# Contents

# 1. Logical axiom schemes

$\boldsymbol{L_1}$ : $B \to (C \to B)$

$\boldsymbol{L_2}$ : $(B \to (C \to D)) \to ((B \to C) \to (B \to D))$

$\boldsymbol{L_3}$ : $B \wedge C \to B$

$\boldsymbol{L_4}$ : $B \wedge C \to C$

$\boldsymbol{L_5}$ : $B \to (C \to B \wedge C)$

$\boldsymbol{L_6}$ : $B \to B \vee C$

$\boldsymbol{L_7}$ : $C \to B \vee C$

$\boldsymbol{L_8}$ : $(B \to D) \to ((C \to D) \to (B \vee C \to D))$

$\boldsymbol{L_9}$ : $(B \to C) \to ((B \to \neg C) \to \neg B)$

$\boldsymbol{L_{10}}$ : $\neg B \to (B \to C)$

$\boldsymbol{L_{11}}$ : $B \vee \neg B$

$\boldsymbol{L_{12}}$ : $\forall x F(x) \to F(t)$ (in particular, $\forall x F(x) \to F(x)$)

$\boldsymbol{L_{13}}$ : $F(t) \to \exists x F(x)$ (in particular, $F(x) \to \exists x F(x)$)

$\boldsymbol{L_{14}}$ : $\forall x (G \to F(x)) \to (G \to \forall x F(x))$

$\boldsymbol{L_{15}}$ : $\forall x (F(x) \to G) \to (\exists x F(x) \to G)$

# 2. Theorems

## 2.1. Prooving directly

$[L_1, L_2, \text{MP}]$:

a) $((A \to B) \to (A \to C)) \to (A \to (B \to C))$. Be careful when assuming hypotheses: assume (A⟶B)⟶(A⟶C), A, B – in this order, no other possibilities!
b) $(A \to B) \to ((B \to C) \to (A \to C))$. It's another version of the **Law of Syllogism** (by Aristotle), or the transitivity property of implication.
c) $(A \to (B \to C)) \to (B \to (A \to C))$. It's another version of the **Premise Permutation Law**. Explain the difference between this formula and Theorem 1.4.3(a): A⟶(B⟶C) ⊢ B⟶(A⟶C).

## 2.2. Deduction theorems

### 2.2.1. Theorem 1.5.1 (Deduction Theorem 1 AKA DT1)

If $T$ is a first order theory, and there is a proof of $[T, \text{MP}] : A_1, A_2, ..., A_n, B \vdash C$, then there is a proof of $[L_1, L_2, T, \text{MP}] : A_1, A_2, ..., A_n \vdash B \to C$.

#### 2.2.2. Theorem 1.5.2 (Deduction Theorem 2 AKA DT2)

If there is a proof $[T, \mathrm{MP}, \mathrm{Gen}] : A_1, A_2, ..., A_n, B \vdash C$, where, after B appears in the proof, Generalization is not applied to the variables that occur as free in $B$, then there is a proof of $[L_1, L_2, L_{14}, T, \mathrm{MP}, \mathrm{Gen}] :$
$A_1, A_2, ..., A_n \vdash B \to C$.

## 2.3. Conjunction

#### 2.3.1. Theorem 2.2.1.

a) (C-introduction): $[L_5, \mathrm{MP}] : A, B \vdash A \wedge B$;
b) (C-elimination): $[L_3, L_4, \mathrm{MP}] : A \wedge B \vdash A, A \wedge B \vdash B$.

#### 2.3.2. Theorem 2.2.2.

a) $[L_1, L_2, L_5, \mathrm{MP}] : (A \to (B \to C)) \leftrightarrow ((A \to B) \to (A \to C))$ (extension of the axiom L_2).
b) $[L_1 - L_4, \mathrm{MP}] : (A \to B) \wedge (B \to C) \to (A \to C)$ (another form of the **Law of Syllogism**, or **transitivity property of implication**).

