Logic
Part I: Classical Logic and Its Semantics

Max Schäfer

Formosan Summer School on Logic, Language, and Computation 2007

July 2, 2007

Principles of Classical Logic

- classical logic seeks to model valid reasoning
- starting from axioms which are evidently true, we try to infer valid (true) conclusions
- a formula of classical logic is perceived to have a definite truth value (true or false) no matter whether we can prove it or not
- example 1: the statement "$\sqrt{2}$ is irrational" is true (and not hard to prove)
- example 2: Fermat's last theorem is true (but was proved only in 1995, 357 years after it was posed)

Outline

Propositional Logic

First Order Logic

Outline

Example of an Informal Proof

Here is a possible proof of the statement "$\sqrt{2}$ is irrational" (abbreviated as Q):

> *Assume $\sqrt{2}$ is rational.*
> *Then we can write it as $\sqrt{2} = \frac{p}{q}$ where p and q are natural numbers without common divisor (except 1).*
> *Then $2 = \frac{p^2}{q^2}$, i.e. $2 \cdot q^2 = p^2$. Hence p^2 is even. But if the square of a natural number is even, then so is the number itself, thus p is even, say $p = 2r$ for some natural number r. This, again, gives $q^2 = 2r^2$, and by the same argument q must be even as well, contradicting our assumption.*
> *Hence $\sqrt{2}$ cannot be rational.*

Example of an Informal Proof

Here is a possible proof of the statement "$\sqrt{2}$ is irrational" (abbreviated as Q):

> *Assume $\sqrt{2}$ is rational.*
> *Then we can write it as $\sqrt{2} = \frac{p}{q}$ where p and q are natural numbers without common divisor (except 1). Then $2 = \frac{p^2}{q^2}$, i.e. $2 \cdot q^2 = p^2$. Hence p^2 is even. But if the square of a natural number is even, then so is the number itself, thus p is even, say $p = 2r$ for some natural number r. This, again, gives $q^2 = 2r^2$, and by the same argument q must be even as well, contradicting our assumption.*
> *Hence $\sqrt{2}$ cannot be rational.*

Observations

- in the proof we have used (among others) the statement "if the square of a natural number is even, then so is the number itself" (abbreviated as P)
- what we have proved is the truth of the *implication* $P \rightarrow Q$
- that is, if P is true then so is Q; but if P is false, our proof is useless (though $P \rightarrow Q$ is still true!)
- in fact, P can be shown to be true; hence the *conjunction* $P \wedge Q$ is true
- if we let R stand for the statement "$\sqrt{2}$ is rational", then P is the *negation* of R (i.e., P expresses that R is false)
- even without any proof, we know that at least one of P and R must be true; thus, the *disjunction* $P \vee R$ is true
- a disjunction does not exclude the possibility that *both* disjuncts are true

The Approach of Propositional Logic

- propositional logic formalizes reasoning about statements
- *propositional letters* represent atomic statements without further structure
- more complex statements can be formed by connectives like $\wedge, \vee, \rightarrow, \neg$
- propositional logic is *not* sufficient to formalize mathematics, but it provides a good starting point

The Language of Propositional Logic

- *formulas* express true or false *propositions* over an alphabet $\mathcal{R}^{(0)}$ of *propositional letters*
- the set PF of propositional formulas is defined inductively:
  - every constant from $\mathcal{R}^{(0)}$ is a formula, called *atomic proposition*
  - if φ, ψ are formulas then
    - $\varphi \wedge \psi$ is a formula
    - $\varphi \vee \psi$ is a formula
    - $\varphi \rightarrow \psi$ is a formula
  - $\perp$ is a formula
- additionally, we define
  - $\neg\varphi := \varphi \rightarrow \perp$
  - $\varphi \leftrightarrow \psi := (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
  - $\top := \perp \rightarrow \perp$

Syntactic Conventions

- we use $\equiv$ to denote syntactic equality of formulas
- to save on parentheses, we take $\leftrightarrow$ to have the lowest precedence, followed by $\rightarrow$, $\vee$, $\wedge$, and $\neg$
- thus, $a \rightarrow b \vee c \leftrightarrow c \rightarrow a \vee b \wedge c$ is to be interpreted as $((a \rightarrow (b \vee c)) \leftrightarrow (c \rightarrow (a \vee (b \wedge c))))$
- $\rightarrow$ associates to the right, i.e. $a \rightarrow b \rightarrow c$ is $a \rightarrow (b \rightarrow c)$
- all the other binary operators associate to the left, i.e. $a \wedge b \wedge c$ is $(a \wedge b) \wedge c$

Subformulas

We define the set of subformulas $\mathrm{Sub}(\varphi)$ for a formula φ by structural induction on φ.

- if φ is atomic or $\varphi \equiv \bot$, then

$$\mathrm{Sub}(\varphi) = \{\varphi\}$$

- if φ is of the form $\vartheta \wedge \psi$, $\vartheta \vee \psi$, or $\vartheta \to \psi$, then

$$\mathrm{Sub}(\varphi) = \mathrm{Sub}(\vartheta) \cup \mathrm{Sub}(\psi) \cup \{\varphi\}$$

  Note that $\mathrm{Sub}(\vartheta)$ and $\mathrm{Sub}(\psi)$ are known by induction hypothesis.

We can now define the set of propositional letters occurring in a formula $\mathrm{PL}(\varphi) := \mathrm{Sub}(\varphi) \cap \mathcal{R}^{(0)}$.

Motivation: Truth Value Semantics

- in propositional logic, we do not care what specific statements the propositional letters stand for
- so we cannot know, e.g., whether p is true
- but for some formulas, it seems clear that they are true, e.g. $p \vee \neg p$ or $p \rightarrow p$
- idea: for a formula to be true means that it is true no matter if the propositional letters express true or false statements

Truth Value Semantics

Interpreting propositional formulas over $\mathbb{B} = \{\mathtt{T}, \mathtt{F}\}$:

- a propositional interpretation $I \colon \mathcal{R}^{(0)} \to \mathbb{B}$ classifies propositional letters as true (those mapped to $\mathtt{T}$) or false (those mapped to $\mathtt{F}$)
- given an interpretation I, we can assign a truth value $[\![\varphi]\!]_I$ to every formula φ:
  1. for $a \in \mathcal{R}^{(0)}$, $[\![a]\!]_I = I(a)$
  2. $[\![\varphi \wedge \psi]\!]_I = \begin{cases} \mathtt{T} & \text{if } [\![\varphi]\!]_I = \mathtt{T} = [\![\psi]\!]_I, \\ \mathtt{F} & \text{otherwise} \end{cases}$
  3. $[\![\varphi \vee \psi]\!]_I = \begin{cases} \mathtt{T} & \text{if } [\![\varphi]\!]_I = \mathtt{T} \text{ or } [\![\psi]\!]_I = \mathtt{T}, \\ \mathtt{F} & \text{otherwise} \end{cases}$
  4. $[\![\varphi \to \psi]\!]_I = \begin{cases} \mathtt{T} & \text{if } [\![\varphi]\!]_I = \mathtt{T} \text{ implies } [\![\psi]\!]_I = \mathtt{T}, \\ \mathtt{F} & \text{if } [\![\varphi]\!]_I = \mathtt{T} \text{ and } [\![\psi]\!]_I = \mathtt{F} \end{cases}$
  5. $[\![\bot]\!]_I = \mathtt{F}$

