

CSE581

Computer Science Fundamentals: Theory

Professor Anita Wasilewska

P1 LOGIC: LECTURE 2a

Chapter 2

Introduction to Classical Logic Languages and Semantics

Chapter 2

Introduction to Classical Logic Languages and Semantics

Lecture 2

Part 1: Classical Logic Model

Part 2: Propositional Language

Part 3: Propositional Semantics

Part 4: Examples of Propositional Tautologies

Lecture 2a

Part 5: Predicate Language

Part 6: Predicate Tautologies- Laws for Quantifiers

Chapter 2

Introduction to Classical Logic Languages and Semantics

Part 5: Predicate Language

Predicate Language

We define a **predicate language** \mathcal{L} following the **pattern** established by the definitions of **symbolic** and **propositional language**.

The predicate language **is much more complicated** in its structure.

Its alphabet \mathcal{A} is **much richer**.

The definition of its set of formulas \mathcal{F} is **more complicated**.

In order to **define** the set \mathcal{F} define an **additional set** \mathbf{T} , called a set of all **terms** of the predicate language \mathcal{L} .

We **single** out this set \mathbf{T} of **terms** not only because **we need** it for the **definition of formulas**, but also because of its role in the **development** of **other notions** of **predicate logic**.

Predicate Language Definition

Definition

By a **predicate language** \mathcal{L} we understand a triple

$$\mathcal{L} = (\mathcal{A}, \mathbf{T}, \mathcal{F})$$

where \mathcal{A} is a predicate **alphabet**

\mathbf{T} is the set of **terms**, and \mathcal{F} is a set of **formulas**

Alphabet Components

Alphabet \mathcal{A}

The components of \mathcal{A} are as follows

1. Propositional connectives

$\neg, \cap, \cup, \Rightarrow, \Leftrightarrow$

2. Quantifiers \forall, \exists

\forall is the **universal** quantifier, and \exists is the **existential** quantifier

3. Parenthesis (and)

Alphabet Components

4. Variables

We assume that we have, as we did in the propositional case a **countably infinite** set **VAR** of **variables**

The variables now have a **different meaning** than they had in the **propositional** case

We hence call them **variables**, or **individual variables**

We put

$$\text{VAR} = \{x_1, x_2, \dots\}$$

5. Constants

The **constants** represent in "real life" concrete **elements of sets**. We assume that we have a **countably infinite** set **C** of constants

$$\text{C} = \{c_1, c_2, \dots\}$$

Alphabet Components

6. Predicate symbols

The **predicate symbols** represent "real life" **relations**

We denote them by **P, Q, R, ...**, with indices, if necessary

We use symbol **P** for the set of all **predicate symbols**

We assume that **P** is countably infinite and write

$$\mathbf{P} = \{P_1, P_2, P_3, \dots\}$$

Alphabet Components

Logic notation

In "real life" we write symbolically $x < y$ to express that element x is smaller then element y according to the two argument relation $<$

In the **predicate language** \mathcal{L} we **represent** the relation $<$ as a two argument predicate $P \in \mathbf{P}$

We write $P(x, y)$ as a **representation** of "real life" $x < y$.

The variables x, y in $P(x, y)$ are **individual variables** from the set **VAR**

Mathematical statements $n < 0$, $1 < 2$, $0 < m$ are **represented** in \mathcal{L} by $P(x, c_1)$, $P(c_2, c_3)$, $P(c_1, y)$, respectively,

where c_1, c_2, c_3 are any **constants** and x, y any **variables**

Alphabet Components

7. Function symbols

The **function symbols** represent "real life" **functions**

We denote function symbols by f, g, h, \dots , with indices, if necessary

We use symbol **F** for the set of all function symbols

We assume that **F** is **countably infinite** and write

$$\mathbf{F} = \{f_1, f_2, f_3, \dots\}$$

Set **T** of Terms

Definition

Terms are expressions built out of **function symbols** and **variables**.

They **describe** how we build **compositions of functions**.

We define the set **T** of all **terms** recursively as follows.

1. All **variables** are **terms**;
2. All **constants** are **terms**;
3. For any **function symbol** $f \in \mathbf{F}$ **representing** a function on n variables, and any terms t_1, t_2, \dots, t_n , the expression $f(t_1, t_2, \dots, t_n)$ is a **term**;
4. The set **T** of all **terms** of the predicate language \mathcal{L} is the smallest set that fulfills the conditions **1. - 3.**

Example

Example

Here are some **terms** of \mathcal{L}

$$h(c_1), f(g(c, x)), g(f(f(c)), g(x, y)),$$

$$f_1(c, g(x, f(c))), g(g(x, y), g(x, h(c))) \dots$$

Observe that to obtain the predicate language **representation** of for example $x + y$ we can first write it as $+(x, y)$ and then replace the addition symbol $+$ by any two argument function symbol $g \in \mathbf{F}$ and get the **term** $g(x, y)$.

Set \mathcal{F} of Formulas

Formulas are build out of elements of the **alphabet** \mathcal{A} and the set **T** of all **terms**.

We denote the **formulas** by A, B, C, \dots , with **indices**, if necessary.

We **build** them, as before in **recursive steps**.

The **first recursive step** says:

all **atomic formulas** are **formulas**.

The **atomic formulas** are the simplest formulas, as the **propositional variables** were in the case of the **propositional language**.