#### 2.3.3. Theorem 2.2.3 (properties of the conjunction connective).

$[L_1 - L_5, \mathrm{MP}]$:

a) $A \wedge B \leftrightarrow B \wedge A$ . Conjunction is commutative.
b) $A \wedge (B \wedge C) \leftrightarrow (A \wedge B) \wedge C$. Conjunction is associative.
c) $A \wedge A \leftrightarrow A$ . Conjunction is idempotent.

#### 2.3.4. Theorem 2.2.4 (properties of the equivalence connective).

$[L_1 - L_5, \mathrm{MP}]$:

a) $A \leftrightarrow A$ (reflexivity),
b) $(A \leftrightarrow B) \to (B \leftrightarrow A)$ (symmetry),
c) $(A \leftrightarrow B) \to ((B \leftrightarrow C) \to ((A \leftrightarrow C))$ (transitivity).

## 2.4. Disjunction

#### 2.4.1. Theorem 2.3.1

a) (D-introduction)$[L_6, L_7, \mathrm{MP}] : A \vdash A \vee B; B \vdash A \vee B$;
b) (D-elimination) If there is a proof $[T, \mathrm{MP}] : A_1, A_2, ..., A_n, B \vdash D$, and a proof $[T, \mathrm{MP}] : A_1, A_2, ..., A_n, C \vdash D$, then there is a proof $[T, L_1, L_2, L_8, \mathrm{MP}] : A_1, A_2, ..., A_n, B \vee C \vdash D$.

#### 2.4.2. Theorem 2.3.2

a) $[L_5, L_6 - L_8, \mathrm{MP}] : A \vee B \leftrightarrow B \vee A$ . Disjunction is commutative. b) $[L_1, L_2, L_5, L_6 - L_8, \mathrm{MP}] : A \vee A \leftrightarrow A$ . Disjunction is idempotent.

#### 2.4.3. Theorem 2.3.3.

Disjunction is associative: $[L_1, L_2, L_5, L_6 - L_8, \mathrm{MP}] : A \vee (B \vee C) \leftrightarrow (A \vee B) \vee C$.

#### 2.4.4. Theorem 2.3.4.

Conjunction is distributive to disjunction, and disjunction is distributive to conjunction:

a) $[L_1 - L_8, \mathrm{MP}] : (A \wedge B) \vee C \leftrightarrow (A \vee C) \wedge (B \vee C)$ .
b) $[L_1 - L_8, \mathrm{MP}] : (A \vee B) \wedge C \leftrightarrow (A \wedge C) \vee (B \wedge C)$ .

#### 2.4.5. Theorem 2.3.4.

Conjunction is distributive to disjunction, and disjunction is distributive to conjunction:

a) $[L_1 - L_8, \mathrm{MP}] : (A \wedge B) \vee C \leftrightarrow (A \vee C) \wedge (B \vee C)$;
b) $[L_1 - L_8, \mathrm{MP}] : (A \vee B) \wedge C \leftrightarrow (A \wedge C) \vee (B \wedge C)$ .

## 2.5. Negation – minimal logic

#### 2.5.1. Theorem 2.4.1.

(N-elimination) If there is a proof

$[T, \mathrm{MP}] : A_1, A_2, ..., A_n, B \vdash C$, and a proof $[T, \mathrm{MP}] : A_1, A_2, ..., A_n, B \vdash \neg C$, then there is a proof $[T, L_1, L_2, L_9, \mathrm{MP}] : A_1, A_2, ..., A_n \vdash \neg B$.

**2.5.2. Theorem 2.4.2.**

a) $[L_1, L_2, L_9, \mathrm{MP}] : A, \neg B \vdash \neg(A \to B)$. What does it mean? b) $[L_1 - L_4, L_9, \mathrm{MP}] : A \wedge \neg B \to \neg(A \to B)$.