Observations about the Semantics

- for any interpretation I, we have
  - $[\![\neg\varphi]\!]_I = \mathtt{T}$ iff $[\![\varphi]\!]_I = \mathtt{F}$
  - $[\![\varphi \leftrightarrow \psi]\!]_I = \mathtt{T}$ iff $[\![\varphi]\!]_I = [\![\psi]\!]_I$
  - $[\![\top]\!]_I = \mathtt{T}$
- observe the connection between $\neg$ and $\rightarrow$:
  - if $[\![\neg\varphi]\!]_I = \mathtt{T}$, then $[\![\varphi \rightarrow \psi]\!]_I = \mathtt{T}$, no matter what ψ is
  - if $[\![\psi]\!]_I = \mathtt{T}$, then $[\![\varphi \rightarrow \psi]\!]_I = \mathtt{T}$, no matter what φ is
  - $[\![\varphi \rightarrow \psi]\!]_I = \mathtt{T}$ iff $[\![\neg\varphi \vee \psi]\!]_I = \mathtt{T}$
- we write $\varphi = \psi$ if, for any interpretation I, $[\![\varphi]\!]_I = [\![\psi]\!]_I$; for example, $\varphi \rightarrow \psi = \neg\varphi \vee \psi$
- "$=$" is an equivalence relation

Satisfiability and Validity

- if φ is true in I, then we write $I \models \varphi$ and say that I *satisfies* φ or that I is a *model* for φ
- a formula φ is *satisfiable* if it has a model
- a formula φ is *valid* (or a *tautology*) if it is satisfied in all interpretations; we then write $\models \varphi$
- for a set Γ of formulas, $I \models \Gamma$ means that I satisfies every formula in Γ
- we write $\Gamma \models \varphi$ to mean that any model for Γ is also a model for φ

Note that $\varphi = \psi$ iff $\models \varphi \leftrightarrow \psi$, and $\models \varphi$ iff $\varphi = \top$.

Important Equivalences

For propositional letters a, b, c, we have:

1. Associativity:
   - $a \wedge (b \wedge c) = (a \wedge b) \wedge c$
   - $a \vee (b \vee c) = (a \vee b) \vee c$
2. Commutativity:
   - $a \wedge b = b \wedge a$
   - $a \vee b = b \vee a$
3. Distributivity:
   - $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
   - $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
4. Absorption:

   $a \wedge (a \vee b) = a = a \vee (a \wedge b)$
5. Complement:
   - $a \vee \neg a = \top$
   - $a \wedge \neg a = \bot$

Further Equivalences

The following equivalences follow from those on the previous slide:

1. Idempotency:

   $a \vee a = a = a \wedge a$

2. Neutrality:
   - $a \vee \bot = a$
   - $a \wedge \top = a$

3. Boundedness:
   - $a \vee \top = \top$
   - $a \wedge \bot = \bot$

4. Switching:
   - $\neg\top = \bot$
   - $\neg\bot = \top$

5. De Morgan Laws:
   - $\neg(a \vee b) = \neg a \wedge \neg b$
   - $\neg(a \wedge b) = \neg a \vee \neg b$

6. Involution:

   $\neg\neg a = a$

Basic Results

Lemma (Replacement)

Let φ be a tautology and $a \in \mathcal{R}^{(0)}$. If we replace every occurrence of a in φ by a formula ψ, then the result is still a tautology.

Lemma (Monotonicity)

If $\Gamma \models \varphi$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \models \varphi$.

Lemma (Satisfiability and Validity)

A formula φ is satisfiable iff $\neg\varphi$ is not valid.

Lemma (Agreement)

For a formula φ and two interpretations I_1, I_2 such that $I_1 \big|_{\mathrm{PL}(\varphi)} = I_2 \big|_{\mathrm{PL}(\varphi)}$, we have $[\![\varphi]\!]_{I_1} = [\![\varphi]\!]_{I_2}$.

Basic Results

Lemma (Replacement)

Let φ be a tautology and $a \in \mathcal{R}^{(0)}$. If we replace every occurrence of a in φ by a formula ψ, then the result is still a tautology.

Lemma (Monotonicity)

If $\Gamma \models \varphi$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \models \varphi$.

Lemma (Satisfiability and Validity)

A formula φ is satisfiable iff $\neg\varphi$ is not valid.

Lemma (Agreement)

For a formula φ and two interpretations I_1, I_2 such that $I_1\big|_{\mathrm{PL}(\varphi)} = I_2\big|_{\mathrm{PL}(\varphi)}$, we have $[\![\varphi]\!]_{I_1} = [\![\varphi]\!]_{I_2}$.

Basic Results

Lemma (Replacement)

Let φ be a tautology and $a \in \mathcal{R}^{(0)}$. If we replace every occurrence of a in φ by a formula ψ, then the result is still a tautology.

Lemma (Monotonicity)

If $\Gamma \models \varphi$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \models \varphi$.

Lemma (Satisfiability and Validity)

A formula φ is satisfiable iff $\neg\varphi$ is not valid.

Lemma (Agreement)

For a formula φ and two interpretations I_1, I_2 such that $I_1 \big|_{\mathrm{PL}(\varphi)} = I_2 \big|_{\mathrm{PL}(\varphi)}$, we have $[\![\varphi]\!]_{I_1} = [\![\varphi]\!]_{I_2}$.

Basic Results

Lemma (Replacement)

Let φ be a tautology and $a \in \mathcal{R}^{(0)}$. If we replace every occurrence of a in φ by a formula ψ, then the result is still a tautology.

Lemma (Monotonicity)

If $\Gamma \models \varphi$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \models \varphi$.

Lemma (Satisfiability and Validity)

A formula φ is satisfiable iff $\neg\varphi$ is not valid.

Lemma (Agreement)

For a formula φ and two interpretations I_1, I_2 such that $I_1\big|_{\mathrm{PL}(\varphi)} = I_2\big|_{\mathrm{PL}(\varphi)}$, we have $[\![\varphi]\!]_{I_1} = [\![\varphi]\!]_{I_2}$.

Decidability of Validity

Theorem (Decidability of Validity)

It is decidable whether a formula φ is valid.

Proof: We only need to check all interpretations of $\mathrm{PL}(\varphi)$.

Corollary (Decidability of Satisfiability)

It is decidable whether a formula φ is satisfiable.

Theorem (NP-completeness of Satisfiability)

It is NP-complete to decide whether a formula φ is satisfiable.

Decidability of Validity

Theorem (Decidability of Validity)

It is decidable whether a formula φ is valid.

Proof: We only need to check all interpretations of $\mathrm{PL}(\varphi)$.

Corollary (Decidability of Satisfiability)

It is decidable whether a formula φ is satisfiable.

Theorem (NP-completeness of Satisfiability)

It is NP-complete to decide whether a formula φ is satisfiable.

Decidability of Validity

Theorem (Decidability of Validity)

It is decidable whether a formula φ is valid.

Proof: We only need to check all interpretations of $\mathrm{PL}(\varphi)$.

Corollary (Decidability of Satisfiability)

It is decidable whether a formula φ is satisfiable.

Theorem (NP-completeness of Satisfiability)

It is NP-complete to decide whether a formula φ is satisfiable.

Truth Tabling

A truth table for a formula φ represents all interpretations $I\big|_{\mathrm{PL}(\varphi)}$ and shows whether φ is true in I.

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

p	q	$p \vee q$
F	F	F
F	T	T
T	F	T
T	T	T

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

Satisfiability and validity of a formula can be read off its truth table.

Truth Tabling: Example

a	b	c	$a \vee b \vee \neg c$	$\neg b \vee \neg(c \vee a)$	$(a \vee b \vee \neg c) \wedge (\neg b \vee \neg(c \vee a)$
F	F	F	T	T	T
F	F	T	F	T	F
F	T	F	T	T	T
F	T	T	T	F	F
T	F	F	T	T	T
T	F	T	T	T	T
T	T	F	T	F	F
T	T	T	T	F	F

Negation Normal Form

A formula φ is in *negation normal form* (NNF) if every negation sign occurs in front of a propositional letter.

