cse352
ARTIFICIAL INTELLIGENCE

Professor Anita Wasilewska

LANGUAGES LECTURE

From Chapter 2 from the book

LOGICS FOR COMPUTER SCIENCE:

Classical and Non-Classical

Anita Wasilewska

Springer 2018

Propositional and Predicate Languages

PART 1: Propositional Languages

PART 2: Predicate Languages

PART 3: Translations to Predicate Languages

PART 1: Propositional Languages

Propositional Language

Definition

A **propositional language** is a pair

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

where $\mathcal{A}, \mathcal{F}$ are called an **alphabet** and a **set of formulas**, respectively

Definition

Alphabet is a set

$$\mathcal{A} = VAR \cup CON \cup PAR$$

VAR, CON, PAR are all disjoint sets of propositional **variables, connectives** and **parenthesis**, respectively

The sets VAR, CON are non-empty

Alphabet Components

VAR is a countably infinite set of **propositional variables**

We denote elements of VAR by a, b, c,d, ... with indices if necessary

$CON \neq \emptyset$ is a finite set of **logical connectives**

We assume that the set CON of logical connectives is non-empty, i.e. that a propositional language always has at least one logical connective

Notation

We denote the language $\mathcal{L}$ with the set of connectives CON by $\mathcal{L}_{CON}$

Observe that propositional languages **differ** only on the choice of the logical connectives hence our notation

Alphabet Components

PAR is a set of **auxiliary symbols**

This set may be empty; for example in case of Polish notation

Assumptions

We assume here that PAR contains only 2 **parenthesis** and

$$PAR = \{(,)\}$$

We also assume that the set CON of logical connectives **contains only unary** and **binary** connectives, i.e.

$$CON = C_1 \cup C_2$$

where C_1 is the set of all unary connectives, and C_2 is the set of all binary connectives

It is possible to create connectives with more then one or two arguments

We consider here only one or two argument connectives

General Principles

Propositional connectives have well established **names** and the way we read them, even if their semantics may differ

We use names **negation, conjunction, disjunction** and **implication** for $\neg$, $\cap$, $\cup$, $\Rightarrow$, respectively

The connective $\uparrow$ is called **alternative negation** and

$A \uparrow B$ reads: not both A and B

The connective $\downarrow$ is called **joint negation**

and $A \downarrow B$ reads: neither A nor B

Some Non-Classical Propositional Connectives

Other most common propositional connectives are **modal** connectives of **possibility** and **necessity**

Standard modal symbols are:

$\Box$ for **necessity** and $\Diamond$ for **possibility**.

The formula $\Diamond A$ reads:

it is **possible** that A or A is **possible**

The formula $\Box A$ reads:

it is **necessary** that A or A is **necessary**

Some Artificial Intelligence Non-Classical Connectives

Knowledge logics also extend the classical logic by adding new **one argument** knowledge connectives

The knowledge connective is often denoted by K

A formula KA reads: it is **known** that A or A is **known**

A language of a **knowledge logic** is for example

$$\mathcal{L}_{\{K, \neg, \cap, \cup, \Rightarrow\}}$$

More Artificial Intelligence Non-Classical Connectives

Autoepistemic logics extend classical logic by adding one argument believe connectives, often denoted by B

A formula BA reads: it is believed that A

A language of an autoepistemic logic is for example

$$\mathcal{L}_{\{\ B,\ \neg,\ \cap,\ \cup,\ \Rightarrow\}}$$

Some Computer Science Non-Classical Connectives

Temporal logics also extend classical logic by adding one argument temporal connectives

Some of temporal connectives are: F, P, G, H.

Their intuitive meanings are:

FA reads A is true at some future time,

PA reads A was true at some past time,

GA reads A will be true at all future times,

HA reads A has always been true in the past

Definition

The set $\mathcal{F}$ of **all formulas** of a propositional language $\mathcal{L}_{CON}$ is build **recursively** from the elements of the alphabet $\mathcal{A}$ as follows.

