

Logic and discrete mathematics (HKGAB4)<http://www.ida.liu.se/~HKGAB4/>

“All rational inquiry depends on logic, on the ability of people to reason correctly most of the time, and, when they fail to reason correctly, on the ability of others to point out the gaps in their reasoning” (Barwise & Etchemendy)

Logic: contents

1. Logic: informal introduction, syntactic and semantic perspective. Meta properties (soundness, completeness).
2. Logical connectives and 0-1 reasoning.
3. Introduction to formal reasoning: Fitch format (notation).
4. Fitch rules for propositional connectives.
5. Quantifiers.
6. Fitch rules for quantifiers.
7. Normal forms for formulas and reasoning by resolution.
8. Logic as a database querying language.
9. Modal logics.

What is logic?

The first approximation: *logic* is the science of correct reasoning, i.e., reasoning based on correct (sound) arguments. A *correct (sound)* argument is one in which anyone who accepts its premises should also accept its conclusions.

To see whether an argument is correct, one looks at the connection between the premisses and the conclusion. One does not judge whether there are good reasons for accepting the premisses, but whether person who accepted the premisses, for whatever reasons, good or bad, ought also accept the conclusion.

Examples

1. Correct arguments:
 - if x is a parent of y , and y is a parent of z , then x is a grandparent of z
 - if A and B is true, then A is true.
2. Incorrect arguments:
 - if A implies B then B implies A
 - if A or B is true, then A is true.
3. are the following arguments correct?
 - if A implies B then not B implies not A
 - if A is true, then A or B is true.

What is logic? – continued

Logical formalisms are applied in many areas of science as a basis for clarifying and formalizing reasoning. Intuitively, a logic is defined by the (family of) language(s) it uses and by its underlying reasoning machinery.

The intensive use of formal reasoning techniques resulted in defining hundreds, if not thousands, of logics that fit nicely to particular application areas.

We then first need to clarify what do we mean by a logic.

In order to make any reasoning fruitful, we have

1. to decide what is the subject of reasoning or, in other words, what are we going to talk about and what language is to be used
2. to associate a precise meaning to basic notions of the language, in order to avoid ambiguities and misunderstandings
3. to state clearly what kind of opinions (sentences) can be formulated in the language we deal with and, moreover, which of those opinions are true (valid), and which are false (invalid).

Now we can investigate the subject of reasoning via the validity of expressed opinions. Such an abstraction defines a specific logic.

What is logic? – continued

Traditionally, there are two methodologies to introduce a logic:

- *syntactically*, via a notion of a proof and proof system
- *semantically*, via a notion of a model, satisfiability and truth.

Both methodologies first require to chose a language that suits best a particular application. For example,

1. talking about politics we use terms “political party”, “prime minister”, “parliament”, “statement”, etc. etc.
2. talking about computer science phenomena we use terms “software”, “program execution”, “statement”, etc. etc.

Of course we use different vocabulary talking about different areas.

Logical language is defined by means of basic concepts, formulas and logical connectives or operators. Connectives and operators have a fixed meaning. Vocabularies reflecting particular application domains are flexible.

What is a logical language?

Language of a logic is a formal language (as defined in the discrete math part of the course), reflecting natural language phenomena and allowing one to formulate sentences (called *formulas*) about a particular domain of interest.

Example

Language of arithmetics of natural numbers consists of:

- constants, e.g., 0, 1, 2, 3, ...
- variables, e.g., l, m, n, k, \dots
- function symbols, e.g., addition (+) or multiplication (*)
- relation symbols, e.g., = or \leq .

Examples of formulas:

$$n \leq 2 * n + 1$$

$$n * (k + 1) = n * k + n$$

$$n * n * n + 2 * n * n + 3 = 20$$

However, $n^3 + 2 * n^2 + 3 = 20$ is a formula only if n^3 and n^2 are operations allowed in the language.

Elements of logical language

Building blocks of logical languages are usually among the following:

- *individual constants* (*constants*, for short), representing particular *individuals*, i.e., elements of the underlying domain
 - examples: 0, 1, *John*
- *variables*, representing a range of individuals,
 - examples: x, y, m, n
- *function symbols*, representing functions,
 - examples: +, *, *father()*
- *relation symbols*, representing relations,
 - examples: =, \leq, \preceq
- *logical constants*: TRUE, FALSE, sometimes also other,
 - examples: UNKNOWN, INCONSISTENT
- *connectives* and *operators*, allowing one to form more complex formulas from simpler formulas,
 - examples of connectives: “and”, “or”, “implies”,
 - examples of operators: “for all”, “exists”, “is necessary”, “always”
- *auxiliary symbols*, making notation easier to understand
 - examples: “(”, “)”, “[”, “]”.

Why “function/relation symbols” instead of “functions/relations”?

In natural language names are not individuals they denote!

Examples

1. Name “John” is not a person named “John”.
2. An individual can have many names,
 - e.g., “John” and “father of Jack” can denote the same person.
3. An individual may have no name – e.g., we do not give separate names for any particle in the universe.
4. Many different individuals may have the same name,
 - e.g., “John” denotes many persons.
5. Some names do not denote any real-world individuals,
 - e.g., “Pegasus”.

In logic *symbols* correspond to names.
A function/relation symbol is not a function/relation, but its name.
However, comparing to natural language, usually **a symbol denotes a unique individual.**

Terms (expressions)

Expressions formed using individual constants, variables and function symbols, are called *terms*. Intuitively, a term represents a value of the underlying domain.
A *ground term* is a term without variables.

Examples

1. In arithmetics the following expressions are terms:

$$1 + 2 + 3 + 4 * 2 \quad \text{– a ground term}$$

$$(n + 2 * k) * 5 + n * n \quad \text{– a term (but not ground)}$$

but the following are not:

$$1 + 2 + 3 + \dots + 10 \quad \text{– “...” is not a symbol}$$

in arithmetics

$$(n + 2 * k) * 5 \leq 20 \quad \text{– “≤” is a relation symbol.}$$

2. If *father* and *mother* are function symbols and *John* is a constant then the following expressions are (ground) terms:

$$father(John)$$

$$mother(father(John))$$

$$father(father(father(John)))$$

$$mother(father(mother(John))).$$

Formulas (sentences)

Expressions formed using logical constants and relation symbols, are called *formulas*. Intuitively, a formula represents an expression resulting in a logical value. A *ground formula* is formula built from relation symbols applied to ground terms only.

Examples

1. In arithmetics the following expressions are formulas:

$1 + 2 + 3 + 4 * 2 < 122$ – a ground formula
 $(n + 2 * k) * 5 + n * n = n + 2$ – a formula (but not ground),

but the following are not:

$1 + 2 + 3 + 4 * 2$
 $(n + 2 * k) * 5 + n * n$

although are well-formed terms.

2. If *father* and *mother* are function symbols, *older* is a relation symbol and *John* is a constant then the following expressions are (ground) formulas:

$older(father(John), John)$
 $older(mother(father(John), father(John)))$
 $older(father(John), mother(John)).$

What is logic? – continued

Having only a language syntax defined, one knows how to formulate sentences (i.e., how to “speak”) correctly, but does not know what do the sentences mean and whether these are true or false.

Examples

- “John Watts is a driver”
– is a correct English sentence, but is it true or false? To check it we have to know to which person “John Watts” refers to (there might be many persons with that name). What is a meaning of “driver”? A professional driver? A person having a driving license? Even if did not drive for the last thirty years?
- “To zdanie jest zbudowane poprawnie”
– is a correct Polish sentence, but what is its meaning? To find it out you should know what is the meaning of particular words and what are grammar rules and maybe even a context.
- “This is the key”
– is a correct English sentence, however the word “key” has may substantially different meanings, like “key to a door” or “key fact” or “key evidence”.

A *meaning* (also called an *interpretation*) is to be attached to any well-formed formula of the logic. The meaning is given either *semantically* (*semantical approach*) or *syntactically* (*syntactical approach*).

Examples

1. Example of semantical reasoning:
consider the sentence “Humans are rational animals”.
– Is it true? Maybe false?

No matter whether we agree or not, in our everyday arguments we would usually have to know the meaning of “humans”, “to be rational” and “animals”.

According to the meaning we have in mind we find this sentence valid or not.

2. Example of syntactical reasoning:
consider the following rule:

“*A* is *B*” and “*B* does *D*” therefore “*A* does *D*”

This rule allows us to infer (prove) facts from other facts, if we agree that it is “sound”.

Consider the previous sentence together with “Rational animals think”. To apply our rule, we interpret *A, B, C*:

- *A* denotes “humans”
- *B* denotes “rational animals”
- *D* denotes “think”

Thus we infer: “Humans think”.

Semantical approach

In the *semantical approach* we attach meaning (“real-world entities”) to symbols:

- individuals to constants
- range of individuals to variables
- functions to function symbols
- relations to relation symbols.

The meaning of connectives, operators and auxiliary symbols is fixed by a given logic.

Example

Consider sentence “John is similar to the Jack’s father”.

In a logical form we would write this sentence more or less as follows:

John is similar to *father* of *Jack*

or, much more often, as: $sim(John, fath(Jack))$, where *sim* and *fath* are suitable abbreviations of similar to and father of.

In order to verify this sentence we have to know the meaning of:

- constants *John, Jack*
- function denoted by symbol *fath*
- relation denoted by symbol *sim*.

“Humans are rational animals” – continued

Analysis of this sentence is not immediate. We have to know the meaning of “humans”, “rational” and “animals”. We can easily agree that:

- **humans** – denotes the set of humans
- **animals** – denotes the set of animals
- **rational** – denotes the set of rational creatures.

Do we have a notion of sets in our logic? In some logics we do, in some we do not.

Assume we have it. Then our sentence can be expressed as follows:

$$humans = rational \cap animals$$

since we want to say that humans are creatures that are both rational and animals.

Now we interpret “humans”, “rational” and “animals” as concrete sets and see whether the equality holds.

More often we consider relations rather than sets. In this case we would have:

- $human(x)$ – meaning that x is a human
- $rational(x)$ – meaning that x is rational
- $animal(x)$ – meaning that x is an animal.

Our sentence can be formulated as:

$$\text{for all } x, human(x) \text{ iff } rational(x) \text{ and } animal(x).$$

Now we have to interpret “human”, “rational” and “animal” as concrete relations, fix a range for variable x and carry out reasoning.

Counterexamples

Quite often, commonsense semantical reasoning depends of searching for *counterexamples*, i.e., individuals falsifying the formula to be proved. If there are no counterexamples then we conclude that the formula is valid.

Example

1. Consider sentence: “thieves are those who steal money”.

It can be formulated in logic as follows:

$$\text{for all } x, thief(x) \text{ iff } steal(x, money).$$

In order to find a counterexample we have to find x which is a thief and does not steal money or steals money but is not a thief.

However, first we have to fix the meaning of “thief”, “steal”, “money” and the range of x (which would be the set of humans).

With the standard meaning we can easily find a person who is a thief and does not steal money (but, e.g., steals jewellery only).

2. For sentence “thieves are those who steal” we would most probably be not able to find a counterexample.

We then conclude that this sentence is valid.

Syntactical approach

In the *syntactical approach* we attach meaning to symbols of the language by fixing *axioms* and *proof rules* (*rules*, in short).

- *Axioms* are facts “obviously true” in a given reality.
- *Rules* allow us to infer new facts on the basis of known facts (axioms, facts derived from axioms, etc.).

Axioms together with proof rules are called *proof systems*.

Example

Consider the following rule (called *modus ponens*):

if A holds
and whenever A holds, B holds, too (i.e., A implies B)
then infer that B holds.

Assume that we have the following two axioms:

I read a good book.
Whenever I read a good book, I learn something new.

Taking

A to be “I read a good book”,
 B to be “I learn something new”,

we have “ A ” and “ A implies B ”, thus applying modus ponens we infer “ B ”, i.e., “I learn something new”.

Discussion

In the semantical reasoning we also apply rules, maybe more implicitly, so what is the difference?

In the syntactical approach we do not care about the meaning of sentences. We just syntactically transform formulas without referring to any specific meaning of items occurring in formulas.

