# Lecture notes
# Einführung in die Logik 2024W

This is a summary of the material discussed in the lecture "Mathematische Logik". It is still a work in progress and there **may me mistakes** in this work. If you find any, feel free to let me know and I will correct them
The content of this script is intended for educational purposes only. It relies on the book [1] for which all rights belong to their respective owners and I do not claim ownership over this content. However the LaTeXcode I wrote is entirely my own work and is open source under the MIT-Licence. Dieses Skript ist noch nicht vollständig und wird regelmäßig aktualisiert.

# Contents



## List of Abbreviations

# CHAPTER 1

# Propositional logic

Language

**Definition 1.1.** `Language of PL:` The Language of Propositional logic is a set containing

- logical symbols: consisting of the **sentential connective** symbols $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ and parenthesis $(,)$
- non-logical symbols: $A_1, A_2, A_3, \ldots$ (also called sentential atoms, variables)

from which we assume (for unique readability) that no symbol is a finite sequence of any other symbols.

Note:

1. The role of the logical symbols doesn't change, the sentential atoms we see as variables, they function as placeholders or variables.
2. we assumed the set of non-logical symbols is countable, for most of our conclusions you could use any set of prop. atoms of any size

expression

**Definition 1.2.** `Expression / prop. sentence:` An **expression** is a any finite sequence of symbols We define **grammatically correct exp.** recursive

1. every prop. atom is a prop. sentence
2. if $\alpha, \beta$ are prop. sentences, then also $\neg\alpha, \alpha \wedge \beta, \alpha \vee \beta, \alpha \rightarrow \beta, \alpha \leftrightarrow \beta$
3. nothing else

prop. sentence
prop. fla.

and call them **prop. sentences.** or **prop. fla.** Equivalently stated every prop. sentence is built up by applying finitly many operations TODO This allows us to symbolize the **expression tree**

construction sequence

**Definition 1.3.** `Construction sequence:` Given a prop. sentence $\alpha$ a **construction sequence** of $\alpha$ is a finite sequence $\langle \alpha_1, \ldots \alpha_{n-1}, \alpha \rangle$ such that for all $i \leq n$ the following holds

- $\alpha_i$ is a sentential atom
- or $\alpha_i = \varepsilon_\neg(\alpha_j)$ for some $j < i$
- or $\alpha_i = \varepsilon_\square(\alpha_j, \alpha_k)$ for some $j, k < i$ and $\square \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$

closure

**Definition 1.4.** : Let $S$ be a set. We say $S$ is **closed** under an $n$-ary operational symbol $f$ iff for all $s_1, s_2, \ldots s_n \in S$ it holds $f(s_1, s_2, \ldots s_n) \in S$

**Induction principle:** Suppose $S$ is a set of prop. sentences containing all prop. atoms and closed under the 5 formula building operations, then $S$ is the set of all prop. sentences.

*Proof.* let $PS =$ set of all prop. sent.

$S \subseteq PS$: is clear

$S \supseteq PS$: let $\alpha \in PS$ then $\alpha$ has a construction seq. $\langle \alpha_1, \ldots \alpha_{n-1}, \alpha \rangle$ and $\alpha_1 \in S$ lets assume that $\alpha_i$ for $i \leq k < n$ is in $S$ then $\alpha_{k+1}$ is either an atom and therefore in $S$ or its obtained by one of the formula building operations from the and therefore $\alpha_{k+1} \in S$

□

## 1.1 TRUTH ASSIGNMENTS

The interpretation of a prop. atom is either true or false, denoted by $0/1$ or $T/F$. A truth assigment is simply any map $\nu : S \mapsto \{0,1\}$, where $S$ is a map of prop. sent. Our goal is going to be to extend any truth assigment $v$ to a function $\overline{v} : \overline{S} \mapsto \{0,1\}$, where $\overline{S}$ is the closure of $S$ under the 5 fla. building functions.

**Definition 1.5. Truth assignment:** Let $\{0,1\}$ be the set of truth values. A truth assignment (TA) for a set $S$ of prop. atoms is a map $\nu : S \to \{0,1\}$

Truth assigment
TA

We now want to extend $\nu$ to $\overline{\nu} : \overline{S} \to \{0,1\}$, where $\overline{S}$ is the closure of $S$ under the 5 fla. building operations such that for all propositional atoms $A \in S$ and propositional formulas $\alpha, \beta$ in $\overline{S}$

1. $\overline{\nu}(A) = \nu(A)$
2. $\overline{\nu}(\neg\alpha) = 1 - \nu(\alpha)$
3. $\overline{\nu}(\alpha \wedge \beta) = \begin{cases} 1 & \text{iff } \overline{\nu}(\alpha) = 1 = \overline{\nu}(\beta) \\ 0 & \text{otherwise} \end{cases}$
4. $\overline{\nu}(\alpha \vee \beta) = \begin{cases} 1 & \text{iff } \overline{\nu}(\alpha) = 1 \text{ or } \overline{\nu}(\beta) = 1 \\ 0 & \text{otherwise} \end{cases}$
5. $\overline{\nu}(\alpha \to \beta) = \begin{cases} 1 & \text{iff } \overline{\nu}(\alpha) = 0 \text{ or } \overline{\nu}(\beta) = 1 \\ 0 & \text{otherwise} \end{cases}$
6. $\overline{\nu}(\alpha \leftrightarrow \beta) = \begin{cases} 1 & \text{iff } \overline{\nu}(\alpha) = \overline{\nu}(\beta) \\ 0 & \text{otherwise} \end{cases}$

We also want the extention to be unique, that is

**Theorem 1.1. Unique readability:** For all TA $\nu$ for a set $S$ $\exists! \overline{\nu} : \overline{S} \to \{0,1\}$ satisfying the above properties

We will proof this later
We will be talking about TA satisfying prop. sent.

**Definition 1.6. Satisfaction:** A TA $\nu$ satisfies a prop. sent. $\alpha$ iff $\overline{\nu}(\alpha) = 1$ (that is, provided that everery atom of $\alpha$ is in the domain of $\nu$). We call $\alpha$ satisfiable iff there exists a TA that satisfies it.

satisfy
satisfiable

**Definition 1.7. Tautological implication:** Let $\Sigma$ be a set of prop. sent. and $\alpha$ a prop. sent. then we say: $\Sigma$ tautologically imlies $\alpha$ iff $\forall$ TA that satisfies $\Sigma$ then $\alpha$ is also satisfied and we write $\Sigma \models \alpha$
If $\Sigma = \{\beta\}$, we simply write $\beta \models \alpha$ If $\Sigma = \varnothing$ then we write $\models \alpha$ for $\varnothing \models \alpha$ and $\alpha$ is called a **tautology**
$\alpha, \beta$ are called **tautologically equivalent** iff $\alpha \models \beta$ and $\beta \models \alpha$, we then write $\alpha \models\!\models \beta$

taut. implication
$\models$

Note: In other words, tautological implication $\Sigma \models \alpha$ means that you can not find a TA, that satisfy all members of $\Sigma$ but not $\alpha$. A tautology is satisfied by every TA. Suppose there is no TA that satisfies $\Sigma$, then we have $\Sigma \models \alpha$ for every prop. sent. $\alpha$

**Example 1.1.** *:* $\{\neg A \vee B\} \models\!\models A \to B$

Note: In order to check if a prop. sent. is satisfiable we need to check $2^N$ TAs, where $N = \#$ of atoms. It is unknown if this can be done by an algorithm in polynomial time. Answering this would settle the debate whether $P = NP$

TODO: Add section here? However we can find a way to reduce satisfiability of an infinite set $\Sigma$ of prop. sent. There later will be a more elementary proof of the compactness theorem, this proof is not part of the exam.

**Theorem 1.2. Compactness theorem:** Let $\Sigma$ be an infinite set of prop. sent. such that

$$\forall \Sigma_0 \subseteq \Sigma, \Sigma_0 \text{finite} \exists \text{ TA satisfying every member of } \Sigma_0 \qquad \text{(finite satisfiability)}$$

then there is a TA satsfying every member of $\Sigma$.