$\mathcal{F} \subseteq \mathcal{A}^*$ and $\mathcal{F}$ is the **smallest** set for which the following conditions are satisfied

(1) $VAR \subseteq \mathcal{F}$

(2) If $A \in \mathcal{F}$, $\triangledown \in C_1$, then $\triangledown A \in \mathcal{F}$

(3) If $A, B \in \mathcal{F}$, $\circ \in C_2$ i.e $\circ$ is a two argument connective, then $(A \circ B) \in \mathcal{F}$

By (1) propositional variables are formulas and they are called **atomic formulas**

The set $\mathcal{F}$ is also called a set of all **well formed formulas** (wff) of the language $\mathcal{L}_{CON}$

Set of Formulas

Observe that the the alphabet $\mathcal{A}$ is countably infinite

Hence the set $\mathcal{A}^*$ of all finite sequences of elements of $\mathcal{A}$ is also countably infinite

By definition $\mathcal{F} \subseteq \mathcal{A}^*$ and hence we get that the set of all formulas $\mathcal{F}$ is also countably infinite

We state as separate fact

Fact

For any propositional language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$, its sets of formulas $\mathcal{F}$ is always a **countably infinite** set

We hence consider here only infinitely countable languages

Exercise 1

Consider a language

$$\mathcal{L} = \mathcal{L}_{\{\neg,\ \Diamond,\ \Box,\ \cup,\ \cap,\ \Rightarrow\}}$$

and a set $S \subseteq \mathcal{A}^*$ such that

$$S = \{\Diamond\neg a \Rightarrow (a \cup b), (\Diamond(\neg a \Rightarrow (a \cup b))), \\ \Diamond\neg(a \Rightarrow (a \cup b))\}$$

1. Determine which of the elements of S are, and which are not well formed formulas (wff) of $\mathcal{L}$
2. If a formula A is a well formed formula, i.e. $A \in \mathcal{F}$, determine its its main connective.
3. If $A \notin \mathcal{F}$ write the correct formula and then determine its **main connective**

Exercise 1 Solution

Solution

The formula $\Diamond\neg a \Rightarrow (a \cup b)$ **is not a well formed formula**

The correct formula is

$$(\Diamond\neg a \Rightarrow (a \cup b))$$

The main connective is $\Rightarrow$

The correct formula says:

If negation of a is possible, then we have a or b

Another correct formula in is

$$\Diamond(\neg a \Rightarrow (a \cup b))$$

The main connective is $\Diamond$

The corrected formula says:

It is possible that not a implies a or b

Exercise 1 Solution

The formula $(\Diamond(\neg a \Rightarrow (a \cup b)))$ **is not correct**

The correct formula is

$$\Diamond(\neg a \Rightarrow (a \cup b))$$

The main connective is $\Diamond$

The correct formula says:

It is possible that not a implies a or b

$\Diamond\neg(a \Rightarrow (a \cup b))$ is a **correct formula**

The main connective is $\Diamond$

The formula says:

It is possible that it is not true that a implies a or b

Language Defined by a set S

Definition

Given a set S of formulas of a language $\mathcal{L}_{CON}$

Let $CS \subseteq CON$ be the set of **all connectives** that appear in formulas of S

A language

$$\mathcal{L}_{CS}$$

is called the **language defined** by the set of formulas S

Example

Let S be a set

$S = \{((a \Rightarrow \neg b) \Rightarrow \neg a),\ \Box(\neg\Diamond a \Rightarrow \neg a)\}$

All connectives appearing in the formulas in S are:

$$\Rightarrow,\quad \neg,\quad \Box,\quad \Diamond$$

The **language defined** by the set S is

$$\mathcal{L}_{\{\neg,\ \Rightarrow,\ \Box,\ \Diamond\}}$$

Exercise 2

Write the following natural language statement:

From the fact that it is possible that Anne is not a boy we deduce that it is not possible that Anne is not a boy or, if it is possible that Anne is not a boy, then it is not necessary that Anne is pretty

in the following two ways

1. As a formula

$A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}}$

2. As a formula

$A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$

Exercise 2 Solution

1.We translate our statement into a formula

$A_1 \in \mathcal{F}_1$ of the language $\mathcal{L}_{\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}}$ as follows

Propositional Variables: a,b

a denotes statement: *Anne is a boy*,

b denotes a statement: *Anne is pretty*

Propositional Modal Connectives: $\Box$, $\Diamond$

$\Diamond$ denotes statement: *it is possible that*

$\Box$ denotes statement: *it is necessary that*

Translation 1: the formula A_1 is

$$(\Diamond\neg a \Rightarrow (\neg\Diamond\neg a \cup (\Diamond\neg a \Rightarrow \neg\Box b)))$$