Theorem

Every formula is semantically equivalent to a formula in NNF.

Proof.

To bring a formula into NNF, push negations inwards using De Morgan, and if necessary eliminate double negations by involution:

$$\begin{aligned}
\neg(a \vee \neg(\neg(\neg b \vee a) \wedge c)) &= \neg a \wedge \neg\neg(\neg(\neg b \vee a) \wedge c) \\
&= \neg a \wedge \neg(\neg b \vee a) \wedge c \\
&= \neg a \wedge \neg\neg b \wedge \neg a \wedge c \\
&= \neg a \wedge b \wedge \neg a \wedge c
\end{aligned}$$

$\square$

Disjunctive Normal Form

- An atomic formula is also called *positive literal*, a negated atom *negative literal*
- A formula φ is in *disjunctive normal form* (DNF) if it is a disjunction of conjunctions of literals, i.e. $\varphi \equiv D_1 \vee \cdots \vee D_n$, where $n \geq 1$ and for any $i \in \{1, \ldots, n\}$ we have $D_i \equiv l_{i,1} \wedge \ldots \wedge l_{i,m_i}$ with $m_i \geq 1$ and all the $l_{i,j}$ being literals. As a limiting case, we also consider $\perp$ to be in DNF.
- A formula φ is in *canonical DNF* if it is in DNF, and every disjunct D_i contains every $a \in \mathrm{PL}(\varphi)$ exactly once. Again, $\perp$ is also considered to be in canonical DNF.

Theorem

Every formula is semantically equivalent to a formula in canonical DNF.

Disjunctive Normal Form

- An atomic formula is also called *positive literal*, a negated atom *negative literal*
- A formula φ is in *disjunctive normal form* (DNF) if it is a disjunction of conjunctions of literals, i.e. $\varphi \equiv D_1 \vee \cdots \vee D_n$, where $n \geq 1$ and for any $i \in \{1, \ldots, n\}$ we have $D_i \equiv l_{i,1} \wedge \ldots \wedge l_{i,m_i}$ with $m_i \geq 1$ and all the $l_{i,j}$ being literals. As a limiting case, we also consider $\perp$ to be in DNF.
- A formula φ is in *canonical DNF* if it is in DNF, and every disjunct D_i contains every $a \in \mathrm{PL}(\varphi)$ exactly once. Again, $\perp$ is also considered to be in canonical DNF.

Theorem

Every formula is semantically equivalent to a formula in canonical DNF.

Disjunctive Normal Form

- An atomic formula is also called *positive literal*, a negated atom *negative literal*
- A formula φ is in *disjunctive normal form* (DNF) if it is a disjunction of conjunctions of literals, i.e. $\varphi \equiv D_1 \vee \cdots \vee D_n$, where $n \geq 1$ and for any $i \in \{1, \ldots, n\}$ we have $D_i \equiv l_{i,1} \wedge \ldots \wedge l_{i,m_i}$ with $m_i \geq 1$ and all the $l_{i,j}$ being literals. As a limiting case, we also consider $\perp$ to be in DNF.
- A formula φ is in *canonical DNF* if it is in DNF, and every disjunct D_i contains every $a \in \mathrm{PL}(\varphi)$ exactly once. Again, $\perp$ is also considered to be in canonical DNF.

Theorem

Every formula is semantically equivalent to a formula in canonical DNF.

Disjunctive Normal Form

- An atomic formula is also called *positive literal*, a negated atom *negative literal*
- A formula φ is in *disjunctive normal form* (DNF) if it is a disjunction of conjunctions of literals, i.e. $\varphi \equiv D_1 \vee \cdots \vee D_n$, where $n \geq 1$ and for any $i \in \{1, \ldots, n\}$ we have $D_i \equiv l_{i,1} \wedge \ldots \wedge l_{i,m_i}$ with $m_i \geq 1$ and all the $l_{i,j}$ being literals. As a limiting case, we also consider $\perp$ to be in DNF.
- A formula φ is in *canonical DNF* if it is in DNF, and every disjunct D_i contains every $a \in \mathrm{PL}(\varphi)$ exactly once. Again, $\perp$ is also considered to be in canonical DNF.

Theorem

Every formula is semantically equivalent to a formula in canonical DNF.

Properties of DNF

Examples:

- $(a \wedge b) \vee (b \wedge \neg c)$ is in (non-canonical) DNF
- $(a \wedge b \wedge c) \vee (a \wedge b \wedge \neg c) \vee (\neg a \wedge b \wedge \neg c)$ is in canonical DNF

Properties of DNF

Examples:

- $(a \wedge b) \vee (b \wedge \neg c)$ is in (non-canonical) DNF
- $\underbrace{(a \wedge b \wedge c)}_{D_1} \vee \underbrace{(a \wedge b \wedge \neg c)}_{D_2} \vee \underbrace{(\neg a \wedge b \wedge \neg c)}_{D_3}$ is in canonical DNF

Properties of DNF

Examples:

- $(a \wedge b) \vee (b \wedge \neg c)$ is in (non-canonical) DNF
- $$\underbrace{(\overbrace{a}^{l_{1,1}} \wedge \overbrace{b}^{l_{1,2}} \wedge \overbrace{c}^{l_{1,3}})}_{D_1} \vee \underbrace{(\overbrace{a}^{l_{2,1}} \wedge \overbrace{b}^{l_{2,2}} \wedge \overbrace{\neg c}^{l_{2,3}})}_{D_2} \vee \underbrace{(\overbrace{\neg a}^{l_{3,1}} \wedge \overbrace{b}^{l_{3,2}} \wedge \overbrace{\neg c}^{l_{3,3}})}_{D_3}$$
  is in canonical DNF

Properties of DNF

Examples:

- $(a \wedge b) \vee (b \wedge \neg c)$ is in (non-canonical) DNF
- $\underbrace{(a \wedge b \wedge c)}_{D_1} \vee \underbrace{(a \wedge b \wedge \neg c)}_{D_2} \vee \underbrace{(\neg a \wedge b \wedge \neg c)}_{D_3}$ is in canonical DNF

Let φ be in DNF and I an interpretation; observe:

- φ is true in I if one (or more) of the D_i are
- some D_i is true in I if I makes its positive literals true and its negative literals false
- φ is unsatisfiable iff it is $\perp$, or every D_i contains both a and $\neg a$ for some $a \in \mathrm{PL}(\varphi)$
- this leads to a method to extract a canonical DNF from a truth table

DNF from Truth Table: Example

a	b	c	$(a \vee b \vee \neg c) \wedge (\neg b \vee \neg(c \vee a))$	contributed disjunct
F	F	F	T	$\neg a \wedge \neg b \wedge \neg c$
F	F	T	F	
F	T	F	T	$\neg a \wedge b \wedge \neg c$
F	T	T	F	
T	F	F	T	$a \wedge \neg b \wedge \neg c$
T	F	T	T	$a \wedge \neg b \wedge c$
T	T	F	F	
T	T	T	F	

Thus, a canonical DNF of $(a \vee b \vee \neg c) \wedge (\neg b \vee \neg(c \vee a))$ is

$$(\neg a \wedge \neg b \wedge \neg c) \vee (\neg a \wedge b \wedge \neg c) \vee (a \wedge \neg b \wedge \neg c) \vee (a \wedge \neg b \wedge c)$$

