrow (\neg P \Longrightarrow R)$$
$$(P \Longrightarrow Q) \Longrightarrow ((Q \Longrightarrow R) \Longrightarrow (P \Longrightarrow R))$$

Satisfiability/Unsatisfiability

– A formula $\phi \in \mathcal{F}$ is satisfiable if and only if:

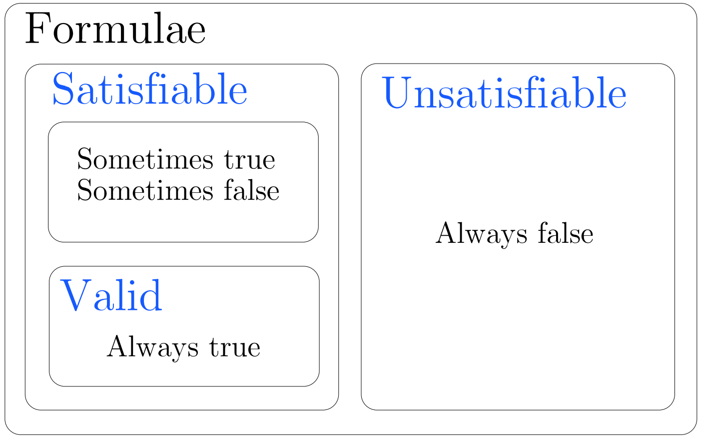
$$\exists \rho \in (\mathrm{V} \mapsto \mathbb{B}) : \mathcal{S}[\![\phi]\!]\rho = \mathtt{tt}$$

– A formula $\phi \in \mathcal{F}$ is unsatisfiable if and only if:

$$\forall \rho \in (\mathrm{V} \mapsto \mathbb{B}) : \mathcal{S}[\![\phi]\!]\rho = \mathtt{tt}$$

(i.e. ϕ is a antilogy, always false)

Satisfiability/Validity/Unsatisfiability



Deductive system for the classical propositional logic

Hilbert deductive system

- Axiom schemata [4]:
  (1) $\phi \vee \phi \Longrightarrow \phi$ [5]
  (2) $\phi \Longrightarrow \phi' \vee \phi$ [6]
  (3) $(\phi \Longrightarrow \phi') \Longrightarrow (\phi'' \vee \phi \Longrightarrow \phi' \vee \phi'')$ [7]
- Inference rule schema [4]:

$$(\mathrm{MP}) \quad \frac{\phi, \ \phi \Longrightarrow \phi'}{\phi'}$$ [8] modus ponens

[4] to be instanciated for all possible formulae $\phi, \phi', \phi'' \in \mathcal{F}$
[5] i.e. $\neg(\neg(\neg\phi \wedge \neg\phi)) \vee \phi)$
[6] i.e. $\neg(\neg\neg\phi \wedge \neg\neg(\neg\phi \wedge \neg\phi'))$
[7] i.e; $\neg(\neg\phi \vee \phi') \vee (\neg(\phi'' \vee \phi) \vee (\phi' \vee \phi''))$ where $\phi_1 \vee \phi_2 \stackrel{\text{def}}{=} \neg(\neg\phi_1 \vee \neg\phi_2)$
[8] i.e. $\dfrac{\phi, \ \neg\phi \vee \phi'}{\phi'}$

Hilbert derivation

- A derivation from a set $\Gamma \in \wp(\mathcal{F})$ of hypotheses is a finite nonempty sequence:

$$\phi_1, \phi_2, \ldots, \phi_n \qquad n \geq 0$$

of formulae such that for each ϕ_i, $i = 1, \ldots, n$, we have:
  - ϕ_i is a element of Γ (hypothesis)
  - ϕ_i is an axiom
  - ϕ_i is the conclusion of an inference rule $\dfrac{\phi_i^1, \ldots, \phi_i^k}{\phi_i}$ such that $\{\phi_i^1, \ldots, \phi_i^k\} \subseteq \{\phi_1, \phi_2, \ldots, \phi_{n-1}\}$ [9]

[9] So that the premises have already been proved.

Hilbert proof

– A proof is a derivation from $\emptyset$

Example of proof

$$
\begin{array}{lll}
 & (\phi \vee \phi \Longrightarrow \phi) \Longrightarrow (\neg\phi \vee (\phi \vee \phi) \Longrightarrow \phi \vee \neg\phi) & \\
 & & \wr[\text{instance of (3)] (a)}\wr \\
 & \phi \vee \phi \Longrightarrow \phi & \wr[\text{instance of (1)] (b)}\wr \\
 & \neg\phi \vee (\phi \vee \phi) \Longrightarrow (\phi \vee \neg\phi) & \wr[\text{(a), (b) and (MP)] (c)}\wr \\
= & (\phi \Longrightarrow (\phi \vee \phi)) \Longrightarrow \phi \vee \neg\phi & \wr\text{def.} \Longrightarrow \text{abbreviation}\wr \\
 & \phi \Longrightarrow (\phi \vee \phi) & \wr[\text{instance of (2)] (d)}\wr \\
 & \phi \vee \neg\phi & \wr[\text{(c), (d) and (MP)]}\wr
\end{array}
$$

Hilbert provability

– $\phi \in \mathrm{F}$ is provable from $\Gamma \in \wp(\mathrm{F})$ (or Γ proves ϕ) iff there is a proof of ϕ from Γ, written:

$$\Gamma \vdash \phi$$

where the deduction system (axioms and inference rules) are understood from the context.

– $\emptyset \vdash \phi$ is written $\vdash \phi$

This is the proof-theoretic semantics of first-order logic.

Example of provability

$$\vdash \neg\phi \vee \neg\neg\phi$$

PROOF.
Replace ϕ by $\neg\phi$ is the previous proof of $\phi \vee \neg\phi$.
$\Box$

Soundness of a deductive system

Provable formulae do hold:

$$\Gamma \vdash \phi \Longrightarrow \Gamma \Vdash \phi$$

PROOF.
The proof for propositional logic is by induction on the length of the formal proof of ϕ from Γ.