Exercise 2 Solution

2. We translate our statement into a formula $A_2 \in \mathcal{F}_2$ of the language $\mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$ as follows

Propositional Variables: a,b

a denotes statement: *it is possible that Anne is not a boy*

b denotes a statement: *it is necessary that Anne is pretty*

Translation 2: the formula A_2 is

$$(a \Rightarrow (\neg a \cup (a \Rightarrow \neg b)))$$

Exercise 3

Exercise 3

Write the following natural language statement:

For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$, OR it is not possible that there is a natural number m, such that $m > 0$

in the following two ways

1. As a formula

$A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$

2. As a formula

$A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$

Exercise 3 Solution

1. We translate our statement into a formula

$A_1 \in \mathcal{F}_1$ of the language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$ as follows

Propositional Variables: a, b

a denotes statement: *For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$*

b denotes a statement: *it is possible that there is a natural number m, such that $m > 0$*

Translation: the formula A_1 is

$$(a \cup \neg b)$$

Exercise 3 Solution

2. We translate our statement into a formula $A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}}$ as follows

Propositional Variables: a, b

a denotes statement: *For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$*

b denotes a statement: *there is a natural number m, such that $m > 0$*

Translation: the formula A_2 is

$$(a \cup \neg\Diamond b)$$

Exercise 4

Write the following natural language statement:

The following statement holds for all natural numbers $n \in N$:

if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$, OR it is not possible that there is a natural number m, such that $m > 0$

in the following two ways

1. As a formula

$A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$

2. As a formula

$A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}}$

Exercise 5

Write the following natural language statement:

From the fact that each natural number is greater than zero we deduce that it is not possible that Anne is a boy or, if it is possible that Anne is not a boy, then it is necessary that it is not true that each natural number is greater than zero

in the following two ways

1. As a formula

$A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$

2. As a formula

$A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$

PART 2: Predicate Languages

Predicate Languages

Predicate Languages are also called First Order Languages

The same applies to the use of terms for Propositional and Predicate Logic

Propositional and **Predicate Logics** called Zero Order and First Order Logics, respectively and we will use both terms equally

We usually work with different predicate languages, depending on what applications we have in mind

All **predicate languages** have some common features, and we begin with these

Predicate Languages Components

Propositional Connectives

Predicate Languages **extend** a notion of the propositional languages so we define the set CON of their propositional connectives as follows

The set CON of **propositional connectives** is a **finite** and **non-empty** and

$$CON = C_1 \cup C_2$$

where C_1, C_2 are the sets of one and two arguments connectives, respectively

Parenthesis

As in the propositional case, we adopt the signs (and) for our parenthesis., i.e. we define a set PAR as

$$PAR = \{ (, \ ) \}$$

Predicate Languages Components

Quantifiers

We adopt two quantifiers; the **universal quantifier** denoted by $\forall$ and the **existential quantifier** denoted by $\exists$, i.e. we have the following set **Q** of quantifiers

$$\mathbf{Q} = \{\forall,\ \exists\}$$

In a case of the classical logic and the logics that **extend it**, it is possible to adopt only one quantifier and to **define the other** in terms of it and propositional connectives

Such definability is impossible in a case of some non-classical logics, for example the intuitionistic logic

But even in the case of **classical logic** the two quantifiers express better the common intuition, so we adopt the both of them

Predicate Languages Components

Variables

We assume that we always have a **countably infinite** set VAR of variables, i.e. we assume that

$$cardVAR = \aleph_0$$

We denote variables by $x, y, z, ...$, with indices, if necessary.

we often express it by writing

$$VAR = \{x_1, x_2, ...\}$$

Note

Predicate Languages Components

The set *CON* of **propositional connectives** defines a propositional part of the **predicate logic language**

Observe that what really differ one **predicate language** from the other is the choice of additional symbols added to the symbols just described

These **additional symbols** are: predicate symbols, function symbols, and constant symbols

A **particular** predicate language is determined by specifying these additional sets of symbols

They are defined as follows

Predicate Languages Components

Predicate symbols

Predicate symbols **represent relations**

Any predicate language must have **at least one** predicate symbol

Hence we assume that any predicate language contains a non empty, finite or countably infinite set

$$\mathbf{P}$$

of **predicate symbols**, i.e. we assume that

$$0 < card\ \mathbf{P} \leq \aleph_0$$

We denote predicate symbols by $P, Q, R, \ldots$, with indices, if necessary

Each predicate symbol $P \in \mathbf{P}$ has a positive integer $\#P$ assigned to it; when $\#P = n$ we **call** P an n-ary (n - place) **predicate (relation) symbol**