Expressibility

- every formula can be expressed in terms of $\neg$, $\vee$ and $\wedge$, thus $\{\neg, \vee, \wedge\}$ is a *functionally complete* set
- but $p \wedge q = \neg\neg(p \wedge q) = \neg(\neg p \vee \neg q)$, hence $\{\neg, \vee\}$ suffice
- other functionally complete sets: $\{\neg, \wedge\}$, $\{\rightarrow, \bot\}$, $\{\neg, \rightarrow\}$, ...
- there are also two operators that are functionally complete by themselves; a nand $b := \neg(a \wedge b)$, a nor $b := \neg(a \vee b)$

Expressibility

- every formula can be expressed in terms of $\neg$, $\vee$ and $\wedge$, thus $\{\neg, \vee, \wedge\}$ is a *functionally complete* set
- but $p \wedge q = \neg\neg(p \wedge q) = \neg(\neg p \vee \neg q)$, hence $\{\neg, \vee\}$ suffice
- other functionally complete sets: $\{\neg, \wedge\}$, $\{\rightarrow, \bot\}$, $\{\neg, \rightarrow\}$, ...
- there are also two operators that are functionally complete by themselves; a nand $b := \neg(a \wedge b)$, a nor $b := \neg(a \vee b)$

Expressibility

- every formula can be expressed in terms of $\neg$, $\vee$ and $\wedge$, thus $\{\neg, \vee, \wedge\}$ is a *functionally complete* set
- but $p \wedge q = \neg\neg(p \wedge q) = \neg(\neg p \vee \neg q)$, hence $\{\neg, \vee\}$ suffice
- other functionally complete sets: $\{\neg, \wedge\}$, $\{\rightarrow, \bot\}$, $\{\neg, \rightarrow\}$, ...
- there are also two operators that are functionally complete by themselves: a nand $b := \neg(a \wedge b)$, a nor $b := \neg(a \vee b)$

Expressibility

- every formula can be expressed in terms of $\neg$, $\vee$ and $\wedge$, thus $\{\neg, \vee, \wedge\}$ is a *functionally complete* set
- but $p \wedge q = \neg\neg(p \wedge q) = \neg(\neg p \vee \neg q)$, hence $\{\neg, \vee\}$ suffice
- other functionally complete sets: $\{\neg, \wedge\}$, $\{\rightarrow, \bot\}$, $\{\neg, \rightarrow\}$, ...
- there are also two operators that are functionally complete by themselves; a nand $b := \neg(a \wedge b)$, a nor $b := \neg(a \vee b)$

Boolean Algebras

A *Boolean algebra* is an algebraic structure $\mathcal{B} = \langle B, \sqcup, \sqcap, -, 0, 1 \rangle$ where

- B is a set, $\sqcup$ and $\sqcap$ are binary operations on B, and $-$ is a unary operation on B, 0 and 1 are distinct elements of B
- $\sqcup$ and $\sqcap$ are associative and commutative
- the *absorption* laws hold:

$$a \sqcup (a \sqcap b) = a \qquad a \sqcap (a \sqcup b) = a$$

- $\sqcup$ distributes over $\sqcap$ and vice versa
- the *complement* laws hold:

$$a \sqcup -a = 1 \qquad a \sqcap -a = 0$$

Examples of Boolean Algebras

- the truth value algebra $\underline{\mathbb{B}} = \langle \mathbb{B}, \vee, \wedge, \neg, \mathtt{F}, \mathtt{T} \rangle$
- $\underline{\mathbf{2}} = \langle \{0, 1\}, \max, \min, (x \mapsto 1 - x), 0, 1 \rangle$
- for any non-empty set X, $\mathbf{P}_X = \langle \mathcal{P}(X), \cup, \cap, \bar{\cdot}, \emptyset, P \rangle$
- thus, Boolean algebras need not be finite; we cannot necessarily use truth tables

Algebraic Semantics of Classical Propositional Logic

Given a Boolean algebra $\mathcal{B}$ and an interpretation $I : \mathcal{R}^{(0)} \to B$, we can assign to every propositional formula φ a value $[\![\varphi]\!]_{\mathcal{B},I}$ in B

1. for $p \in V$, $[\![p]\!]_{\mathcal{B},I} = I(p)$
2. $[\![\bot]\!]_{\mathcal{B},I} = 0$
3. $[\![\varphi \wedge \psi]\!]_{\mathcal{B},I} = [\![\varphi]\!]_{\mathcal{B},I} \sqcap [\![\psi]\!]_{\mathcal{B},I}$
4. $[\![\varphi \vee \psi]\!]_{\mathcal{B},I} = [\![\varphi]\!]_{\mathcal{B},I} \sqcup [\![\psi]\!]_{\mathcal{B},I}$
5. $[\![\varphi \to \psi]\!]_{\mathcal{B},I} = -[\![\varphi]\!]_{\mathcal{B},I} \sqcup [\![\psi]\!]_{\mathcal{B},I}$

Universality of the Truth Value Algebra

We can generalize satisfaction and validity:

- define $I \models_{\mathcal{B}} \varphi$ to mean that $[\![\varphi]\!]_{\mathcal{B},I} = 1$
- $\models_{\mathcal{B}} \varphi$, $\Gamma \models_{\mathcal{B}} \varphi$ are defined analogously

The truth value algebra $\underline{\mathbb{B}}$ is universal:

Theorem

For any formula φ, we have $\models_{\underline{\mathbb{B}}} \varphi$ iff $\models_{\mathcal{B}} \varphi$ for all Boolean Algebras $\mathcal{B}$.

Outline

Propositional Logic

First Order Logic

Motivation: First Order Logic

- in mathematics, we want to express propositions about individuals, e.g.

  For every n, if $n > 0$ then for all m we have $m + n > m$.

- in the example, the individuals are numbers, ranged over by variables n, m
- we use constants (like 0) and functions (like $+$, arity 2) to construct *terms*
- relations (like $>$, arity 2) can be used to form *atomic propositions* about terms
- atomic propositions are used to construct more complex propositions
- first order logic (FOL) formalizes such statements in an abstract setting

The Approach of First Order Logic

- first order logic formalizes reasoning about statements that can refer to individuals through *individual variables*
- a fixed set of function symbols acts on the individuals
- a fixed set of relation symbols expresses predicates on the individuals
- more complex statements can be formed by connectives like $\wedge, \vee, \rightarrow, \neg$ and the quantifiers $\forall, \exists$
- first order logic is sufficient to formalize great parts of mathematics, for example arithmetic (but not analysis)

The Language of FOL

- a first order *signature* $\Sigma = \langle \mathcal{F}, \mathcal{R} \rangle$ describes a language with
  - *function constants* $f \in \mathcal{F}$ with arity $\alpha(f) \in \mathbb{N}$
  - *relation constants* $r \in \mathcal{R}$ with arity $\alpha(r) \in \mathbb{N}$
- we write f/n to mean $\alpha(f) = n$, and $\mathcal{F}^{(n)} := \{f/n \mid f \in \mathcal{F}\}$, same for $\mathcal{R}^{(n)}$.
- terms $\mathcal{T}(\Sigma, \mathcal{V})$ over Σ and a set $\mathcal{V}$ of *individual variables* are inductively defined:
  - $\mathcal{V} \subseteq \mathcal{T}(\Sigma, \mathcal{V})$
  - for $f/n \in \mathcal{F}$, $t_1, \ldots, t_n \in \mathcal{T}(\Sigma, \mathcal{V})$, also $f(t_1, \ldots, t_n) \in \mathcal{T}(\Sigma, \mathcal{V})$
- for a 0-ary constant d, we write $d()$ simply as d

Example

Consider the signature $\Sigma = \langle \mathcal{F}, \mathcal{R} \rangle$ with $\mathcal{F} = \{0/0, s/1, +/2\}$ and $\mathcal{R} = \{\approx/2, \leq/2, </2\}$.