A proof of length one, can only use a formula ϕ in Γ which is assumed to hold (i.e. $\mathcal{S}[\![\phi]\!]\rho = \mathrm{tt}$) or an axiom that does hold as shown below.

– $\mathcal{S}[\![\phi \vee \phi \Longrightarrow \phi]\!]\rho$
 $= \mathcal{S}[\![\neg(\neg(\neg\phi \wedge \neg\phi))]\!]\rho$ def. $\vee$
 $= \overline{\neg}(\overline{\neg}(\overline{\neg}(\mathcal{S}[\![\phi]\!]\rho)\overline{\&}\overline{\neg}(\mathcal{S}[\![\phi]\!]\rho)))$ def. $\mathcal{S}$
 $= \overline{\neg}(\mathcal{S}[\![\phi]\!]\rho)\overline{\&}\overline{\neg}(\mathcal{S}[\![\phi]\!]\rho)$ def. $\overline{\neg}$
 $= \overline{\neg}(\mathrm{ff})$ def. $\overline{\&}$

 $= \mathrm{tt}$ def. $\overline{\neg}$

– The proof is similar for the other two axioms.

A proof of length $n+1$, $n \geq 1$ is an initial proof $\phi_0, \ldots, \phi_{n-1}$ of length n followed by a formula ϕ_n. By induction hypothesis, we have $\mathcal{S}[\![\phi_i]\!]\rho = \mathrm{tt}$, $i = 1, \ldots, n-1$.

If $\phi_n \in \Gamma$ or ϕ_n is an axiom then $\mathcal{S}[\![\phi_n]\!]\rho = \mathrm{tt}$ as shown above.

Otherwise, ϕ_n is derived by the modus ponens inference rule (MP). In that case, we have k, $0 \leq k < n$ such that $\mathcal{S}[\![\phi_k]\!]\rho = \mathrm{tt}$ and $\mathcal{S}[\![\phi_k \Longrightarrow \phi_n]\!]\rho = \mathrm{tt}$ so $(\mathcal{S}[\![\phi_k]\!]\rho \overline{\Longrightarrow} \mathcal{S}[\![\phi_n]\!]\rho) = \mathrm{tt}$ where the truth table of $\overline{\Longrightarrow}$ is derived from the definition of $\Longrightarrow$ and that of $\overline{\neg}$ and $\overline{\wedge}$ as follows:

$\overline{\Longrightarrow}$	ff	tt
ff	tt	tt
tt	ff	tt

Since $\mathcal{S}[\![\phi_k]\!]\rho = \mathrm{tt}$ the truth table of $\overline{\Longrightarrow}$ shows than the only possibility for $(\mathcal{S}[\![\phi_k]\!]\rho \overline{\Longrightarrow} \mathcal{S}[\![\phi_n]\!]\rho) = \mathrm{tt}$ is $\mathcal{S}[\![\phi_n]\!]\rho = \mathrm{tt}$. $\square$

Consistency of a deductive system

Absence of contradictory proofs

$$\neg(\exists \Gamma : \Gamma \vdash \phi \overline{\wedge} \Gamma \vdash \neg\phi)$$

A sound deductive system is consistent.

PROOF.
By reduction ad absurdum assume inconsistency $\exists \Gamma : \Gamma \vdash \phi \overline{\wedge} \Gamma \vdash \neg\phi$. By soundness $\Gamma \Vdash \phi \overline{\wedge} \Gamma \Vdash \neg\phi$ whence for all ρ such that $\forall \phi' \in \Gamma : \rho \Vdash \phi'$, we have $\mathcal{S}[\![\phi]\!]\rho = \mathrm{tt}$ and $\mathcal{S}[\![\neg\phi]\!]\rho = \mathrm{tt} = \overline{\neg}\mathcal{S}[\![\phi]\!]\rho = \overline{\neg}\mathrm{tt} = \mathrm{ff}$ which is the desired contradiction since $\mathrm{tt} \neq \mathrm{ff}$. $\square$

Negative normal form

A formula is in negative normal form iff it can be parsed by the following grammar:

$$\begin{aligned} \phi &::= \phi \vee \phi \\ &\quad\mid\ \phi \wedge \phi \\ &\quad\mid\ \varphi \\ \varphi &::= X \\ &\quad\mid\ \neg X \end{aligned}$$

Normalization in negative normal form

$$
\begin{aligned}
\mathrm{nnf}(\neg\phi) &\stackrel{\mathrm{def}}{=} \overline{\mathrm{nnf}}(\phi)\\
\mathrm{nnf}(\phi_1 \vee \phi_2) &\stackrel{\mathrm{def}}{=} \mathrm{nnf}(\phi_1) \vee \mathrm{nnf}(\phi_2)\\
\mathrm{nnf}(\phi_1 \wedge \phi_2) &\stackrel{\mathrm{def}}{=} \mathrm{nnf}(\phi_1) \wedge \mathrm{nnf}(\phi_2)\\
\overline{\mathrm{nnf}}(\neg\phi) &\stackrel{\mathrm{def}}{=} \mathrm{nnf}(\phi)\\
\overline{\mathrm{nnf}}(\phi_1 \vee \phi_2) &\stackrel{\mathrm{def}}{=} \overline{\mathrm{nnf}}(\phi_1) \wedge \overline{\mathrm{nnf}}(\phi_2)\\
\overline{\mathrm{nnf}}(\phi_1 \wedge \phi_2) &\stackrel{\mathrm{def}}{=} \overline{\mathrm{nnf}}(\phi_1) \vee \overline{\mathrm{nnf}}(\phi_2)\\
\mathrm{nnf}(X) &\stackrel{\mathrm{def}}{=} X\\
\overline{\mathrm{nnf}}(X) &\stackrel{\mathrm{def}}{=} \neg X
\end{aligned}
$$

A formula ϕ is equivalent to its negative normal form $\mathrm{nnf}(\phi)$ is that:

$$\vdash \phi \quad \text{if and only if} \quad \vdash \mathrm{nnf}(\phi)$$

Conjunctive normal form

A formula is in conjunctive normal form iff it can be parsed by the following grammar:

$$
\begin{aligned}
\phi &::= \phi^{\wedge}\\
\phi^{\wedge} &::= \phi^{\wedge} \wedge \phi^{\wedge}\\
&\mid\ \phi^{\vee}\\
\phi^{\vee} &::= \phi^{\vee} \vee \phi^{\vee}\\
&\mid\ \varphi\\
\varphi &::= X\\
&\mid\ \neg X
\end{aligned}
$$