We define the **atomic formulas** as follows.

Atomic Formulas

Definition

An **atomic formula** is any expression of the form

$$R(t_1, t_2, \dots, t_n),$$

where **R** is any n-argument predicate $R \in \mathbf{P}$ and t_1, t_2, \dots, t_n are **terms**, i.e. $t_1, t_2, \dots, t_n \in \mathbf{T}$.

Some **atomic formulas** of \mathcal{L} are:

$$Q(c), Q(x), Q(g(x_1, x_2)),$$

$$R(c, d), R(x, f(c)), R(g(x, y), f(g(c, z))), \dots$$

Set \mathcal{F} of Formulas

Definition

The set \mathcal{F} of formulas of predicate language \mathcal{L} is the smallest set meeting the following conditions.

1. All **atomic formulas** are **formulas**;
2. If A, B are formulas, then $\neg A, (A \cap B), (A \cup B), (A \Rightarrow B), (A \Leftrightarrow B)$ are formulas;
3. If A is a formula, then $\forall xA, \exists xA$ are formulas for any variable $x \in VAR$.

Set \mathcal{F} of Formulas

Example

Some formulas of \mathcal{L} are:

$$\begin{aligned} &R(c, d), \quad \exists y R(y, f(c)), \quad R(x, y), \\ &(\forall x R(x, f(c)) \Rightarrow \neg R(x, y)), \quad (R(c, d) \cap \forall z R(z, f(c))), \\ &\forall y R(y, g(c, g(x, f(c))))), \quad \forall y \neg \exists x R(x, y) \end{aligned}$$

Set \mathcal{F} of Formulas

Let's look now closer at the following formulas.

$$R(c_1, c_2), \quad R(x, y), \quad ((R(y, d) \Rightarrow R(a, z)),$$

$$\exists x R(x, y), \quad \forall y R(x, y), \quad \exists x \forall y R(x, y).$$

Observations

1. Some formulas are **without quantifiers**:

$$R(c_1, c_2), \quad R(x, y), \quad (R(y, d) \Rightarrow R(a, z)).$$

A formula **without quantifiers** is called an **open formula**

Variables x, y in $R(x, y)$ are called **free variables**.

The variable y in $R(y, d)$ and z in $R(a, z)$ are also **free**.

Set \mathcal{F} of Formulas

Observations

2. Quantifiers **bind variables** within formulas.

The variable x is **bounded** by $\exists x$ in the formula $\exists x R(x, y)$, the variable y is **free**.

The variable y is **bounded** by $\forall y$ in the formula $\forall y R(x, y)$, the variable x is **free**.

3. The formula $\exists x \forall y R(x, y)$ **does not** contain any **free** variables, **neither does** the formula $R(c_1, c_2)$.

4. A formula **without** any **free variables** is called a **closed formula** or a **sentence**.

Mathematical Statements

We often use **logic symbols**, while writing **mathematical statements** in a more symbolic way.

For example, **mathematicians** to say "all natural numbers are greater than zero and some integers are equal 1" often write

$$x \geq 0, \quad \forall_{x \in \mathbb{N}} \quad \text{and} \quad \exists_{y \in \mathbb{Z}}, \quad y = 1$$

Some of them who are more "**logic oriented**" would write it as

$$\forall_{x \in \mathbb{N}} x \geq 0 \cap \exists_{y \in \mathbb{Z}} y = 1,$$

or even as

$$(\forall_{x \in \mathbb{N}} x \geq 0 \cap \exists_{y \in \mathbb{Z}} y = 1).$$

Observe that **none** of the above **symbolic statement** are formulas of the **predicate language**.

These are **mathematical statements** written with **mathematical** and **logic symbols**. They are written with different degree of "**logical precision**", the last being, from a **logician point of view** **the most precise**.

Mathematical Statements

Our goal now is to “translate ” **mathematical** and **natural language** statement into correct **formulas** of the predicate language \mathcal{L} .

Let's start with some **observations**.

O1 The quantifiers in $\forall_{x \in N}, \exists_{y \in Z}$ **are not** the one used in **logic**.

O2 The predicate language \mathcal{L} **admits only** quantifiers $\forall x, \exists y$, for any variables $x, y \in VAR$.

O3 The quantifiers $\forall_{x \in N}, \exists_{y \in Z}$ are called **quantifiers with restricted domain**.

The **restriction** of the **quantifier domain** can, and often is given by more **complicated** statements.

Quantifiers with Restricted Domain

The quantifiers $\forall_{A(x)}$ and $\exists_{A(x)}$ are called quantifiers with **restricted domain**, or **restricted quantifiers**, where $A(x) \in \mathcal{F}$ is any formula with a free variable $x \in \text{VAR}$.

Definition

$\forall_{A(x)} B(x)$ stands for a formula $\forall x(A(x) \Rightarrow B(x)) \in \mathcal{F}$.

$\exists_{A(x)} B(x)$ stands for a formula $\exists x(A(x) \cap B(x)) \in \mathcal{F}$.

We write it as the following **transformations rules** for **restricted quantifiers**

$$\forall_{A(x)} B(x) \equiv \forall x(A(x) \Rightarrow B(x))$$

$$\exists_{A(x)} B(x) \equiv \exists x(A(x) \cap B(x))$$

Translations to Formulas of \mathcal{L}

Given a **mathematical statement** \mathbf{S} written with **logical symbols**.