*Proof.* using topology: We have our infinite set of prop. sent. which satisfies above condition. One way to look at TA is as a sequence of $0, 1$, Let $\mathcal{A} = \{A_0, A_1, \dots\}$ be the set of all prop. atoms. We are going to identify TAs with elements in $\{0,1\}^{\mathcal{A}} := \{f : \mathcal{A} \to \{0,1\}\}$ (set of all TAs) This is a topological space with product topology, which we will view The basic open sets (called cylinders) will be

- fix finitly many places and set TV on them,
- others beliebig

$U \subseteq \{0,1\}^{\mathcal{A}}$ such that $p_n(U) = \{0,1\}$ for all but finite many $n$, where $p_n$ is the $n$-th projection. Note: basic open sets are also closed. We now define the open sets as unions of basic open sets. The idea is to use Tychonoffs Thm. which tells us that $\{0,1\}^{\mathcal{A}}$ is compact. i.e. the intersection of a family of closed subsets w/ the finite intersection property (FIP) is non-empty finite intersection property means the intersection of finitly many sets is non-empty.

For $\alpha \in \Sigma$ let $T_\alpha \subseteq \{0,1\}^{\mathcal{A}}$ be the set of TA that satisfy $\alpha$. This $T_\alpha$ is a finite union of cylinders, bc. it only depends on finitly many assigments, hence closed. The family $\{T_\alpha : \alpha \in \Sigma\}$ of closed sets with FIP. Tychonoff tells us, that $\bigcup_{\alpha\in\Sigma} T_\alpha \neq \varnothing$ so there is a TA satisfying $\Sigma$. □

useful might be book p. 26-27

## 1.2 A PARSING ALGORITHM

To prove Theorem 1.1 we essentially need to show that we have enough parenthesis to make the reading of a prop. sent. unique. TODO Bsp

**Lemma 1.1.** : Every prop. sent. has the same number of left and right parenthesis.

*Proof.* Let $M =$ set of prop. sent. w/ # left parenthesis = # right parenthesis and $PS =$ set of all prop. sent. We have $M \subseteq PS$. Since atoms have no parenthesis, they are in $M$. we just need to show that $M$ is closed under the 5 construction operations.
$\varepsilon_\neg = (\neg\alpha) \dots$ □

**Lemma 1.2.** : No proper initial segment of a prop. sent. is itself a prop. sent.

*Proof.* Let $\alpha = \alpha_1\alpha_2\dots\alpha_n$ be a prop. sent. By proper initial segment we understand $\beta = \alpha_1 \dots \alpha_i$ for $1 \leq i < n$. We will prove that every proper initial segment has an excess of left parenthesis, then we use the previous lemma. Let $PS =$ set of all prop. sent. and $PF =$ set of prop. sent. s.t. no proper initial segment has # left parenthesis = # right parenthesis, we will prove that these sets are the same.
Let $\alpha \in PF$. By induction over the fla. building operations

- Atoms: since the empty sequence is no prop. sent. they have no proper initial segment.
- If the above is true for $\alpha, \beta$ then the proper initial segments of $(\neg\alpha)$ are of the form

  $(\neg\alpha$
  $(\neg\alpha'$ where $\alpha'$ is a propper initial segment of $\alpha$
  $($ or
  $(\neg$

  Therefore $\varepsilon_\neg$ preserves this property and under $\varepsilon_\wedge, \varepsilon_\vee, \varepsilon_\to, \varepsilon_\leftrightarrow$ this is also the case.

□

**Parsing algorithm**

We now give a parsing algorithm procedure. For input we take some expression $\tau$ and the algorithm will determine if $\tau$ is a prop. sent. If so, it will generate a unique construction tree (in form of a rooted tree) for $\tau$. (i.e. the construction tree gives us a unique readability) That there is a unique way to perform the algorithm is implied by Lemma 1.2

0. create the root and label it $\tau$
1. HALT if all leaves are labled w/ prop. atom and return: "$\tau$ is a prop. sent."
2. select a leaf of the graph which is not labled w/ prop. atom
3. if the first symbol of label under consideration is not a left parenthesis, then halt and return: "$\tau$ is not a prop. sent."
4. if the second symbol of the label is "$\neg$" then GOTO 6.
5. scan the expression from left to right
   if we reach a proper initial segment of the form "$(\beta$" where $\#lp(\beta) = \#rp(\beta)$ and $\beta$ is followed by one of thesection $\wedge, \vee, \to, \leftrightarrow$ and the remainder of the expression is of the form $\beta'$), where $\#lp(\beta') = \#rp(\beta')$

   Then: create two child nodes (left,right) to the selected element and label them (left $:= \beta$, right $:= \beta'$) GOTO 1.

   Else: HALT and return "$\tau$ is not a prop. sent."
6. if the expression is of the form $(\neg\beta)$ where $\#lp(\beta) = \#rp(\beta)$

   Then: construct one childnode and label it $\beta$ and GOTO 1.

   Else: HALT and return: "$\tau$ is not a prop. sent."

**Example 1.2.** *:* TODO The parsing algorithm applied to $((\neg(A_1 \to A_2)) \vee A_3)$ returns the following construction tree.

### Correctness of the parsing algorithm

- The algorithm always halts, because the length of a childs label is less than the label of a parent.
- If the algorithm halts with the conclusion that $\tau$ is a prop. sent. then we can prove inductively (starting from the leaves) that each label is a prop. sent
- Unique way to make choices in the algorithm: in particular $\beta, \beta'$ in step 5. If there was a shorter choice for $\beta$ it would be a proper initial segment of $\beta$ but such prop. sent. can not exist. (This also works under the assumption that a longer choice exists).
- rejections are made correctly

Back to proving the existence and uniqueness of $\overline{\nu}$ in Theorem 1.1. Let $\alpha$ be a prop. sent. of $\overline{S}$. We apply the parsing algorithm to $\alpha$ to get a unique construction tree For the leaves, use $\nu$ go get the truth values then work our way up using the conditions (1-6) in Definition 1.5.

### A more formal notation

TODO

## 1.3 Induction and recursion

**Generalization of induction principle:**

Let $\mathcal{U}$ be a set and $B \subseteq \mathcal{U}$ our initial set. $\mathcal{F} = \{f, g\}$ a class of functions containing just $f$ and $g$, where

$$f : \mathcal{U} \times \mathcal{U} \to \mathcal{U}, \qquad g : \mathcal{U} \to \mathcal{U}$$

We want to construct the smallest subset $\mathcal{C} \subseteq \mathcal{U}$ such that $B \subseteq \mathcal{C}$ and $\mathcal{C}$ is closed under all elements of $\mathcal{F}$.

**Definition 1.8.** `Closedness, Inductiveness:` We say $\mathcal{S} \subseteq \mathcal{U}$ is

- **closed** under $f$ and $g$ iff for all $x, y \in \mathcal{S}$ it holds $f(x, y) \in \mathcal{S}$ and $g(x) \in \mathcal{S}$
- **inductive** if $B \subseteq \mathcal{S}$ and $\mathcal{S}$ is closed under $\mathcal{F}$

One way is from the top down $\mathcal{C}^* := \bigcap_{\mathcal{S}\text{inductive}} S$ Another is from bottom up: We call $c_1 := \mathcal{B}$,

$$C_i := C_{i-1} \cup \{f(x, y) : x, y \in C_{i-1}\} \cup \{g(x) : x \in C_{i-1}\}$$

and $C_* := \bigcup_{n \geq 1} C_n$ Exercise: show that $C^* = C_* =: C$. Example:

**Example 1.3.** *:*

1. Let $\mathcal{U}$ be the set of all expressions, $\mathcal{B}$ the set of atoms and $\mathcal{F} = \{\varepsilon_\square : \square in\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}\}$ Then $\mathcal{C}$ would be the set of all propositional formulas.
2. Let $\mathcal{U}$ be $\mathbb{R}$, $\mathcal{B}$ the set containing 0 and $\mathcal{F} = \{S\}$, $S(x) = x + 1$ Then $\mathcal{C}$ would be the set of the natural numbers.