The meaning is then given by facts that are true or false.

Example

Consider the following rule:

if a person x is a parent of a person y
then x is older than y .

Assume we have the following axioms:

John is a person.
Eve is a person.
Marc is a person.
John is a parent of Marc.
Eve is a parent of Marc.

Applying the considered rule we can infer:

John is older than Marc.
Eve is older than Marc.

This reasoning gives us a (partial) meaning of “to be older than”.

What is logic? – Conclusion

By a *logic* we shall understand any triple $\langle \text{TV}, \mathcal{L}, \mathcal{I} \rangle$, where

- TV is the set of *truth values*, e.g., $\text{TV} = \{\text{TRUE}, \text{FALSE}\}$
- \mathcal{L} is a set of formulas, e.g., defined using formal grammars
- \mathcal{I} is an “oracle” assigning meaning to all formulas of the logic, $\mathcal{I} : \mathcal{L} \rightarrow \text{TV}$, i.e., for any formula $A \in \mathcal{L}$, the value $\mathcal{I}(A)$ is a truth value (e.g., TRUE or FALSE).

Remarks

1. We do not explicitly consider terms in the above definition. However, the meaning of terms can be provided by formulas of the form *term* = *value*. If such a formula is TRUE then the value of term *term* is *value*.
2. As discussed earlier the “oracle” can be defined syntactically or semantically. However, we do not exclude other possibilities here, although these are used only in rather theoretical investigations.
3. Truth values can also be UNKNOWN, INCONSISTENT, in some logics even all real values from the interval $[0, 1]$ (e.g., in fuzzy logics).
4. In logic the definition of the “oracle” is not required to be constructive, although automated as well as practical everyday reasoning, is to be based on some constructive machinery.

Example

Let us fix a language for arithmetics of real numbers and define the “oracle”, saying, among others, that formulas of the form “for every real number x property $A(x)$ holds are TRUE iff

for any real number substituting x in $A(x)$,
property $A(x)$ holds.

This is highly non-constructive!

In everyday practice one uses more constructive techniques based on manipulating terms and formulas.

For example, in order to prove that for every real number x we have that $x < x + 1$ one does not pick all real numbers one by one and check whether a given number is less than the number increased by 1, like e.g., checking:

$$\begin{aligned} 2.5 &< 2.5 + 1 \\ \sqrt{5} &< \sqrt{5} + 1 \\ 1238 &< 1238 + 1 \\ &\dots \end{aligned}$$

One rather observes that

1. $0 < 1$
2. adding x to both sides of any inequality preserves the inequality and obtains that $x + 0 < x + 1$, i.e. $x < x + 1$.

Some meta-properties

A *meta-property* is a property of logic rather than of the reality the logic describes.

There are two important meta-properties relating syntactical and semantical approaches, namely *soundness* (also called *correctness*) and *completeness* of a proof system wrt a given semantics.

Assume a logic is given via its semantics \mathcal{S} and via a proof system \mathcal{P} . Then we say that:

- proof system \mathcal{P} is *sound* (*correct*) wrt the semantics \mathcal{S} iff every property that can be inferred using \mathcal{P} is true under semantics \mathcal{S} ,
- proof system \mathcal{P} is *complete* wrt the semantics \mathcal{S} iff every property that is true under semantics \mathcal{S} can be inferred using \mathcal{P} .

In practical reasoning:

- **soundness is required**
- **completeness is desirable.**

What you should have learnt from Lecture I?

- what is correct argument and correct reasoning?
- what is logic?
- what is a logical language?
- what are the typical “building blocks” for expressions?
- what are function and relation symbols?
- what are terms? what are ground terms?
- what are formulas?
- what is the semantical approach?
- what is the syntactical approach?
- what is a proof system?
- what is soundness and completeness of proof systems?

Truth tables

Truth tables provide us with a semantics of logical connectives.

Truth table consists of columns representing arguments of a connective and one (last) column representing the logical value of the sentence built from arguments, using the connective.

Connectives: not

Connective *not*, denoted by \neg , expresses the *negation* of a sentence.

Truth table for negation

A	$\neg A$
FALSE	TRUE
TRUE	FALSE

Connectives: and

Connective *and*, denoted by \wedge , expresses the *conjunction* of sentences.

Truth table for conjunction

A	B	$A \wedge B$
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Examples

- “Eve is a student and works part-time in a lab.”
This sentence is true when both “Eve is a student” and “Eve works part-time in a lab” are true.
If one of these sentences is false, the whole conjunction is false, too.
- “Students and pupils read books” translates into “Students read books and pupils read books.”
- “Marc is a driver who likes his job” translates into the conjunction “Marc is a driver and Marc likes his job”, so connective “and” does not have to be explicit.

Connectives: or

Connective *or*, denoted by \vee , expresses the *disjunction* of sentences.

Truth table for disjunction

A	B	$A \vee B$
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Examples

- Sentence “Marc went to a cinema or to a shop” is false only when Marc went neither to a cinema nor to a shop.
- “Students **and** pupils read books” translates into “If a person is a student **or** is a pupil then (s)he reads books” – compare this with the translation given in the previous slide and try to check whether these translations are equivalent

Connectives: equivalent to

Connective *equivalent to*, denoted by \leftrightarrow , expresses the *equivalence* of sentences.

$A \leftrightarrow B$ is also read as “A if and only if B”.

Truth table for equivalence

A	B	$A \leftrightarrow B$
FALSE	FALSE	TRUE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Observe that definitions have a form of equivalence.
For example,

Humans are rational animals

translates into

to be a human is equivalent to be both rational and animal.

Connectives: implies

Connective *implies*, denoted by \rightarrow , expresses the *implication* of sentences. Implication $A \rightarrow B$ is also read “if A then B ”.

Truth table for implication

A	B	$A \rightarrow B$
FALSE	FALSE	TRUE
FALSE	TRUE	TRUE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

Why FALSE implies everything?

The fact that everything followed from a single contradiction (i.e., FALSE) has been noticed by Aristotle.

The great logician Bertrand Russell once claimed that he could prove anything if given that (in the standard arithmetic of natural numbers) $1 + 1 = 1$.

So one day, somebody asked him, OK. Prove that you're the Pope. He thought for a while and proclaimed,

<p>I am one. The Pope is one. One plus one is one.</p>	}	Therefore, the Pope and I are one.
--	---	------------------------------------

Example of application of propositional reasoning

Consider the following sentences:

1. if the interest rate of the central bank will not be changed then government expenses will be increased or new unemployment will arise
2. if the government expenses will not be increased then taxes will be reduced
3. if taxes will be reduced and the interest rate of the central bank will not be changed then new unemployment will not arise
4. if the interest rate of the central bank will not be changed then the government expenses will be increased.

The task is to check whether:

- (i) the conjunction of (1), (2) and (3) implies (4)
- (ii) the conjunction of (1), (2) and (4) implies (3).

The task is not very complex, but without a formal technique one might be helpless here.

We have the following atomic sentences:

- *ir* – standing for “interest rate of the central bank will be changed”
- *ge* – standing for “government expenses will be increased”
- *un* – standing for “new unemployment will arise”
- *tx* – standing for “taxes will be reduced”

The considered sentences are translated into:

1. $(\neg ir) \rightarrow (ge \vee un)$
2. $(\neg ge) \rightarrow tx$
3. $(tx \wedge \neg ir) \rightarrow \neg un$
4. $(\neg ir) \rightarrow ge$.

Our first task is then to check whether the conjunction of (1), (2) and (3) implies (4), i.e., whether:

$$\left. \begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(tx \wedge \neg ir) \rightarrow \neg un] \end{array} \right\} \rightarrow [(\neg ir) \rightarrow ge]$$

The second task is to check whether the conjunction of (1), (2) and (4) implies (3), i.e., whether:

$$\left. \begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(\neg ir) \rightarrow ge] \end{array} \right\} \rightarrow [(tx \wedge \neg ir) \rightarrow \neg un]$$

This is still not an immediate task, but at least it is formulated formally and we can apply one of many techniques to verify the above implications.

We will return to them in one of the next lectures.

– How many logicians does it take to replace a light-bulb?
– Doesn't matter: They can't do it, but they can prove that it can be done.

So, how do logicians prove things?

There are many methods and techniques.

One of these techniques depends on observing a correspondence between connectives and operations on sets.

For any sentence (formula) A , let $S(A)$ be the set of individuals for which A is true. Then:

- $S(\neg A) = \neg S(A)$
- $S(A \wedge B) = S(A) \cap S(B)$
- $S(A \vee B) = S(A) \cup S(B)$.

We also have:

- $A(Obj)$ holds iff $Obj \in S(A)$
- $A \rightarrow B$ holds iff $S(A) \subseteq S(B)$
- $A \leftrightarrow B$ holds iff $S(A) = S(B)$.

Examples

1. Consider sentence:

X is not a student.

The set of persons for which the sentence is true, is the complement of the set of students.

2. Consider sentence:

X is a driver and X is young.

The set of persons for which the sentence is true, is the intersection of all drivers and all young persons.

3. Consider sentence:

X is 23 or 24 years old.

The set of persons for which the sentence is true, is the union of all persons of age 23 and of age 24.

4. Consider sentence

Basketball players are tall.

Interpreting this sentence in logic gives rise to implication:

if x is a basketball player then x is tall,

which means that the set of basketball players is included in the set of tall persons.

5. Consider sentence

To be a human is equivalent to be a child or a woman or a man.

It means that the set of humans is equal to the union of sets of children, women and men.

Example of a proof via sets

Consider the following reasoning:

1. Students who like to learn read many books.
2. John does not read many books.
3. Therefore John does not like to learn.

Is the conclusion valid?

We isolate atomic sentences:

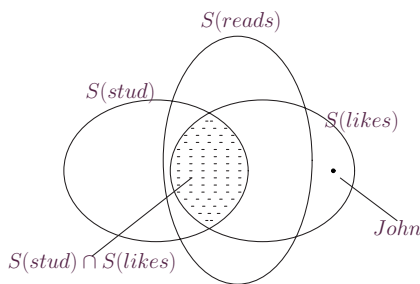
- X is a student, abbreviated by $stud(X)$
- X likes to learn, abbreviated by $likes(X)$
- X reads many books, abbreviated by $reads(X)$.

The above reasoning can now be translated as follows:

Sentence	Formula in logic/Set expression
1	$[stud(X) \wedge likes(X)] \rightarrow reads(X)$ $[S(stud) \cap S(likes)] \subseteq S(reads)$
2	$\neg reads(John)$ $\neg John \in S(reads)$ i.e., $John \notin S(reads)$
3	$\neg likes(John)$ $\neg John \in S(likes)$ i.e., $John \notin S(likes)$

We then check correctness of the following reasoning expressed in terms of set expressions:

1. $[S(stud) \cap S(likes)] \subseteq S(reads)$
2. $John \notin S(reads)$
3. therefore $John \notin S(likes)$.



Conclusion: Our reasoning has been **wrong!**

The conclusion would have been right if the second premise would have been:

2'. John is a student who does not read many books.

Why?

Let us now show the proof method based on truth tables (it is also called the *0-1 method*, since truth values FALSE and TRUE are often denoted by **0** and **1**, respectively).

We construct a truth table for a given formula as follows:

- columns:
 - first columns correspond to all “atomic sentences”, i.e., sentences not involving connectives
 - next columns correspond to sub-formulas involving one connective, if any
 - next columns correspond to sub-formulas involving two connectives, if any
 - etc.
 - the last column corresponds to the whole formula.
- rows correspond to all possible assignments of truth values to atomic sentences.

- A *tautology* is a formula which is TRUE for all possible assignments of truth values to atomic sentences.
- A formula is *satisfiable* if it is TRUE for at least one such assignment.
- A formula which is FALSE for all such assignments is called a *counter-tautology*.

Example

Consider sentence

It is not the case that John does not have a daughter.
Therefore John has a daughter.

The structure of this sentence is

$$[\neg(\neg A)] \rightarrow A,$$

where A stands for “John has a daughter”.