Normalization in conjunctive normal form

Any formula ϕ can be put in equivalent conjunctive normal form by applying the following transformations to $\mathrm{nnf}(\phi)$:

$$
\begin{aligned}
\phi' \vee (\phi_1 \wedge \phi_2) &\rightsquigarrow (\phi' \wedge \phi_1) \vee (\phi' \wedge \phi_2)\\
(\phi_1 \vee \phi_2) \wedge \phi' &\rightsquigarrow (\phi_1 \vee \phi') \wedge (\phi_2 \vee \phi')
\end{aligned}
$$

A formula ϕ is equivalent to its conjunctive normal form $\phi^{\wedge}$ in that:

$$\vdash \phi \quad \text{if and only if} \quad \vdash \phi^{\wedge}$$

Completeness of a deductive system

Formulae which hold are provable:

$$\Gamma \Vdash \phi \Longrightarrow \Gamma \vdash \phi$$

The very first proof for propositional logic was given by Bernays (a student of Hilbert) [2]. The better known proof is that of Post [3].

Reference

[2] Richard Zach. "Completeness before Post: Bernays, Hilbert, and the development of propositional logic", Bulletin of Symbolic Logic 5 (1999) 331–366.

[3] Ryan Stansifer. "Completeness of Propositional Logic as a Program", Florida Institute of Technology, Melbourne, Florida, March 2001.

Bernay's proof can be sketched as follows. Every formula is interderivable with its conjunctive normal form. A conjuction is provable if and only if each of its conjuncts is provable. A disjunction of propositional variables or negations of proprositional variables if and only if it contains a variable and its negation, and conversely, every such disjunction is provable. So a formula is provable if and only if every conjunct in its normal form contains a variable and its negation. Now suppose that ϕ is a valid ($\Vdash \phi$) but underivable formula. Its conjunctive normal form $\phi^\wedge$ is also underivable, so it must contain a conjunct ϕ' where every variable occurs only negated or unnegated but not both. If ϕ where added as a new axiom (so that $\Vdash \phi$ implies soundness of the new deductive system), then $\phi^\wedge$ and ϕ' would also be derivable. By substituting X for every unnegated variable and $(\neg X)$ for every negated variable in ϕ', we would obtain X as a derivable formula (after some simplification), and the system would be inconsistent, which is the desired contradiction.

Classical first-order logic

Syntax of the classical first-order logic

Lexems

The lexems are the basic constituants of the formal language.

- symbols: (, ,,), $\wedge$, $\neg$, $\forall$, ...
- constants: $a, b \ldots \in \mathcal{C}$ denote individual objects of the universe of discourse
- variables: $x, y, \ldots \in \mathcal{V}$ denote unknown but fixed[10] objects of the universe of discourse

[10] Different instances of the same variable in a given scope of a formula always denote the same unkown individal object of the universe of discourse. This is not true of imperative computer programs.

- function symbols: $f\backslash n, g\backslash n, \ldots \in \mathcal{F}^n$ denote fonctions of arity n. We let $\mathcal{F}^0 \stackrel{\text{def}}{=} \mathcal{C}$ and $\mathcal{F} = \bigcup_{n\in\mathbb{N}} F^n$. For short we write f instead of $f\backslash n$ when the arity n is understood
- relation symbols: $r\backslash n, \rho\backslash n, \ldots \in \mathcal{R}^n$ denote fonctions of arity n. We let $\mathbb{B} \stackrel{\text{def}}{=} \{\mathtt{tt}, \mathtt{ff}\}$ and $\mathcal{R} = \bigcup_{n\in\mathbb{N}} R^n$. For short we write r instead of $r\backslash n$ when the arity n is understood

Terms

Terms $t \in \mathcal{T}$ denote individual objects of the universe of discourse computed by applying fonctions to constants or variables:

$$
\begin{array}{rcl}
t & ::= & c \\
 & | & x \\
 & | & f\backslash n(t_1, \ldots, t_n)
\end{array}
$$

Atomic formulæ

Atomic formulæ $A \in \mathcal{A}$ are used to state elementary facts about objects of the universe of discourse:

$$A ::= r\backslash n(t_1, \ldots, t_n)$$

Example:

- z is a variable whence a term
- $*\backslash 2(+\backslash 2(x, 1), y)$ is a term
- $\leq\backslash 2$ is a relation symbol whence $\leq\backslash 2(*\backslash 2(+\backslash 2(x, 1), y), z)$[11] is an atomic formula

[11] written $((x + 1) * y) \leq z$ in infix form

First-order formulae

The set $\Phi \in \mathcal{L}$ of first-order formulae (of the first-order language $\mathcal{L}$) is defined by the following grammar

$$
\begin{array}{rlll}
\Phi ::= & A & & A \in \mathcal{A} \\
\mid & \forall x : \Phi & & x \in \mathcal{V} \\
\mid & \Phi_1 \vee \Phi_2 & & \\
\mid & \neg \Phi & &
\end{array}
$$

$\exists x : \Phi$ is a shorthand for $\neg(\forall x : (\neg \Phi))$

Free variables

Free variables are not bound by a quantifier:

$$
\begin{aligned}
\mathrm{fv}(\forall x : \Phi) &\stackrel{\text{def}}{=} \mathrm{fv}(\Phi) \setminus \{x\} \\
\mathrm{fv}(\Phi_1 \vee \Phi_2) &\stackrel{\text{def}}{=} \mathrm{fv}(\Phi_1) \cup \mathrm{fv}(\Phi_2) \\
\mathrm{fv}(\neg \Phi) &\stackrel{\text{def}}{=} \mathrm{fv}(\Phi) \\
\mathrm{fv}(r\backslash n(t_1, \ldots, t_n)) &\stackrel{\text{def}}{=} \bigcup_{i=1}^{n} \mathrm{fv}(t_i) \\
\mathrm{fv}(c) &\stackrel{\text{def}}{=} \emptyset \\
\mathrm{fv}(x) &\stackrel{\text{def}}{=} \{x\} \\
\mathrm{fv}(f\backslash n(t_1, \ldots, t_n) &\stackrel{\text{def}}{=} \bigcup_{i=1}^{n} \mathrm{fv}(t_i)
\end{aligned}
$$