### Induction principle

$\mathcal{C}$ generated from $\mathcal{B}$ by use of elements of $\mathcal{F}$ if $S \subseteq C$ such that $\mathcal{B} \subseteq S$ and $S$ is closed under all elements of $\mathcal{F}$, then $S = C$ proof: $S \subseteq C$ is clear. $S$ is inductive, so $C \subseteq S$. Question: under what conditions do we get "generalized unique readability?" The goal would be to define a function on $\mathcal{C}$ recursively i.e. to have rules for computing $\overline{h}(x)$ for $x \in B$ with some rules of computing $\overline{h}(f(x, y))$ and $\overline{h}(g(x))$ from $\overline{h}(x)$ and $\overline{h}(y)$.

**Example 1.4.** *:* Suppose that $G$ is some additive group, generated from $\mathcal{B}$ (the set of generators), $h = \mathcal{B} \rightarrow H$ where $(H, \cdot, ^{-1}, 1)$ a group. When is there an extention $\overline{h}$ of $h$ s.th. $\overline{h} : G \rightarrow H$ is a grouphomomorphism.

- $\overline{h}(0) = 1$
- $\overline{h}(a + b) = \overline{h}(a) \cdot \overline{h}(b)$
- $\overline{h}(-a) = \overline{h}(a)^{-1}$

This is not always possible. **Note:** that it is possible if $G$ is generated freely by the elements of $\mathcal{B}$ and the set of atoms is independent (one element of $\mathcal{B}$ cannot be generated in finitly many steps by other elements of $\mathcal{B}$).

**Definition 1.9.** *:* $\mathcal{C}$ is freely generated from $\mathcal{B}$ by $f, g$ if

- $\mathcal{C}$ is generated from $\mathcal{B}$ by $f, g$
- $f|_{C^2}$ and $g|_C$ are such that
  - $f|_{C^2}$ and $g|_C$ are one-to-one (injective)
  - $rng(f|_{C^2})$ and $rng(g|_C)$ and $\mathcal{B}$ are p.w. disjoint

**Theorem 1.3.** `recursion Theorem:` $C \subseteq U$ freely generated from $B$ by $f, g$ and $V$ a set and $h : B \rightarrow V$, $F : V^2 \rightarrow V$, $G : V \rightarrow V$ Then $\exists! \overline{h} : \mathcal{C} \rightarrow V$ s.that

- for all $a$ in $\mathcal{B}$ it holds $\overline{h}(a) = h(a)$
- for all $x, y$ in $\mathcal{C}$ it holds
  1. $\overline{h}(f(x, y)) = F(\overline{h}(x), \overline{h}(y))$
  2. $\overline{h}(g(x)) = G(\overline{h}(x))$

Note: if given conditions are satisfied then $h$ extends uniquely to a homomorphism $(C, f, g) \rightarrow (V, F, G)$

Before we proof the recursion theorem, we will show how unique readability easily follows from it. Note: Recusion Theorem implies unique readability for propositional formulas. What we need to check is that **Claim:** The Assumptions of recursion theorem are satisfied.

*proof of claim.* $\mathcal{F}_\vee$ is one to one, suppose $(\alpha \vee \beta) = (\delta \vee \gamma)$ then $\alpha \vee \beta) = \delta \vee \gamma)$ And $\alpha, \delta$ are prop. formulas, so they equal to each other (else one is an initial segment of the other, hence not a prop. fla.) By the same argument we get $\beta$ is equal to $\gamma$. ⊠

**Claim:** Disjointment of ranges
*proof of claim.*
- if $(\alpha \vee \beta)$ is $A$ then $A$ starts with (
- if $(\alpha \vee \beta)$ is $(\gamma \to \delta)$ then by the same argument $\alpha$ is $\gamma$ but $\vee$ and $\to$ are diffrent
-

⊠

**Proof of the Rec Thm.**

$v : C \to V$ TODO partial function pfeil nur oben is called acceptable if $\forall x, y \in C$

1. if $x \in B \cap dom(v)$ then $v(x) = h(x)$
2. if $f(x, y) \in dom(v)$ then $x, y \in dom(v)$ and similarly for $g$
   - $v(f(x, y)) = F(v(x), v(y))$
   - $v(g(x)) = G(v(x))$

And when $\mathcal{U} = \{\Gamma_v : v\text{acceptable}\}$, we define $\overline{h} :=$ function w/ graph $\bigcup \mathcal{U}$
**Claim 1:** $\overline{h}$ is a function.
*proof of claim.*

$$S := \{x \in \mathcal{C} : \exists \text{at most one } y \text{ w}/(x, y) \in \bigcup \mathcal{U}\}$$

We want $S = C$, we have $S \subseteq C$, it is enough to show that $S$ is inductive.

- $x \in B \cap dom(v)$ for some $v$ acceptable. then $v(x) = h(x)$ by 1. also $x \notin rngf|_{C^2}$ and $x \notin rngg|_C$
- $x, yin\mathcal{S}$ We want $f(x, y), g(x) \in S$ there are $v_1, v_2$ acceptable s.t. $f(x, y) \in dom(v_1) \cap dom(v_2)$

⊠

**Claim 2:** $\overline{h}$ is acceptable.
*proof of claim.* $\overline{h} : C \rightharpoonup V$ by definition. if $x \in B \cap dom\overline{h}$ then there is a $v$ acceptable, s.t. $x \in dom(v)$ then $\overline{h}(x) = v(x) = h(x)$ if $f(x, y) \in dom\overline{h}$ then $f(x, y) \in dom(v)$ form some $v$ acceptable. Hence $x, y \in dom(v)$ and therefore $x, y \in dom(\overline{h})$ and we have

$$\overline{h}(f(x, y)) = v(f(x, y)) = F(v(x), v(y)) = F(\overline{h}(x), \overline{h}(y))$$

⊠

**Claim 3:** The domain of $\overline{h}$ equals $\mathcal{C}$.
*proof of claim.* it is enough to show that the domain of $\overline{h}$ is inductive. $B \subseteq dom(\overline{h})$ bc. $B \subseteq dom(h)$ where $h$ is acceptable. Now we need to show closure under $f, g$. suppose $x', y' \in dom(\overline{h})$ then $x' \in dom(v_1)$ for some acceptable $v_i$ lets assume $f(x', y') \notin dom(\overline{h})$ then we extend $\overline{h}$ to a function with the same graph as $\overline{h}$. Then $\Gamma \cup \{(f(x', y'), F(\overline{x'}, \overline{y'}))\}$ is the graph of an acceptable function. ⊠

**Claim 4:** Suppose both $\overline{h}, \overline{\overline{h}}$ work, we schow that $S = \{x \in C : \overline{h}(x), \overline{\overline{h}}(x)\}$ is the whole set $C$. it is enough to show that $S$ is inductive.

Let $x \in B$ then $\overline{h}(x) = h(x) = \overline{\overline{h}}(x)$. Then for $x, y \in S$

$$\overline{h}(f(x, y)) = F(\overline{h}(x), \overline{h}(y)) = F(\overline{\overline{h}}(x), \overline{\overline{h}}(y)) = \overline{\overline{h}} \ldots$$

## 1.4 SENTENTIAL CONNECTIVES

**Definition 1.10.** `Tautological equivalence relation:` For $\alpha, \beta$ prop. sent. we define $\alpha \sim \beta$ iff $\alpha \models\!\!\models \beta$. This defines an equivalent relation.