We obtain a formula $A \in \mathcal{F}$ that is a **translation** of \mathbf{S} into \mathcal{L} by conducting a following **sequence** of steps.

Step 1 We **identify** **basic statements** in \mathbf{S} , i.e. mathematical statements that **involve only relations**. They are to be translated into **atomic formulas**.

We **identify** the **relations** in the basic statements and **choose** the **predicate symbols** as their names.

We **identify** all **functions** and **constants** (if any) in the basic statements and **choose** the **function symbols** and **constant symbols** as their names.

Step 2 We **write** the **basic statements** as **atomic formulas** of \mathcal{L} .

Translations to Formulas of \mathcal{L}

Remember that in the predicate language \mathcal{L} we write a function symbol **in front** of the function arguments **not between** them as we write in mathematics.

The same applies to **relation symbols**.

For example we re-write a basic mathematical statement $x + 2 > y$ as $> (+ (x, 2), y)$

Then we write it as an **atomic formula** $P(f(x, c), y)$

$P \in \mathbf{P}$ stands for two argument relation $>$,

$f \in \mathbf{F}$ stands for two argument function $+$,

and $c \in \mathbf{C}$ stands for the **number 2**.

Translations to Formulas of \mathcal{L}

Step 3 We **write** the statement **S** a **formula** with **restricted quantifiers** (if needed)

Step 4. We **apply** the **transformations rules** for **restricted quantifiers** to the **formula** from Step 3 and **obtain** a proper formula **A** of \mathcal{L} as a result, i.e. as a **translation** of the given **mathematical statement S**

In case of a translation from mathematical statement written **without logical symbols** we **add** a following step.

Step 0 We **identify** **propositional connectives** and **quantifiers** and use them to re-write the statement in a form that is as close to the structure of a **logical formula** as possible

Translations Examples

Exercise

Given a **mathematical statement** **S** written with **logical symbols**

$$(\forall_{x \in \mathbb{N}} x \geq 0 \cap \exists_{y \in \mathbb{Z}} y = 1)$$

1. Translate it into a proper **logical formula** with **restricted quantifiers** i.e. into a formula of \mathcal{L} that **uses** the restricted domain quantifiers.

2. Translate your **restricted quantifiers formula** into a correct formula **without** restricted domain quantifiers, i.e. into a **proper formula** of \mathcal{L}

A **long** and **detailed solution** is given in the book **Chapter 2**, pages 43-45.

A **short statement** of the exercise and a **short solution** follows

Translations Examples

Exercise

Given a **mathematical statement** **S** written with **logical symbols**

$$(\forall_{x \in N} x \geq 0 \cap \exists_{y \in Z} y = 1)$$

Translate it into a proper formula of \mathcal{L} .

Short Solution

The **basic statements** in **S** are: $x \in N$, $x \geq 0$, $y \in Z$, $y = 1$

The corresponding **atomic formulas** of \mathcal{L} are:

$N(x)$, $G(x, c_1)$, $Z(y)$, $E(y, c_2)$, for

$n \in N$, $x \geq 0$, $y \in Z$, $y = 1$, respectively.

The statement **S** becomes **restricted quantifiers** formula

$$(\forall_{N(x)} G(x, c_1) \cap \exists_{Z(y)} E(y, c_2))$$

By the **transformation rules** we get $A \in \mathcal{F}$:

$$(\forall x(N(x) \Rightarrow G(x, c_1)) \cap \exists y(Z(y) \cap E(y, c_2)))$$

Translations Examples

Exercise

Here is a **mathematical statement S**:

"For all real numbers x the following holds: If $x < 0$, then there is a natural number n , such that $x + n < 0$."

1. **Re-write S** as a **symbolic** mathematical statement **SF** that only uses **mathematical** and **logical symbols**.
2. **Translate** the symbolic statement **SF** into to a corresponding formula $A \in \mathcal{F}$ of the predicate language \mathcal{L}

Translations Examples

Solution

The statement **S** is:

"For all real numbers x the following holds: If $x < 0$, then there is a natural number n , such that $x + n < 0$."

S becomes a **symbolic** mathematical statement **SF**

$$\forall_{x \in R} (x < 0 \Rightarrow \exists_{n \in N} x + n < 0)$$

We write **R**(x) for $x \in R$, **N**(y) for $y \in N$,

We use the constant **c** for the number 0

We use **L** \in **P** to denote the relation $<$

We use **f** \in **F** to denote the function $+$

The statement $x < 0$ becomes an **atomic formula** **L**(x , **c**).

The statement $x + n < 0$ becomes **L**(**f**(x , y), **c**)

Translations Examples

Solution c.d.

The **symbolic** mathematical statement **SF**

$$\forall_{x \in R} (x < 0 \Rightarrow \exists_{n \in N} x + n < 0)$$

becomes a **restricted quantifiers** formula

$$\forall_{R(x)} (L(x, c) \Rightarrow \exists_{N(y)} L(f(x, y), c))$$

We apply now the **transformation rules** and get a corresponding formula $A \in \mathcal{F}$:

$$\forall x (R(x) \Rightarrow (L(x, c) \Rightarrow \exists y (N(y) \cap L(f(x, y), c))))$$

PROPOSITIONAL LANGUAGE TRANSLATION

ATTENTION

The **TRANSLATION** of the same mathematical statement **S**:

"For all real numbers x the following holds: If $x < 0$, then there is a natural number n , such that $x + n < 0$."

into a **PROPOSITIONAL LANGUAGE** is a formula

a

where a is any propositional variable denoting the statement

"For all real numbers x the following holds: If $x < 0$, then there is a natural number n , such that $x + n < 0$."