Predicate Languages Components

Function symbols

We assume that any predicate language contains a finite (may be empty) or countably infinite set **F** of **function symbols**

I.e. we assume that

$$0 \leq card\ \mathbf{F} \leq \aleph_0$$

When the set **F** is empty we say that we deal with a **language without functional symbols**

We denote functional symbols by $f, g, h, \ldots$ with indices, if necessary

Similarly, as in the case of predicate symbols, each **function symbol** $f \in \mathbf{F}$ has a positive integer $\#f$ assigned to it; if $\#f = n$ then f is called an n-ary (n - place) **function symbol**

Predicate Languages Components

Constant symbols

We also assume that we have a finite (may be empty) or countably infinite set

$$\mathbf{C}$$

of **constant symbols**

I.e. we assume that

$$0 \leq card\ \mathbf{C} \leq \aleph_0$$

The elements of **C** are **denoted** by $c, d, e...$, with indices, if necessary

We often express it by putting

$$\mathbf{C} = \{c_1, c_2, ...\}$$

When the set **C** is **empty** we say that we deal with a language without constant symbols

Alphabet of Predicate Languages

Sometimes the **constant symbols** are defined as **0-ary function symbols**, i.e. we have that

$$\mathbf{C} \subseteq \mathbf{F}$$

We single them out as a separate set for our convenience

We assume that all of the above sets of symbols are **disjoint**

Alphabet

The union of all of above disjoint sets of symbols is called the **alphabet** $\mathcal{A}$ of the **predicate language**, i.e. we **define**

$$\mathcal{A} = VAR \cup CON \cup PAR \cup \mathbf{Q} \cup \mathbf{P} \cup \mathbf{F} \cup \mathbf{C}$$

Predicate Languages Notation

Observe, that once the set of propositional connectives is fixed, the **predicate language** is determined by the sets **P, F** and **C**

We use the **notation**

$$\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$$

for the **predicate language** $\mathcal{L}$ **determined** by **P, F, C**

If there is no danger of confusion, we may **abbreviate** $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ to just $\mathcal{L}$

If the set of propositional connectives involved is not fixed, we also use the notation

$$\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$$

to denote the **predicate language** $\mathcal{L}$ **determined** by **P, F, C** and the set of propositional connectives *CON*

Predicate Languages Notation

We sometimes allow the same symbol to be used as an n-place relation symbol, and also as an m-place one; no confusion should arise because the different uses can be told apart easily

Example

If we write $P(x, y)$, the symbol P denotes **2-argument** predicate symbol

If we write $P(x, y, z)$, the symbol P denotes **3-argument** predicate symbol

Similarly for **function symbols**

Two more Predicate Language Components

Having defined the alphabet we now complete the formal **definition of the predicate language** by defining two more components:

the set $\mathbf{T}$ of all **terms** and

the set $\mathcal{F}$ of all **well formed formulas**

of the **language** $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$

Set of Terms

Terms

The set $\mathbf{T}$ of **terms** of the **predicate language** $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is the **smallest** set

$$\mathbf{T} \subseteq \mathcal{A}^*$$

meeting the conditions:

1. any variable is a **term**, i.e. $VAR \subseteq \mathbf{T}$
2. any constant symbol is a **term,** i.e. $\mathbf{C} \subseteq \mathbf{T}$
3. if f is an n-place **function symbol**, i.e. $f \in \mathbf{F}$ and $\# f = n$ and $t_1, t_2, ..., t_n \in T$, then $f(t_1, t_2, ..., t_n) \in \mathbf{T}$

Terms Examples

Example 1

Let $f \in \mathbf{F}, \# f = 1$, i.e. f is a 1-place **function symbol**

Let x, y be **variables**, c, d be **constants**, i.e.
$x, y \in VAR, c, d \in \mathbf{C}$

Then the following expressions are **terms**:

$$x,\ y,\ f(x),\ f(y),\ f(c), f(d),\ f(f(x)),\ f(f(y)),\ f(f(c)),\ f(f(d)), \ldots$$

Example 2

Let $\mathbf{F} = \emptyset, \mathbf{C} = \emptyset$

In this case terms consists of **variables only**, i.e.