**Example 1.5.** *:* $A \to B \models\!\!\models \neg A \vee B$

Note: A $k$-place boolean function is a functon of the form $f : \{0,1\}^k \to \{0,1\}$ and we define $0, 1$ as the 0-place boolean functions.
If $\alpha$ is a prop. sent. then it determines a $k$-place boolean function, where $k$ is the number of atoms, $\alpha$ is built up from. If $\alpha$ is $A_1 \vee \neg A_2$ then $B_\alpha : \{0,1\}^2 \to \{0,1\}$ and asign its values corresponding a truth table. TODO extend / rearange function

**Theorem 1.4.** *:* If $\alpha, \beta$ are prop. sent. with at most $n$ prop. Atoms (combined), then

1. $\alpha \models \beta$ iff $\forall x \in \{0,1\}^n$ it holds $B_\alpha(x) \leq B_\beta(x)$
2. $\alpha \models\!\!\models \beta$ iff $\forall x \in \{0,1\}^n$ it holds $B_\alpha(x) = B_\beta(x)$
3. $\models \alpha$ iff $\forall x \in \{0,1\}^n$ it holds $B_\alpha(x) = 1$

**Theorem 1.5.** `Realisation:` Let $G$ be an $n$-ary boolean function for $n \geq 1$. Then there is a prop. sent. $\alpha$ such that. $B_\alpha = G$. We say $\alpha$ realizes $G$.

*Proof.* 1. if $G$ is constantly equal to 0 then set $\alpha$ to $A_1 \wedge \neg A_1$.

2. Otherwise the set of inputs $\{\vec{x}_1, \vec{x}_2, \dots \vec{x}_k\}$ for which $G(\vec{x}_i) = 1$ holds is not empty. We denote $\vec{x}_i = (x_{i1}, x_{i2}, \dots x_{in})$ and define a matrix $(x_{ij})_{k \times n}$ We further set $\beta_{ij} = \begin{cases} A_j & \text{iff } x_{ij} = 1 \\ \neg A_j & \text{iff } x_{ij} = 0 \end{cases}$
**Example:**
$$(x_{ij}) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \neg A_1 & A_2 & \neg A_3 \\ A_1 & A_2 & \neg A_3 \end{pmatrix} = (\beta_{ij})$$
We define $\gamma_i$ as $\beta_{i1} \wedge \beta_{i2} \wedge \dots \beta_{in}$ for $1 \leq i \leq k$
and $\alpha$ as $\gamma_1 \vee \gamma_2 \vee \dots \gamma_k = \vee_{i=1}^k \gamma_i$ Then $B_\alpha = G$ is fulfilled.

□

Note: $\alpha$ as constructed in the proof is in the so-called Disjunctive normal form (DNF).

**Corollary 1.5.** Every prop. sent. is tautologically equivalent to a sentence in DNF

**Corollary 1.5.** $\{\neg, \wedge, \vee\}$ is a complete set of logical connectives, i.e. every prop. sent. is tautologically equivalent to a sentence built up from atoms and $\neg, \wedge, \vee$.

**Theorem 1.6.** *:* Both $\{\neg, \wedge\}$ and $\{\neg, \vee\}$ are complete.

*Proof.* Its sufficient to show that every $k$-place boolean function is realisable by a prop. sent. built up using only $\neg$ and $\wedge$. This is, because $\alpha \wedge \beta \models\!\!\models \neg(\neg\alpha \vee \neg\beta)$ We prove this by induction over the number of disjuctions of a prop. sent. $\alpha$ in DNF. Suppose the statement is true for $k \leq n$. For $n+1$ and $\alpha = \bigvee_{j=1}^{n+1} \gamma_j$ there exists an $\alpha' \models\!\!\models \bigvee_{j=1}^{n} \gamma_j$ and

$$\alpha = \bigvee_{j=1}^{n+1} \gamma_j \models\!\!\models \alpha' \vee \gamma_{n+1} \models\!\!\models \neg(\neg\alpha' \wedge \neg\gamma_{n+1})$$

□

Note: We used the observation that, if $\alpha \models\!\!\models \beta$ and we replace a subsequence of $\alpha$ by a so called tautological equivalence then the result is also tautologically equivalent to $\beta$
TODO S.10

**Example 1.6.** $\{\to, \wedge\}$ `is not complete.:` Let $\alpha \in PS$ built up from only $\to, \wedge$ from the atoms $A_1, \dots A_n$ then we claim

$$A_1 \wedge A_2 \wedge \dots \wedge A_n \models \alpha$$

We can also say $\{\to, \wedge\}$ is not complete bc. $\neg A$ is not tautological equivalent to a sent. built up from $\to, \wedge$

*Proof.* Let $C := \{\alpha \in PS$ built up from $\to, \wedge$ and $A_1, \dots A_n$ for which $\bigwedge_{i=1}^{n} A_i \models \alpha\}$ we want to show that $C = \{\alpha \in PS$ built up from $\to, \wedge$ and $A_1, \dots A_n\}$

- We have $\{A_1, A_2 \dots, A_n\} \subseteq C$
- for $\alpha, \beta \in C$ it holds
  (1) $A_1 \wedge \dots \wedge A_n \models \alpha \to \beta$
  (2) $A_1 \wedge \dots \wedge A_n \models \alpha \wedge \beta$

Therefore $C$ is closed under the fla. building operations and we have proven our claim. □

Note: $\{\wedge, \vee, \to, \leftrightarrow\}$ is still not complete.
Note: The number of $n$-ary boolean functions existing is $2^{2^n}$ We define a notation for $n = 0$: $\bot$ (for TV $= 0$) and $\top$ (for TV $= 1$) We can conclude that $\{\neg, \to\}$ and $\{\to, \bot\}$ are both complete, it holds $\neg A \dashv\models A \to \bot$

**Definition 1.11.** Satisfiability:
A set of prop. sent. $\Sigma$ is called **satisfiable** iff $\exists$ TA that satisfies every member of $\Sigma$.

## 1.5 Compactness Theorem

**Theorem 1.7.** Compactness Theorem: $\Sigma$ is satisfiable iff every finite subset $\Sigma_0 \subseteq \Sigma$ is satisfiable. (i.e. $\Sigma$ is finitely satisfied)

*Proof.* Let $\Sigma$ be a finitely satisfiable set of prop. sent. Outline of the proof:

1. extend $\Sigma$ to a maximal finitely satisfiable set $\Delta$ of prop. sent.
2. construct a thruth assigment using $\Delta$

1. Let $\alpha_1, \alpha_2, \dots$ be an enumeration of all prop. sent. and define $\Delta_n$ inductively by $\Delta_0 := \Sigma$

$$\Delta_{n+1} := \begin{cases} \Delta_n \cup \{\alpha_{n+1}\} & \text{if satisfiable} \\ \Delta_n \cup \{\neg\alpha_{n+1}\} & \text{otherwise} \end{cases}$$

**Claim:** $\Delta_n$ is finitely satisfiable for each $n$

*proof of claim.* By regular induction over $n$. $\Delta_0$ is finitely satisfiable. Let us assume $\Delta_n$ is finitely satisfiable. If $\Delta_{n+1} = \Delta_n \cup \{\alpha_{n+1}\}$ then we are finished. Otherwise let $\Delta' \subseteq \Delta_n$ be a finite set that $\Delta' \cup \{\alpha_{n+1}\}$ is not satisfiable. It holds $\Delta' \models \neg\alpha_{n+1}$. We assume that $\Delta_n \cup \{\neg\alpha_{n+1}\}$ is not finitely satisfiable. Then there exists a finite subset $\Delta'' \subseteq \Delta_n$ such that $\Delta'' \cup \{\neg\alpha_{n+1}\}$ is (finite and) not satisfiable. It therefore holds $\Delta'' \models \alpha_{n+1}$ But $\Delta' \cup \Delta''$ is a finite subset of $\Delta_n$ and by above observations $\Delta' \cup \Delta'' \models \alpha_{n+1}$ and $\Delta' \cup \Delta'' \models \neg\alpha_{n+1}$ A contradiction to the assumption that $\Delta_n$ is finitely satisfiable. ⊠

We set $\Delta := \bigcup_{i \in \mathbb{N}} \Delta_i$ and get

(a) $\Sigma \subseteq \Delta$
(b) (Maximality): for every prop. sent. $\alpha$ it holds $\alpha \in \Delta$ or $\neg\alpha \in \Delta$
(c) (Satisfiability): $\Delta$ is finitely satisfiable. For every finite subset there exists a $\Delta_n$ which is a superset.