Translations from Natural Language

Exercise

Translate a natural language statement

S: "Any friend of Mary is a friend of John and Peter is not John's friend. Hence Peter is not Mary's friend"

into a formula $A \in \mathcal{F}$ of the predicate language \mathcal{L} .

Solution

1. We identify the **basic relations** and **functions** (if any) and **translate** them into atomic formulas

We have only **one relation** of "being a friend".

We **translate** it into an **atomic formula** $F(x, y)$,
where $F(x, y)$ stands for "x is a friend of y"

Translations from Natural Language

S: "Any friend of Mary is a friend of John and Peter is not John's friend. Hence Peter is not Mary's friend"

We use **constants** m, j, p for **Mary, John, and Peter**, respectively

We hence have the following **atomic formulas**:

$F(x, m), F(x, j), F(p, j)$, where

$F(x, m)$ stands for "x is a friend of Mary",

$F(x, j)$ stands for "x is a friend of John", and

$F(p, j)$ stands for "Peter is a friend of John"

Translations from Natural Language

2. Statement "Any friend of Mary is a friend of John"

translates into a **restricted quantifier** formula $\forall_{F(x,m)} F(x,j)$

"Peter is not John's friend" **translates** into $\neg F(p,j)$, and

"Peter is not Mary's friend" **translates** into $\neg F(p,m)$

3. **Restricted** quantifiers formula for **S** is

$$((\forall_{F(x,m)} F(x,j) \cap \neg F(p,j)) \Rightarrow \neg F(p,m))$$

and the formula $A \in \mathcal{F}$ of \mathcal{L} is

$$((\forall x (F(x,m) \Rightarrow F(x,j)) \cap \neg F(p,j)) \Rightarrow \neg F(p,m))$$

Rules of Translations

Rules of translation from **natural** language to the **predicate** language \mathcal{L}

1. Identify the basic **relations** and **functions** (if any) and **translate** them into **atomic formulas**
2. Identify **propositional connectives** and use symbols $\neg, \cup, \cap, \Rightarrow, \Leftrightarrow$ for them
3. Identify **quantifiers**: restricted $\forall_{A(x)}, \exists_{A(x)}$, and non-restricted $\forall x, \exists x$
4. Use the **symbols** from 1. - 3. and **restricted quantifiers transformation rules** to write $A \in \mathcal{F}$ of the predicate language \mathcal{L}

Translation Example

Exercise

Given a natural language statement

S: "For any bird one can find some birds that white"

Show that the **translation** of **S** into a formula of the predicate language \mathcal{L} is $\forall x(B(x) \Rightarrow \exists x(B(x) \cap W(x)))$

Solution

We follow the **rules of translation** to **verify** the **correctness** of the translation

1. Atomic formulas: $B(x)$, $W(x)$.

$B(x)$ stands for "x is a bird" and $W(x)$ stands for "x is white"

2. There is **no propositional connectives** in **S**

Translation Example

3. Restricted quantifiers:

$\forall_{B(x)}$ for "any bird " and

$\exists_{B(x)}$ for "one can find some birds".

Restricted quantifiers formula for **S** is

$$\forall_{B(x)} \exists_{B(x)} W(x)$$

4. By the **transformation rules** we get a required formula of the predicate language \mathcal{L} :

$$\forall x (B(x) \Rightarrow \exists x (B(x) \cap W(x)))$$

Translation Example

Exercise

Translate into \mathcal{L} a natural language statement

S: "Some patients like all doctors."

Solution

1. Atomic formulas: $P(x)$, $D(x)$, $L(x, y)$.

$P(x)$ stands for "x is a patient",

$D(y)$ stands for "y is a doctor", and

$L(x,y)$ stands for "x likes y"

2. There is no propositional connectives in **S**

Translation Example

3. Restricted quantifiers:

$\exists_{P(x)}$ for "some patients" and $\forall_{D(y)}$ for "all doctors"

Observe that we **can't** write $L(x, D(y))$ for "x likes doctor y"
 $D(y)$ is a predicate, **not a term**, and hence $L(x, D(y))$ **is not a formula**

We have to express the statement "x likes all doctors y" in terms of **restricted quantifiers** and the predicate $L(x,y)$ only

Translation Example

Observe that the statement "x likes all doctors y" means also "all doctors y are liked by x"

We can **re-write** it as "for all doctors y, x likes y" what translates to a formula $\forall_{D(y)} L(x, y)$

Hence the statement **S** translates to

$$\exists_{P(x)} \forall_{D(y)} L(x, y)$$

4. By the **transformation rules** we get the following **translation** of **S** into \mathcal{L}

$$\exists x (P(x) \cap \forall y (D(y) \Rightarrow L(x, y)))$$

Chapter 2

Introduction to Classical Logic Languages and Semantics

Part 6: Predicate Tautologies- Laws for Quantifiers

Predicate Tautologies

The notion of **predicate** tautology is much more **complicated** than that of the **propositional**

We **define** it **formally** in later chapters

Predicate tautologies are also called **valid formulas**, or **laws of quantifiers** to **distinguish them** from the **propositional** case

We **provide** here a **motivation**, **examples** and an **intuitive definitions**

We also **list** and **discuss** the most used and useful **tautologies** and **equational laws** of quantifiers

Interpretation

The formulas of the **predicate** language \mathcal{L} have a **meaning** only when an **interpretation** is given for its **symbols**

We **define** the **interpretation** I in a set $U \neq \emptyset$ by **interpreting** **predicate** and **functional** symbols of \mathcal{L} as concrete **relations** and **functions** defined in the set U

We **interpret** **constants** symbols as **elements** of the set U

The set U is called the **universe** of the **interpretation** I .