$$\mathbf{T} = VAR = \{x_1, x_2, ...\ \}$$

Terms Examples

Directly from the **Example 2** we get the following

REMARK

For any predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, the set $\mathbf{T}$ of its **terms** is always **non-empty**

Example 3

Let $f \in \mathbf{F}, \#f = 1, g \in \mathbf{F}, \#g = 2, x, y \in VAR, c, d \in \mathbf{C}$

Some of the **terms** are the following:

$$f(g(x, y)),\ f(g(c, x)),\ g(f(f(c)), g(x, y)),$$

$$g(c, g(x, f(c))), g(f(g(x, y)), g(x, f(c)))\ ....$$

Terms Notation

From time to time, the logicians are and we may be informal about how we write terms

Example

If we **denote** a 2- place function symbol g by $+$, we **may write** $x + y$ instead $+(x, y)$

Because in this case we can think of $x + y$ as an unofficial way of designating the **"real" term** $g(x, y)$

Atomic Formulas

Before we define the **set of formulas**, we need to define one more set; the set of **atomic**, or **elementary** formulas

Atomic formulas are the simplest formulas as the propositional variables were in the case of **propositional languages**

Atomic Formulas

Definition

An **atomic formula** of a **predicate language** $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is any element of $\mathcal{A}^*$ of the form

$$R(t_1, t_2, ..., t_n)$$

where $R \in \mathbf{P}, \#R = n$ and $t_1, t_2, ..., t_n \in \mathbf{T}$

I.e. R is n-ary **relational symbol** and $t_1, t_2, ..., t_n$ are **any terms**

The set of all **atomic formulas** is denoted by $A\mathcal{F}$ and is defined as

$$A\mathcal{F} = \{R(t_1, t_2, ..., t_n) \in \mathcal{A}^* : \quad R \in \mathbf{P},\ t_1, t_2, ..., t_n \in \mathbf{T},\ n \geq 1\}$$

Atomic Formulas Examples

Example 1

Consider a language $\mathcal{L}(\{P\}, \emptyset, \emptyset)$, for #P = 1

Our language

$$\mathcal{L} = \mathcal{L}(\{P\}, \emptyset, \ \emptyset)$$

is a language **without** neither **functional**, nor **constant** symbols, and with one, 1-place **predicate** symbol P

The set of **atomic formulas** contains all formulas of the form $P(x)$, for x any variable, i.e.

$$A\mathcal{F} = \{P(x) : x \in VAR\}$$

Atomic Formulas Examples

Example 2

Let now consider a **predicate language**

$$\mathcal{L} = \mathcal{L}(\{R\}, \{f, g\}, \ \{c, d\})$$

for $\#f = 1, \#g = 2, \#R = 2$

The language $\mathcal{L}$ has **two functional symbols:** 1-place symbol f and 2-place symbol g, one 2-place **predicate symbol** R, and two **constants:** c,d

Some of the **atomic formulas** in this case are the following.

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

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

Set of Formulas Definition

Now we are ready to define the set $\mathcal{F}$ of all **well formed formulas** of any **predicate language** $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$

Definition

The set $\mathcal{F}$ of all **well formed formulas**, called shortly **set of formulas**, of the language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is the smallest set meeting the following **four conditions:**

1. Any **atomic formula** of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is a **formula** , i.e.

$$A\mathcal{F} \subseteq \mathcal{F}$$

2. If A is a formula of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, $\triangledown$ is an one argument **propositional connective**, then $\triangledown A$ is a formula of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. the following **recursive condition** holds

$$\text{if } A \in \mathcal{F}, \triangledown \in C_1 \text{ then } \triangledown A \in \mathcal{F}$$

Set of Formulas Definition

3. If A, B are **formulas** of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ and $\circ$ is a two argument **propositional connective**, then $(A \circ B)$ is a **formula** of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. the following **recursive condition** holds

$$\text{If } \quad A \in \mathcal{F}, \nabla \in C_2, \quad \text{then} \quad (A \circ B) \in \mathcal{F}$$

4. If A is a **formula** of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ and x is a **variable**, $\forall, \exists \in \mathbf{Q}$, then $\forall xA$, $\exists xA$ are **formulas** of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. the following recursive condition holds

$$\text{If } \quad A \in \mathcal{F}, \; x \in VAR, \quad \forall, \exists \in \mathbf{Q}, \quad \text{then} \quad \forall xA, \; \exists xA \in \mathcal{F}$$

Scope of the Quantifier

Another important notion of the **predicate language** is the notion of **scope of a quantifier**

It is defined as follows

Definition

Given formulas $\forall x A$, $\exists x A$, the formula A is said to be in the **scope of the quantifier** $\forall$, $\exists$, respectively.