**2.5.3. Theorem 2.4.3.**

$[L_1, L_2, L_9, \mathrm{MP}] : (A \to B) \to (\neg B \to \neg A)$. What does it mean? It's the so-called **Contraposition Law**.

Note. The following rule form of Contraposition Law is called **Modus Tollens**: $[L_1, L_2, L_9, \mathrm{MP}] : A \to B, \neg B \vdash \neg A, \vee$ , $\frac{A \to B; \neg B}{\neg A}$

**2.5.4. Theorem 2.4.4.**

$[L_1, L_2, L_9, \mathrm{MP}] : A \to \neg\neg A$.

**2.5.5. Theorem 2.4.5.**

a) $[L_1, L_2, L_9, \mathrm{MP}] : \neg\neg\neg A \leftrightarrow \neg A$. b) $[L_1, L_2, L_6, L_7, L_9, \mathrm{MP}] : \neg\neg(A \vee \neg A)$. What does it mean? This is a "weak form" of the **Law of Excluded Middle** that can be proved constructively. The formula $\neg\neg(A \vee \neg A)$ can be proved in the constructive logic, but $A \vee \neg A$ can't – as we will see in Section 2.8.

**2.5.6. Theorem 2.4.9.**

a) $[L_1, L_2, L_8, L_9, \mathrm{MP}] : \neg A \vee \neg B \to \neg(A \wedge B)$ . It's the constructive half of the so-called **First de Morgan Law**. What does it mean?
b) $[L_1 - L_9, \mathrm{MP}] : \neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$. It's the so-called **Second de Morgan Law**.

## 2.6. Negation – constructive logic

**2.6.1. Theorem 2.5.1.**

a) $[L_1, L_8, L_{10}, \mathrm{MP}] : \neg A \vee B \to (A \to B)$.
b) $[L_1, L_2, L_6, \mathrm{MP}] : A \vee B \to (\neg A \to B) \vdash \neg A \to (A \to B)$ . It means that the "natural" rule $A \vee B; \neg A \vdash B$ implies $L_{10}$!

**2.6.2. Theorem 2.5.2.**

$[L_1 - L_{10}, \mathrm{MP}]$:

a) $(\neg\neg A \to \neg\neg B) \to \neg\neg(A \to B)$. It's the converse of Theorem 2.4.7(b). Hence, $[L_1 - L_{10}, \mathrm{MP}] : \vdash \neg\neg(A \to B) \leftrightarrow (\neg\neg A \to \neg\neg B)$.
b) $\neg\neg A \to (\neg A \to A)$. It's the converse of Theorem 2.4.6(a). Hence, [L1-L10, MP]: $\neg\neg A \leftrightarrow (\neg A \to A)$.
c) $A \vee \neg A \to (\neg\neg A \to A)$ .
d) $\neg\neg(\neg\neg A \to A)$. What does it mean? It's a "weak" form of the Double Negations Law – provable in constructive logic.

## 2.7. Negation – classical logic

**2.7.1. Theorem 2.6.1. (Double Negation Law)**

$[L_1, L_2, L_8, L_{10}, L_{11}, \mathrm{MP}] : \neg\neg A \to A$. Hence, $[L_1 - L_{11}, \mathrm{MP}] : \neg\neg A \leftrightarrow A$.

**2.7.2. Theorem 2.6.2.**

$[L_8, L_{11}, \mathrm{MP}] : A \to B, \neg A \to B \vdash B$. Or, by Deduction Theorem 1, $[L_1, L_2, L_8, L_{11}, \mathrm{MP}] : (A \to B) \to ((\neg A \to B) \to B)$.

**2.7.3. Theorem 2.6.3.**

$[L_1 - L_{11}, \mathrm{MP}] : (\neg B \to \neg A) \to (A \to B)$. Hence, $[L_1 - L_{11}, \mathrm{MP}] : (A \to B) \leftrightarrow (\neg B \to \neg A)$.