Truth table for $(\neg\neg A) \rightarrow A$:

A	$\neg A$	$\neg(\neg A)$	$[\neg(\neg A)] \rightarrow A$
FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	TRUE	TRUE

Thus our initial formula is a tautology.

The same applies to any formula of the shape

$$[\neg(\neg A)] \rightarrow A,$$

no matter what A is!

E.g.,

“It is not true that I do not say that John is a good candidate for this position”
implies
“I do say that John is a good candidate for this position”.

Exercise:

Convince yourselves that $[\neg(\neg A)] \leftrightarrow A$ is also a tautology.

Example: checking satisfiability

Recall that a formula is satisfiable if there is at least one row in its truth table, where it obtains value TRUE.

As an example we check whether the following formula is satisfiable:

$$A \rightarrow [(A \vee B) \wedge \neg B] \tag{1}$$

We construct the following truth table:

A	B	$\neg B$	$A \vee B$	$(A \vee B) \wedge \neg B$	(1)
FALSE	FALSE	TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	TRUE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE	FALSE	FALSE

There are rows, where formula (1) obtains value TRUE, thus it is satisfiable.

The only row where (1) is not true is the last one, i.e., when A and B are both true, thus (1) is, in fact, equivalent to $\neg(A \wedge B)$.

Example: verifying counter-tautologies

Recall that a formula is a counter-tautology if in all rows in its truth table it obtains value FALSE.

As an example we check whether the following formula is a counter-tautology:

$$\neg[A \rightarrow (B \rightarrow A)] \tag{2}$$

We construct the following truth table:

A	B	$B \rightarrow A$	$A \rightarrow (B \rightarrow A)$	(2)
FALSE	FALSE	TRUE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	FALSE
TRUE	FALSE	TRUE	TRUE	FALSE
TRUE	TRUE	TRUE	TRUE	FALSE

Formula (2) obtains value FALSE in all rows, thus it is a counter-tautology.

In fact, its negation, i.e., formula

$$A \rightarrow (B \rightarrow A)$$

is a tautology.

Example: DeMorgan's laws

DeMorgan's laws are:

$$\neg(A \wedge B) \leftrightarrow (\neg A \vee \neg B) \tag{3}$$

$$\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B) \tag{4}$$

The following truth table proves (3), where T, F abbreviate TRUE and FALSE, respectively:

A	B	$\neg A$	$\neg B$	$A \wedge B$	$\neg(A \wedge B)$	$\neg A \vee \neg B$	(3)
F	F	T	T	F	T	T	T
F	T	T	F	F	T	T	T
T	F	F	T	F	T	T	T
T	T	F	F	T	F	F	T

The next truth table proves (4):

A	B	$\neg A$	$\neg B$	$A \vee B$	$\neg(A \vee B)$	$\neg A \wedge \neg B$	(4)
F	F	T	T	F	T	T	T
F	T	T	F	T	F	F	T
T	F	F	T	T	F	F	T
T	T	F	F	T	F	F	T

Example: implication laws

1. Consider:

$$(A \rightarrow B) \leftrightarrow (\neg A \vee B) \tag{5}$$

Let us check whether it is a tautology.

The following table does the job.

A	B	$\neg A$	$A \rightarrow B$	$\neg A \vee B$	(5)
FALSE	FALSE	TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE	FALSE	TRUE
TRUE	TRUE	FALSE	TRUE	TRUE	TRUE

2. Consider:

$$\neg(A \rightarrow B) \leftrightarrow (A \wedge \neg B) \tag{6}$$

Let us check whether it is a tautology.

The following table does the job.

A	B	$\neg B$	$A \rightarrow B$	$\neg(A \rightarrow B)$	$A \wedge \neg B$	(6)
FALSE	FALSE	TRUE	TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE
TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE

Example: an application of the second implication law

Consider sentence:

“if $\overbrace{\text{John walks}}^A$ then $\underbrace{\text{he is close to his office}}_B$ then $\underbrace{\text{he has a break}}_C$.”

Assume the above sentence does not hold. We are interested whether the following sentences:

- “John walks.”
- “John is not close to his office.”
- “John does not have a break.”

are consequences of its negation.

Consider the negation of the initial sentence:

$$\neg[A \rightarrow (B \rightarrow C)]$$

According to the second implication law the above formula is equivalent to:

$$A \wedge \neg(B \rightarrow C),$$

i.e., to

$$A \wedge B \wedge \neg C.$$

Thus sentences 1 and 3 are consequences of the negation of the initial sentence, while 2 is not.

Example: three atomic sentences

Let us return to the question about equivalence of two translations of the sentence:

“Students and pupils read books”.

The two translations were:

- “Students read books and pupils read books”
- “If a person is a student or is a pupil then (s)he reads books”.

Translating those sentences into logic results in:

1'. $[s(x) \rightarrow r(x)] \wedge [p(x) \rightarrow r(x)]$

2'. $[s(x) \vee p(x)] \rightarrow r(x),$

where $s(x)$ stands for “ x is a student”, $p(x)$ stands for “ x is a pupil” and $r(x)$ stands for “ x reads books”.

We try to convince ourselves that the above formulas are equivalent no matter what x is. The following truth table does the job (s, p, r, F, T abbreviate $s(x), p(x), r(x), \text{FALSE}, \text{TRUE}$).

s	p	r	$s \rightarrow r$	$p \rightarrow r$	$s \vee p$	1'	2'	$1' \leftrightarrow 2'$
F	F	F	T	T	F	T	T	T
T	F	F	F	T	T	F	F	T
F	T	F	T	F	T	F	F	T
T	T	F	F	F	T	F	F	T
F	F	T	T	T	F	T	T	T
T	F	T	T	T	T	T	T	T
F	T	T	T	T	T	T	T	T
T	T	T	T	T	T	T	T	T

What you should have learnt from Lecture II?

- what are connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$?
- what is a truth table for a connective?
- what are truth tables for connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$?
- what is a truth table for a formula?
- what is a 0-1 method?
- what is a tautology, satisfiable formula and counter-tautology?
- how to use truth tables to check whether a formula is a tautology or is satisfiable or is a counter-tautology?

Introduction to proof systems

Observe that:

1. the semantic definition of tautologies does not provide us with any tools for systematically proving that a given formula is indeed a tautology of a given logic or is a consequence of a set of formulas
2. truth table method can be quite complex when many atomic sentences are involved; moreover, it works only in the simplest cases when we do not have more complex logical operators.

We shall then define proof systems as a tool for proving or disproving validity of formulas.

There are many ways to define proof systems. Here we only mention the most frequently used:

- *Hilbert-like* systems
- *Gentzen-like* proof systems (natural deduction)
- *resolution* (Robinson)
- *analytic tableaux*. (Beth, Smullyan)

What is a proof system?

We shall first illustrate the idea via *Hilbert-like* proof systems.

Hilbert-like proof systems consist of:

- a set of *axioms* (i.e. “obvious” formulas accepted without proof)
- a set of *proof rules* (called also *derivation* or *inference* rules), where any rule allows one to prove new formulas on the basis of formulas proved already.

Proof rules are usually formulated according to the following scheme:

if all formulas from a set of formulas F are proved then formula A is proved, too.

Such a rule is denoted by: $F \vdash A$, often by $\frac{F}{A}$

or, in Fitch notation which we shall frequently use, by

$$\left| \begin{array}{l} F \\ \hline A \end{array} \right.$$

Formulas from set F are called *premises (assumptions)* and formula A is called the *conclusion* of the rule.

What is a formal proof?

The set of *provable formulas* is defined as the (smallest) set of formulas satisfying the following conditions:

- every axiom is provable
- if the premises of a rule are provable then its conclusion is provable, too.

One can then think about proof systems as (nondeterministic) procedures, since the process of proving theorems can be formulated as follows, where formula A is the one to be proved valid:

- if A is an axiom, or is already proved, then the proof is finished,
- otherwise:
 - select (nondeterministically) a set of axioms or previously proved theorems
 - select (nondeterministically) an applicable proof rule
 - apply the selected rule and accept its conclusion as a new theorem
 - repeat the described procedure from the beginning.

Fitch format (notation)

We will mainly use so-called *Fitch format (notation)*, introduced by Frederic Fitch.

In Fitch format we use the following notation:

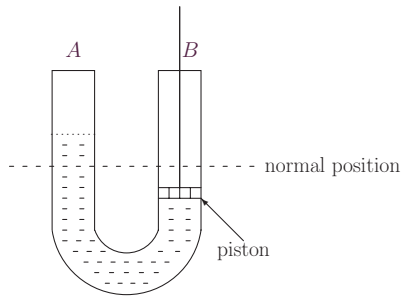
$$\left| \begin{array}{ll} P_1 & \text{Premise 1} \\ \vdots & \\ P_k & \text{Premise } k \\ \hline J_1 & \text{Justification 1} \\ \vdots & \\ J_n & \text{Justification } n \\ \hline C & \text{Conclusion} \end{array} \right.$$

where:

- the vertical bar indicates steps of a single proof
- the horizontal bar provides a distinction between *premises* P_1, \dots, P_k (also called *assumptions*) and claims that follow from premises
- *justifications* J_1, \dots, J_n serve as intermediate facts, applied to derive the *conclusion* C .

An introductory example

Consider the tube filled with water, as shown in figure below.



We have the following physical laws describing the world of tube:

1. if the piston remains in the normal position,
then the water level in *A* part of the tube is in the same position
2. if the piston is pushed down,
then the water level in *A* part of the tube is moving up
3. if the if the piston is pushed up,
then the water level in *A* part of the tube is moving down
4. if water level is up then it is not down.

In order to formalize these rules we use Fitch notation:

$$\frac{\text{piston in normal position}}{\text{water in normal position}} \quad (7)$$

$$\frac{\text{piston down}}{\text{water up}} \quad (8)$$

$$\frac{\text{piston up}}{\text{water down}} \quad (9)$$

$$\frac{\text{water up}}{\neg \text{water down}} \quad (10)$$

Example of reasoning, assuming that piston is pushed down:

$$\frac{\begin{array}{l} \text{piston down} \quad \text{assumption} \\ \text{water up} \quad (8) - \text{justification} \\ \neg \text{water down} \quad (10) - \text{conclusion} \end{array}}{\quad} \quad (11)$$

We then conclude that whenever the piston is down, water level is not down.

Examples of known rules

- *Modus ponens* formalizes reasoning of the form
 - John is a student.
 - If John is a student then John learns a lot.
 - Therefore, John learns a lot.

In Fitch notation:

$$\frac{\begin{array}{l} A \\ A \rightarrow B \end{array}}{B}$$

- *Modus tollens* formalizes reasoning of the form
 - if there is fire here, then there is oxygen here.
 - There is no oxygen here.
 - Therefore, there is no fire here.

In Fitch notation:

$$\frac{\begin{array}{l} A \rightarrow B \\ \neg B \end{array}}{\neg A}$$

Examples of invalid rules

- *Affirming the consequent*:

$$\frac{\begin{array}{l} A \rightarrow B \\ B \end{array}}{A}$$

Example:

- if there is fire here, then there is oxygen here.
- There is oxygen here.
- Therefore, there is fire here.

- *Denying the antecedent*:

$$\frac{\begin{array}{l} A \rightarrow B \\ \neg A \end{array}}{\neg B}$$

Example:

- if there is fire here, then there is oxygen here.
- There is no fire here.
- Therefore, there is no oxygen here.

Subproofs

Structuring proofs often requires to isolate their particular parts as *subproofs*:

P_1	premise 1
⋮	
P_k	premise k
┌	
S_1	subproof 1
C_1	conclusion of a sub-proof 1
└	
⋮	
┌	
S_n	subproof n
C_n	conclusion of a sub-proof n
└	
C	conclusion

- Subproofs look like proofs and can be arbitrarily nested, i.e., any subproof can have its subproofs, etc.
- The meaning of a subproof is that its conclusion is a justification in the proof.

Applications of proof rules often requires to carry out certain subproofs.

- Actually modus ponens is frequently used in the following form (in the case of the classical logic we use here these formulations lead to the same valid conclusions).

A
┌
A
B
└
B

(let A holds; if assuming A you can prove B then conclude B).