- examples for terms over Σ and $\mathcal{V} := \{x, y\}$ are 0, $s(0)$, $s(s(0))$, ..., $s(x)$, $+(s(x), y)$, $s(+(x, y))$, ...
- but not $0(0)$ or $+(1)$
- $+(x, y)$ is usually written infix as $x + y$, but this is purely syntactic sugar

Example

Consider the signature $\Sigma = \langle \mathcal{F}, \mathcal{R} \rangle$ with $\mathcal{F} = \{0/0, s/1, +/2\}$ and $\mathcal{R} = \{\approx/2, \leq/2, </2\}$.

- examples for terms over Σ and $\mathcal{V} := \{x, y\}$ are 0, $s(0)$, $s(s(0))$, $\ldots$, $s(x)$, $+(s(x), y)$, $s(+(x, y))$, $\ldots$
- but not $0(0)$ or $+(1)$
- $+(x, y)$ is usually written infix as $x + y$, but this is purely syntactic sugar

Example

Consider the signature $\Sigma = \langle \mathcal{F}, \mathcal{R} \rangle$ with $\mathcal{F} = \{0/0, s/1, +/2\}$ and $\mathcal{R} = \{\approx/2, \leq/2, </2\}$.

- examples for terms over Σ and $\mathcal{V} := \{x, y\}$ are 0, $s(0)$, $s(s(0))$, ..., $s(x)$, $+(s(x), y)$, $s(+(x, y))$, ...
- but not $0(0)$ or $+(1)$
- $+(x, y)$ is usually written infix as $x + y$, but this is purely syntactic sugar

The Language of FOL (II)

- an *atom* is of the form $r(t_1, \ldots, t_n)$, where $r/n \in \mathcal{R}$, $t_1, \ldots, t_n \in \mathcal{T}(\Sigma, \mathcal{V})$; like before we write just r if $\alpha(r) = 0$
- formulas are inductively defined:
  - every atom is a formula
  - if φ, ψ are formulas then
    - $\varphi \wedge \psi$ is a formula
    - $\varphi \vee \psi$ is a formula
    - $\varphi \rightarrow \psi$ is a formula
  - if $x \in \mathcal{V}$ and φ is a formula, then
    - $\forall x.\varphi$ is a formula
    - $\exists x.\varphi$ is a formula
  - $\perp$ is a formula

The quantifiers $\forall$ and $\exists$ have the lowest precedence of all connectives.

Example

Taking the signature Σ and $\mathcal{V}$ from before, the following are atoms (again, we use infix notation):

- $x \approx y$
- $x < s(x)$
- $x + y \approx y + x$

And here are some formulas:

- $\neg(x \approx s(x))$
- $(x < y) \rightarrow (s(x) < y \vee s(x) \approx y)$
- $\forall n.n > 0 \rightarrow (\forall m.m < m + n)$

Example

Taking the signature Σ and $\mathcal{V}$ from before, the following are atoms (again, we use infix notation):

- $x \approx y$
- $x < s(x)$
- $x + y \approx y + x$

And here are some formulas:

- $\neg(x \approx s(x))$
- $(x < y) \rightarrow (s(x) < y \vee s(x) \approx y)$
- $\forall n.n > 0 \rightarrow (\forall m.m < m + n)$

Free and Bound Variables

- an appearance of an individual variable is called *bound* if it is within the scope of a quantifier, otherwise it is *free*
- the same variable can appear both free and bound:

$$(\forall x.R(x,z) \rightarrow (\exists y.S(y,x))) \wedge T(x)$$

- a formula is called *closed* when no variable occurs free in it
- the names of bound variables only serve to connect them with their quantifier, one name is as good as another (details later)

The Set of Free Variables

- definition of the set of free variables:
  1. $\mathrm{FV}(x) = \{x\}$ for $x \in \mathcal{V}$
  2. $\mathrm{FV}(f(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
  3. $\mathrm{FV}(r(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
  4. $\mathrm{FV}(\bot) = \emptyset$
  5. $\mathrm{FV}(\varphi \wedge \psi) = \mathrm{FV}(\varphi \vee \psi) = \mathrm{FV}(\varphi \rightarrow \psi) = \mathrm{FV}(\varphi) \cup \mathrm{FV}(\psi)$
  6. $\mathrm{FV}(\forall x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$
  7. $\mathrm{FV}(\exists x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$

For example:

- $\mathrm{FV}(s(x) \approx 0 \vee x \approx x) = \{x\}$
- $\mathrm{FV}(\forall m.\exists n.m < n) = \emptyset$
- $\mathrm{FV}(\exists x.x < y) = \{y\}$

The Set of Free Variables

- definition of the set of free variables:
  1. $\mathrm{FV}(x) = \{x\}$ for $x \in \mathcal{V}$
  2. $\mathrm{FV}(f(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
  3. $\mathrm{FV}(r(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
  4. $\mathrm{FV}(\bot) = \emptyset$
  5. $\mathrm{FV}(\varphi \wedge \psi) = \mathrm{FV}(\varphi \vee \psi) = \mathrm{FV}(\varphi \rightarrow \psi) = \mathrm{FV}(\varphi) \cup \mathrm{FV}(\psi)$
  6. $\mathrm{FV}(\forall x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$
  7. $\mathrm{FV}(\exists x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$

For example:

- $\mathrm{FV}(s(x) \approx 0 \vee x \approx x) = \{x\}$
- $\mathrm{FV}(\forall m.\exists n.m < n) = \emptyset$
- $\mathrm{FV}(\exists x.x < y) = \{y\}$

The Set of Free Variables

- definition of the set of free variables:
  1. $\mathrm{FV}(x) = \{x\}$ for $x \in \mathcal{V}$
  2. $\mathrm{FV}(f(t_1, \dots, t_n)) = \bigcup_{i \in \{1,\dots,n\}} \mathrm{FV}(t_i)$
  3. $\mathrm{FV}(r(t_1, \dots, t_n)) = \bigcup_{i \in \{1,\dots,n\}} \mathrm{FV}(t_i)$
  4. $\mathrm{FV}(\perp) = \emptyset$
  5. $\mathrm{FV}(\varphi \wedge \psi) = \mathrm{FV}(\varphi \vee \psi) = \mathrm{FV}(\varphi \rightarrow \psi) = \mathrm{FV}(\varphi) \cup \mathrm{FV}(\psi)$
  6. $\mathrm{FV}(\forall x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$
  7. $\mathrm{FV}(\exists x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$

For example:

- $\mathrm{FV}(s(x) \approx 0 \vee x \approx x) = \{x\}$
- $\mathrm{FV}(\forall m.\exists n.m < n) = \emptyset$
- $\mathrm{FV}(\exists x.x < y) = \{y\}$

The Set of Free Variables

- definition of the set of free variables:
    1. $\mathrm{FV}(x) = \{x\}$ for $x \in \mathcal{V}$
    2. $\mathrm{FV}(f(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
    3. $\mathrm{FV}(r(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} \mathrm{FV}(t_i)$
    4. $\mathrm{FV}(\bot) = \emptyset$
    5. $\mathrm{FV}(\varphi \wedge \psi) = \mathrm{FV}(\varphi \vee \psi) = \mathrm{FV}(\varphi \rightarrow \psi) = \mathrm{FV}(\varphi) \cup \mathrm{FV}(\psi)$
    6. $\mathrm{FV}(\forall x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$
    7. $\mathrm{FV}(\exists x.\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$