**2.7.4. Theorem 2.6.3.**

*(another one with the same number of because numbering error (it seems like it))*

$[L_1 - L_9, L_{11}, \mathrm{MP}] : \vdash \neg(A \wedge B) \to \neg A \vee \neg B$ . Hence, $[L_1 - L_9, L_{11}, \mathrm{MP}] : \vdash \neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$ .

**2.7.5. Theorem 2.6.4.**
$[L_1 - L_8, L_{11}, \mathrm{MP}] : (A \to B) \to \neg A \vee B$. Hence, (I-elimination) $[L_1 - L_{11}, \mathrm{MP}] : (A \to B) \leftrightarrow \neg A \vee B$.

**2.7.6. Theorem 2.6.5.**
$[L_1 - L_{11}, \mathrm{MP}] : \neg(A \to B) \to A \wedge \neg B$.

**2.7.7. Theorem 2.7.1 (Glivenko's Theorem).**
$[L_1 - L_{11}, \mathrm{MP}] : \vdash A$ if and only if $[L_1 - L_{10}, \mathrm{MP}] : \vdash \neg\neg A$.

## 2.8. Axiom independence

**2.8.1. Theorem 2.8.1.**
The axiom $L_9$: $(A \to B) \to ((A \to \neg B) \to \neg A)$ can be proved in $[L_1, L_2, L_8, L_{10}, L_{11}, \mathrm{MP}]$.

**2.8.2. Theorem 2.8.2.**
The axiom $L_9$ cannot be proved in $[L_1 - L_8, L_{10}, \mathrm{MP}]$.

## 2.9. Replacement Theorem 1
Let us consider three formulas: $B$, $B'$, $C$, where $B$ is a sub-formula of $C$, and $o(B)$ is a propositional occurrence of $B$ in $C$. Let us denote by $C'$ the formula obtained from $C$ by replacing $o(B)$ by $B'$. Then, in the minimal logic,

$[L_1 - L_9, \mathrm{MP}] : B \leftrightarrow B' \vdash C \leftrightarrow C'$.

## 2.10. Replacement Theorem 2
Let us consider three formulas: $B$, $B'$, $C$, where $B$ is a sub-formula of $C$, and $o(B)$ is any occurrence of $B$ in $C$. Let us denote by $C'$ the formula obtained from $C$ by replacing $o(B)$ by B'. Then, in the minimal logic,

$[L_1 - L_9, L_{12} - L_{15}, \mathrm{MP}, \mathrm{Gen}] : B \leftrightarrow B' \vdash C \leftrightarrow C'$.

## 2.11. Theorem 3.1.1.
$[L_1, L_2, L_{12}, L_{13}, \mathrm{MP}] : \forall x B(x) \to \exists x B(x)$ . What does it mean? It prohibits "empty domains".

Theorem 3.1.2.

a) [L1, L2, L12, L14, MP, Gen]: ∀x(B⟶C)⟶(∀x B⟶∀xC).
b) [L1, L2, L12-L15, MP, Gen]: ∀x(B⟶C)⟶(∃z(x+z+1=y).x B⟶∃z(x+z+1=y).xC).

## 2.12. Theorems 3.1.3.
If F is any formula, then:

a) (U-introduction) [Gen]: F(x) ⊢∀x F(x) .
b) (U-elimination) [L12, MP, Gen]: ∀x F(x) ⊢F(x) . What does it mean?
c) (E-introduction) [L13, MP, Gen]: F(x) ⊢∃z(x+z+1=y).x F(x) . What does it mean?

## 2.13. Theorems 3.1.4.
If F is any formula, and G is a formula that does not contain free occurrences of x, then:

a) (U2-introduction) [L14, MP, Gen] G ⟶F (x) ⊢G ⟶∀x F (x) . What does it mean?
b) (E2-introduction) [L15, MP, Gen]: F (x)⟶G ⊢∃z(x+z+1=y).x F (x)⟶G . What does it mean?