2. Let $\nu$ be a TA for the prop. atoms $A_1, A_2, \dots$ such that $\nu(A) = 1$ iff $A \in \Delta$
**Claim:** For every prop. sent. $\varphi$ it holds $\overline{\nu}(\varphi) = 1$ iff $\varphi \in \Delta$.
*proof of claim.* Let $S = \{\varphi \in PS$ s.t. $\overline{\nu}(\varphi) = 1$ iff $\varphi \in \Delta\}$.

- $PS \supseteq S$ is clear.
- $PS \subseteq S$
  (a) $\{A_1, A_2 \dots\} \subseteq S$ by definition of $\nu$

(b) closure under $\epsilon_\neg$: Let $\varphi \in S$ then we get by maximality and satisfiability of $\Delta$:

$$\begin{aligned} & \overline{\nu}(\neg\varphi) = 1 \\ \text{iff} \quad & \overline{\nu}(\varphi) = 0 \\ \text{iff} \quad & \varphi \notin \Delta \\ \text{iff} \quad & (\neg\varphi) \in \Delta \end{aligned}$$

closure under $\epsilon_\to$: Let $\varphi_1, \varphi_2 \in S$ similiarly

$$\begin{aligned} & \overline{\nu}(\varphi_1 \to \varphi_2) = 0 \\ \text{iff} \quad & \overline{\nu}(\varphi_1) = 1 \text{ and } \overline{\nu}(\varphi_2) = 0 \\ \text{iff} \quad & \varphi_1 \in \Delta \text{ and } \varphi_2 \notin \Delta \\ \text{iff} \quad & (\varphi_1 \to \varphi_2) \notin \Delta \end{aligned}$$

The closure under the other fla. building operations are similar. ⊠

By this claim $\overline{\nu}$ satisfies $\Sigma$. □

**Corollary 1.7.** If $\Sigma \models \tau$ then there exists a finite subset $\Sigma' \subseteq \Sigma$ s.t. $\Sigma' \models \tau$

*Proof.* Recall: $\Sigma \models \tau$ iff $\Sigma \cup \{\neg\tau\}$ is not satisfiable. Suppose $\Sigma \models \tau$ but no finite subset does.
Then $\forall \Sigma' \subseteq \Sigma$ finite $\Sigma' \cup \{\neg\tau\}$ is satisfiable. By the compactness theorem $\Sigma \cup \{\neg\tau\}$ is satisfiable which is a contradiction to $\Sigma \models \tau$. □

Note: Theorem 1.7 and Corollary 1.7 are equivalent.

# CHAPTER 2

# Predicate - / first order logic

**Definition 2.1.** **A First order Language:** consists of infinetely many distinct symbols such that no symbol is a proper initial segment of another symbol and the symbols are divided into 2 groups:

1. logical symbols (These elements have a fixed meaning and the equivalence symbol $=$ is optional)

   $(,), \neg, \to, v_1, v_2, \ldots, =$

2. parameters

   - quantifier symbol: $\forall$ (the range is subject of interpretation)
   - predicate symbols: for every $n > 0$ we have a set of $n$-ary predicates $P$
   - constant symbols: Some set of constants (could also be $\varnothing$)
   - function symbols: for every $n > 0$ we have a set of $n$-ary function symbols

Note:

- We could drop constants and instead introduce 0-ary function symbols
- to specify language we need to specify the parameters and say if $=$ is included

**Example 2.1.** *:*

- $\mathcal{L}_{\text{set}} = \{\in\}$, $=$ included and the binary predicate symbol $\in$ "element in"
- $\mathcal{L}_{\text{arith}} = \{<, 0, S, E, +, \cdot\}$

  $=$ included
  $<$ is a binary rel. symbol
  $0$ is a constant
  $S$ is a unary function symbol
  $E$ exponentiation TODO
  $+, \cdot$ binary function symbols
- $\mathcal{L}_{\text{ring}} = \{=, +, \cdot, -, 0, 1\}$

  $=$ included
  $0, 1$ are constants
  $-$ is a unary function symbol (additive inverse)
  $+, \cdot$ binary function symbols

## 2.1 Formulas

**Definition 2.2.** `Expression:` An **expression** is any finite sequence of symbols. There exist two kinds of expressions that makes sense "grammatically"

Terms:
- points to an object
- they are built up from variables and constants using function symbols

Formulas:
- They express assertions about objects,
- they are built up from atomic formulas
- atomic formulas these are built up from terms using predicate symbols and $=$, if included

**Definition 2.3.** `Term Building Operations:` For every $n > 0$ and for every $n$-place function symbol $f$ let $\mathcal{F}_f$ be an $n$-place term building operation, that is $\mathcal{F}_f(t_1, \dots t_n) := ft_1, \dots t_n$ (polish notation for $f(t_1, \dots t_n)$). The Set of Terms we then define as the set of expressions that are built up from variables and constants by applying the term building operations finitely many times.

**Example 2.2.** *:* Let $\mathcal{L} = \mathcal{L}_{arith}$ then the set of terms will contain 0, $v_{42}$, $S0$, $SSS0$, $Sv_1$, $+SOv_1$

**Definition 2.4.** `Atomic formula:` Any expression of the form

$= t_1 t_2$ or $Pt_1, \dots t_n$, where $t_1, \dots t_n$ are terms and $P$ is an $n$-ary predicate symbol

Note: Atomic formulas are not defined inductively.

**Example 2.3.** *:* *cont.* $= v_1 v_{42}$, $< S0SS0$ are atomic formulas, but $\neg = v_1 v_{42}$ is not.

**Definition 2.5.** `Formulas:` We define $\varepsilon_\neg$, $\varepsilon_\to, Q_i$ to be the fla. building operations, defined as follows $\varepsilon_\neg(\alpha) := (\neg\alpha)$, $\varepsilon_\to := (\alpha \to \beta)$ and $Q_i(\gamma) := \forall v_i \gamma$. The set of formulas is the set of expressions built up from atomic formulas by applying the fla. building operations finitely many times.

**Example 2.4.** *:* *cont.* $\forall v_1(= Sv_1 0)$ is a formula we get by applying $Q_1$ on the atomic formula $= Sv_1 0$.

### Free variables

$\exists$ quantifier

bounded variable

**Example 2.5.** *:* We introduce the $\exists$ quantifier by defining $\exists y\alpha$ means $\neg\forall y\neg\alpha$.
"Every non-zero natual number is a succsesor" $\forall x(x \neq 0 \to \exists y S(y) = x)$ is different then "if a number is not 0, then it is a succsesor" $x \neq 0 \to \exists y S(y) = x$. $x$ occurs bounded in the first formula, for the latter $x$ occures free in the fla.

If you have an expression without free variables, it is either true or false, on the other hand if a variable occurs free in a formula, the truth value of it depends on the variable itself.

**Definition 2.6.** `Free variables:` Let $x$ be a variable. $x$ occurs **free in** $\varphi$ is defined inductively as follows:

1. If $\varphi$ is an atomic fla. then $x$ occurs **free** in $\varphi$ iff $x$ occurs in $\varphi$
2. If $\varphi = (\neg\alpha)$ then $x$ occurs free in $\varphi$ iff $x$ occurs free in $\alpha$
3. If $\varphi = (\alpha \to \beta)$ then $x$ occurs free in $\varphi$ iff $x$ occurs free in $\alpha$ or $\beta$
4. If $\varphi = \forall v_i \alpha$ then $x$ occurs free in $\varphi$ iff $x$ occurs free in $\alpha$ and $x \neg v_i$

A formula $\alpha$ is called a sentence, if no variable occurs free in $\alpha$

Note: The above definition makes sense thanks to the recursion theorem. def function $h$ on the set of atoms: $h(\alpha)$ = the set of var occ in fla $\alpha$, which is the set of all variables $v_i$ that occur free in $\alpha$. we now want to extend $h$ to $\overline{h}$, which is the set of all formulas.