Bound variables

Bound variables appear under the scope of a quantifier:

$$
\begin{aligned}
\mathrm{bv}(\forall x : \Phi) &\stackrel{\text{def}}{=} \{x\} \cup \mathrm{bv}(\Phi) \\
\mathrm{bv}(\Phi_1 \vee \Phi_2) &\stackrel{\text{def}}{=} \mathrm{bv}(\Phi_1) \cup \mathrm{bv}(\Phi_2) \\
\mathrm{bv}(\neg \Phi) &\stackrel{\text{def}}{=} \mathrm{bv}(\Phi) \\
\mathrm{bv}(r\backslash n(t_1, \ldots, t_n)) &\stackrel{\text{def}}{=} \emptyset \\
\mathrm{bv}(c) &\stackrel{\text{def}}{=} \emptyset \\
\mathrm{bv}(x) &\stackrel{\text{def}}{=} \emptyset \\
\mathrm{bv}(f\backslash n(t_1, \ldots, t_n) &\stackrel{\text{def}}{=} \emptyset
\end{aligned}
$$

Theories

- The set of variables of a formula is $\mathrm{var}(\Phi) \stackrel{\text{def}}{=} \mathrm{bv}(\Phi) \cup \mathrm{fv}(\Phi)$
- A closed sentence (or ground formula) is a formula Φ with no free variable (so that $\mathrm{fv}(\Phi) = \emptyset$)
- A theory is a set of closed sentences

Substitution

– Substitution is a syntactic replacement of a variable by a term, may be with appropriate renaming of bound variables, so as to avoid capturing the term free variables, as in

$$\begin{array}{rl} & \exists x : x = y + 1[y := x] \\ \not\to & \exists x : x = x + 1 \end{array}$$

but should be

$$\to \quad \exists x' : x' = x + 1$$

A substitution $\sigma \in \mathcal{V} \mapsto \mathcal{T}$ is a function from variables to terms with finite domain:

$$\begin{array}{rcll} \mathrm{dom}(\sigma) & \stackrel{\mathrm{def}}{=} & \{x \in \mathcal{V} \mid x \neq \sigma(x)\} & \text{(finite domain)} \\ \mathrm{rng}(\sigma) & \stackrel{\mathrm{def}}{=} & \{\sigma(x) \mid x \in \mathrm{dom}(\sigma)\} & \text{(range)} \\ \mathrm{yld}(\sigma) & \stackrel{\mathrm{def}}{=} & \bigcup\{\mathrm{fv}(t) \mid t \in \mathrm{rng}(\sigma)\} & \text{(yield)} \end{array}$$

We write σ as:

$$[x_1 \leftarrow \sigma(x_1), \ldots, x_n \leftarrow \sigma(x_n)]$$

where $\mathrm{dom}(\sigma) = \{x_1, \ldots, x_n\}$.

Application of a substitution to a term

$$\begin{array}{rcll} \sigma(c) & \stackrel{\mathrm{def}}{=} & c & \\ \sigma(y) & \stackrel{\mathrm{def}}{=} & y & \text{iff } y \not\in \mathrm{dom}(\sigma) \\ \sigma(f(t_1, \ldots, t_n)) & \stackrel{\mathrm{def}}{=} & f(\sigma(t_1), \ldots, \sigma(t_n)) & \\ \sigma(r(t_1, \ldots, t_n)) & \stackrel{\mathrm{def}}{=} & r(\sigma(t_1), \ldots, \sigma(t_n)) & \\ \sigma(\neg\Phi) & \stackrel{\mathrm{def}}{=} & \neg\sigma(\Phi) & \\ \sigma(\Phi_1 \vee \Phi_2) & \stackrel{\mathrm{def}}{=} & \sigma(\Phi_1) \vee \sigma(\Phi_2) & \\ \sigma(\forall x : \Phi) & \stackrel{\mathrm{def}}{=} & \forall x' : \sigma(\Phi[x := x']) & \text{where} \\ & & & x' \not\in \mathrm{yld}(\sigma) \cup (\mathrm{fv}(\Phi) \setminus \{x\}) \end{array}$$

Example of substitution in a term

$$\begin{array}{rl} & (\exists x : x = y + 1)[y := x] \\ = & \exists x' : ((x = y + 1)[x := x'])[y := x] \\ = & \exists x' : ((x)[x := x'] = (y)[x := x'] + (1)[x := x'])[y := x] \\ = & \exists x' : (x' = y + 1)[y := x] \\ = & \exists x' : ((x')[y := x] = (y)[y := x] + (1)[y := x]) \\ = & \exists x' : ((x')[y := x] = (y)[y := x] + (1)[y := x]) \end{array}$$

Semantics of the classical first-order logic

Interpretation

An interpretation I is defined by:

- A domain of discourse D_I (or domain of interpretation)
- An interpretation $I[\![f]\!] \in D_I^m \mapsto D_I$ for each function symbol $f \in \mathcal{F}^m$, $m \geq 0$ (including constants)
- An interpretation $I[\![r]\!] \in D_I^m \mapsto \mathbb{B}$ for each relation symbol $r \in \mathcal{R}^m$, $m \geq 0$

Environment/Assignment

- An environment/assignment $\rho \in \mathcal{V} \mapsto D_I$ assigns a value $\rho(x)$ to each variable $x \in \mathcal{V}$

Assignment notation: if $f \in A \mapsto B$, $a \in A$, $b \in B$ then $f[a := b] = f' \in A \mapsto B$ such that:

$$\begin{array}{ll} f'(a) = b & \text{i.e. } f[a := b](a) = b \\ f'(x) = f(x) & \text{whenever } x \neq a \text{ i.e. } f[a := b](x) = f(x) \end{array}$$