Example 3

Let $\mathcal{L}$ be a language of the previous **Example 2** with the set of connectives $\{\cap, \cup, \Rightarrow, \neg\}$, i.e. let's consider

$$\mathcal{L} = \mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}(\{f, g\}, \{R\}, \{c, d\})$$

for #f = 1, #g = 2 , #R = 2

Some of the formulas of $\mathcal{L}$ are the following.

$$R(c, d), \;\; \exists y R(y, f(c)), \;\; \neg R(x, y),$$

$$(\exists x R(x, f(c)) \Rightarrow \neg R(x, y)), \quad (R(c, d) \cap \forall z R(z, f(c))),$$

$$\forall y R(y, \; g(c, g(x, f(c)))), \quad \forall y \neg \exists x R(x, y)$$

Scope of Quantifiers

The formula $R(x, f(c))$ is in **scope of the quantifier** $\exists$ in the formula

$$\exists x R(x, f(c))$$

The formula $(\exists x\ R(x, f(c)) \Rightarrow \neg R(x, y))$ **is not in scope of any quantifier**

The formula $(\exists x R(x, f(c)) \Rightarrow \neg R(x, y))$ is in **scope** of quantifier $\forall$ in the formula

$$\forall y(\exists x R(x, f(c)) \Rightarrow \neg R(x, y))$$

Predicate Language Definition

Now we are ready to define formally a **predicate language**

Let $\mathcal{A}, T, \mathcal{F}$ be the **alphabet** , the set of **terms** and the set of **formulas** as already defined

Definition

A **predicate language** $\mathcal{L}$ is a triple

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

As we have said before, the language $\mathcal{L}$ is determined by the **choice** of the symbols of its **alphabet**, namely of the **choice** of **connectives**, **predicates**, **functions** , and **constants** symbols

If we want specifically mention these **choices**, we write

$$\mathcal{L} = \mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C}) \quad \text{or} \quad \mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$$

Free and Bound Variables

Given a **predicate language** $\mathcal{L} = (\mathcal{A}, T, \mathcal{F})$, we must distinguish between formulas like

$$P(x,y), \quad \forall x P(x,y) \quad \text{and} \quad \forall x \exists y P(x,y)$$

This is done by introducing the notion of **free** and **bound** variables, and **open** and **closed** formulas

Closed formulas are also called **sentences**

Informally, in the formula

$$P(x,y)$$

both variables x and y are called **free** variables

They **are not** in the **scope** of any quantifier

The formula of that type, i.e. formula **without quantifiers** is an **open formula**

Free and Bound Variables

In the formula

$$\forall y P(x, y)$$

the variable x is **free**, the variable y is **bounded** by the the quantifier $\forall$

In the formula

$$\forall z P(x, y)$$

both x and y are **free**

In the formulas

$$\forall z P(z, y), \quad \forall x P(x, y)$$

only the variable y is **free**

Free and Bound Variables

In the formula

$$\forall x(P(x) \Rightarrow \exists y Q(x,y))$$

there is **no free variables**

In the formula

$$(\forall x P(x) \Rightarrow \exists y Q(x,y))$$

the variable x (in $Q(x,y)$) is **free**

Sometimes in order to distinguish more easily **which** variable is **free** and which is **bound** in the formula we might use the bold face type for the quantifier bound variables, i.e. to write the last formulas as

$$(\forall \mathbf{x} P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(x,\mathbf{y}))$$