## 2.14. Theorem 3.1.5.
a) [L1, L2, L5, L12, L14, MP, Gen]: $\forall x \forall y B(x, y) \leftrightarrow \forall y \forall x B(x, y)$
b) [L1, L2, L5, L13, L15, MP, Gen]: $\exists x \exists y B(x, y) \leftrightarrow \exists y \exists x B(x, y)$.
c) [L1, L2, L12-L15, MP, Gen]: $\exists x \forall y B(x, y) \leftrightarrow \forall y \exists x B(x, y)$.

Theorem 3.1.6. If the formula B does not contain free occurrences of x, then [L1-L2, L12-L15, MP, Gen]: (∀x B)↔B; (∃z(x+z+1=y).x B)↔B , i.e., quantifiers ∀x ;∃z(x+z+1=y). x can be dropped or introduced as needed.

Theorem 3.2.1. In the classical logic, [L1-L15, MP, Gen]: ¬ x¬B ∀ ↔ xB.

## 2.15. Theorem 3.3.1.
a) [L1-L5, L12, L14, MP, Gen]: ∀x(B∧C)↔∀x B∧∀xC .

b) [L1, L2, L6-L8, L12, L14, MP, Gen]: ⊢∀x B∨∀xC ⟶∀x(B∨C) . The converse formula ∀x(B∨C)⟶∀x B∨∀xC cannot be true. Explain, why.

### 2.16. Theorem 3.3.2.

a) [L1-L8, L12-L15, MP, Gen]: ∃z(x+z+1=y).x(B∨C)↔∃z(x+z+1=y). x B∨∃z(x+z+1=y).xC . b) [L1-L5, L13-L15, MP, Gen]: ∃z(x+z+1=y).x(B∧C)⟶∃z(x+z+1=y). x B∧∃z(x+z+1=y).xC . The converse implication ∃z(x+z+1=y).x B∧∃z(x+z+1=y). xC ⟶∃z(x+z+1=y). x(B∧C) cannot be true. Explain, why. Exercise 3.3.3. a) Prove (a⟶) of Theorem 3.3.2. (Hint: start by assuming B∨C , apply D-elimination, etc., and finish by E2-introduction.)

# 3. Three-valued logic

This is a general scheme (page 74) to define a three valued logic.

For example, let us consider a kind of "three-valued logic", where 0 means "false", 1 – "unknown" (or NULL – in terms of SQL), and 2 means "true". Then it would be natural to define "truth values" of conjunction and disjunction as

$A \wedge B = \min(A, B)$ ;

$A \vee B = \max(A, B)$ .

But how should we define "truth values" of implication and negation?

| $A$ | $B$ | $A \wedge B$ | $A \vee B$ | $A \to B$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $i_1$ |
| 0 | 1 | 0 | 1 | $i_2$ |
| 0 | 2 | 0 | 2 | $i_3$ |
| 1 | 0 | 0 | 1 | $i_4$ |
| 1 | 1 | 1 | 1 | $i_5$ |
| 1 | 2 | 1 | 2 | $i_6$ |
| 2 | 0 | 0 | 2 | $i_7$ |
| 2 | 1 | 1 | 2 | $i_8$ |
| 2 | 2 | 2 | 2 | $i_9$ |

| $A$ | $\neg A$ |
|---|---|
| 0 | $i_{10}$ |
| 1 | $i_{11}$ |
| 2 | $i_{12}$ |

# 4. Model interpreation

## 4.1. Interpretation of a language

#### 4.1.1. The language-specific part

Let L be a predicate language containing:

- (a possibly empty) set of object constants $c_1, ..., c_k, ...$;
- (a possibly empty) set of function constants $f_1, ..., f_m, ...,$;
- (a non empty) set of predicate constants $p_1, ..., p_n, ....$

An interpretation $J$ of the language $L$ consists of the following two entities (a set and a mapping):

a) A non-empty finite or infinite set DJ – the domain of interpretation (it will serve first of all as the range of object variables). (For infinite domains, set theory comes in here.)
b) A mapping intJ that assigns:
   - to each object constant $c_i$ – a member $\text{int}_J(c_i)$ of the domain $D_J$ [contstant corresponds to an object from domain];
   - to each function constant $f_i$ – a function $\text{int}_J(f_i)$ from $D_J \times ... \times D_J$ into $D_J$ [],

- to each predicate constant $p_i$ – a predicate $\mathrm{int}_J(p_i)$ on $D_J$.