- *Modus tollens* can be also formalized as follows:

$\neg B$
┌
A
B
└
$\neg A$

(let $\neg B$ holds; if assuming A you can prove B then conclude $\neg A$).

Proofs by cases

Proofs by cases are based on the following principle:

- if $A \vee B$ holds,
 A implies C and
 B implies C
- then conclude that $A \vee B$ implies C .

Fitch-like rule formalizing this reasoning:

$A_1 \vee A_2 \dots \vee A_n$	assumption
┌	
A_1	
C	
└	
┌	
A_2	
C	
└	
⋮	
┌	
A_n	
C	
└	
C	conclusion

The proof by cases strategy is then the following:

- Suppose a disjunction holds, then we know that at least one of the disjuncts is true.
- We do not know which one. So we consider the individual "cases" (represented by disjuncts), separately.
- We assume the first disjunct and derive our conclusion from it.
- We repeat this procedure for each disjunct.
- So no matter which disjunct is true, we get the same conclusion. Thus we may conclude that the conclusion follows from the entire disjunction.

Example: proof by cases

$Mary\ will\ go\ to\ a\ cinema \vee Mary\ will\ go\ to\ a\ theater$
┌
Mary will go to a cinema
Mary will have a good time
└
┌
Mary will go to a theater
Mary will have a good time
└
Mary will have a good time

Example: proof by cases

Let us prove that the square of a real number is always non-negative (i.e., ≥ 0). Formally, $x^2 \geq 0$.

Proof:

there are three possible cases for x :

- positive ($x > 0$)
- zero ($x = 0$)
- negative ($x < 0$).

$x > 0 \vee x = 0 \vee x < 0$	
$x > 0$	– the first case
$x^2 = x * x > 0$	
$x^2 \geq 0$	
$x = 0$	– the second case
$x^2 = x * x = 0$	
$x^2 \geq 0$	
$x < 0$	– the third case
$x^2 = x * x > 0$	
$x^2 \geq 0$	
$x^2 \geq 0$	conclusion.

Reduction to absurdity (proof by contradiction)

Reduction to absurdity (proof by contradiction)

To prove that formula A holds:

1. state the opposite of what you are trying to prove ($\neg A$)
2. for the sake of argument assume $\neg A$ is true
3. prove contradiction (e.g., FALSE, or $B \wedge \neg B$, for some formula B)
4. if you can prove a false conclusion from the assumption $\neg A$ and other true statements, you know that the assumption $\neg A$ must be false, thus $\neg(\neg A)$ must be true. But $\neg(\neg A)$ is just A .

Then one can conclude A . Here we applied tautology (one could also use modus tollens – how?)

$$[(\neg A) \rightarrow \text{FALSE}] \rightarrow A.$$

In Fitch notation:

$\neg A$	
FALSE	
A	

Example: proof by contradiction

Consider reasoning:

A committee meeting takes place (M) if all members of the committee are informed in advance (A) and there are at least 50% committee members present (P).

Members are informed in advance if post works normally (N). Therefore, if the meeting was canceled, there were fewer than 50% members present or post did not work normally.

We want to prove

$$(\neg M) \rightarrow (\neg P \vee \neg N),$$

provided that

- $(A \wedge P) \rightarrow M$
- $N \rightarrow A$.

We negate $(\neg M) \rightarrow (\neg P \vee \neg N)$.

By implication laws, the negation

$$\neg[(\neg M) \rightarrow (\neg P \vee \neg N)]$$

is equivalent to

$$(\neg M) \wedge \neg(\neg P \vee \neg N)$$

which, by DeMorgan's law, is equivalent to

$$(\neg M) \wedge P \wedge N.$$

Proof of our claim in Fitch notation:

1	$(\neg M) \wedge P \wedge N$	
2	$(A \wedge P) \rightarrow M$	
3	$N \rightarrow A$	
4	$\neg M$	– from 1
5	P	– from 1
6	N	– from 1
7	$(\neg M) \rightarrow (\neg A \vee \neg P)$	– from 2 – why?
8	$\neg A \vee \neg P$	– from 4, 7 and modus ponens
9	A	– from 6, 3 and modus ponens
10	$A \wedge P$	– from 5 and 9
11	FALSE	– by DeMorgan law, we have that 8 and 10 contradict each other.

Therefore $(\neg M) \wedge P \wedge N$ is FALSE, which proves our initial implication.

Some derivations are still informal since we have no rules for them (e.g., conclusions 4, 5, 6). All necessary rules will be introduced later (in lecture IV).

Rules for identity (equality)

Identity introduction, denoted by (=intro):

$$\left| \begin{array}{l} x = x \end{array} \right.$$

The above rule allows one to add, in any proof, line $x = x$.

Identity elimination, denoted by (=elim):

$$\left| \begin{array}{l} A(x) \\ x = y \\ \hline A(x = y) \end{array} \right.$$

where $A(x = y)$ denotes formula obtained from A by replacing some (or all) occurrences of x with y .

Example

Expression $(x + z = x + y)(x = y)$ stands for any of formulas:

- ▷ $y + z = x + y$,
- ▷ $x + z = y + y$,
- ▷ $y + z = y + y$.

Thus, e.g., we have:

$$\left| \begin{array}{l} x + z = x + y \\ x = y \\ \hline x + z = y + y \end{array} \right.$$

Examples of proofs

1. identity elimination:

$$\begin{array}{l|l} 1 & x < x + 1 \\ 2 & x = \sqrt{2} \\ \hline 3 & \sqrt{2} < \sqrt{2} + 1 \quad (=elim): 1, 2 \end{array}$$

2. identity elimination:

$$\begin{array}{l|l} 1 & likes(John, Mary) \\ 2 & Mary = Anne \\ \hline 3 & likes(John, Anne) \quad (=elim): 1, 2 \end{array}$$

3. *symmetry of identity*:

$$\begin{array}{l|l} 1 & x = y \\ 2 & x = x \quad (=intro) \\ \hline 3 & y = x \quad (=elim): 1, 2, \text{ where } A(x) \leftrightarrow x = x \end{array}$$

4. *transitivity of identity*:

$$\begin{array}{l|l} 1 & x = y \\ 2 & y = z \\ \hline 3 & y = x \quad \text{symmetry of identity applied to 1} \\ 4 & x = z \quad (=elim): 2, 3 \end{array}$$

Symmetry and transitivity of identity, just proved, lead to the following rules, frequently used in reasoning.

Symmetry of identity

The *symmetry of identity* means that: whenever $x = y$ then also $y = x$.

The following rule (further denoted by (=symm)) reflects this principle:

$$\left| \begin{array}{l} x = y \\ \hline y = x \end{array} \right.$$

Transitivity of identity

The *transitivity of identity* means that: whenever $x = y$ and $y = z$ then also $x = z$.

The following rule (further denoted by (=trans)) reflects this principle:

$$\left| \begin{array}{l} x = y \\ y = z \\ \hline x = z \end{array} \right.$$

What you should have learnt from Lecture III?

- what is a proof system?
- what is a formal proof?
- what is the Fitch format (notation)?
- what is a subproof?
- what is Fitch representation of subproofs?
- what are proofs by cases?
- what is reduction to absurdity (proof by contradiction)?
- what are rules for equality?

FALSE elimination

FALSE *elimination rule*, denoted by (\perp elim):

$$\left| \begin{array}{l} \text{FALSE} \\ \vdots \\ P \end{array} \right.$$

(\perp elim) reflects the principle that FALSE implies any formula P (more precisely, $\text{FALSE} \rightarrow P$).

FALSE introduction

FALSE *introduction rule*, denoted by (\perp intro):

$$\left| \begin{array}{l} P \\ \vdots \\ \neg P \\ \vdots \\ \text{FALSE} \end{array} \right.$$

(\perp intro) reflects the principle that P and $\neg P$ imply FALSE, i.e., are inconsistent (more precisely, $(P \wedge \neg P) \rightarrow \text{FALSE}$).

Negation elimination

Negation *elimination rule*, denoted by (\neg elim):

$$\left| \begin{array}{l} \neg\neg P \\ \vdots \\ P \end{array} \right.$$

(\neg elim) reflects the principle that double negation is equivalent to no negation (more precisely, $(\neg\neg P) \rightarrow P$).

Negation introduction

Negation *introduction rule*, denoted by (\neg intro):

$$\left| \begin{array}{l} \left| \begin{array}{l} P \\ \vdots \\ \text{FALSE} \end{array} \right. \\ \vdots \\ \neg P \end{array} \right.$$

(\neg intro) reflects the principle that formula which implies FALSE, is FALSE itself (more precisely, $(P \rightarrow \text{FALSE}) \rightarrow \neg P$).

Examples

1. From P one can derive $\neg\neg P$:

$$\begin{array}{l|l} 1 & P \\ 2 & \neg P \\ 3 & \text{FALSE} \quad (\perp\text{intro}): 1, 2 \\ 4 & \neg\neg P \quad (\neg\text{intro}): 2, 3 \end{array}$$

2. If from P one can derive $\neg P$ then, in fact, $\neg P$ holds. We apply the proof by contradiction, i.e., we assume that the negation of $\neg P$ holds:

$$\begin{array}{l|l} 1 & \neg\neg P \quad \text{assumption} \\ 2 & P \quad (\neg\text{elim}): 1 \\ 3 & \neg P \quad 2, \text{ assumption that } P \text{ implies } \neg P \\ 4 & \text{FALSE} \quad (\perp\text{intro}): 2, 3 \end{array}$$

Thus the assumption that the negation of $\neg P$ holds leads to contradiction. In consequence we have $\neg P$.

Justifying the rules

We justify rules (of the classical logic) by showing that assumptions imply conclusion. More formally, if A_1, \dots, A_k are assumptions and C is a conclusion then we have to prove that implication $(A_1 \wedge \dots \wedge A_k) \rightarrow C$, is a tautology.

Examples

1. Consider (\perp intro).

We have to prove that $(P \wedge \neg P) \rightarrow \text{FALSE}$:

P	$\neg P$	$P \wedge \neg P$	$(P \wedge \neg P) \rightarrow \text{FALSE}$
FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	FALSE	TRUE

2. Consider (\neg intro).

We have to prove that $(P \rightarrow \text{FALSE}) \rightarrow \neg P$:

P	$\neg P$	$P \rightarrow \text{FALSE}$	$(P \rightarrow \text{FALSE}) \rightarrow \neg P$
FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE

Conjunction elimination

Conjunction elimination rule, denoted by $(\wedge\text{elim})$:

$$\left| \begin{array}{l} P_1 \wedge \dots \wedge P_i \wedge \dots \wedge P_n \\ \vdots \\ P_i \end{array} \right. \quad \text{where } 1 \leq i \leq n$$

$(\wedge\text{elim})$ reflects the principle that conjunction implies each of its conjuncts
 (more precisely, $(P_1 \wedge \dots \wedge P_i \wedge \dots \wedge P_n) \rightarrow P_i$ for any $1 \leq i \leq n$).

Example

We show that $A \wedge B$ and $\neg A$ are contradictory:

$$\begin{array}{l|l} 1 & A \wedge B \\ 2 & \neg A \\ \hline 3 & A \quad (\wedge\text{elim}): 1 \\ 4 & \text{FALSE} \quad (\perp\text{intro}): 2, 3 \end{array}$$

Conjunction introduction

Conjunction introduction rule, denoted by $(\wedge\text{intro})$:

$$\left| \begin{array}{l} P_1 \\ P_2 \\ \vdots \\ P_n \end{array} \right. \quad \text{for } 1 \leq i \leq n, \text{ all } P_i\text{'s appear}$$

$$P_1 \wedge P_2 \wedge \dots \wedge P_n$$

$(\wedge\text{intro})$ reflects the principle that in order to prove conjunction one has to prove all its conjuncts first.