Bound Variables, Sentence, Open Formula

Bound variables: a variable is called **bound** if it is not free

Sentence: a formula with **no free variables** is called a sentence

Open formula: a formula with **no bound variables** is called an open formula

Example

The formulas

$$\exists x Q(c, g(x, d)), \qquad \neg\forall x(P(x) \Rightarrow \exists y(R(f(x), y) \cap \neg P(c)))$$

are **sentences**

The formulas

$$Q(c, g(x, d)), \qquad \neg(P(x) \Rightarrow (R(f(x), y) \cap \neg P(c)))$$

are **open formulas**

Examples

Example

The formulas

$$\exists x Q(c, g(x, y)), \qquad \neg(P(x) \Rightarrow \exists y(R(f(x), y) \cap \neg P(c)))$$

are **neither** sentences **nor** open formulas

They contain **some free** and **some bound** variables;

the variable y is **free** in $\exists \mathbf{x} Q(c, g(\mathbf{x}, y))$

the variable x is **free** in $\neg(P(x) \Rightarrow \exists \mathbf{y}(R(f(x), \mathbf{y}) \cap \neg P(c)))$

Notations

Notation: It is common practice to use the notation

$$A(x_1, x_2, ..., x_n)$$

to indicate that

$$FreeVariables(A) \subseteq \{x_1, x_2, ..., x_n\}$$

without implying that **all of** $x_1, x_2, ..., x_n$ are actually **free** in A

This is similar to the practice in **algebra** of writing $w(a_0, a_1, ..., a_n) = a_0 + a_1 x + ... + a_n x^n$ for a polynomial w without implying that **all** of the coefficients $a_0, a_1, ..., a_n$ are nonzero

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 then zero and some integers are equal 1" often write

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

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

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

or even as

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

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 Translations

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

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 $x \in VAR$.

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 $\mathcal{L}$

Translations to Formulas of $\mathcal{L}$

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

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

Step 1 We **identify** basic statements in **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 $> (+(x, 2), y)$, and 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

Exercise

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

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

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 Chapter 2, page 28.

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 $n \in N$, a constant c for the number 0. We use $L \in \mathbf{P}$ to denote the relation $<$ We use $f \in \mathbf{F}$ 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 $A \in \mathcal{F}$:

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

PART 3: Translations to Predicate Languages

Translations Exercises

Exercise 1

Given a Mathematical Statement written with logical symbols

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

1. Translate it into a proper logical formula with restricted domain quantifiers

2. Translate your restricted domain quantifiers logical formula into a correct logical formula **without** restricted domain quantifiers

Exercise 1 Solution

1. We translate the Mathematical Statement

$$\forall_{x \in R} \, \exists_{n \in N} (x + n > 0 \Rightarrow \exists_{m \in N} (m = x + n))$$

into a proper **logical formula** with restricted domain quantifiers as follows

Step 1

We identify all **predicates** and use their **symbolic** representation as follows:

R(x) for $x \in R$

N(x) for $n \in N$

G(x,y) for relation $>$, E(x,y) for relation $=$

Exercise 1 Solution

Step 2

We identify all **functions** and **constants** and their **symbolic** representation as follows:

f(x,y) for the function $+$, c for the constant 0

Step 3

We write **mathematical** expressions in as **symbolic logic** formulas as follows:

G(f(x,y), c) for $x + n > 0$ and E(z, f(x,y)) for $m = x + n$

Step 4

We identify logical **connectives** and **quantifiers** and write the **logical formula** with restricted domain quantifiers as follows

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

Exercise 1 Solution

2. We translate the **logical formula** with restricted domain quantifiers

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

into a correct **logical formula** **without** restricted domain quantifiers as follows

$$\forall x(R(x) \Rightarrow \exists_{N(y)}(G(f(x,y),c) \Rightarrow \exists_{N(z)} E(z, f(x,y))))$$

$$\equiv \ \forall x(R(x) \Rightarrow \exists y(N(y) \cap (G(f(x,y),c) \Rightarrow \exists_{N(z)} E(z, f(x,y)))))$$

$$\equiv \ \forall x(R(x) \Rightarrow \exists y(N(y) \cap (G(f(x,y),c) \Rightarrow \exists z(N(z) \cap E(z, f(x,y))))))$$

Correct **logical formula** is:

$$\forall x(R(x) \Rightarrow \exists y(N(y) \cap (G(f(x,y),c) \Rightarrow \exists z(N(z) \cap E(z, f(x,y))))))$$

Translations Exercises

Exercise 2

Here is a mathematical statement **S**:

For all natural numbers n the following holds:

If $n < 0$, **then** *there is a natural number m, such that* $m + n < 0$

P1. Re-write **S** as a Mathematical Statement "formula" **MSF** that only uses **mathematical** and **logical symbols**

P2. Translate your Mathematical Statement "formula" **MSF** into to a correct **predicate language formula** **LF**