These two items specify a **model structure** for \mathcal{L}

We write it as a pair $\mathbf{M} = (U, I)$

Model Structure

Given a formula A of \mathcal{L} , and the **model structure** $\mathbf{M} = (U, I)$

We **denote** by A_I a **statement** written with logical symbols **determined** by the formula A and the **interpretation** I in the **universe** U

When A **is a sentence**, it means it is a formula **without free** variables, A_I **represents** a proposition that is **true** or **false**

When A **is not a sentence**, it contains **free** variables and may be **satisfied** (i.e. true) for **some** values in the universe U and **not satisfied** (i.e. false) for **the others**

Lets look at **few simple** examples

Examples

Example

Let **A** be a formula $\exists x P(x, c)$

Consider a **model structure** $\mathbf{M}_1 = (N, I_1)$

The **universe** of the **interpretation** I_1 is the set **N** of natural numbers

We **define** the **interpretation** I_1 as follows

We **interpret** the two argument predicate **P** as a relation $=$ and the constant **c** as number **5**, i.e we put

$$P_{I_1} := \text{and} \quad c_{I_1} : 5$$

Examples

The formula $A: \exists x P(x, c)$ under the interpretation I_1 becomes a mathematical statement $\exists x x = 5$ defined in the set \mathbf{N} of natural numbers

We write it for short

$$A_{I_1} : \exists_{x \in \mathbf{N}} x = 5$$

A_{I_1} is obviously a **true** mathematical statement

In this case we say: the formula $A: \exists x P(x, c)$ is **true** under the interpretation I_1 in the universe of \mathbf{M}_1 , or for short:

A is **true** in the model structure \mathbf{M}_1 .

We write it **symbolically** as

$$\mathbf{M}_1 \models \exists x P(x, c)$$

and say: \mathbf{M}_1 is a **model** for the formula A

Examples

Example

Consider now a **model structure** $\mathbf{M}_2 = (N, I_2)$ and the formula $A: \exists x P(x, c)$.

We interpret now the predicate P as relation $<$ in the set N of natural numbers and the constant c as number 0

We write it as

$$P_{I_2} : < \quad \text{and} \quad c_{I_2} : 0$$

Examples

The formula $A: \exists x P(x, c)$ under the interpretation I_2 becomes a mathematical statement $\exists x x < 0$ defined in the set \mathbb{N} of natural numbers

We write it for short

$$A_{I_2} : \exists_{x \in \mathbb{N}} x < 0$$

A_{I_2} is obviously a **false** mathematical statement.

We say: the formula $A: \exists x P(x, c)$ is **false** under the interpretation I_2 in \mathbf{M}_2 , or we say for short: A is **false** in \mathbf{M}_2

We write it **symbolically** as

$$\mathbf{M}_2 \not\models \exists x P(x, c)$$

and say that \mathbf{M}_2 is a **counter-model** for the formula A

Examples

Example

Consider now a **model structure**

$\mathbf{M}_3 = (Z, I_3)$ and the formula $A: \exists x P(x, c)$

We **define** an interpretation I_3 in the set of all **integers** Z exactly as the interpretation I_1 was defined, i.e. we put

$$P_{I_3} : < \text{ and } c_{I_3} : 0$$

Examples

In this case we get

$$A_{I_3} : \exists_{x \in \mathbb{Z}} x < 0$$

Obviously A_{I_3} is a **true** mathematical statement

The formula A is **true** under the interpretation I_3 in \mathbf{M}_3

(A is **satisfied, true** in \mathbf{M}_3)

We write it symbolically as

$$\mathbf{M}_3 \models \exists x P(x, c)$$

\mathbf{M}_3 is yet another **model** for the formula A

Examples

When a formula **is not a closed** (not a sentence) the situation gets more complicated

Given a model structure $\mathbf{M} = (U, I)$,
a formula can be **satisfied** (i.e. true) for **some values** in the universe U and **not satisfied** (i.e. false) for the others

Examples

Example

Given the formulas:

1. $A_1 : R(x, y)$, 2. $A_2 : \forall y R(x, y)$, 3. $A_3 : \exists x \forall y R(x, y)$

Let $\mathbf{M} = (N, I)$ be a **model structure**

where the predicate R is **interpreted** as a relation \leq , i.e.

$$R_I : \leq$$

1. Consider the formula $A_1 : R(x, y)$. Obviously,

$$A_{1I} : x \leq y$$

and $A_1 : R(x, y)$ is **satisfied** in $\mathbf{M} = (N, I)$ by all $n, m \in N$ such that $n \leq m$