Example

$$\begin{array}{l|l} 1 & A \wedge B \\ \hline 2 & B \quad (\wedge\text{elim}): 1 \\ 3 & A \quad (\wedge\text{elim}): 1 \\ 4 & B \wedge A \quad (\wedge\text{intro}): 2, 3 \end{array}$$

Disjunction elimination

Disjunction elimination rule, denoted by $(\vee\text{elim})$:

$$\left| \begin{array}{l} P_1 \vee \dots \vee P_n \\ \left| \begin{array}{l} P_1 \\ \hline S \end{array} \right. \\ \vdots \\ \left| \begin{array}{l} P_n \\ \hline S \end{array} \right. \\ \vdots \\ S \end{array} \right. \quad \text{for } 1 \leq i \leq n, \text{ all } P_i\text{'s appear}$$

$(\vee\text{elim})$ reflects the principle of proof by cases.

Disjunction introduction

Disjunction introduction rule, denoted by $(\vee\text{intro})$:

$$\left| \begin{array}{l} P_i \quad \text{for some } 1 \leq i \leq n \\ \vdots \\ P_1 \vee \dots \vee P_i \vee \dots \vee P_n \end{array} \right.$$

$(\vee\text{intro})$ reflects the principle that any disjunct implies the whole disjunction
 (more precisely, $P_i \rightarrow (P_1 \vee \dots \vee P_i \vee \dots \vee P_n)$ for any $1 \leq i \leq n$).

Example

$$\begin{array}{l|l} 1 & (A \wedge B) \vee C \\ 2 & A \wedge B \\ \hline 3 & A \quad (\wedge\text{elim}): 2 \\ 4 & A \vee C \quad (\vee\text{intro}): 3 \\ \hline 5 & C \\ \hline 6 & A \vee C \quad (\vee\text{intro}): 5 \\ 7 & A \vee C \quad (\vee\text{elim}): 1, 2-4, 5-6 \end{array}$$

Example: proof of a DeMorgan's law

1		$\neg(A \wedge B)$					
2			$\neg(\neg A \vee \neg B)$ (proof by contradiction)				
3				$\neg A$			
4					$\neg A \vee \neg B$ (\vee intro): 3		
5					FALSE (\perp intro): 2, 4		
6					$\neg\neg A$ (\neg intro): 3-5		
7					A (\neg elim): 6		
8						$\neg B$	
9							$\neg A \vee \neg B$ (\vee intro): 8
10							FALSE (\perp intro): 2, 9
11							$\neg\neg B$ (\neg intro): 8-10
12							B (\neg elim): 11
13							$A \wedge B$ (\wedge intro): 7, 12
14							FALSE (\perp intro): 1, 13
15							$\neg\neg(\neg A \vee \neg B)$ (\neg intro): 2-14
16							$\neg A \vee \neg B$ (\neg elim): 15

Implication revisited

Given an implication $A \rightarrow B$, we say that A is a *sufficient condition* for B and B is a *necessary condition* for A .

Example

Consider sentence:

If it rains, the streets are wet.

It is formulated as

$\text{rains} \rightarrow \text{streets are wet.}$

Thus "to rain" is a sufficient condition for streets to be wet.

On the other hand, "wet streets" is a necessary condition for raining, since if streets are not wet, then it is not raining.

Thus streets must be wet when it rains.

In other words, it is necessary that streets are wet when it rains.

Observe that the following law is reflected by the definition of necessary conditions:

$$(A \rightarrow B) \leftrightarrow (\neg B \rightarrow \neg A).$$

Implication elimination

Implication elimination rule, denoted by (\rightarrow elim):

	$P \rightarrow Q$
	\vdots
	P
	\vdots
	Q

(\rightarrow elim) reflects the modus ponens rule (P is a sufficient condition for Q).

Example

1		$\text{piston down} \rightarrow \text{water level up}$	
2		piston down	
3		water level up	(\rightarrow elim): 1-2

Implication introduction

Implication introduction rule, denoted by (\rightarrow intro):

	P
	\vdots
	Q
	$P \rightarrow Q$

(\rightarrow intro) reflects the principle that whenever assuming P we can prove Q , we can also prove $P \rightarrow Q$.

Example

We prove that $P \rightarrow (Q \vee P)$.

1			P	
2			$Q \vee P$	(\vee intro): 1
3		$P \rightarrow (Q \vee P)$		(\rightarrow intro): 1-2

Example: transitivity of implication

We want to prove the following rule, denoted by (\rightarrow tr):

$$\begin{array}{|l} P \rightarrow Q \\ \vdots \\ Q \rightarrow R \\ \hline P \rightarrow R \end{array}$$

Proof:

$$\begin{array}{|l} 1 \quad P \rightarrow Q \\ 2 \quad Q \rightarrow R \\ 3 \quad \begin{array}{|l} P \\ \hline Q \end{array} \quad (\rightarrow\text{elim}): 1, 3 \\ 4 \quad R \quad (\rightarrow\text{elim}): 2, 4 \\ 5 \quad P \rightarrow R \quad (\rightarrow\text{intro}): 3-5 \end{array}$$

Equivalence elimination

Equivalence elimination rule, denoted by (\leftrightarrow elim):

$$\begin{array}{|l} P \leftrightarrow Q \quad (\text{or } Q \leftrightarrow P) \\ \vdots \\ P \\ \vdots \\ Q \end{array}$$

(\leftrightarrow elim) reflects the principle that formulas equivalent to previously proved, are proved, too.

Example

$$\begin{array}{|l} 1 \quad human(x) \leftrightarrow [rational(x) \wedge animal(x)] \\ 2 \quad rational(x) \rightarrow thinks(x) \\ 3 \quad \begin{array}{|l} rational(x) \wedge animal(x) \\ \hline rational(x) \end{array} \\ 4 \quad rational(x) \quad (\wedge\text{elim}): 3 \\ 5 \quad [rational(x) \wedge animal(x)] \rightarrow rational(x) \quad (\rightarrow\text{intro}): 3,4 \\ 6 \quad human(x) \rightarrow rational(x) \quad (\leftrightarrow\text{elim}): 1,5 \\ 7 \quad human(x) \rightarrow thinks(x) \quad (\rightarrow\text{tr}): 6,2 \end{array}$$

Equivalence introduction

Equivalence introduction rule, denoted by (\leftrightarrow intro):

$$\begin{array}{|l} \begin{array}{|l} P \\ \hline \vdots \\ Q \end{array} \\ \vdots \\ \begin{array}{|l} Q \\ \hline \vdots \\ P \end{array} \\ \hline P \leftrightarrow Q \end{array}$$

(\leftrightarrow intro) reflects the principle that in order to prove equivalence one has to prove implication from P to Q as well as from Q to P (more precisely, $(P \leftrightarrow Q) \leftrightarrow [(P \rightarrow Q) \wedge (Q \rightarrow P)]$).

What you should have learnt from Lecture IV?

- what are rules for elimination and introduction of FALSE?
- what are rules for propositional connectives:
 - negation
 - conjunction
 - disjunction
 - implication
 - equivalence?
- how to apply those rules?
- how to justify those rules?

Quantifiers

In the classical logic we have two *quantifiers*:

- the *existential quantifier*
“there exists an individual x such that ...”,
denoted by $\exists x$
- the *universal quantifier*
“for all individuals x ...”,
denoted by $\forall x$.

Examples

1. “Everybody loves somebody”:

$$\forall x \exists y \text{ loves}(x, y)$$

2. “Relation R is reflexive”:

$$\forall x R(x, x)$$

3. “Relation R is symmetric”:

$$\forall x \forall y R(x, y) \rightarrow R(y, x)$$

4. “Relation R is transitive”:

$$\forall x \forall y \forall z ((R(x, y) \wedge R(y, z)) \rightarrow R(x, z))$$

It is important that variables under quantifiers represent individuals. In fact, we have the following hierarchy of logics:

0. *zero-order logic* (propositional), where we do not allow neither variables representing domain elements nor quantifiers
1. *first-order logic*, where we allow variables representing domain elements and quantifiers over domain elements (in this case quantifiers are called *first-order quantifiers*)
2. *second-order logic*, where we additionally allow variables representing sets of domain elements and quantifiers over sets of elements (so-called *second-order quantifiers*),
3. *third-order logic*, where we additionally allow variables representing sets of sets of domain elements and quantifiers over sets of sets of elements (so-called *third-order quantifiers*)
4. etc.

In the above, “sets” can be replaced by “functions” or “relations”, since any set can be represented by a function or a relation and vice versa, any function and relation can be represented as a set of tuples.

Examples

1. “To like is to be pleased with”:

$$\text{likes} \rightarrow \text{pleased}$$

– a formula of zero-order logic

2. “John likes everybody”:

$$\forall x \text{ likes}(\text{John}, x)$$

– a formula of first-order logic

3. “John has no relationships with anybody”:

$$\neg \exists R \exists x R(\text{John}, x)$$

– a formula of second-order logic

4. “No matter how the word *good* is understood, John has no good relationships with anybody”:

$$\forall \text{Good} \neg \exists R \exists x \text{ Good}(R) \wedge R(\text{John}, x)$$

– a formula of third-order logic.

The scope of quantifiers

The *scope of a quantifier* is the portion of a formula where it binds its variables.

Examples

- 1.

$$\forall x \left[R(x) \vee \overbrace{\exists y (R(x) \rightarrow S(x, y))}^{\text{the scope of } \exists y} \right]$$

the scope of $\forall x$

- 2.

$$\forall x \left[R(x) \vee \overbrace{\exists y \exists x (R(x) \rightarrow S(x, y))}^{\text{the scope of } \exists y} \right]$$

the scope of $\forall x$

in which quantifier’s scope occurs variable x in $S(x, y)$
– $\forall x$ or $\exists x$?

Free variables are variables which are not in the scope of a quantifier. If a variable is in the scope of a quantifier then it is *bound* by the quantifier.

Restricted quantifiers

The four *Aristotelian forms* of quantification are:

1. “all P 's are Q 's”
i.e., all individuals satisfying P satisfy Q – this form is also called the *restricted universal quantification*
2. “some P 's are Q 's”
i.e., some individuals satisfying P satisfy Q – this form is also called the *restricted existential quantification*
3. “no P 's are Q 's”
i.e., no individuals satisfying P satisfy Q
4. “some P 's are not Q 's”
i.e., some individuals satisfying P do not satisfy Q .

Examples: Aristotelian forms

1. “All students have to learn.”
2. “Some students have jobs.”
3. “No students are illiterate.”
4. “Some students do not like math.”

Translation of Aristotelian forms into quantified formulas

1. “all P 's are Q 's” translates into $\forall x [P(x) \rightarrow Q(x)]$
2. “some P 's are Q 's” translates into $\exists x [P(x) \wedge Q(x)]$
3. “no P 's are Q 's” translates into $\forall x [P(x) \rightarrow \neg Q(x)]$
4. “some P 's are not Q 's” translates into $\exists x [P(x) \wedge \neg Q(x)]$.

Examples

1. “All students have to learn” translates into:

$$\forall x [\text{student}(x) \rightarrow \text{hasToLearn}(x)]$$

2. “Some students work” translates into:

$$\exists x [\text{student}(x) \wedge \text{hasJob}(x)]$$

3. “No students are illiterate” translates into:

$$\forall x [\text{student}(x) \rightarrow \neg \text{illiterate}(x)]$$

4. “Some students do not like math” translates into:

$$\exists x [\text{student}(x) \wedge \neg \text{likes}(x, \text{math})]$$

Translating natural language sentences into quantified formulas

When we translate natural language sentences into quantified formulas, we integrate information from two sources:

1. vocabulary (words in the sentence); here we get:
 - constants (names of individuals), e.g., “Eve”, “John”
 - variables (representing indefinite individuals), e.g., corresponding to “a”, “one”, or appearing within quantifiers “all”, “some”, etc.
 - concepts (sets of individuals), e.g., “man”, “animal”, “furniture”; these are represented by one argument relation symbols (e.g., $\text{man}(x)$ means that x is a man)
 - relations (other than those formalizing concepts) showing the relationships between constant and concepts.
2. syntactic structure of the sentence; here we get:
 - knowledge about logical operators appearing in the sentence, like connectives and quantifiers
 - knowledge how to combine constants, variables, concepts and other relations.

Remark:

Not all natural language sentences can be properly translated into the quantified formulas of the classical logic we deal with!