- $\overline{h}(\neg\alpha) = \overline{h}(\alpha)$
- $\overline{h}(\alpha \to \beta) = \overline{h}(\alpha) \cup \overline{h}(\beta)$
- $\overline{h}(Q_i(\alpha)) = \overline{h}(\alpha) \backslash \{v_i\}$

We say $x$ occurs free in $\alpha$ iff $x \in \overline{h}(\alpha)$.
Note: We will now use $\neg, \wedge, \vee, \to, \leftrightarrow, \exists v_i$ (all can be expressed in terms of $\neg, \to, Q_i$.) We will sometimes drop the $(,)$ and not always be using polish notation.

## 2.2 SEMANTICS OF FIRST ORDER LOGIC

The equivalent scheme to our TA in predicate logic. The meaning of formulas is given by *structures*, which also determine the scope of the quantifier $\forall$, the meaning of all parameters.

**Definition 2.7. structure:** A **structure** $\mathcal{A}$ for a first order language $\mathcal{L}$ is a non-empty set set $A$ called **universe** or **underlying set of** $\mathcal{A}$ together with an interpretation of each parameters of $\mathcal{L}$ i.e.

- $\forall$ ranges over the universe $A$
- for an $n$-ary pred. symbol $P \in \mathcal{L}$ its interpretation $P^{\mathcal{A}}$ is a subset of $A^n$
- for a constant $c \in \mathcal{L}$ its interpretation $c^{\mathcal{A}}$ is an element of $A$
- for an $n$-ary function symbol $f \in \mathcal{L}$ its interpretation $f^{\mathcal{A}}$ is a total function $f^{\mathcal{A}} : A^n \to A$

Note: $A \neq \varnothing$, and all functions $f^{\mathcal{A}}$ are total.

**Example 2.6.** *:* Let $\mathcal{L} = \{\in\}$ where $\in$ is a binary relation " An example of an $\mathcal{L}$ structure is $(\mathbb{N}, \in^{\mathbb{N}})$ where $\in^{\mathbb{N}} = \{(x, y) \in \mathbb{N}^2 : x < y\}$

**Definition 2.8.** : Let $\varphi$ be a $\mathcal{L}$-fla. and $\mathcal{A}$ a $\mathcal{L}$-structure. Let $V$ be the set of all variables in $\mathcal{L}$ and $s : V \to A$ an assignment. We define the extention $\overline{s}$ of $s$ to the set of all $\mathcal{L}$-terms by

- $x \in V$ then $\overline{s}(x) := s(x)$
- $c \in \mathcal{L}$ a constant symbol, then $\overline{s}(c) := c^{\mathcal{A}}$
- $t_1, \dots t_n$ $\mathcal{L}$-terms and $f \in \mathcal{L}$ an $n$-ary function symbol, then $\overline{s}(ft_1 \dots t_n) := f^{\mathcal{A}}(\overline{s}(t_1), \dots \overline{s}(t_n))$

Note: in the previous definition point 3. for $n = 1$ yields a commutative diagram.

**Theorem 2.1.** : For any given assignment $s$ there exists a unique extention $\overline{s}$ as in the previous definition.

*Proof.* will follow from recursion theorem and unique decomposition of terms. □

## 2.3 DEFINITION OF TRUTH

**Definition 2.9.** : We define '$\mathcal{A}$ satisfy $\varphi$ with $s$' and write $\mathcal{A} \models \varphi[s]$ inductively over the complexity of the formula$\varphi$

- if $\varphi$ is atomic:
  - $\mathcal{A} \models= t_1, t_2[s]\overline{s}(t_1) = \overline{s}(t_2)$
  - $\mathcal{A} \models Pt_1, \dots t_n[s](\overline{s}(t_1), \dots \overline{s}(t_2)) \in P^{\mathcal{A}}$
- suppose $\mathcal{A} \models \varphi[s]$ and $\mathcal{A} \models \psi[s]$ are defined, then

- $\mathcal{A} \models \neg\varphi[s]$ iff $\mathcal{A} \not\models \varphi[s]$
- $\mathcal{A} \models \varphi \to \psi[s]$ iff $\mathcal{A} \models \psi[s]$ or $\mathcal{A} \not\models \varphi[s]$
- $\mathcal{A} \models \forall x\varphi[s]$ iff for all $a \in A$ $\mathcal{A} \models \varphi[s(x|a)]$ where

$$s(x|a)(v) = \begin{cases} s(v) \text{ if } v \neq x \\ a \text{ if } v = x \end{cases}$$

**Example 2.7.** : $L = \{\forall, \leq, S, 0\}$ a $L$-structure then could be $\mathcal{N} = (\mathbb{N}, \leq^{\mathcal{N}}, S^{\mathcal{N}}, 0^{\mathcal{N}})$ and TODO $s : v_n \mapsto n - 1$ then $s(v_1) = 0$

- $\overline{s}(0) = 0^{\mathcal{N}}$ (a constant is always mapped to its realisation, the interpretation of constant 0 in the structure $\mathcal{N}$)
- $\overline{s}(Sv_1) = S^{\mathcal{N}}(\overline{s}(v_1)) = S^{\mathcal{N}}(0) = 1$
- $\mathcal{N} \models \forall v_1(S(v_1) \neq v_1)[s]$
  iff for all $a \in \mathbb{N}$ we have that $\mathcal{N} \models (S(v_1) \neq v_1)[s(v_1|a)]$
  iff ...
  iff for all $a \in \mathbb{N}$ we have $S^{\mathcal{A}}(a) \neq a$, which is true in our structure of the natural numbers.
- Is it true in $\mathcal{N}$ that $\mathcal{N} \models S(0) \leq S(v_1)[s]$? Yes because

$$\mathcal{N} \models S(0) \leq S(v_1)[s]$$
$$iff 1 \leq 1$$

Note: To know wheter $\mathcal{A} \models \varphi[s]$ it suffices to know where $s$ maps the variables that are free in $\varphi$

**Theorem 2.2.** : Suppose $s_1, s_2 : V \to A$ agree on all variables that occur free in $\phi$ then

$$\mathcal{A} \models \varphi[s_1] \text{iff } \mathcal{A} \models \varphi[s_2]$$

*Proof.* By complexity of $\varphi$

- if $\varphi$ is $Pt_1, \ldots t_n$ note: any var that occur in $\varphi$ occur free in $\varphi$, so $s_1, s_2$ agree on all variables that occur in the terms $t_1, \ldots t_n$.
  So we Claim: for $t$ a term, $s_1, s_2$ assignments that agree on all variables of $t$ then $\overline{s}_1(t) = \overline{s}_2(t)$
  *proof of claim.* By complexity of $t$

$t = v_m$ then $\overline{s}_1(t) = s_1(v_m) = s_2(v_m) = \overline{s}_2(t)$
$t = c$ then $\overline{s}_1(t) = c^{\mathcal{A}} = \overline{s}_2(t)$
$t = ft_1 \ldots t_n$ inductively, assume $\overline{s}_1(t_i) = \overline{s}_2(t_i)$ for all $1 \leq i \leq n$ then TODO

⊠

- $\varphi := t_1, t_2$ is similar
- $\varphi : \neg\alpha$ then $\mathcal{A} \models \neg\alpha[s_1]$ iff $\mathcal{A} \vDash \alpha[s_1]$iff $\mathcal{A} \vDash \alpha[s_2]$ iff $\mathcal{A} \models \neg\alpha[s_1]$
- $\varphi : \alpha \to \beta$ then $\mathcal{A} \models \alpha \to \beta[s_1]$ iff .. or .. iff for s2 iff ... or ..
- $\varphi : \forall x\alpha$ then the assumption is that $s_1, s_2$ .. the free variables of $\alpha$ are the free variables of $\varphi$ exept for $x$. but because $s_1(x|a) = s_2(x|a)$ they both agree on all free variables of $\alpha$.

$$\mathcal{A} \models \forall x\varphi[s_1] iff for all a \in A \mathcal{A} \models \varphi[s_1(x|a)]$$
$$iff for all a \in A \mathcal{A} \models \varphi[s_2(x|a)]$$
$$iff \mathcal{A} \models \forall x\varphi[s_2]$$

□

Notation: $\mathcal{A} \models \varphi$TODO means that all free variables of $\varphi$ are among $v_1, \dots v_n$ and $\mathcal{A} \models \varphi[s]$ whenever $s(v_i) = a_i$ for all $1 \leq i \leq n$.