Examples

2. Consider the formula $A_2 : \forall y R(x, y)$

By definition of $\mathbf{M} = (N, I)$ we have that

$$R_I : \leq$$

and hence

$$A_{2I} : \forall y \in N \ x \leq y$$

and the formula $A_2 : \forall y R(x, y)$ is **satisfied** in the model structure $\mathbf{M} = (N, I)$ by the natural number 0 only

Examples

3. Consider the formula $A_3 : \exists x \forall y R(x, y)$

3. $A_{3I} : \exists x \in \mathbb{N} \forall y \in \mathbb{N} x \leq y$ asserts that there is a smallest natural number what is a **true** statement, i.e. \mathbf{M} is a **model** for A_3

Observe that changing the universe of $\mathbf{M} = (\mathbb{N}, I)$ to the set of all **Integers** \mathbb{Z} , we get a different a model structure $\mathbf{M}_1 = (\mathbb{Z}, I)$.

In this case $A_{3I} : \exists x \in \mathbb{Z} \forall y \in \mathbb{Z} x \leq y$

asserts that there is a smallest integer and A_3 is a **false** sentence in \mathbf{M}_1 , i.e. \mathbf{M}_1 is a **counter-model** for A_3

Predicate Tautology Definition

We want the **predicate** language **tautologies** to have the same property as the **propositional**, namely to be **always true**

In this case, we **intuitively agree** that it means that we want the **predicate tautologies** to be formulas that are **true** under **any interpretation** in **any possible universe**

A **rigorous definition** of the **predicate tautology** is provided in a later chapter on **Predicate Logic**

Predicate Tautology Definition

We construct the **rigorous definition** in the following steps.

1. We first define **formally** the notion of **interpretation** I of symbols of \mathcal{L} in a set $U \neq \emptyset$, i.e. in the **model structure** $\mathbf{M} = (U, I)$ for the predicate language \mathcal{L} .
2. Then we define **formally** a notion "a formula A of \mathcal{L} is **true** in $\mathbf{M} = (U, I)$ "

We write it symbolically

$$\mathbf{M} \models A$$

and call the model structure $\mathbf{M} = (U, I)$ a **model** for A

3. We define a notion " **A is a predicate tautology**" as follows.

Predicate Tautology Definition

Defintion For any formula A of predicate language \mathcal{L} ,
 A is a **predicate tautology (valid formula)** if and only if

$$\mathbf{M} \models A$$

for all model structures $\mathbf{M} = (U, I)$ for \mathcal{L}

4. Directly from the above definition we get the following
definition of a notion " A is not a predicate tautology"

Defintion

For any formula A of predicate language \mathcal{L} ,

A is not a **predicate tautology** if and only if **there is** a
model structure $\mathbf{M} = (U, I)$ for \mathcal{L} , such that

$$\mathbf{M} \not\models A$$

We call such model structure \mathbf{M} a **counter-model** for A

Predicate Tautology Definition

The definition of a notion "A is not a predicate tautology" says:
to prove that **A is not** a predicate tautology **one has to show**
a **counter-model**

It means **one has** to **define** a non-empty set **U** and define an
interpretation **I**, such that **we can prove** that **A_I** is **false**

Predicate Tautology Definition

We use terms **predicate tautology** or **valid formula** instead of just saying a **tautology** in order to **distinguish** tautologies belonging to **two very different** languages

For the same reason we **usually reserve** the symbol \models for **propositional** case

Sometimes we use symbols \models_p or \models_f to **denote predicate tautologies**

p stands for **predicate** and **f** stands **first order**.

The **predicate tautologies** are also called **laws of quantifiers**

We will use **both** names

Predicate Tautologies Examples

Here are some **examples** of **predicate tautologies** and **counter models** for formulas that are **not tautologies**.

Example

For any formula $A(x)$ with a free variable x :

$$\models_p (\forall x A(x) \Rightarrow \exists x A(x))$$

Observe that the following formula B

$$B : (\forall x A(x) \Rightarrow \exists x A(x))$$

represents an **infinite number** of formulas

Hence the formula B a **tautology** for **any** formula $A(x)$ of \mathcal{L} with a free variable x

Predicate Tautologie Examples

The **inverse** implication to $(\forall x A(x) \Rightarrow \exists x A(x))$ is **not** a predicate tautology, i.e.

$$\not\models_p (\exists x A(x) \Rightarrow \forall x A(x))$$

To **prove it** we have to **provide an example** of a concrete formula $A(x)$ and **construct** a **counter-model** $\mathbf{M} = (U, I)$ for the formula $F : (\exists x A(x) \Rightarrow \forall x A(x))$

Let $A(x)$ be an **atomic** formula $P(x, c)$

We define $\mathbf{M} = (N, I)$ for N set of natural numbers and $P_I : <, c_I : 3$

The formula F becomes an obviously **false** mathematical statement

$$F_I : (\exists_{n \in N} n < 3 \Rightarrow \forall_{n \in N} n < 3)$$

Restricted Quantifiers Laws

We have to be **very careful** when we deal with quantifiers with **restricted domain**.

For example, the **most basic predicate tautology**

$(\forall x A(x) \Rightarrow \exists x A(x))$ **fails** when written with the **restricted domain** quantifiers.

Example

We show that $\not\models_p (\forall_{B(x)} A(x) \Rightarrow \exists_{B(x)} A(x))$.