Examples

1. “Eve is a student”

- constants: *Eve*, variables: none
- concepts: *student*
- other relations: none.

Translation: $\text{student}(\text{Eve})$.

2. “Eve is a better student than John”

- constants: *Eve*, *John*, variables: none
- concepts: none
- other relations: two-argument relation *betterStudent*.

Translation: $\text{betterStudent}(\text{Eve}, \text{John})$.

3. “Eve is the best student”

- constants: *Eve*, variable: x (needed in quantification)
- concepts: *student*
- other relations: two-argument relation *betterStudent*.

Translation:

$$\text{student}(\text{Eve}) \wedge \neg \exists x [\text{student}(x) \wedge \text{betterStudent}(x, \text{Eve})]$$

4. "Eve likes John"
 - constants: *Eve, John*, variables: none
 - concepts: none
 - other relations: two-argument relation *likes*.
 Translation $likes(Eve, John)$.
5. "Eve and John like each other"
 - constants, variables, concepts and other relations are as before.
 Translation $likes(Eve, John) \wedge likes(John, Eve)$.
6. "Eve likes somebody"
 - constants: *Eve*, variables: x (corresponding to "somebody")
 - concepts and other relations as before.
 Translation: $\exists x likes(Eve, x)$.
7. "Somebody likes Eve"
 - constants, variables, concepts and other relations as before.
 Translation: $\exists x likes(x, Eve)$.
8. "Eve likes everybody"
 - constants, variables, concepts and other relations as before.
 Translation: $\forall x likes(Eve, x)$.
9. "Everybody likes Eve"
 - constants, variables, concepts and other relations as before.
 Translation: $\forall x likes(x, Eve)$.

10. "If one is a good last year student then (s)he has only good grades"
 - constants: none, variable: x (corresponding to "one")
 - concepts: *goodStudent, lastYear*
 - other relations: one-argument relations *hasGoodGrades* and *hasBadGrades*.
 Translation:

$$[goodStudent(x) \wedge lastYear(x)] \rightarrow [hasGoodGrades(x) \wedge \neg hasBadGrades(x)].$$
 By convention sentences are implicitly universally quantified over variables which are not in scope of quantifiers. The above sentence is then equivalent to

$$\forall x \{ [goodStudent(x) \wedge lastYear(x)] \rightarrow [hasGoodGrades(x) \wedge \neg hasBadGrades(x)] \}.$$
 Would this be equivalent to

$$\forall x \{ goodStudent(x) \rightarrow [\neg hasBadGrades(x)] \}?$$
11. Another translation of the above sentence can be based on the following choice:
 - constants: none, variables: x, y
 - concepts: *goodStudent, goodGrade, lastYear*
 - other relations: two-argument relation *hasGrade*.
 Translation:

$$[goodStudent(x) \wedge lastYear(x)] \rightarrow [hasGrade(x, y) \rightarrow goodGrade(y)].$$

To understand the meaning of a sentence (and then translate it into logic), one often rephrases the sentence and makes it closer to logical form.

Examples

	Original sentence	Rephrased sentence
1	"If you don't love yourself, you can't love anybody else"	"If you don't love you, there does not exist a person p , such that you love p "
2	"Every day is better than the next"	"For all days d , d is better than $next(d)$ "
3	"Jack is the best student"	"Jack is a student and for all students s different from Jack, Jack is better than s "

Translations of the above sentences:

1. $\neg love(you, you) \rightarrow \neg \exists p [love(you, p)]$
2. $\forall d [better(d, next(d))]$
3. $student(Jack) \wedge \forall s \{ [student(s) \wedge s \neq Jack] \rightarrow better(Jack, s) \}$

Extensional versus intensional relations

By an *extensional relation* we mean any relation which allows us to replace its arguments by equal elements. *Intensional* relations are those that are not extensional. More precisely, extensional relations are those for which rule (=elim) can be applied.

To test whether a relation, say $R(x_1, x_2, \dots, x_k)$, is extensional or intensional consider its arguments x_1, x_2, \dots, x_k , on by one. The test for the i -th argument, x_i , where $1 \leq i \leq k$, runs as follows:

– if $x_i = e$ implies that

$$R(x_1, \dots, x_i, \dots, x_k) \text{ and } R(x_1, \dots, e, \dots, x_k)$$

are equivalent, then the test is passed.

If for all arguments the test is passed then the relation is extensional. Otherwise (if there is at least one argument where the test is not passed) the relation is intensional.

For the purpose of the test one can use Fitch notation.

Examples of extensional relations

1. All relations in traditional mathematics are extensional, e.g., \leq , \geq , “to be parallel”, to be perpendicular, etc.
For example, *parallel* is extensional since the following reasoning is valid:

$parallel(x, y)$	– lines x and y are parallel
$x = l_1$	– lines x and l_1 are equal
$y = l_2$	– lines x and l_2 are equal
$parallel(x, l_2)$	– lines x and l_2 are parallel
$parallel(l_1, y)$	– lines l_1 and y are parallel

2. many relations in natural language are extensional, e.g., “likes”, “has”, “betterThan”, etc.. For example, *likes* is extensional since the following reasoning is valid:

$likes(x, y)$	
$x = Eve$	
$y = John$	
$likes(x, John)$	
$likes(Eve, y)$	

Examples of intensional relations

1. Consider sentence “Electra knows that Orestes is her brother. The sentence contains relation “P knows Q” with meaning “P knows that Q is her brother”. It appears to be an intensional relation, as the following paradox of Orestes and Electra shows (the paradox is attributed to Eubulides), where MM is a man in a mask, so that Electra is unable to recognize whether MM is or not is *Orestes*:

$knows(Electra, Orestes)$	Electra knows that Orestes is her brother
$Orestes = MM$	Orestes and the masked man are the same person
$knows(Electra, MM)$	

so we conclude that Electra knows the masked man, contrary to our assumption that she cannot recognize him. The above reasoning is invalid. In consequence “knows” is not extensional.

2. The following relations are intensional: “P believes in Q”, “it is necessary that P holds”, “P is obligatory”, “P is allowed” — why?
Which of the above relations are first-order and which are second-order?
3. Give another examples of intensional relations.

DeMorgan’s laws for quantifiers

DeMorgan laws for quantifiers are:

- $[\neg\neg\forall x A(x)] \leftrightarrow [\exists x \neg A(x)]$
- $[\neg\exists x A(x)] \leftrightarrow [\forall x \neg A(x)]$

Examples

1. “It is not the case that all animals are large” is equivalent to “Some animals are not large”.
2. “It is not the case that some animals are plants” is equivalent to “All animals are not plants”.
3. $\neg\forall x\exists y\forall z [friend(x, y)\wedge likes(y, z)]$ is equivalent to $\exists x\forall y\exists z \neg[friend(x, y)\wedge likes(y, z)]$ i.e., to $\exists x\forall y\exists z [\neg friend(x, y)\vee \neg likes(y, z)]$.
4. What is the negation of:
 - “Everybody loves somebody”?
 - “Everybody loves somebody sometimes”?

Analogies between quantifiers and propositional connectives

“For all individuals x relation $R(x)$ holds” is equivalent to
 R holds for the first individual
 and R holds for the second individual
 and R holds for the third individual
 and ...

Given a fixed set of individuals $U = \{u_1, u_2, u_3, \dots\}$ we then have that:

$$\forall x \in U [R(x)] \leftrightarrow [R(u_1)\wedge R(u_2)\wedge R(u_3)\wedge \dots]$$

Similarly, “Exists an individual x such that relation $R(x)$ holds” is equivalent to

R holds for the first individual
 or R holds for the second individual
 or R holds for the third individual
 or ...

Given a fixed set of individuals $U = \{u_1, u_2, u_3, \dots\}$ we then have that:

$$\exists x \in U [R(x)] \leftrightarrow [R(u_1)\vee R(u_2)\vee R(u_3)\vee \dots]$$

What are then the differences between quantifiers and connectives?

Other laws for quantifiers

1. Null quantification.

Assume variable x is not free in formula P . Then:

- (a) $\forall x[P]$ is equivalent to P
- (b) $\exists x[P]$ is equivalent to P
- (c) $\forall x[P \vee Q(x)]$ is equivalent to $P \vee \forall x[Q(x)]$
- (d) $\exists x[P \wedge Q(x)]$ is equivalent to $P \wedge \exists x[Q(x)]$.

2. Pushing quantifiers past connectives:

- (a) $\forall x[P(x) \wedge Q(x)]$ is equivalent to $\forall x[P(x)] \wedge \forall x[Q(x)]$
- (b) $\exists x[P(x) \vee Q(x)]$ is equivalent to $\exists x[P(x)] \vee \exists x[Q(x)]$

However observe that:

- 1. $\forall x[P(x) \vee Q(x)]$ is not equivalent to $\forall x[P(x)] \vee \forall x[Q(x)]$
- 2. $\exists x[P(x) \wedge Q(x)]$ is not equivalent to $\exists x[P(x)] \wedge \exists x[Q(x)]$

Examples

- 1. "all students in this room are tall or medium" is not equivalent to "all students in this room are tall or all students in this room are medium"
- 2. Find an example showing the second inequivalence.

What you should have learnt from Lecture V?

- what are quantifiers?
- what is the scope of a quantifier?
- what are free and bound variables?
- what are Aristotelian forms of quantification?
- what are restricted quantifiers?
- how to translate natural language into logic?
- what are extensional and intensional relations?
- what are DeMorgan's laws for quantifiers?
- what are some other laws for quantifiers?

Universal elimination

Universal elimination rule, denoted by $(\forall\text{elim})$:

$$\begin{array}{l} \forall x A(x) \\ \vdots \\ A(c) \end{array} \quad \text{where } c \text{ is any constant}$$

$(\forall\text{elim})$ reflects the principle that an universally valid formula is valid for any particular individual, too.

Examples

- 1. From the sentence "All humans think" we deduce that "John thinks":

1	$\forall x \text{thinks}(x)$	
2	$\text{thinks}(\text{John})$	$(\forall\text{elim}): 1$
- 2. From assumption that "Everybody likes somebody" we deduce that "Eve likes somebody":

1	$\forall x \exists y \text{likes}(x, y)$	
2	$\exists y \text{likes}(\text{Eve}, y)$	$(\forall\text{elim}): 1$

Universal introduction

Universal introduction rule, denoted by $(\forall\text{intro})$, where notation \boxed{c} indicates that constant c stands for an arbitrary "anonymous" object and, moreover, that c can appear only in the subproof where it has been introduced:

$$\begin{array}{l} \boxed{c} \quad (c \text{ stands for arbitrary object}) \\ \vdots \\ A(c) \\ \forall x A(x) \end{array}$$

$(\forall\text{intro})$ reflects the principle that whenever we prove a formula for an "anonymous" individual, the formula is universally valid.

Example

Consider an universe consisting of ants.

$$\begin{array}{l} \boxed{\text{any}} \quad \text{any is a constant} \\ \text{small}(\text{any}) \\ \forall x \text{small}(x) \end{array}$$

Example: distribution of universals over conjunction

1	$\forall x [A(x) \wedge B(x)]$	
2	\boxed{c}	
3	$A(c) \wedge B(c)$	(\forall elim): 1
4	$A(c)$	(\wedge elim): 3
5	$\forall x A(x)$	(\forall intro): 2-4
6	\boxed{d}	
7	$A(d) \wedge B(d)$	(\forall elim): 1
8	$B(d)$	(\wedge elim): 7
9	$\forall x B(x)$	(\forall intro): 6-8
10	$\forall x A(x) \wedge \forall x B(x)$	(\wedge intro): 5, 9
11	$\forall x A(x) \wedge \forall x B(x)$	
12	$\forall x A(x)$	(\wedge elim): 11
13	$\forall x B(x)$	(\wedge elim): 11
14	\boxed{e}	
15	$A(e)$	(\forall elim): 12
16	$B(e)$	(\forall elim): 13
17	$A(e) \wedge B(e)$	(\wedge intro): 15, 16
18	$\forall x [A(x) \wedge B(x)]$	(\forall intro): 14-17
19	$\forall x [A(x) \wedge B(x)] \leftrightarrow [\forall x A(x) \wedge \forall x B(x)]$	(\leftrightarrow intro): 1-10, 11-18

Generalized universal introduction

Generalized universal introduction rule, also denoted by (\forall intro):

$\boxed{c} A(c)$	($A(c)$ holds for any object c)
\vdots	
$B(c)$	
$\forall x [A(x) \rightarrow B(x)]$	

Generalized (\forall intro) reflects the principle of general conditional proof:

- assume $A(c)$ holds for c denoting arbitrary object and prove that $B(c)$ holds
- then you can conclude that $A(c)$ implies $B(c)$, i.e., $A(c) \rightarrow B(c)$ holds for arbitrary individual c .