For example:

- $\mathrm{FV}(s(x) \approx 0 \vee x \approx x) = \{x\}$
- $\mathrm{FV}(\forall m.\exists n.m < n) = \emptyset$
- $\mathrm{FV}(\exists x.x < y) = \{y\}$

Substitution in Terms and Formulas

- the operation of substituting a term t for a variable x in a term s (written $[x := t]s$) is defined as follows:
  1. $[x := t]y = \begin{cases} t & \text{if } x \equiv y, \\ y & \text{otherwise} \end{cases}$
  2. $[x := t](f(t_1, \ldots, t_n)) = f([x := t]t_1, \ldots, [x := t]t_n)$
- on formulas, the definition is
  1. $[x := t](r(t_1, \ldots, t_n)) = r([x := t]t_1, \ldots, [x := t]t_n)$
  2. $[x := t]\bot = \bot$
  3. $[x := t](\varphi \circ \psi) = ([x := t]\varphi) \circ ([x := t]\psi)$, for $\circ \in \{\land, \lor, \to\}$
  4. $[x := t](Qy.\varphi) = \begin{cases} Qy.\varphi & \text{if } x \equiv y, \\ Qy.([x := t]\varphi) & \text{if } x \not\equiv y,\ y \notin \mathrm{FV}(t) \end{cases}$, $Q \in \{\forall, \exists\}$

Note that substitution on formulas is a partial operation.

Example

- $[x := s(0)](s(x) \approx 0 \lor x \approx x) \equiv (s(s(0)) \approx 0 \lor s(0) \approx s(0))$
- $[y := s(0)](s(x) \approx 0 \lor x \approx x) \equiv (s(x) \approx 0 \lor x \approx x)$
- $[m := 0](\forall m.\exists n.m < n) \equiv (\forall m.\exists n.m < n)$
- $[y := x](\exists x.x < y)$ is not defined
- $[n := m](\forall m.\exists n.m < n)$ is not defined

Example

- $[x := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(s(0)) \approx 0 \vee s(0) \approx s(0))$
- $[y := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(x) \approx 0 \vee x \approx x)$
- $[m := 0](\forall m.\exists n.m < n) \equiv (\forall m.\exists n.m < n)$
- $[y := x](\exists x.x < y)$ is not defined
- $[n := m](\forall m.\exists n.m < n)$ is not defined

Example

- $[x := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(s(0)) \approx 0 \vee s(0) \approx s(0))$
- $[y := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(x) \approx 0 \vee x \approx x)$
- $[m := 0](\forall m.\exists n.m < n) \equiv (\forall m.\exists n.m < n)$
- $[y := x](\exists x.x < y)$ is not defined
- $[n := m](\forall m.\exists n.m < n)$ is not defined

Example

- $[x := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(s(0)) \approx 0 \vee s(0) \approx s(0))$
- $[y := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(x) \approx 0 \vee x \approx x)$
- $[m := 0](\forall m.\exists n.m < n) \equiv (\forall m.\exists n.m < n)$
- $[y := x](\exists x.x < y)$ is not defined
- $[n := m](\forall m.\exists n.m < n)$ is not defined

Example

- $[x := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(s(0)) \approx 0 \vee s(0) \approx s(0))$
- $[y := s(0)](s(x) \approx 0 \vee x \approx x) \equiv (s(x) \approx 0 \vee x \approx x)$
- $[m := 0](\forall m.\exists n.m < n) \equiv (\forall m.\exists n.m < n)$
- $[y := x](\exists x.x < y)$ is not defined
- $[n := m](\forall m.\exists n.m < n)$ is not defined

Alpha Equivalence

- for a quantifier $Q \in \{\forall, \exists\}$, $Qx.\varphi$ *alpha reduces* to $Qy.\varphi'$ if $\varphi' \equiv [x := y]\varphi$
- φ is called *alpha equivalent* to ψ (written $\varphi =_\alpha \psi$), if ψ results from φ by any number of alpha reductions on subformulas of φ
- Examples:
  - $(\forall x.R(x, x)) =_\alpha (\forall y.R(y, y))$
  - $(\forall x.\exists x.S(x)) =_\alpha (\forall y.\exists x.S(x)) =_\alpha (\forall y.\exists z.S(z))$
  - $(\forall x.\exists y.T(x, y)) \neq_\alpha (\forall x.\exists x.T(x, x))$

Notice that alpha reduction *never* changes the names of free variables.

Alpha Equivalence

- for a quantifier $Q \in \{\forall, \exists\}$, $Qx.\varphi$ *alpha reduces* to $Qy.\varphi'$ if $\varphi' \equiv [x := y]\varphi$
- φ is called *alpha equivalent* to ψ (written $\varphi =_\alpha \psi$), if ψ results from φ by any number of alpha reductions on subformulas of φ
- Examples:
  - $(\forall x.R(x,x)) =_\alpha (\forall y.R(y,y))$
  - $(\forall x.\exists x.S(x)) =_\alpha (\forall y.\exists x.S(x)) =_\alpha (\forall y.\exists z.S(z))$
  - $(\forall x.\exists y.T(x,y)) \neq_\alpha (\forall x.\exists x.T(x,x))$

Notice that alpha reduction *never* changes the names of free variables.

Alpha Equivalence

- for a quantifier $Q \in \{\forall, \exists\}$, $Qx.\varphi$ *alpha reduces* to $Qy.\varphi'$ if $\varphi' \equiv [x := y]\varphi$
- φ is called *alpha equivalent* to ψ (written $\varphi =_\alpha \psi$), if ψ results from φ by any number of alpha reductions on subformulas of φ
- Examples:
  - $(\forall x.R(x,x)) =_\alpha (\forall y.R(y,y))$
  - $(\forall x.\exists x.S(x)) =_\alpha (\forall y.\exists x.S(x)) =_\alpha (\forall y.\exists z.S(z))$
  - $(\forall x.\exists y.T(x,y)) \neq_\alpha (\forall x.\exists x.T(x,x))$

Notice that alpha reduction *never* changes the names of free variables.

Alpha Equivalence

- for a quantifier $Q \in \{\forall, \exists\}$, $Qx.\varphi$ *alpha reduces* to $Qy.\varphi'$ if $\varphi' \equiv [x := y]\varphi$
- φ is called *alpha equivalent* to ψ (written $\varphi =_\alpha \psi$), if ψ results from φ by any number of alpha reductions on subformulas of φ
- Examples:
  - $(\forall x.R(x,x)) =_\alpha (\forall y.R(y,y))$
  - $(\forall x.\exists x.S(x)) =_\alpha (\forall y.\exists x.S(x)) =_\alpha (\forall y.\exists z.S(z))$
  - $(\forall x.\exists y.T(x,y)) \neq_\alpha (\forall x.\exists x.T(x,x))$

Notice that alpha reduction *never* changes the names of free variables.

Alpha Equivalence

- for a quantifier $Q \in \{\forall, \exists\}$, $Qx.\varphi$ *alpha reduces* to $Qy.\varphi'$ if $\varphi' \equiv [x := y]\varphi$
- φ is called *alpha equivalent* to ψ (written $\varphi =_\alpha \psi$), if ψ results from φ by any number of alpha reductions on subformulas of φ
- Examples:
    - $(\forall x.R(x,x)) =_\alpha (\forall y.R(y,y))$
    - $(\forall x.\exists x.S(x)) =_\alpha (\forall y.\exists x.S(x)) =_\alpha (\forall y.\exists z.S(z))$
    - $(\forall x.\exists y.T(x,y)) \neq_\alpha (\forall x.\exists x.T(x,x))$

Notice that alpha reduction *never* changes the names of free variables.