**Corollary 2.2.** If $\sigma$ is a sentence then $\mathcal{A} \models \varphi[s]$ for all $s : V \to A$ or $\mathcal{A} \vDash \varphi[s]$ for all $s : V \to A$.

Notation: $\mathcal{A} \models \sigma$ "$\sigma$ is true in $\mathcal{A}$, $\mathcal{A}$ is a model of $\sigma$ or $\sigma$ holds in $\mathcal{A}$.

Note: If $\sigma$ is a sentence then we can not have $\mathcal{A} \models \sigma$ and $\mathcal{A} \vDash \sigma$ because $A \neq \varnothing$.

**Definition 2.10.** : $\mathcal{A}$ is a model of a set of sentences $\Sigma$ iff for every sentence $\sigma \in \Sigma$ it holds $\mathcal{A} \models \sigma$

**Example 2.8.** : $\mathcal{L} = \{0, 1, +, -, \cdot\}$ A realisation could be $\mathcal{R} = (\mathbb{R}, 0, 1, +, -, \cdot)$ or $\mathcal{C} = (\mathbb{C}, 0, 1, +, -, \cdot)$ then the sentence $\sigma$ : $\exists x(x \cdot x = -1)$ then $\mathcal{R} \vDash \sigma$ but $\mathcal{C} \models \sigma$

Note: $\wedge, \vee, \leftrightarrow, \exists$ work as expected. That is $\mathcal{A} \models (\alpha \wedge \beta)[s]$ iff $\mathcal{A} \models \alpha[s]$ and $\mathcal{A} \models \beta[s]$
$\mathcal{A} \models (\alpha \vee \beta)[s]$ iff $\mathcal{A} \models \alpha[s]$ or $\mathcal{A} \models \beta[s]$ $\mathcal{A} \models \exists x \alpha[s]$ iff $\mathcal{A} \models \neg\forall x \neg\alpha[s]$
iff $\mathcal{A} \vDash \forall x \neg\alpha[s]$
iff it is not true that forall $a \in A$ $\mathcal{A} \models \neg\alpha[s(x|a)]$
iff there is $a \in A$ such that $\mathcal{A} \models \alpha[s(x|a)]$

## 2.4 Logical implication

Let $\Gamma$ be a set of $\mathcal{L}$-formulas, $\varphi$ a $\mathcal{L}$-formula.

**Definition 2.11.** : $\Gamma \models \varphi$ "$\Gamma$ logically implies $\varphi$" if for every $L$-structure $\mathcal{A}$ and for every $s : V \to A$
if $\mathcal{A} \models \gamma[s]$for every $\gamma \in \Gamma$ then $\mathcal{A} \models \varphi[s]$

**Definition 2.12.** : $\varphi, \psi$ are called logically equivalent if $\varphi \models \psi$ and $\psi \models \varphi$.

**Definition 2.13.** : $\varphi$ is called valid iff $\models \varphi$ i.e. $\varnothing \models \varphi$ i.e. for every $\mathcal{L}$-structure $\mathcal{A}$ and every $s : V \to A$ it is $\mathcal{A} \models \varphi[s]$

**Example 2.9.** :

1. $\forall x_1 P x_1 \models P x_2$
   Suppose $\mathcal{A} \models \forall x_1 P x_1[s]$. then for all $a \in A$ it is $\mathcal{A} \models P x_1[s(x_1|a)]$ in particular, $a \in P^{\mathcal{A}}$ for $a = s(x_2)$
2. $\forall P x_2 \vDash \forall x_1 P x_1$
   We need a counterexample to $\forall P x_2 \models \forall x_1 P x_1$. Let $A = \{a_1, a_2\}$ $s(x_2) = a_1$ and $P^{\mathcal{A}} = \{a_1\}$ then $\mathcal{A} \models P x_2[s]$.
3. Is the following valid? $\models \exists x(Px \to \forall x Px)$

## 2.5 definability in a structure

## 2.6 Homomorphisms of structures

## 2.7 A parsing algorithm for first order logic

## 2.8 Unique readability for terms

## 2.9 Deductions (formal proofs)

## 2.10 Generalization and deduction theorem

TODO evt noch sectioons

# CHAPTER 3

# Boolean Algebra

**Definition 3.1.** `Boolean Algebra:` A **boolean algebra** is a set $B$ with

- distinguished elements $0, 1$ (called zero and unit of $B$)
- a unary operation $'$ on $B$ (called **complementation** )
- two binary operations $\vee$ called **join** and $\wedge$ called **meet** s.t. for all $x, y, z \in B$

1. $x \vee 0 = x \qquad x \wedge 1 = x$
2. $x \vee x' = 1 \qquad x \wedge x' = 0$
3. $x \vee y = y \vee x \qquad x \wedge y = y \wedge x$
4. $(x \vee y) \vee z = x \vee (y \vee z) \qquad (x \wedge y) \wedge z = x \wedge (y \wedge z)$
5. $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \qquad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$

**Example 3.1.** *:* Let $S$ be a set, $B := \mathcal{P}(S)$ the power set of $S$, $0 := \varnothing$ and $1 := S$,

$$' : \mathcal{P}(S) \to \mathcal{P}(S), x' := S \backslash x \qquad x \vee y := x \cup y, \quad x \wedge y := x \cap y \text{ for } x, y \in \mathcal{P}(S)$$

**Lemma 3.1.** : Let $(B, ', \vee, \wedge, 0, 1)$ be a boolean algebra. Then it holds

a) $0' = 1$, $1' = 0$

b) $x \vee x = x$, $x \wedge x = x$

c) $(x')' = x$

d) $(x \vee y)' = x' \wedge y'$, $(x \wedge y)' = x' \vee y'$

e) $x \vee y = y$ iff $x \wedge y = x$

**Lemma 3.2.** :

a) $x \leq y :\Leftrightarrow x \vee y = y$ defines a partial ordering on $B$ (inclusion) and it holds

b) $x \vee y$ is the least upper bound of $\{x, y\}$ in $B$
$x \wedge y$ is the greatest lower bound of $\{x, y\}$ in $B$

c) $0 \leq x \leq 1$ for all $x \in B$

Note**:** A boolean algebra is a complemented distributive lattice.

**Definition 3.2.** `Opposite of boolean algebra:` Let $(B, ', \vee, \wedge, 0, 1)$ be a boolean algebra. The boolean algebra $B^{\text{op}}$ is defined by

$$B^{\text{op}} := B, \quad 0^{\text{op}} := 1, \quad 1^{\text{op}} := 0, \quad ' \text{ stayes the same as for} B, \quad \vee^{\text{op}} := \wedge, \quad \wedge^{\text{op}} := \vee$$

Note: $(B^{\text{op}})^{\text{op}} = B$

**Definition 3.3.** `Subalgebra:` A **subalgebra** of $B$ is a subset $A \subseteq B$ s.t. $0, 1 \in A$ and $A$ is closed under $', \wedge, \vee$. The subalgebra generated by $P \subseteq B$ is defined to be the smallest subalgebra containing $P$. Equivalently it is the intersection of all Subalgebras of $B$ that contain $P$.

**Example 3.2.** *Power set algebra:* Let $S$ be a set then $\mathcal{P}(S)$ defines a boolean algebra on $S$. $B := \{x \in \mathcal{P}(S) : x \text{is finite or cofinite}\}$ is a subalgebra of $\mathcal{P}(S)$ w/ set of generators $\{\{s\} : s \in S\}$

Note: We will prove the Tarski-Stone Theorem: every boolean algebra is isomorphic to an algebra on a set.