In consequence,

$$\forall x [A(x) \rightarrow B(x)].$$

Examples

1. $\forall x [A(x) \rightarrow (A(x) \vee B(x))]$:

1	$\boxed{d} A(d)$	
2	$A(d) \vee B(d)$	(\vee intro): 1
3	$\forall x [A(x) \rightarrow (A(x) \vee B(x))]$	generalized (\forall intro): 1-2

2. Transitivity of implication:

1	$\forall x [A(x) \rightarrow B(x)]$	
2	$\forall y [B(y) \rightarrow C(y)]$	
3	$\boxed{d} A(d)$	
4	$A(d) \rightarrow B(d)$	(\forall elim): 1
5	$B(d)$	(\rightarrow elim): 3, 4
6	$B(d) \rightarrow C(d)$	(\forall elim): 2
7	$C(d)$	(\rightarrow elim): 5, 6
8	$\forall x [A(x) \rightarrow C(x)]$	generalized (\forall intro): 3-7

Existential elimination

Existential elimination rule, denoted by (\exists elim):

$\exists x A(x)$	
\vdots	
$\boxed{c} A(c)$	
\vdots	
B	
B	

What principle is reflected by this rule?

Example

1	$\exists x \text{ student}(x)$	
2	$\forall x [\text{student}(x) \rightarrow \text{learns}(x)]$	
3	$\boxed{\text{Mary}} \text{ student}(\text{Mary})$	
4	$\text{student}(\text{Mary}) \rightarrow \text{learns}(\text{Mary})$	(\forall elim): 2
5	$\text{learns}(\text{Mary})$	(\rightarrow elim): 3-4
6	$\exists x \text{ learns}(x)$	(\exists intro): 5
7	$\exists x \text{ learns}(x)$	(\exists elim): 1-6

Existential introduction

Existential introduction rule, denoted by (\exists intro):

$$\begin{array}{|l} A(c) \\ \vdots \\ \exists x A(x) \end{array}$$

(\exists intro) reflects the principle that if a formula holds for an individual, then it is existentially valid.

Examples

1.

$$\begin{array}{|l} \text{1} \quad \text{human}(\text{John}) \\ \text{2} \quad \exists x \text{human}(x) \quad (\exists\text{intro}): 1 \end{array}$$
2.

$$\begin{array}{|l} \text{1} \quad \text{wise}(\text{Eve}) \wedge \text{logician}(\text{Eve}) \\ \text{2} \quad \exists x [\text{wise}(x) \wedge \text{logician}(x)] \quad (\exists\text{intro}): 1 \end{array}$$

Proving facts about real-world phenomena

In practical applications one often formalizes reasoning using implication. Implication is then represented as a rule, reflecting a specific situation and not applicable in general. Consider implication $A \rightarrow B$. We have:

$$\begin{array}{|l} \text{1} \quad A \rightarrow B \\ \text{2} \quad A \\ \hline \text{3} \quad B \quad (\rightarrow\text{elim}): 1, 2 \end{array}$$

Assuming that $A \rightarrow B$ holds in the considered reality, we often abbreviate this reasoning using rule:

$$\begin{array}{|l} A \\ \hline B \end{array}$$

Example

Consider adults. Then the following implication:

$$\text{man}(x) \rightarrow \neg \text{woman}(x)$$

is represented as rule:

$$\begin{array}{|l} \text{man}(x) \\ \hline \neg \text{woman}(x) \end{array}$$

even if this rule is not valid in general (why?).

A cube example

Rule:

$$\begin{array}{|l} \text{leftOf}(x, y) \\ \hline \text{rightOf}(y, x) \end{array}$$

reflects implication $\forall x [\text{leftOf}(x, y) \rightarrow \text{rightOf}(y, x)]$

We can apply this rule in reasoning:

- $$\begin{array}{|l} \text{1} \quad \forall x [\text{cube}(x) \rightarrow \text{large}(x)] \\ \text{2} \quad \forall x [\text{large}(x) \rightarrow \text{leftOf}(x, \text{redCirc})] \\ \text{3} \quad \exists x \text{cube}(x) \\ \text{4} \quad \boxed{c} \text{cube}(e) \\ \text{5} \quad \text{cube}(e) \rightarrow \text{large}(e) \quad (\forall\text{elim}): 1 \\ \text{6} \quad \text{large}(e) \quad (\rightarrow\text{elim}): 4, 5 \\ \text{7} \quad \text{large}(e) \rightarrow \text{leftOf}(e, \text{redCirc}) \quad (\forall\text{elim}): 2 \\ \text{8} \quad \text{leftOf}(e, \text{redCirc}) \quad (\rightarrow\text{elim}): 6, 7 \\ \text{9} \quad \text{rightOf}(\text{redCirc}, e) \\ \text{10} \quad \forall x [\text{cube}(x) \rightarrow \text{rightOf}(\text{redCirc}, x)] \quad (\forall\text{intro}): 4-9 \end{array}$$

DeMorgan law: example of a complex proof

We prove that:

$$\neg \forall x A(x) \rightarrow \exists x \neg A(x)$$

- $$\begin{array}{|l} \text{1} \quad \neg \forall x A(x) \\ \text{2} \quad \neg \exists x \neg A(x) \\ \text{3} \quad \boxed{c} \\ \text{4} \quad \neg A(c) \\ \text{5} \quad \exists x \neg A(x) \quad (\exists\text{intro}): 4 \\ \text{6} \quad \text{FALSE} \quad (\perp\text{intro}): 5, 2 \\ \text{7} \quad \neg \neg A(c) \quad (\neg\text{intro}): 4-6 \\ \text{8} \quad A(c) \quad (\neg\text{elim}): 7 \\ \text{9} \quad \forall x A(x) \quad (\forall\text{intro}): 3-8 \\ \text{10} \quad \text{FALSE} \quad (\perp\text{intro}): 9, 1 \\ \text{11} \quad \exists x \neg A(x) \quad (\neg\text{intro}): 2-10 \end{array}$$

Numerical quantifiers

If n is a natural number, then we often consider the following quantifiers, called the *numerical quantifiers*:

- $\exists^{\leq n} x A(x)$ – there are at most n objects x satisfying $A(x)$
- $\exists^{\geq n} x A(x)$ – there are at least n objects x satisfying $A(x)$
- $\exists^n x A(x)$ (also denoted by $\exists^n A$) – there are exactly n objects x satisfying $A(x)$.

Examples

1. $\exists^{\leq 5} x \text{ cube}(x)$ – there are at most 5 cubes
2. $\exists^{\geq 5} x \text{ cube}(x)$ – there are at least 5 cubes
3. $\exists^5 x \text{ cube}(x)$ – there are exactly 5 cubes
4. $\forall x \exists^{\leq 3} y \text{ color}(x, y)$
– there are at most 3 colors of considered objects
5. $\forall x \exists^{\leq 1} y \text{ married}(x, y)$
– everybody is married to at most one person
6. $\forall x \exists^{\leq 4} y [\text{student}(x) \wedge \text{book}(y) \wedge \text{borrows}(x, y)]$
– what is the meaning of the above formula?

Interpreting numerical quantifiers in terms of classical quantifiers

Numerical quantifiers can be expressed by classical quantifiers:

$$\begin{aligned} \exists^{\geq n} x A(x) &\leftrightarrow \exists x_1, x_2, \dots, x_n \\ &\quad [x_1 \neq x_2 \wedge \dots \wedge x_{n-1} \neq x_n \wedge \\ &\quad A(x_1) \wedge \dots \wedge A(x_n)] \end{aligned}$$

$$\begin{aligned} \exists^{\leq n} x A(x) &\leftrightarrow \neg \exists x_1, x_2, \dots, x_n, x_{n+1} \\ &\quad [x_1 \neq x_2 \wedge \dots \wedge x_n \neq x_{n+1} \wedge \\ &\quad A(x_1) \wedge \dots \wedge A(x_{n+1})] \end{aligned}$$

$$\exists^n x A(x) \leftrightarrow [\exists^{\leq n} x A(x) \wedge \exists^{\geq n} x A(x)]$$

Examples

1. $\exists^{\geq 3} x \text{ cube}(x) \leftrightarrow$
 $\exists x_1, x_2, x_3$
 $[x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_2 \neq x_3 \wedge \text{cube}(x_1) \wedge \text{cube}(x_2) \wedge \text{cube}(x_3)]$
2. $\exists^{\leq 3} x \text{ cube}(x) \leftrightarrow$
 $\neg \exists x_1, x_2, x_3, x_4$
 $[x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge$
 $\wedge x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge$
 $x_3 \neq x_4 \wedge$
 $\text{cube}(x_1) \wedge \text{cube}(x_2) \wedge \text{cube}(x_3) \wedge \text{cube}(x_4)]$

Examples of applications of numerical quantifiers: Russellian analysis of definite descriptions

Bertrand Russell considered sentences with *definite descriptions* (like “the” and its equivalents), for example:

1. “The A is a B ” translates into
“There is exactly one A and it is a B ”
Example: “The *cube* is *small*” translates into:
“There is exactly one *cube* in question and it is *small*.”
2. “Both A ’s are B ’s” translates into
“There is exactly two A ’s and each of them is a B ”
Example: “Both *cubes* are *small*” translates into:
“There are exactly two *cubes* in question and each of them is *small*.”
3. “Neither A is a B ” translates into
“There are exactly two A ’s and none of them is a B ”
Example: “Neither *cube* is *small*” translates into:
“There are exactly two *cubes* in question and none of them is *small*.”

What you should have learnt from Lecture VI?

- what are rules for universal quantifiers?
- what are rules for existential quantifiers?
- how to apply those rules in practical reasoning?
- what are numerical quantifiers?
- what is a definite description and how to translate it into logic?

NNF: Negation Normal Form

A *literal* is a formula of the form A or $\neg A$, where A is an atomic formula. A literal of the form A is called *positive* and of the form $\neg A$ is called *negative*.

We say that formula A is in *negation normal form*, abbreviated by NNF, iff it contains no other connectives than \wedge, \vee, \neg , and the negation sign \neg appears in literals only.

Examples

1. $(p \vee \neg q \wedge s) \vee \neg t$ is in NNF
2. $(p \vee \neg \neg q \wedge s) \vee \neg t$ is not in NNF
3. $R(x, y) \wedge \neg Q(x, z) \wedge \forall z [\neg S(z) \wedge T(z)]$ is in NNF
4. $R(x, y) \wedge \neg Q(x, z) \wedge \neg \forall z [\neg S(z) \wedge T(z)]$ is not in NNF

Any classical first-order formula can be equivalently transformed into the NNF

Examples

- 2'. $(p \vee q \wedge s) \vee \neg t$, equivalent to formula 2. above, is in NNF
- 4'. $R(x, y) \wedge \neg Q(x, z) \wedge \exists z [S(z) \vee \neg T(z)]$, equivalent to formula 4. above, is in NNF.

Transforming formulas into NNF

Replace subformulas according to the table below, until NNF is obtained.