Motivation: Semantics of FOL

- like in propositional logic, in FOL we do not care what functions or relations the symbols in Σ stand for
- thus, we do not know if $\forall m.\exists n.m < n$ is true
- but some sentences are intuitively true, e.g.

$$\begin{array}{ll} & (\forall x.\neg R(x,x)) \\ \wedge & (\forall y.\forall z.R(y,z) \to R(z,y)) \\ \wedge & (\forall x.\forall y.\forall z.R(x,y) \wedge R(y,z) \to R(x,z)) \\ \to & \neg(\exists x.\exists y.R(x,y)) \end{array}$$

- how do we evaluate, e.g., $\forall x.\neg R(x,x)$?
  - we need to know the range of x and the interpretation of R on this range
  - then we would like to evaluate $\neg R(x,x)$, where x is bound to any of its possible values
- thus, we need to consider not only the interpretation of the function and relation symbols, but also variable bindings

Motivation: Semantics of FOL

- like in propositional logic, in FOL we do not care what functions or relations the symbols in Σ stand for
- thus, we do not know if $\forall m.\exists n.m < n$ is true
- but some sentences are intuitively true, e.g.

$$\begin{array}{ll} & (\forall x.\neg R(x,x)) \\ \wedge & (\forall y.\forall z.R(y,z) \rightarrow R(z,y)) \\ \wedge & (\forall x.\forall y.\forall z.R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \\ \rightarrow & \neg(\exists x.\exists y.R(x,y)) \end{array}$$

- how do we evaluate, e.g., $\forall x.\neg R(x,x)$?
  - we need to know the range of x and the interpretation of R on this range
  - then we would like to evaluate $\neg R(x,x)$, where x is bound to any of its possible values
- thus, we need to consider not only the interpretation of the function and relation symbols, but also variable bindings

Motivation: Semantics of FOL

- like in propositional logic, in FOL we do not care what functions or relations the symbols in Σ stand for
- thus, we do not know if $\forall m.\exists n.m < n$ is true
- but some sentences are intuitively true, e.g.

$$\begin{array}{rl} & (\forall x.\neg R(x,x)) \\ \wedge & (\forall y.\forall z.R(y,z) \rightarrow R(z,y)) \\ \wedge & (\forall x.\forall y.\forall z.R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \\ \rightarrow & \neg(\exists x.\exists y.R(x,y)) \end{array}$$

- how do we evaluate, e.g., $\forall x.\neg R(x,x)$?
  - we need to know the range of x and the interpretation of R on this range
  - then we would like to evaluate $\neg R(x,x)$, where x is bound to any of its possible values
- thus, we need to consider not only the interpretation of the function and relation symbols, but also variable bindings

Motivation: Semantics of FOL

- like in propositional logic, in FOL we do not care what functions or relations the symbols in Σ stand for
- thus, we do not know if $\forall m.\exists n.m < n$ is true
- but some sentences are intuitively true, e.g.

$$
\begin{array}{ll}
 & (\forall x.\neg R(x,x)) \\
\wedge & (\forall y.\forall z.R(y,z) \rightarrow R(z,y)) \\
\wedge & (\forall x.\forall y.\forall z.R(x,y) \wedge R(y,z) \rightarrow R(x,z)) \\
\rightarrow & \neg(\exists x.\exists y.R(x,y))
\end{array}
$$

- how do we evaluate, e.g., $\forall x.\neg R(x,x)$?
  - we need to know the range of x and the interpretation of R on this range
  - then we would like to evaluate $\neg R(x,x)$, where x is bound to any of its possible values
- thus, we need to consider not only the interpretation of the function and relation symbols, but also variable bindings

Semantics: Structures, Interpretations and Assignments

- a (first order) structure $\mathcal{M} = \langle D, I \rangle$ for a signature Σ consists of
  - a non-empty set D, the *domain*
  - an interpretation $I = \langle [\![\cdot]\!]_{\mathcal{F}}, [\![\cdot]\!]_{\mathcal{R}} \rangle$ such that
    - for every $f \in \mathcal{F}^{(n)}$, $[\![f]\!]_{\mathcal{F}} \colon D^n \to D$
    - for every $r \in \mathcal{R}^{(n)}$, $[\![r]\!]_{\mathcal{R}} \colon D^n \to \mathbb{B}$
- a *variable assignment* on I is a function $\sigma \colon \mathcal{V} \to D$
  We write $\sigma[x := t]$ for the assignment

$$y \mapsto \begin{cases} t & \text{if } x \equiv y \\ \sigma(y) & \text{otherwise} \end{cases}$$

Semantics: Interpreting Terms and Formulas

- interpretation of terms over $\mathcal{M}$ and σ:
  - $[\![x]\!]_{\mathcal{M},\sigma} = \sigma(x)$
  - $[\![f(t_1, \ldots, t_n)]\!]_{\mathcal{M},\sigma} = [\![f]\!]_{\mathcal{F}}([\![t_1]\!]_{\mathcal{M},\sigma}, \ldots, [\![t_n]\!]_{\mathcal{M},\sigma})$
- interpretation of formulas:
  - $[\![r(t_1, \ldots, t_n)]\!]_{\mathcal{M},\sigma} = [\![r]\!]_{\mathcal{R}}([\![t_1]\!]_{\mathcal{M},\sigma}, \ldots, [\![t_n]\!]_{\mathcal{M},\sigma})$
  - $[\![\bot]\!]_{\mathcal{M},\sigma}$, $[\![\varphi \wedge \psi]\!]_{\mathcal{M},\sigma}$, etc.: as before
  - $[\![\forall x.\varphi]\!]_{\mathcal{M},\sigma} = \begin{cases} \mathtt{T} & \text{if, for all } d \in D\text{, } [\![\varphi]\!]_{\mathcal{M},\sigma[x:=d]} = \mathtt{T}, \\ \mathtt{F} & \text{otherwise} \end{cases}$
  - $[\![\exists x.\varphi]\!]_{\mathcal{M},\sigma} = \begin{cases} \mathtt{T} & \text{if there is } d \in D \text{ with } [\![\varphi]\!]_{\mathcal{M},\sigma[x:=d]} = \mathtt{T}, \\ \mathtt{F} & \text{otherwise} \end{cases}$

Satisfiability and Validity

- $\mathcal{M}, \sigma \models \varphi$ stands for $[\![\varphi]\!]_{\mathcal{M},\sigma} = \mathtt{T}$
- $\mathcal{M} \models \varphi$ means that $\mathcal{M}, \sigma \models \varphi$ for any σ; $\mathcal{M}$ is called a model for φ
- we write $\models \varphi$ (and call φ valid) if $\mathcal{M} \models \varphi$ for any structure $\mathcal{M}$
- $\Gamma \models \varphi$ now means that any $\mathcal{M}$ and σ such that $[\![\gamma]\!]_{\mathcal{M},\sigma} = \mathtt{T}$ for every $\gamma \in \Gamma$ also gives $[\![\varphi]\!]_{\mathcal{M},\sigma} = \mathtt{T}$
- analogously, $\varphi = \psi$ means that $[\![\varphi]\!]_{\mathcal{M},\sigma} = [\![\psi]\!]_{\mathcal{M},\sigma}$ for any $\mathcal{M}$ and σ

Example

Consider the structure $\mathcal{M} = \langle \mathbb{N}, \langle [\![\cdot]\!]_{\mathcal{F}}, [\![\cdot]\!]_{\mathcal{R}} \rangle \rangle$ for signature Σ as before:

- $[\![0]\!]_{\mathcal{F}} = 0$
- $[\![s]\!]_{\mathcal{F}}(n) = n + 1$
- $[\![+]\!]_{\mathcal{F}}(m, n) = m + n$
- $[\![\approx]\!]_{\mathcal{R}} = \{(n, n) \mid n \in \mathbb{N}\}$
- $[\![\leq]\!]_{\mathcal{R}} = \{(m, n) \mid m, n \in \mathbb{N}, m \leq n\}$
- $[\![<]\!]_{\mathcal{R}} = \{(m, n) \mid m, n \in \mathbb{N}, m < n\}$

Then $\mathcal{M} \models \forall n. n > 0 \rightarrow (\forall m. m < m + n)$.