Having an interpretation $J$ of the language $L$, we can define the notion of **true formulas** (more precisely – the notion of formulas that are true under the interpretation $J$).

**Example.** The above interpretation of the "language about people" put in the terms of the general definition:

a) $D = \{\mathrm{br}, \mathrm{jo}, \mathrm{pa}, \mathrm{pe}\}$.
b) $\mathrm{int}_J(\mathrm{Britney}) = \mathrm{br}, \mathrm{int}_J(\mathrm{John}) = \mathrm{jo}, \mathrm{int}_J(\mathrm{Paris}) = \mathrm{pa}, \mathrm{int}_J(\mathrm{Peter}) = \mathrm{pe}$.
c) $\mathrm{int}_J(\mathrm{Male}) = \{\mathrm{jo}, \mathrm{pe}\}; \mathrm{int}_J(\mathrm{Female}) = \{\mathrm{br}, \mathrm{pa}\}$.
d) $\mathrm{int}_J(\mathrm{Mother}) = \{(\mathrm{pa}, \mathrm{br}), (\mathrm{pa}, \mathrm{jo})\}; \mathrm{int}_J(\mathrm{Father}) = \{(\mathrm{pe}, \mathrm{jo}), (\mathrm{pe}, \mathrm{br})\}$.
e) $\mathrm{int}_J(\mathrm{Married}) = \{(\mathrm{pa}, \mathrm{pe}), (\mathrm{pe}, \mathrm{pa})\}$.
f) $\mathrm{int}_J(=) = \{(\mathrm{br}, \mathrm{br}), (\mathrm{jo}, \mathrm{jo}), (\mathrm{pa}, \mathrm{pa}), (\mathrm{pe}, \mathrm{pe})\}$.

### 4.1.2. Interpretations of languages – the standard common part

Finally, we define the notion of **true formulas** of the language $L$ under the interpretation $J$ (of course, for a fixed combination of values of their free variables – if any):

a) Truth-values of the formulas: $\neg B, B \wedge C, B \vee C, B \to C$ [those are not examples] must be computed. This is done with the truth-values of $B$ and $C$ by using the well-known classical truth tables (see Section 4.2).

b) The formula $\forall x B$ is true under $J$ if and only if $B(c)$ is true under $J$ for all members $c$ of the domain $D_J$.

c) The formula $\exists x B$ is true under $J$ if and only if there is a member c of the domain $D_J$ such that $B(c)$ is true under $J$.

**Example.** In first order arithmetic, the formula

$$y((x = y + y) \vee (x = y + y + 1))$$

is intended to say that "x is even or odd". Under the standard interpretation S of arithmetic, this formula is true for all values of its free variable x.

Similarly, $\forall x \forall y(x + y = y + x)$ is a closed formula that is true under this interpretation. The notion "a closed formula F is true under the interpretation J" is now precisely defined.

**Important – non-constructivity!** It may seem that, under an interpretation, any closed formula is "either true or false". However, note that, for an infinite domain DJ, the notion of "true formulas under J" is extremely non-constructive.

### 4.1.3. Example of building of an interpretation

In Section 1.2, in our "language about people" we used four names of people (Britney, John, Paris, Peter) as object constants and the following predicate constants:

a) Male $(x)$ – means "x is a male person";
b) Female $(x)$ – means "x is a female person";
c) Mother $(x, y)$ – means "x is mother of y";
d) Father $(x, y)$ – means "x is father of y";
e) Married $(x, y)$ – means "x and y are married";
f) $x = y$ – means "x and y are the same person".