Semantics of the first-order logic

Given an interpretation I, the semantics is:

$$\mathcal{S}^I[\![t]\!] \in (\mathcal{V} \mapsto D_I) \mapsto D_I$$

$$\begin{array}{rcl} \mathcal{S}^I[\![c]\!]\rho & \stackrel{\text{def}}{=} & I[\![c]\!] \\ \mathcal{S}^I[\![x]\!]\rho & \stackrel{\text{def}}{=} & \rho(x) \\ \mathcal{S}^I[\![f(t_1, \ldots, t_n)]\!]\rho & \stackrel{\text{def}}{=} & I[\![f]\!](\mathcal{S}^I[\![t_1]\!]\rho, \ldots, \mathcal{S}^I[\![t_n]\!]\rho) \end{array}$$

$$\mathcal{S}^I[\![A]\!] \in (\mathcal{V} \mapsto D_I) \mapsto \mathbb{B}$$

$$\mathcal{S}^I[\![r(t_1,\ldots,t_n)]\!]\rho \stackrel{\text{def}}{=} I[\![r]\!](\mathcal{S}^I[\![t_1]\!]\rho,\ldots,\mathcal{S}^I[\![t_n]\!]\rho)$$

$$\mathcal{S}^I[\![\Phi]\!] \in (\mathcal{V} \mapsto D_I) \mapsto \mathbb{B}$$

$$\mathcal{S}^I[\![\neg\Phi]\!]\rho \stackrel{\text{def}}{=} \overline{\neg}(\mathcal{S}^I[\![g\Phi]\!]\rho)$$

$$\mathcal{S}^I[\![\Phi_1 \vee \Phi_2]\!]\rho \stackrel{\text{def}}{=} \mathcal{S}^I[\![\Phi_1]\!]\rho \,\overline{\vee}\, \mathcal{S}^I[\![\Phi_2]\!]\rho$$

$$\mathcal{S}^I[\![\forall x : \Phi]\!]\rho \stackrel{\text{def}}{=} \overline{\bigwedge_{v \in D_I}}{}^{12}\, \mathcal{S}^I[\![\Phi]\!]\rho[x := v]$$

12 If $S \subseteq \mathbb{B}$ then $\overline{\bigwedge} S \stackrel{\text{def}}{=} (S \subseteq \{\mathtt{tt}\})$.

It follows that for the abbreviations, we have:

$$\mathcal{S}^I[\![\Phi_1 \Longrightarrow \Phi_2]\!]\rho \stackrel{\text{def}}{=} \mathcal{S}^I[\![\Phi_1]\!]\rho \,\overline{\Longrightarrow}\, \mathcal{S}^I[\![\Phi_2]\!]\rho$$

$$\mathcal{S}^I[\![\exists x : \Phi]\!]\rho \stackrel{\text{def}}{=} \overline{\bigvee_{v \in D_I}}\, \mathcal{S}^I[\![\Phi]\!]\rho[x := v]$$

where:

$\overline{\Longrightarrow}$	ff	tt
ff	tt	tt
tt	ff	tt

$\overline{\vee}$	ff	tt
ff	ff	tt
tt	tt	tt

and if $S \subseteq \mathbb{B}$ then $\overline{\bigvee} S \stackrel{\text{def}}{=} (S \cap \{\mathtt{tt}\} \neq \emptyset)$.

Semantics of substitution

Assignment is the semantic counterpart of syntactic substitution:

$$\mathcal{S}^I[\![\sigma(\Phi)]\!]\rho = \mathcal{S}^I[\![\Phi]\!]\rho'$$

$$\text{where} \quad \forall x \in \mathcal{V} : \rho'(x) = \mathcal{S}^I[\![\sigma(x)]\!]\rho$$

Lemma

If $x \notin \text{fv}[\![t]\!]$ then
$\forall \rho \in \mathcal{V} \mapsto D_I : \forall v \in D_I : \mathcal{S}^I[\![t]\!]\rho = \mathcal{S}^I[\![t]\!]\rho[x := v]$

PROOF.

- The case $t = x$ is disallowed by $x \notin \text{fv}[\![x]\!] = \{x\}$
- If $y \neq x$ then $x \notin \text{fv}[\![y]\!] = \{y\}$ and $\mathcal{S}^I[\![y]\!]\rho = \rho(y) = \rho[x := v](y) = \mathcal{S}^I[\![y]\!]\rho[x := v]$
- $\mathcal{S}^I[\![f(t_1,\ldots,t_n)]\!]\rho$
  $= I[\![f]\!](\mathcal{S}^I[\![t_1]\!]\rho,\ldots,\mathcal{S}^I[\![t_n]\!]\rho)$
  $= I[\![f]\!](\mathcal{S}^I[\![t_1]\!]\rho[x := v],\ldots,\mathcal{S}^I[\![t_n]\!]\rho[x := v])$ by induction hypothesis since $\forall i : x \notin \text{fv}[\![t_i]\!]$
  $= \mathcal{S}^I[\![f(t_1,\ldots,t_n)]\!]\rho[x := v]$
- $\mathcal{S}^I[\![r(t_1,\ldots,t_n)]\!]\rho$
  $= I[\![r]\!](\mathcal{S}^I[\![t_1]\!]\rho,\ldots,\mathcal{S}^I[\![t_n]\!]\rho)$

$= I[\![r]\!](\mathcal{S}^I[\![t_1]\!]\rho[x := v], \ldots, \mathcal{S}^I[\![t_n]\!]\rho[x := v])$ by induction hypothesis since $\forall i : x \notin \mathrm{fv}[\![t_i]\!]$

$= \mathcal{S}^I[\![r(t_1, \ldots, t_n)]\!]\rho[x := v]$

$\square$

Proof of the theorem

PROOF.