To **prove** this we have to show that corresponding formula of \mathcal{L} obtained by the restricted quantifiers **transformations rules** **is not** a predicate tautology, i.e. to prove:

$$\not\models_p (\forall x(B(x) \Rightarrow A(x)) \Rightarrow \exists x(B(x) \cap A(x))).$$

Restricted Quantifiers Laws

We construct a **counter-model M** for the formula

$F : (\forall x(B(x) \Rightarrow A(x)) \Rightarrow \exists x(B(x) \cap A(x)))$ as follows

We take $\mathbf{M} = (N, I)$, where N is the set of natural numbers, we take as $B(x), A(x)$ atomic formulas $Q(x, c), P(x, c)$, and the interpretation I is defined as $Q_I : <, P_I : >, c_I : 0$

The formula F becomes a **mathematical statement**

$$F_I : (\forall_{n \in N} (x < 0 \Rightarrow n > 0) \Rightarrow \exists_{n \in N} (n < 0 \cap n > 0))$$

F_I is a **false** because the statement $n < 0$ is **false** for all natural numbers and the implication **false** $\Rightarrow B$ is **true** for any logical value of B

Hence $\forall_{n \in N} (n < 0 \Rightarrow n > 0)$ is a **true** statement and $\exists_{n \in N} (n < 0 \cap n > 0)$ is obviously **false**

Restricted Quantifiers Laws

Restricted Quantifiers Law corresponding to the basic **predicate tautology**

$$(\forall x A(x) \Rightarrow \exists x A(x))$$

is:

$$\models_p (\forall_{B(x)} A(x) \Rightarrow (\exists x B(x) \Rightarrow \exists_{B(x)} A(x))).$$

It means that we **prove** that the corresponding proper formula of \mathcal{L} obtained by the **restricted quantifiers** transformations rules is a predicate tautology, i.e. we **prove** that

$$\models_p (\forall x (B(x) \Rightarrow A(x)) \Rightarrow (\exists x B(x) \Rightarrow \exists x (B(x) \cap A(x))))$$

Quantifiers Laws

Another **basic predicate tautology** called a **dictum de omni** law is:

For any formulas $A(x), A(y)$ with free variables $x, y \in VAR$,

$$\models_p (\forall x A(x) \Rightarrow A(y))$$

The corresponding **restricted quantifiers law** is:

$$\models_p (\forall_{B(x)} A(x) \Rightarrow (B(y) \Rightarrow A(y))),$$

where $y \in VAR$

Quantifiers Laws

The next important laws are the **Distributivity Laws**

Distributivity of **existential quantifier** over **conjunction** holds only in **one direction**, namely the following is a predicate tautology.

$$\models_p (\exists x (A(x) \wedge B(x)) \Rightarrow (\exists x A(x) \wedge \exists x B(x))),$$

where $A(x), B(x)$ are **any formulas** with a free variable x

The **inverse implication** **is not** a **predicate tautology**, i.e. we have to **find** concrete **formulas** $A(x), B(x) \in \mathcal{F}$ and find a model structure $\mathbf{M} = (U, I)$ with the interpretation I of all **predicate**, **functional**, and **constant** symbols in the formulas $A(x), B(x)$, such that $\mathbf{M} = (U, I)$ is **counter-model** for the **inverse implication**

$$F : ((\exists x A(x) \wedge \exists x B(x)) \Rightarrow \exists x (A(x) \wedge B(x)))$$

Quantifiers Laws

Let F be a formula

$$F : ((\exists x A(x) \wedge \exists x B(x)) \Rightarrow \exists x (A(x) \wedge B(x)))$$

Counter - Model for F is as follows

Take $\mathbf{M} = (R, I)$ where R is the set of real numbers.

Let $A(x), B(x)$ be atomic formulas $Q(x, c), P(x, c)$

$$F : ((\exists x Q(x, c) \wedge \exists x P(x, c)) \Rightarrow \exists x (Q(x, c) \wedge P(x, c)))$$

We define the interpretation I as $Q_I : >, P_I : <, c_I : 0$.

The formula F becomes an obviously **false mathematical statement**

$$F_I : ((\exists_{x \in R} x > 0 \wedge \exists_{x \in R} x < 0) \Rightarrow \exists_{x \in R} (x > 0 \wedge x < 0))$$

Quantifiers Laws

Distributivity of **universal quantifier** over **disjunction** holds only on **one direction**, namely the following is a predicate tautology for any formulas $A(x), B(x)$ with a free variable x .

$$\models_p ((\forall x A(x) \cup \forall x B(x)) \Rightarrow \forall x (A(x) \cup B(x))).$$

The **inverse implication** **is not** a predicate tautology, i.e. **there are** formulas $A(x), B(x)$ with a free variable x , such that

$$\not\models_p (\forall x (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x)))$$

Quantifiers Laws

It means that we have to find concrete formulas

$A(x), B(x) \in \mathcal{F}$ and a model structure $\mathbf{M} = (U, I)$ that is a **counter-model** for the formula

$$F : (\forall x (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x))).$$

Take $\mathbf{M} = (R, I)$ where R is the set of real numbers, and $A(x), B(x)$ are atomic formulas $Q(x, c), R(x, c)$.

We define $Q_I : \geq, R_I : <, c_I : 0$.