Now, let us consider the following interpretation of the language – a specific "small four person world":

The domain of interpretation – and the range of variables – is: $D = \{\mathrm{br}, \mathrm{jo}, \mathrm{pa}, \mathrm{pe}\}$ (no people, four character strings only!).

Interpretations of predicate constants are defined by the following truth tables:

| x | Male(x) | Female(x) |
|---|---|---|
| br | false | true |
| jo | true | false |
| pa | false | true |

| pe | true | false |
|---|---|---|

| x | y | Father(x,y) | Mother(x,y) | Married(x,y) | x=y |
|---|---|---|---|---|---|
| br | br | false | false | false | true |
| br | jo | false | false | false | false |
| br | pa | false | false | false | false |
| br | pe | false | false | false | false |
| jo | br | false | false | false | false |
| jo | jo | false | false | false | true |
| jo | pa | false | false | false | false |
| jo | pe | false | false | false | false |
| pa | br | false | true | false | false |
| pa | jo | false | true | false | false |
| pa | pa | false | false | false | true |
| pa | pe | false | false | true | false |
| pe | br | true | false | false | false |
| pe | jo | true | false | false | false |
| pe | pa | false | false | true | false |
| pe | pe | false | false | false | true |

## 4.2. Three kinds of formulas

If one explores some formula F of the language L under various interpretations, then three situations are possible:

a) $F$ is true in all interpretations of the language $L$. Formulas of this kind are called **logically valid formulas** (LVF, Latv. **LVD**).

b) $F$ is true in some interpretations of $L$, and false – in some other interpretations of $L$.

c) F is false in all interpretations of L Formulas of this kind are called **unsatisfiable formulas** (Latv. **neizpildāmas funkcijas**).

Formulas that are "not unsatisfiable" (formulas of classes (a) and (b)) are called, of course, satisfiable formulas: a formula is satisfiable, if it is true in at least one interpretation [**satisfiable functions** (Latv. **izpildāmas funkcijas**)].

### 4.2.1. Prooving an F is LVF (Latv. LVD)

First, we should learn to prove that some formula is (if really is!) logically valid. Easiest way to do it by reasoning from the opposite: suppose that exists such interpretation J, where formula is false, and derive a contradiction from this. Then this will mean that formula is true in all interpretations, and so logically valid. Check pages 125-126 of the book for example of such proof (there is proven that axiom L12 is true in all interpretations). Definitely check it, because in such way you will need to solve tasks in homeworks and tests.

### 4.2.2. Prooving an F is satisfiable but NOT LVF

As an example, let us verify that the formula

$$\forall x(p(x) \vee q(x)) \rightarrow \forall x p(x) \vee \forall x q(x)$$

is not logically valid (p, q are predicate constants). Why it is not? Because the truth-values of p(x) and q(x) may behave in such a way that $p(x) \vee q(x)$ is always true, but neither $\forall x p(x)$, nor $\forall x q(x)$ is true. Indeed, let us take the domain $D = \{a, b\}$, and set (in fact, we are using one of two possibilities):

| x | p(x) | q(x) |
|---|---|---|
| b | false | true |
| a | true | false |

In this interpretation, $p(a) \vee q(a) = \text{true}$ , $p(b) \vee q(b) = \text{true}$, i.e., the premise $\forall x(p(x) \vee q(x))$ is true. But the formulas$\forall p(x), \forall q(x)$ both are false. Hence, in this interpretation, the conclusion $\forall x p(x) \vee \forall x q(x)$ is false, and $\forall x(p(x) \vee q(x)) \to \forall x p(x) \vee \forall x q(x)$ is false. We have built an interpretation, making the formula false. Hence, it is not logically valid.