By structural induction on formulae

- $\mathcal{S}^I[\![\sigma(c)]\!]\rho = \mathcal{S}^I[\![c]\!]\rho = I[\![c]\!] = \mathcal{S}^I[\![c]\!]\rho'$
- $\mathcal{S}^I[\![\sigma(x)]\!]\rho = \rho'(x) = \mathcal{S}^I[\![x]\!]\rho'$
- $\mathcal{S}^I[\![\sigma(f(t_1, \ldots, t_n))]\!]\rho$
  $= \mathcal{S}^I[\![f(\sigma(t_1), \ldots, \sigma(t_n))]\!]\rho$
  $= I[\![f]\!](\mathcal{S}^I[\![\sigma(t_1)]\!]\rho, \ldots, \mathcal{S}^I[\![\sigma(\sigma(t_n)]\!]\rho)$
  $= I[\![f]\!](\mathcal{S}^I[\![t_1]\!]\rho', \ldots, \mathcal{S}^I[\![t_n]\!]\rho'$
  $= \mathcal{S}^I[\![f(t_1, \ldots, t_n)]\!]\rho')$

proving that $\forall t : \mathcal{S}^I[\![\sigma(t)]\!]\rho \;=\; \mathcal{S}^I[\![t]\!]\rho'$

- $\mathcal{S}^I[\![\sigma(r(t_1, \ldots, t_n))]\!]\rho$
  $= \mathcal{S}^I[\![r(\sigma(t_1), \ldots, \sigma(t_n))]\!]\rho$
  $= I[\![r]\!](\mathcal{S}^I[\![\sigma(t_1)]\!]\rho, \ldots, \mathcal{S}^I[\![\sigma(\sigma(t_n)]\!]\rho)$
  $= I[\![r]\!](\mathcal{S}^I[\![t_1]\!]\rho', \ldots, \mathcal{S}^I[\![t_n]\!]\rho'$

$= \mathcal{S}^I[\![r(t_1, \ldots, t_n)]\!]\rho')$

proving that $\forall A : \mathcal{S}^I[\![\sigma(A)]\!]\rho \;=\; \mathcal{S}^I[\![A]\!]\rho'$

- $\mathcal{S}^I[\![\sigma(\neg\Phi)]\!]\rho = \mathcal{S}^I[\![\neg\sigma(\Phi)]\!]\rho = \overline{\neg}(\mathcal{S}^I[\![\sigma(\Phi)]\!]\rho) = \overline{\neg}(\mathcal{S}^I[\![\Phi]\!]\rho') = \mathcal{S}^I[\![\neg\Phi]\!]\rho'$
- $\mathcal{S}^I[\![\sigma(\Phi_1 \vee \Phi_2)]\!]\rho = \mathcal{S}^I[\![\sigma(\Phi_1) \vee \sigma(\Phi_2)]\!]\rho = \mathcal{S}^I[\![\sigma(\Phi_1)]\!]\rho \,\overline{\vee}\, \mathcal{S}^I[\![\sigma(\Phi_2)]\!]\rho = \mathcal{S}^I[\![\Phi_1]\!]\rho' \,\overline{\vee}\, \mathcal{S}^I[\![\Phi_2]\!]\rho' = \mathcal{S}^I[\![\Phi_1 \vee \Phi_2]\!]\rho'$
- $\mathcal{S}^I[\![\sigma(\forall x : \Phi)]\!]\rho$

  $= \mathcal{S}^I[\![\forall x' : \sigma(\Phi[x := x'])]\!]\rho$

  $\wr$where $x' \notin \mathrm{yld}(\sigma) \cup (\mathrm{fv}(\Phi) \setminus \{x\})$$\wr$

  $= \mathcal{S}^I[\![\forall x' : \sigma([x \leftarrow x']$ [13] $(\Phi))]\!]\rho$

  $= \mathcal{S}^I[\![\forall x' : (\sigma \circ [x \leftarrow x'])$ [14] $(\Phi))]\!]\rho$

  $= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![(\sigma \circ [x \leftarrow x'])(\Phi))]\!]\rho[x' := v]$

  $= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!]\left(\lambda y \cdot \mathcal{S}^I[\![(\sigma \circ [x \leftarrow x'])(y))]\!]\rho[x' := v]\right)$ $\wr$by induction hypothesis$\wr$

$= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\lambda y \cdot (\!| y = x \mathrel{?} \mathcal{S}^I[\![(\sigma \circ [x \leftarrow x'])(y))]\!]\rho[x' := v] \mathrel{¿} \mathcal{S}^I[\![(\sigma \circ [x \leftarrow x'])(y))]\!]\rho[x' := v] |\!)$ [15] $)$

$= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\lambda y \cdot (\!| y = x \mathrel{?} \mathcal{S}^I[\![\sigma(x')]\!]\rho[x' := v] \mathrel{¿} \mathcal{S}^I[\![\sigma(y)]\!]\rho[x' := v] |\!))$

$= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\lambda y \cdot (\!| y = x \mathrel{?} \mathcal{S}^I[\![x']\!]\rho[x' := v] \mathrel{¿} \mathcal{S}^I[\![\sigma(y)]\!]\rho[x' := v] |\!))$

$\wr$since $x' \notin \mathrm{yld}(\sigma)$ so that $\sigma(x') = x'$$\wr$

$= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\lambda y \cdot (\!| y = x \mathrel{?} v \mathrel{¿} \mathcal{S}^I[\![y]\!]\rho' |\!))$

$\wr$since

- $\mathcal{S}^I[\![x']\!]\rho[x' := v] = \rho[x' := v](x') = v$
- $x' \notin \mathrm{yld}(\sigma)$ so that $x' \in \mathrm{fv}[\![\sigma(y)]\!]$ hence, by the lemma, $\mathcal{S}^I[\![\sigma(y)]\!]\rho[x' := v] = \mathcal{S}^I[\![\sigma(y)]\!]\rho = \mathcal{S}^I[\![y]\!]\rho'$ by induction hypothesis

$\wr$

$$
\begin{aligned}
&= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\lambda y \cdot (\!| y = x \mathrel{?} v \mathrel{:} \rho'(y) |\!))\\
&= \overline{\bigwedge_{v \in D_I}} \mathcal{S}^I[\![\phi]\!](\rho'[x := v])\\
&= \mathcal{S}^I[\![\forall x : \phi]\!]\rho'
\end{aligned}
$$