Example

Consider the structure $\mathcal{M} = \langle \mathbb{N}, \langle [\![\cdot]\!]_{\mathcal{F}}, [\![\cdot]\!]_{\mathcal{R}} \rangle \rangle$ for signature Σ as before:

- $[\![0]\!]_{\mathcal{F}} = 0$
- $[\![s]\!]_{\mathcal{F}}(n) = n + 1$
- $[\![+]\!]_{\mathcal{F}}(m, n) = m + n$
- $[\![\approx]\!]_{\mathcal{R}} = \{(n, n) \mid n \in \mathbb{N}\}$
- $[\![\leq]\!]_{\mathcal{R}} = \{(m, n) \mid m, n \in \mathbb{N}, m \leq n\}$
- $[\![<]\!]_{\mathcal{R}} = \{(m, n) \mid m, n \in \mathbb{N}, m < n\}$

Then $\mathcal{M} \models \forall n. n > 0 \rightarrow (\forall m. m < m + n)$.

Basic Results

From now on, we fix some signature Σ and a set $\mathcal{V}$ of variables.

Lemma

Let $\mathcal{M}$ be a structure for Σ, φ a formula, and σ, σ' variable assignments that agree on $\mathrm{FV}(\varphi)$. Then φ is true over $\mathcal{M}$ and σ iff it is true over $\mathcal{M}$ and σ'.

Corollary

The interpretation of a closed formula is independent of variable assignments.

Lemma

Alpha equivalent formulas evaluate to the same truth value.

Hence we usually identify alpha equivalent formulas; substitution is then always defined.

Basic Results

From now on, we fix some signature Σ and a set $\mathcal{V}$ of variables.

Lemma

Let $\mathcal{M}$ be a structure for Σ, φ a formula, and σ, σ' variable assignments that agree on $\mathrm{FV}(\varphi)$. Then φ is true over $\mathcal{M}$ and σ iff it is true over $\mathcal{M}$ and σ'.

Corollary

The interpretation of a closed formula is independent of variable assignments.

Lemma

Alpha equivalent formulas evaluate to the same truth value.

Hence we usually identify alpha equivalent formulas; substitution is then always defined.

Basic Results

From now on, we fix some signature Σ and a set $\mathcal{V}$ of variables.

Lemma

Let $\mathcal{M}$ be a structure for Σ, φ a formula, and σ, σ' variable assignments that agree on $\mathrm{FV}(\varphi)$. Then φ is true over $\mathcal{M}$ and σ iff it is true over $\mathcal{M}$ and σ'.

Corollary

The interpretation of a closed formula is independent of variable assignments.

Lemma

Alpha equivalent formulas evaluate to the same truth value.

Hence we usually identify alpha equivalent formulas; substitution is then always defined.

Some Equivalences of FOL

1. $(\forall x.\varphi) = \neg(\exists x.\neg\varphi)$
2. $(\forall x.\varphi \wedge \psi) = (\forall x.\varphi) \wedge (\forall x.\psi)$
3. $(\exists x.\varphi \vee \psi) = (\exists x.\varphi) \vee (\exists x.\psi)$
4. $(\forall x.\forall y.\varphi) = (\forall y.\forall x.\varphi)$
5. $(\exists x.\exists y.\varphi) = (\exists y.\exists x.\varphi)$
6. $(\exists x.\forall y.\varphi) \rightarrow (\forall y.\exists x.\varphi)$, but *not* vice versa

In general we have *neither* $(\forall x.\varphi \vee \psi) = (\forall x.\varphi) \vee (\forall x.\psi)$ *nor* $(\exists x.\varphi \wedge \psi) = (\exists x.\varphi) \wedge (\exists x.\psi)$.

However, if $x \notin \mathrm{FV}(\varphi)$ we have:

1. $(\forall x.\varphi \vee \psi) = \varphi \vee (\forall x.\psi)$
2. $(\exists x.\varphi \wedge \psi) = \varphi \wedge (\exists x.\psi)$

Prenex Normal Form

A formula φ is in *prenex normal form* (PNF) if it is of the form $Q_1 x_1 \dots Q_n x_n \varphi'$, where $n \geq 0$, $Q_i \in \{\forall, \exists\}$, and φ' does not contain quantifiers.

Theorem

For every formula of FOL there is an equivalent one in PNF.

Proof.

Quantifiers can be pulled outwards over negations. For a formula $(\forall x.\varphi) \vee \psi$, we can choose $x \notin \mathrm{FV}(\psi)$ and then rewrite to $\forall x.(\varphi \vee \psi)$; likewise for the other cases. $\square$

Examples

$$
\begin{array}{rl}
& \forall n.n > 0 \to (\forall m.m < n + m) \\
= & \forall n.\neg(n > 0) \vee (\forall m.m < n + m) \\
= & \forall n.\forall m.\neg(n > 0) \vee m < n + m \\
= & \forall n.\forall m.n > 0 \to m < n + m
\end{array}
$$

$$
\begin{array}{rl}
& ((\forall x.p(x)) \vee (\forall x.q(x))) \to (\forall x.p(x) \vee q(x)) \\
= & ((\forall x.p(x)) \vee (\forall y.q(y))) \to (\forall z.p(z) \vee q(z)) \\
= & (\forall x.\forall y.p(x) \vee q(y)) \to (\forall z.p(z) \vee q(z)) \\
= & \exists x.\exists y.((p(x) \vee p(y)) \to (\forall z.p(z) \vee q(z))) \\
= & \exists x.\exists y.\forall z.(p(x) \vee p(y) \to p(z) \vee q(z))
\end{array}
$$

Examples

$$
\begin{aligned}
& \forall n.n > 0 \to (\forall m.m < n + m) \\
= \quad & \forall n.\neg(n > 0) \lor (\forall m.m < n + m) \\
= \quad & \forall n.\forall m.\neg(n > 0) \lor m < n + m \\
= \quad & \forall n.\forall m.n > 0 \to m < n + m
\end{aligned}
$$

$$
\begin{aligned}
& ((\forall x.p(x)) \lor (\forall x.q(x))) \to (\forall x.p(x) \lor q(x)) \\
= \quad & ((\forall x.p(x)) \lor (\forall y.q(y))) \to (\forall z.p(z) \lor q(z)) \\
= \quad & (\forall x.\forall y.p(x) \lor q(y)) \to (\forall z.p(z) \lor q(z)) \\
= \quad & \exists x.\exists y.((p(x) \lor p(y)) \to (\forall z.p(z) \lor q(z))) \\
= \quad & \exists x.\exists y.\forall z.(p(x) \lor p(y) \to p(z) \lor q(z))
\end{aligned}
$$

Undecidability of FOL

Theorem

It is undecidable for a formula φ of FOL whether it is a tautology.

This theorem can be proved, for example, by encoding Post's Correspondence Problem in first order logic.