**Example 3.3.** *Lindenbaum Algebra of $\Sigma$:* Let $A$ be a set of prop. atoms, $\text{Prop}(A)$ the set of prop. generated by $A$. Further let $\Sigma \subseteq \text{Prop}(A)$ and $p, q, r$ range over $\text{Prop}(A)$.
We say $p$ is $\Sigma$-equivalent to $q$ iff $\Sigma \models_{\text{taut}} p \leftrightarrow q$ $\Sigma$-Equivalence is an equivalent relation on $\text{Prop}(A)$ and $\text{Prop}(A)/\Sigma$ is a boolean algebra with

$$0 := \bot/\Sigma, \quad 1 := \top/\Sigma, \quad (p/\Sigma)' := (\neg p)/\Sigma, \quad (p/\Sigma \vee q/\Sigma) := (p \vee q)/\Sigma, \quad (p/\Sigma \wedge q/\Sigma) := (p \wedge q)/\Sigma$$

a set of generators is $\{a/\Sigma : a \in A\}$

**Definition 3.4.** *Homomorphisms of boolean algebras:* Let $B, C$ be boolean algebras. A map $\phi : B \to C$ is a (homo)morphism of boolean algebras iff $\forall x, y \in B$ it holds

- $\phi(0_B) = 0_C$
- $\phi(x') = \phi(x)'$
- $\phi(x \vee y) = \phi(x) \vee \phi(y)$
- $\phi(x \wedge y) = \phi(x) \wedge \phi(y)$

If $\phi : B \to C$ is bijective too , we call $\phi$ an isomorphism and $\phi^{-1} : C \to B$ is also a morphism of boolean algebras.

Note: $\phi(B)$ is subalgebra of $C$

**Example 3.4.** *:* Let $S, T$ be sets then a function $f : S \to T$ induces a morphism of boolean algebras $\mathcal{P}(T) \to \mathcal{P}(S) : y \mapsto f^{-1}(y)$ If $S \subseteq T$ and $f$ the inclusion map $S \hookrightarrow T$ then we get a boolean algebra morphism $Y \to Y \cap S$.
• $id_B : B \to B$ • $x \mapsto x' : B \to B^{\text{op}}$ are both isomorphism

Note: A boolean algebra morphism $\phi : B \to C$ is injective iff $\ker f = 0_B$

**Lemma 3.3.** *:* Let $X_1, \dots X_m \subseteq S$ and $\mathcal{A}$ a boolean algebra on $S$ generated by $\{X_1, \dots X_m\}$. Then $\mathcal{A}$ is finite and isomorphic to $\mathcal{P}(\{1, 2, \dots n\})$ for some $n \leq 2^m$.

*Proof.* TODO □

**Definition 3.5.** *Trivial algebras:*

- $B$ is trivial if $|B| = 1$ (equivalently $0 = 1 \in B$) according to Lemma 3.3 $B$ is isomorphic to $\mathcal{P}(\varnothing)$
- If $|S| = 1$ then $|\mathcal{P}(S)| = 2$ TODO

**Definition 3.6.** *Ideal:* An ideal of $B$ is a subset of $I \subseteq B$ s.t.

(I1) $0 \in I$

(I2) $\forall a, b \in B$ it holds $\quad a \leq b \text{ and } b \in I \implies a \in I \quad$ and $\quad a, b \in I \implies a \vee b \in I$

**Example 3.5.** *:* $F_{\text{in}} = \{F \subseteq S : F \text{ finite}\}$ is ideal in $\mathcal{P}(S)$.

Note: If $I$ is an ideal of $B$ then $I \vee b := \{x \in B : x = a \vee b \text{ for some } a \in I\}$ is the smallest ideal w/ respect of $\subseteq$ of $B$ that contains $I \cup \{b\}$.

**Example 3.6.** *:*

- For a boolean algebra morphism $\phi : B \to C$ the kernel $\ker(\phi)$ is an ideal in $B$.
- If $I$ is an ideal in $B$ then $a =_I b :\Leftrightarrow a \vee x = b \vee x$ for some $x \in I$ defines an equivalent relation and $B/_{=_I}$ is a boolean algebra w/

  $0 := 0/_{=_I} \quad 1 := 1/_{=_I} \quad (a/_{=_I})' := a'/_{=_I} \quad a/_{=_I} \vee b/_{=_I} := (a \vee b)/_{=_I} \quad a/_{=_I} \wedge b/_{=_I} := (a \wedge b)/_{=_I}$

  Then $\phi : B \to B/_{=_I} : b \mapsto b/_{=_I}$ is a boolean algebra morphism w/ $\ker(\phi) = I$

# CHAPTER 4

# Set Theory

**Example 4.1.** *Russel's paradox:* Let $A = \{a : a \notin a\}$. If any collection of elements is a set, then $A$ would be a set. Question: is $A \in A$? if yes, then $A \notin A$, if not then $A \in A$

Trying to resolve this, we will introduce the ZFC (Zermelo-Frankel axioms w/ choice) System. Let $\mathcal{L} = \{\in\}$ be a Language of first order, where $\in$ ... binary relation "beeing element of" For $(\mathcal{U}, \in)$ If $(\mathcal{U}, \in) \models$ ZFC, then the elements of the universe $\mathcal{U}$ are called sets.

TODO

## 4.1 AXIOMS OF ZFC

**Definition 4.1.** *Axiom of extensionality:*

$$\forall x \forall y (x = y \leftrightarrow \forall u (u \in x \leftrightarrow u \in y))$$

**Definition 4.2.** *Pairing Axiom:* for any two sets $a, b$ one can form a set whose elements are precicely $a, b$

$$\forall x \forall y \exists z (u \in z \leftrightarrow (u = x \vee u = y))$$

Our notation will be $z = \{x, y\}$

Note: $\{x, y\}$ is unique by Definition 4.1

**Lemma 4.1.** : Let $x, y$ be sets. We define $(x, y) := \{\{x\}, \{x, y\}\}$. Then it holds $(x, y) = (a, b)$ iff $x = a$ and $y = b$

*Proof.*
- if $x = y$, then $(x, y) = \{\{x\}\}$ therefore $a = b$ and by Definition 4.1 it holds $x = a$.
- if $x \neq y$, then $\{\{x\}, \{x, y\}\} = \{\{a\}, \{a, b\}\}$ iff $\{x\} = \{a\}$ and $\{x, y\} = \{a, b\}$. That is, iff $x = a$ and $y = b$.

□

TODO oredered n-tuples

**Definition 4.3.** *Union Axiom:* For every set $x$ there is a set $z$ consisting of all elements of the elements of $x$.

$$\forall x \exists z \forall y (y \in z \leftrightarrow (\exists u u \in x \wedge y \in u))$$

We call $z$ the union of $x$, notation: $\bigcup_x := z$

**Definition 4.4.** *Power set Axiom:* Let $x \subseteq y$ be the abbreviation for $\forall z (z \in x \rightarrow z \in y)$ The **Powerset Axiom** states, that for every set $x$ there exists a set $z$ consisting of all subsetes $y \subseteq x$ that are themselve sets.

$$\forall x \exists z \forall y (y \in z \leftrightarrow y \subseteq x)$$

Notation: $\mathcal{P}(x) := z$.

TODO class relations

**Definition 4.5.** *Axiom of replacement / substitution:* Let $\varphi(x, y, \underline{a})$ a $\mathcal{L}$-fla., w/ free variables among $x, y$ and set-parameters $\underline{a}$. Suppose $\varphi$ defines a class function on $\mathcal{U}$, than the followoing is an axiom:

$$\forall u \exists z \forall y (y \in z \leftrightarrow \exists x (x \in u \wedge \varphi(x, y, \underline{a})))$$

i.e. the image of a set under a class function is a set.

**Definition 4.6.** *Axiom scheme of comprehension:* TODO

# List of definitions

# Bibliography

[1] Herbert B Enderton and Herbert Enderton. *A Mathematical Introduction to Logic*. eng. United States: Elsevier Science and Technology, 2001. ISBN: 0122384520.