$\square$

[13] The function $[x \leftarrow x']$ is the substitution of x' for x

[14] $\circ$ is function composition $f \div compg(x) \stackrel{\text{def}}{=} f(g(x))$

[15] The conditional is $(\!| \mathrm{tt} \mathrel{?} a : b |\!) = a$ and $(\!| \mathrm{ff} \mathrel{?} a : b |\!) = b$ and $(\!| a \mathrel{?} b \,\|\, c \mathrel{?} d : e |\!) = (\!| a \mathrel{?} b : (\!| c \mathrel{?} d : e |\!) |\!)$

Deductive system for the classical first-order logic

Deduction system for first-order logic (H)

– Axioms (for all instances of formulae Φ, Φ', Φ', variable x and term t):

$$
\begin{array}{ll}
(1) & \Phi \vee \Phi \Longrightarrow \Phi\\
(2) & \Phi \Longrightarrow \Phi' \vee \Phi\\
(3) & (\Phi \Longrightarrow \Phi') \Longrightarrow (\Phi'' \vee \Phi \Longrightarrow \Phi' \vee \Phi'')\\
(4) & \forall x : \Phi \Longrightarrow \Phi[x := t]\\
(5) & (\forall x : \Phi \vee \Phi') \Longrightarrow \Phi \vee \forall x : \Phi' \text{ when } x \notin \mathrm{fv}(\Phi)
\end{array}
$$

– Inference rules (for all instances of formulae Φ, Φ' and variable x):

$$
\begin{array}{lcl}
(\mathrm{MP}) & \dfrac{\Phi,\ \Phi \Longrightarrow \Phi'}{\Phi'} & \text{Modus Ponens}\\[2ex]
(\mathrm{Gen}) & \dfrac{\Phi}{\forall x : \Phi} & \text{Generalization}
\end{array}
$$

Example 1 of proof

$$\Phi[x := t] \Longrightarrow \neg\forall x : \neg\Phi \quad \text{(i.e. } \exists x : \Phi\text{)}$$

Proof. (assuming tautologies for short)

(a) $\forall x : \neg\Phi \Longrightarrow (\neg\Phi)[x := t]$ ≀instance of (4)∫

(b) $(\Phi \Longrightarrow \Phi') \Longrightarrow (\neg\Phi' \Longrightarrow \neg\Phi)$ ≀contraposition tautology∫

(b') $(\forall x : \neg\Phi \Longrightarrow (\neg\Phi)[x := t]) \Longrightarrow \neg((\neg\Phi)[x := t]) \Longrightarrow \neg\forall x : \neg\Phi$ ≀tautology, instance of (b)∫

(c) $\neg((\neg\Phi)[x := t]) \Longrightarrow \neg\forall x : \neg\Phi$ ≀(a), (b') and (MP)∫

(c') $\neg\neg(\Phi[x := t]) \Longrightarrow \neg\forall x : \neg\Phi$ ≀def. substitution∫

(d) $(\neg\neg\Phi \Longrightarrow \neg\Phi') \Longrightarrow (\Phi \Longrightarrow \neg\Phi')$ ≀tautology∫

(d') $(\neg\neg(\Phi[x := t]) \Longrightarrow \neg\forall x : \neg\Phi) \Longrightarrow (\Phi[x := t] \Longrightarrow \neg\forall x : \neg\Phi)$ ≀tautology∫

(e) $\Phi[x := t] \Longrightarrow \neg\forall x : \neg\Phi$ ≀(c), (d') and (MP)∫

$\square$

Example 2 of proof

$$\{\Phi \Longrightarrow \Phi'\} \vdash \neg\forall x : \neg\Phi \Longrightarrow \Phi' \quad \text{when } x \notin \mathrm{fv}(\Phi')$$

Proof. (assuming tautologies for short)

(a) $\Phi \Longrightarrow \Phi'$ ≀hypothesis∫

(b) $(\Phi \Longrightarrow \Phi') \Longrightarrow (\neg\Phi' \Longrightarrow \neg\Phi)$ ≀contraposition tautology∫

(c) $\neg\Phi' \Longrightarrow \neg\Phi$ ≀(a), (b) and (MP)∫

(c') $\neg\neg\Phi' \vee \neg\Phi$ ≀def. abbreviation $\Longrightarrow$∫

(d) $\forall x : (\neg\neg\Phi' \vee \neg\Phi)$ ≀(c'), (Gen)∫

(e) $\neg\neg\Phi' \vee \forall x : \neg\Phi$ ≀(d), (5), $x \notin \mathrm{fv}(\neg\neg\Phi') = \mathrm{fv}(\Phi')$∫

(f) $\neg\Phi' \Longrightarrow \forall x : \neg\Phi$ ≀def. abbreviation $\Longrightarrow$∫

(g) $(\neg\Phi' \Longrightarrow \forall x : \neg\Phi) \Longrightarrow (\neg\forall x : \neg\Phi \Longrightarrow \neg\neg\Phi')$ ≀contraposition tautology∫

(h) $\neg\forall x : \neg\Phi \Longrightarrow \neg\neg\Phi'$ ≀(f), (g) and (MP)∫

(i) $(\Phi \Longrightarrow \neg\neg\Phi') \Longrightarrow (\Phi \Longrightarrow \Phi')$ ≀tautology∫

(i') $(\neg\forall x : \neg\Phi \Longrightarrow \neg\neg\Phi') \Longrightarrow (\neg\forall x : \neg\Phi \Longrightarrow \Phi')$ ≀tautology, instance of (i)∫

(j) $\neg\forall x : \neg\Phi \Longrightarrow \Phi'$ ≀(h), (i') and (MP)∫

$\square$

Extension of the deduction system (H) for first-order logic

These theorems are often incorporated to the deductive system as an axiom

$$\Phi[x := t] \Longrightarrow \exists x : \Phi$$

and a generalization rule:

$$\frac{\Phi \Longrightarrow \Phi'}{(\exists x : \Phi) \Longrightarrow \Phi'} \quad \text{when } x \notin \mathrm{fv}(\Phi)$$