The formula F becomes an obviously **false** mathematical statement

$$F_I : (\forall_{x \in R} (x \geq 0 \cup x < 0) \Rightarrow (\forall_{x \in R} x \geq 0 \cup \forall_{x \in R} x < 0)).$$

Logical Equivalence

The most frequently used laws of quantifiers have a form of a **logical equivalence**, symbolically written as \equiv .

Remember that \equiv not a new logical connective.

This is a very **useful symbol**. It **says** that two formulas always have the **same logical value**, hence it can be used in the same way we the equality symbol $=$.

Formally we define it as follows.

Definition

For any formulas $A, B \in \mathcal{F}$ of the **predicate language** \mathcal{L} ,

$$A \equiv B \text{ if and only if } \models_p (A \leftrightarrow B).$$

We have also a similar definition for the **propositional language** and **propositional tautology**.

Equational Laws for Quantifiers

De Morgan

For any formula $A(x) \in \mathcal{F}$ with a free variable x ,

$$\neg \forall x A(x) \equiv \exists x \neg A(x), \quad \neg \exists x A(x) \equiv \forall x \neg A(x)$$

Definability

For any formula $A(x) \in \mathcal{F}$ with a free variable x ,

$$\forall x A(x) \equiv \neg \exists x \neg A(x), \quad \exists x A(x) \equiv \neg \forall x \neg A(x)$$

Equational Laws for Quantifiers

Renaming the Variables

Let $A(x)$ be any formula with a **free** variable x
and let y be a variable that **does not occur** in $A(x)$.

Let $A(x/y)$ be a result of **replacement** of **each** occurrence of x by y , then the following holds.

$$\forall x A(x) \equiv \forall y A(y), \quad \exists x A(x) \equiv \exists y A(y)$$

Alternations of Quantifiers

Let $A(x, y)$ be any formula with a **free** variables x and y .

$$\forall x \forall y (A(x, y)) \equiv \forall y \forall x (A(x, y)),$$

$$\exists x \exists y (A(x, y)) \equiv \exists y \exists x (A(x, y))$$

Equational Laws for Quantifiers

Introduction and Elimination Laws

If B is a formula such that B **does not contain** any **free** occurrence of x , then the following logical equivalences hold.

$$\forall x(A(x) \cup B) \equiv (\forall xA(x) \cup B),$$

$$\exists x(A(x) \cup B) \equiv (\exists xA(x) \cup B),$$

$$\forall x(A(x) \cap B) \equiv (\forall xA(x) \cap B),$$

$$\exists x(A(x) \cap B) \equiv (\exists xA(x) \cap B)$$

Equational Laws for Quantifiers

Introduction and Elimination Laws

If B is a formula such that B **does not contain** any **free** occurrence of x , then the following logical equivalences hold.

$$\forall x(A(x) \Rightarrow B) \equiv (\exists x A(x) \Rightarrow B),$$

$$\exists x(A(x) \Rightarrow B) \equiv (\forall x A(x) \Rightarrow B),$$

$$\forall x(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall x A(x)),$$

$$\exists x(B \Rightarrow A(x)) \equiv (B \Rightarrow \exists x A(x))$$

Equational Laws for Quantifiers

Distributivity Laws

Let $A(x), B(x)$ be any formulas with a **free** variable x .

Distributivity of **universal quantifier** over **conjunction**.

$$\forall x (A(x) \cap B(x)) \equiv (\forall x A(x) \cap \forall x B(x))$$

Distributivity of **existential quantifier** over **disjunction**.

$$\exists x (A(x) \cup B(x)) \equiv (\exists x A(x) \cup \exists x B(x))$$

Equational Laws for Quantifiers

We also define the notion of logical equivalence \equiv for the formulas of the **propositional language** and its semantics

For any formulas $A, B \in \mathcal{F}$ of the **propositional language** \mathcal{L} ,

$$A \equiv B \quad \text{if and only if} \quad \models (A \Leftrightarrow B)$$

Moreover, we prove that **any substitution** of **propositional tautology** by a formulas of the **predicate language** is a **predicate tautology**

The same holds for the **logical equivalence**

Equational Laws for Quantifiers

In particular, we transform the **propositional tautologies** into the following corresponding **predicate equivalences**.

For any formulas A, B of the **predicate language** \mathcal{L} ,

$$(A \Rightarrow B) \equiv (\neg A \cup B),$$

$$\neg\neg A \equiv A$$

We use them to prove the following **De Morgan Laws** for **restricted quantifiers**.

Equational Laws for Quantifiers

Restricted De Morgan

For any formulas $A(x), B(x) \in \mathcal{F}$ with a **free** variable x ,

$$\neg \forall_{B(x)} A(x) \equiv \exists_{B(x)} \neg A(x), \quad \neg \exists_{B(x)} A(x) \equiv \forall_{B(x)} \neg A(x).$$

Here is a poof of first equality. The proof of the second one is similar and is left as an exercise.

$$\begin{aligned} \neg \forall_{B(x)} A(x) &\equiv \neg \forall x (B(x) \Rightarrow A(x)) \\ &\equiv \neg \forall x (\neg B(x) \cup A(x)) \\ &\equiv \exists x \neg (\neg B(x) \cup A(x)) \equiv \exists x (\neg \neg B(x) \cap \neg A(x)) \\ &\equiv \exists x (B(x) \cap \neg A(x)) \equiv \exists_{B(x)} \neg A(x). \end{aligned}$$