On the other hand, this formula is satisfiable – there is an interpretation under which it is true. Indeed, let us take $D = \{a\}$ as the domain of interpretation, and let us set $p(a) = q(a) = \text{true}$. Then all the formulas

$$\forall x(p(x) \vee q(x)), \forall x p(x), \forall x q(x)$$

become true, and so becomes the entire formula.

## 4.3. Gödel's Completeness Theorem

**Theorem 4.3.1.** In classical predicate logic $[L_1 - L_{15}, \text{MP}, \text{Gen}]$ all logically valid formulas can be derived.

**Theorem 4.3.3.** All formulas that can be derived in classical predicate logic $[L_1 - L_{15}, \text{MP}, \text{Gen}]$ are logically valid. In this logic it is not possible to derive contradictions, it is consistent.

### 4.3.1. Gödel's theorem usage for task solving

This theorem gives us new method to conclude that some formula $F$ is derivable in classical predicate logic: instead of trying to derive $F$ by using axioms, rules of inference, deduction theorem, T 2.3.1 and other helping tools, we can just prove that $F$ is logically valid (by showing that none of interpretations can make it false). If we manage to do so, then we can announce: according to Gödel's theorem, $F$ is derivable in classical predicate logic $[L_1 - L_{15}, \text{MP}, \text{Gen}]$.

# 5. Tableaux algorithm

## 5.1. Step 1.

We will solve the task from the opposite: append to the hypotheses $F_1, ...F_n$ negation of formula $G$, and obtain the set $F_1, ..., F_n, \neg G$. When you will do homework or test, you shouldn't forget this, because if you work with the set $F_1, ..., F_n, G$, then obtained result will not give an answer whether $G$ is derivable or not. You should keep this in mind also when the task has only one formula, e.g., verify, whether formula $(A \to B) \to ((B \to C) \to (A \to C))$ is derivable. Then from the beginning you should append negation in front: ¬((A⟶B)⟶((B⟶C)⟶(A⟶C))) and then work further. Instead of the set $F_1, ..., F_n, \neg G$ we can always check one formula $F_1 \wedge ... \wedge F_n \wedge \neg G$. Therefore, our task (theoretically) is reducing to the task: given some predicate language formula F, verify, whether it is satisfiable or not.

## 5.2. Step 2.

Before applying the algorithm, you first should translate formula to the so-called negation normal form. We can use the possibilities provided by Substitution theorem. First, implications are replaced with negations and disjunctions:

$$(A \to B) \leftrightarrow \neg A \vee B$$

Then we apply de Morgan laws to get negations close to the atoms:

$$\neg(A \vee B) \leftrightarrow \neg A \wedge \neg B \equiv$$
$$\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$$

In such way all negations are carried exactly before atoms. After that we can remove double negations:

$$\neg\neg A \leftrightarrow A$$

Example: $(p \to q) \to q$.

First get rid of implications: $\neg(\neg p \vee q) \vee q$.

Then apply de Morgan law: $(\neg\neg p \wedge \neg q) \vee q$.

Then get rid of double negations: $(p \wedge \neg q) \vee q$.

Now we have obtained equivalent formula in negation normal form – formula only has conjunctions and disjunctions, and all negations appear only in front of atoms.

## 5.3. Step 3.

Next, we should build a tree, vertices of which are formulas. In the root of the tree we put our formula. Then we have two cases.

a) If vertex is formula A∧B, then each branch that goes through this vertex is extended with vertices A and B.
b) If vertex is a formula A∨B, then in place of continuation we have branching into vertex A and vertex B.

In both cases, the initial vertex is marked as processed. Algorithm continues to process all cases 1 and 2 until all non-atomic vertices have been processed.

## 5.4. Step 4.

When the construction of the tree is finished, we need to analyze and make conclusions. When one branch has some atom both with and without a negation (e.g., $A$ and $\neg A$), then it is called closed branch. Other branches are called open branches.

**Theorem.** If in constructed tree, there exists at least one open branch, then formula in the root is satisfiable. And vice versa – if all branches in the tree are closed, then formula in the root is unsatisfiable.