Logical equivalences involving quantifiers and negations

- $\neg\forall x : \Phi \iff \exists x : \neg Phi$ De Morgan laws
- $\neg\exists x : \Phi \iff \forall x : \neg\Phi$
- $(\forall x : \Phi \wedge \forall x : \Phi) \iff \forall x : (\Phi \wedge \Phi')$
- $(\exists x : \Phi \vee \forall x : \Phi) \iff \exists x : (\Phi \vee \Phi')$
- $(\Phi \Longrightarrow \Phi') \Longrightarrow (\exists x : \Phi \Longrightarrow \Phi')$ when $x \notin \mathrm{fv}(\Phi')$
- $(\Phi \Longrightarrow \Phi') \Longrightarrow (\Phi \Longrightarrow \forall x : \Phi')$ when $x \notin \mathrm{fv}(\Phi')$
- $\forall x : (\Phi \vee \Phi') \iff (\forall x : \Phi) \vee \Phi'$ when $x \notin \mathrm{fv}(\Phi')$
- $\exists x : (\Phi \wedge \Phi') \iff (\exists x : \Phi) \wedge \Phi'$ when $x \notin \mathrm{fv}(\Phi')$
- $\Phi \iff \forall x : \Phi$ when $x \notin \mathrm{fv}(\Phi')$
- $\Phi \iff \exists x : \Phi$ when $x \notin \mathrm{fv}(\Phi')$

Properties of the deduction system (H) for first-order logic

- The Hilbert style deductive system (H) is sound, consistent, compact [16] and complete [4] for the first-order-logic.

Reference

[4] Kurt Gödel. "Die Vollständigkeit der Axiome des logischen Funktionen-kalküls", *Monatshefte für Mathematik und Physik* 37 (1930), 349-360.

[16] $\Gamma \vdash \Phi$ if and only if $\Gamma' \vdash \Phi$ for a finite subset Γ' of Γ.

- The Hilbert style deductive system (H) is not decidable [5].
- Proofs cannot be fully automated: there is no terminating algorithm that, given a first-order formula Φ as input, returns true whenever Φ is classically valid.

Reference

[5] Kurt Gödel. "Über Formal Unentscheidbare Sätze der Principia Mathematica und Verwandter Systeme, I". *Monatshefte für Mathematik und Physik* 38, 173–198, 1931.

The theory axiomatizing equality

Writing $= \backslash 2(A, B)$ as $A = B$, the theory axiomatizing equality is first-order logic plus the following axioms:

- $\forall x : x = x$ reflexivity
- $\forall x : \forall y : (x = y) \Longrightarrow (y = x)$ symmetry
- $\forall x_1 : \ldots \forall x_n : \forall y_1 : \ldots \forall y_n : (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Longrightarrow (f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n))$ Leibnitz functional congruence
- $\forall x_1 : \ldots \forall x_n : \forall y_1 : \ldots \forall y_n : (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \Longrightarrow (r(x_1, \ldots, x_n) = r(y_1, \ldots, y_n))$ Leibnitz relational congruence
- $\forall x : \forall y : \forall z : (x = y \wedge y = z) \Longrightarrow (x = z)$ transitivity

Peano arithmetic [6]

– Constant symbols: 0
– Functional symbols: s (sucessor), $+$, $\times$
– Relation symbols: $=$, $\leq$
– Axioms:
  - $\forall x : x = x$ reflexivity
  - $\forall x : \forall y : (x = y) \Longrightarrow (y = x)$ symmetry
  - $\forall x : \forall y : \forall z : (x = y \wedge y = z) \Longrightarrow (x = z)$ transitivity

  - $\forall x : \forall y : (x = y) \Longrightarrow (s(x) = s(y))$ congruence
  - $\forall x : \forall y : \forall z : \forall t : (x = z \wedge t = t) \Longrightarrow (x + y = z + t)$
  - $\forall x : \forall y : \forall z : \forall t : (x = z \wedge t = t) \Longrightarrow (x \times y = z \times t)$
  - $\forall x : \forall y : \forall z : \forall t : (x = z \wedge t = t) \Longrightarrow (x \leq y = z \leq t)$
  - $\forall x : (x = 0) \vee (\exists y : x = s(y))$ every natural but 0 is a successor
  - $\forall x : \neg(s(x) = 0)$ 0 is not a successor
  - $\forall x : \forall y : (s(x) = s(y)) \Longrightarrow (x = y)$ s is injective so every nonzero natural has a unique predecessor
  - $\forall x : x + 0 = x$ def. addition
  - $\forall x : \forall y : s + s(y) = s(x + y)$

  - $\forall x : x \times 0 = 0$ def. multiplication
  - $\forall x : \forall y : x \times s(y) = (x \times y) + x$
  - $((\Phi)[x := 0] \wedge (\forall x : \Phi \Longrightarrow (\Phi)[x := s(x)]) \Longrightarrow (\forall x : \Phi)$ recurrence (for all instances of Φ)

Reference

[6] Giuseppe Peano. Arithmetices principia, nova methodo exposita. Augustae Taurinorum, Ed. Fratres Bocca, 1889. – XVI, 20 p.

Non standard integers

This axiomatization formalizes natural numbers but does not exclude "non standard models" of the form:

$0\ 1\ 2\ 3 \ldots \ \ldots -2^0\ -1^0\ 0^0\ 1^0\ 2^0 \ldots\ -2^1\ -1^1\ 0^1\ 1^1\ 2^1 \ldots\ -2^2\ -1^2\ 0^2\ 1^2\ 2^2 \ldots\ \ldots$

Excluded by the second-order logic induction axiom [17]:

$$\forall P : (P(0) \wedge (\forall x : P(x) \Longrightarrow P(s(x)))) \Longrightarrow \forall x : P$$

[17] The difference is that there is a denumerable infinity of instances of Φ while there can be a non-denumerable infinity of Ps, see G.S.Boolos and R.C.Jeffrey, "Computability and Logic", Cambridge University Press, 1974, 1980, 1989, Section 17, pp.193-195.

THE END

My MIT web site is http://www.mit.edu/~cousot/

The course web site is http://web.mit.edu/afs/athena.mit.edu/course/16/16.399/www/.