P3. Argue whether the statement **S** it true of false

P4. Give an **interpretattion** of the **predicate language formula** **LF** under which it is **false**

Exercise 2 Solution

P1. We **re-write** mathematical statement **S**

For all natural numbers n the following holds:

If $n < 0$, **then** *there is a natural number m, such that* $m + n < 0$

as a Mathematical Statement "formula" **MSF** that only uses mathematical and logical symbols as follows

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

Exercise 2 Solution

P2. We translate the **MSF** "formula"

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

into a correct **predicate language** **formula** using the following **5** steps

Step 1

We identify **predicates** and write their **symbolic** representation as follows

We write N(x) for $x \in N$ and L(x,y) for relation $<$

Step 2

We identify **functions** and **constants** and write their **symbolic** representation as follows

f(x,y) for the function $+$ and c for the constant 0

Exercise 2 Solution

Step 3

We write the mathematical expressions in **S** as atomic formulas as follows:

L(f(y,c), c) for $m + n < 0$

Step 4

We identify logical **connectives** and **quantifiers** and write the **logical formula** with restricted domain quantifiers as follows

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

Exercise 2 Solution

Step 5

We translate the above into a correct **logical formula**

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

P3 Argue whether the statement **S** it true of false

Statement $\forall_{n \in N}(n < 0 \Rightarrow \exists_{m \in N}(m + n < 0))$ is TRUE as the statement $n < 0$ is FALSE for all $n \in N$ and the classical implication FALSE $\Rightarrow$ Anyvalue is always TRUE

Exercise 2 Solution

P4. Here is an **interpretation** in a non-empty set X under which the **predicate language formula**

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

is false

Take a set $X = \{1,2\}$

We **interpret** N(x) as $x \in \{1,2\}$, L(x, y) as $x > y$, and constant c as 1

We **interpret** f as a two argument function f_I defined on the set X by a formula $f_I(y,x) = 1$ for all $y, x \in \{1,2\}$

The mathematical statement

$$\forall_{x\in\{1,2\}}(x > 1 \Rightarrow \exists_{y\in\{1,2\}}(f_I(y,x) > 1))$$

is a **false statement** when $x = 2$

In this case we have $2 > 1$ is **true** and as $f_I(y,2) = 1$ for all $y \in \{1,2\}$ we get that $\exists_{y\in\{1,2\}}(f_I(y,2) > 1))$ is **false** as $1 > 1$ is **false**

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 May'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 May'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(x) stands for " x 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 $\forall_{D(x)}$ 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 $\mathcal{L}$

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

Translation in Artificial Intelligence

Translation in Artificial Intelligence

In AI we use **intended names** for relations, functions and constants

The symbolic language we use is still a symbolic language, even if the **intended names** are used.

In the AI we write, for example

Like(John, Mary)

instead of a formula $L(c_1, c_2)$ in logic.

We write

$$greater(x, y) \quad \text{or} \quad >(x, y)$$

instead of $R(x, y)$ in logic.

Example

AI **intended interpretation** formulas corresponding to a statement

S: "For every student there is a student that is an elephant"

are as follows

Restricted quantifiers AI formula:

$$\forall_{Student(x)} \exists_{Student(x)}\ Elephant(x)$$

Non-restricted quantifiers AI formula:

$$\forall x(Student(x) \Rightarrow \exists x(Student(x) \cap Elephant(x)))$$

Translation in Artificial Intelligence

Observe that a proper formulas of the LOGIC language corresponding the statement

"For every student there is a student that is an elephant" are the same as the formulas corresponding to the natural language statement

For any bird one can find some birds that white", namely

Restricted quantifiers logic formula:

$$\forall_{P(x)} \exists_{P(x)} R(x)$$

Non-restricted quantifiers logic formula:

$$\forall x(P(x) \Rightarrow \exists x(P(x) \cap R(x)))$$

Translation in Artificial Intelligence

Statement

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

translates as

Restricted quantifier AI formula:

$$((\forall_{Friend(x,Mary)} Friend(x, John) \ \cap \neg Friend(Peter, John))$$

$$(\Rightarrow \neg Friend(Peter, Mary))$$

Non-restricted AI formula:

$$((\forall x(Friend(x, Mary) \Rightarrow Friend(x, John)) \ \cap \neg Friend(Peter, John))$$

$$\Rightarrow \neg Friend(Peter, Mary))$$