Rule	Subformula	Replaced by
1	$A \leftrightarrow B$	$(\neg A \vee B) \wedge (A \vee \neg B)$
2	$A \rightarrow B$	$\neg A \vee B$
3	$\neg \neg A$	A
4	$\neg(A \vee B)$	$\neg A \wedge \neg B$
5	$\neg(A \wedge B)$	$\neg A \vee \neg B$
6	$\neg \forall x A$	$\exists x \neg A$
7	$\neg \exists x A$	$\forall x \neg A$

Example

$$\neg \{ \forall x [A(x) \vee \neg B(x)] \rightarrow \exists y [\neg A(y) \wedge C(y)] \} \xrightarrow{(2)}$$

$$\neg \{ \neg \forall x [A(x) \vee \neg B(x)] \vee \exists y [\neg A(y) \wedge C(y)] \} \xrightarrow{(4)}$$

$$\{ \neg \neg \forall x [A(x) \vee \neg B(x)] \} \wedge \{ \neg \exists y [\neg A(y) \wedge C(y)] \} \xrightarrow{(3)}$$

$$\forall x [A(x) \vee \neg B(x)] \wedge \{ \neg \exists y [\neg A(y) \wedge C(y)] \} \xrightarrow{(7)}$$

$$\forall x [A(x) \vee \neg B(x)] \wedge \forall y \neg [\neg A(y) \wedge C(y)] \xrightarrow{(5)}$$

$$\forall x [A(x) \vee \neg B(x)] \wedge \forall y [\neg \neg A(y) \vee \neg C(y)] \xrightarrow{(3)}$$

$$\forall x [A(x) \vee \neg B(x)] \wedge \forall y [A(y) \vee \neg C(y)].$$

CNF: Conjunctive Normal Form

- A *clause* is any formula of the form:

$$A_1 \vee A_2 \vee \dots \vee A_k,$$
 where $k \geq 1$ and A_1, A_2, \dots, A_k are literals
- A *Horn clause* is a clause in which at most one literal is positive.

Let A be a quantifier-free formula. We say A is in *conjunctive normal form*, abbreviated by CNF, if it is a conjunction of clauses.

Examples

1. $A(x) \vee B(x, y) \vee \neg C(y)$ is a clause but not a Horn clause
2. $\neg A(x) \vee B(x, y) \vee \neg C(y)$ as well as $\neg A(x) \vee \neg B(x, y) \vee \neg C(y)$ are Horn clauses
3. $(P \vee Q \vee T) \wedge (S \vee \neg T) \wedge (\neg P \vee \neg S \vee \neg T)$ is in CNF
4. $[\neg A(x) \vee B(x, y) \vee \neg C(y)] \wedge [A(x) \vee B(x, y) \vee \neg C(y)]$ is in CNF
5. $(P \wedge Q \vee T) \wedge (S \vee \neg T) \wedge (\neg P \vee \neg S \vee \neg T)$ is not in CNF
6. $\forall x [\neg A(x) \vee B(x, y) \vee \neg C(y)] \wedge [A(x) \vee B(x, y) \vee \neg C(y)]$ is not in CNF, since it is not quantifier-free.

Why Horn clauses are important?

Any Horn clause can be transformed into the form of implication:

$$[A_1 \wedge A_2 \wedge \dots \wedge A_l] \rightarrow B,$$

where all A_1, A_2, \dots, A_l, B are positive literals. Such implications are frequently used in everyday reasoning, expert systems, deductive database queries, etc.

Examples

1. $[temp \geq 100^\circ C \wedge time \geq 6min] \rightarrow eggHardBoiled$
2. $[snow \wedge ice] \rightarrow safeSpeed \leq 30 \frac{km}{h}$
3. $[day = Monday \wedge 7 \leq week \leq 12] \rightarrow lecture(13:00)$
4. $[starterProblem \wedge temp \leq -30^\circ C] \rightarrow chargeBattery.$

In the case of two or more positive literals, any clause is equivalent to:

$$[A_1 \wedge A_2 \wedge \dots \wedge A_l] \rightarrow [B_1 \vee B_2 \vee \dots \vee B_n],$$

where all literals $A_1, A_2, \dots, A_l, B_1, B_2, \dots, B_n$ are positive.

Disjunction on the righthand side of implication causes serious complexity problems.

Transforming formulas into CNF

Any quantifier-free formula can be equivalently transformed into the CNF.

1. Transform the formula into NNF
2. Replace subformulas according to the table below, until CNF is obtained.

Rule	Subformula	Replaced by
8	$(A \wedge B) \vee C$	$(A \vee C) \wedge (B \vee C)$
9	$C \vee (A \wedge B)$	$(C \vee A) \wedge (C \vee B)$

Example

$$\begin{aligned}
 &(\neg \text{fastFood} \wedge \neg \text{restaurant}) \vee (\text{walk} \wedge \text{park}) && \stackrel{(8)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \vee (\text{walk} \wedge \text{park})) \wedge (\neg \text{restaurant} \vee (\text{walk} \wedge \text{park})) && \stackrel{(9)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \vee \text{walk}) \wedge (\neg \text{fastFood} \vee \text{park}) \wedge \\
 &\quad (\neg \text{restaurant} \vee (\text{walk} \wedge \text{park})) && \stackrel{(9)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \vee \text{walk}) \wedge (\neg \text{fastFood} \vee \text{park}) \wedge \\
 &\quad (\neg \text{restaurant} \vee \text{walk}) \wedge (\neg \text{restaurant} \vee \text{park})
 \end{aligned}$$

We then have the following conjunction of implications:

$$\bigwedge \begin{cases} \text{fastFood} \rightarrow \text{walk} \\ \text{fastFood} \rightarrow \text{park} \\ \text{restaurant} \rightarrow \text{walk} \\ \text{restaurant} \rightarrow \text{park}. \end{cases}$$

Applications of CNF

- rule-based reasoning
- logic programming
- deductive databases
- expert systems
- tautology checking.

A formula in CNF can easily be tested for validity, since in this case we have the following simple criterion:

if each clause contains a literal together with its negation, then the formula is a tautology, otherwise it is not a tautology.

Examples

1. $(p \vee q \vee \neg p) \wedge (p \vee \neg q \vee r \vee q)$ is a tautology
2. $(p \vee q \vee \neg p) \wedge (p \vee \neg q \vee r)$ is not a tautology.

DNF: Disjunctive Normal Form

A *term* is any formula of the form:

$$A_1 \wedge A_2 \wedge \dots \wedge A_k,$$

where $k \geq 1$ and A_1, A_2, \dots, A_k are literals

Let A be a quantifier-free formula. We say A is in *disjunctive normal form*, abbreviated by DNF, if it is a disjunction of terms.

Examples

1. $A(x) \wedge B(x, y) \wedge \neg C(y)$ is a term
2. $(P \wedge Q \wedge T) \vee (S \wedge \neg T) \vee (\neg P \wedge \neg S \wedge \neg T)$ is in DNF
3. $[\neg A(x) \wedge B(x, y) \wedge \neg C(y)] \vee [A(x) \wedge B(x, y) \wedge \neg C(y)]$ is in DNF
4. $(P \wedge Q \vee T) \wedge (S \vee \neg T) \wedge (\neg P \vee \neg S \vee \neg T)$ is not in DNF
5. $\forall x [\neg A(x) \wedge B(x, y) \wedge \neg C(y)] \vee [A(x) \wedge B(x, y) \wedge \neg C(y)]$ is not in DNF, since it is not quantifier-free.

Transforming formulas into DNF

Any quantifier-free formula can be equivalently transformed into the DNF.

1. Transform the formula into NNF
2. Replace subformulas according to the table below, until DNF is obtained.

Rule	Subformula	Replaced by
10	$(A \vee B) \wedge C$	$(A \wedge C) \vee (B \wedge C)$
11	$C \wedge (A \vee B)$	$(C \wedge A) \vee (C \wedge B)$

Example

$$\begin{aligned}
 &(\neg \text{fastFood} \vee \neg \text{restaurant}) \wedge (\text{walk} \vee \text{park}) && \stackrel{(10)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \wedge (\text{walk} \vee \text{park})) \vee (\neg \text{restaurant} \wedge (\text{walk} \vee \text{park})) && \stackrel{(11)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \wedge \text{walk}) \vee (\neg \text{fastFood} \wedge \text{park}) \vee \\
 &\quad (\neg \text{restaurant} \wedge (\text{walk} \vee \text{park})) && \stackrel{(11)}{\Leftrightarrow} \\
 &(\neg \text{fastFood} \wedge \text{walk}) \vee (\neg \text{fastFood} \wedge \text{park}) \vee \\
 &\quad (\neg \text{restaurant} \wedge \text{walk}) \vee (\neg \text{restaurant} \wedge \text{park})
 \end{aligned}$$

Now we have a formula representing various cases.

Satisfiability test for formulas in DNF

Satisfiability criterion for formulas in DNF:
if each term contains a literal together with its negation, then the formula is not satisfiable, otherwise it is satisfiable.

Examples

1. $(p \wedge q \wedge \neg p) \vee (p \wedge \neg q \wedge \neg r)$ is satisfiable
2. $(p \wedge q \wedge \neg p) \vee (p \wedge \neg q \wedge q)$ is not satisfiable
3. $[R(x) \wedge P(x, y) \wedge \neg T(x, y) \wedge \neg P(x, y)] \vee [R(y) \wedge \neg T(y, z) \wedge \neg R(y)]$ is not satisfiable
4. $[R(x) \wedge P(x, y) \wedge \neg T(x, y) \wedge \neg P(x, z)] \vee [R(y) \wedge \neg T(y, z) \wedge \neg R(y)]$ is satisfiable – why?

PNF: Prenex Normal Form

We say A is in *prenex normal form*, abbreviated by PNF, if all its quantifiers (if any) are in its prefix, i.e., it has the form:

$$Q_1 x_1 Q_2 x_2 \dots Q_n x_n [A(x_1, x_2, \dots, x_n)],$$

where $n \geq 0$, Q_1, Q_2, \dots, Q_n are quantifiers (\forall, \exists) and A is quantifier-free.

Examples

1. $A(x) \wedge B(x, y) \wedge \neg C(y)$ is in PNF
2. $\forall x \exists y [A(x) \wedge B(x, y) \wedge \neg C(y)]$ is in PNF
3. $A(x) \wedge \forall x [B(x, y) \wedge \neg C(y)]$ as well as $\forall x [A(x) \wedge B(x, y) \wedge \neg \exists y C(y)]$ are not in PNF.

Transforming formulas into PNF

Any classical first-order formula can be equivalently transformed into the PNF.

1. Transform the formula into NNF
2. Replace subformulas according to the table below, until PNF is obtained, where Q denotes any quantifier, \forall or \exists .

Rule	Subformula	Replaced by
12	$Qx A(x)$, $A(x)$ without variable x	$Qz A(z)$
13	$\forall x A(x) \wedge \forall x B(x)$	$\forall x [A(x) \wedge B(x)]$
14	$\exists x A(x) \vee \exists x B(x)$	$\exists x [A(x) \vee B(x)]$
15	$A \vee Qx B$, where A contains no x	$Qx (A \vee B)$
16	$A \wedge Qx B$, where A contains no x	$Qx (A \wedge B)$

Example

$$\begin{aligned}
 & A(z) \vee \forall x [B(x, u) \wedge \exists y C(x, y) \wedge \forall z D(z)] && (15) \\
 \Leftrightarrow & \forall x \{ A(z) \vee [B(x, u) \wedge \exists y C(x, y) \wedge \forall z D(z)] \} && (16) \\
 \Leftrightarrow & \forall x \{ A(z) \vee \exists y [B(x, u) \wedge C(x, y) \wedge \forall z D(z)] \} && (15) \\
 \Leftrightarrow & \forall x \exists y \{ A(z) \vee [B(x, u) \wedge C(x, y) \wedge \forall z D(z)] \} && (16) \\
 \Leftrightarrow & \forall x \exists y \{ A(z) \vee \forall z [B(x, u) \wedge C(x, y) \wedge D(z)] \} && (12) \\
 \Leftrightarrow & \forall x \exists y \{ A(z) \vee \forall t [B(x, u) \wedge C(x, y) \wedge D(t)] \} && (15) \\
 \Leftrightarrow & \forall x \exists y \forall t \{ A(z) \vee [B(x, u) \wedge C(x, y) \wedge D(t)] \}. &&
 \end{aligned}$$

Resolution rule for propositional calculus

Resolution method has been introduced by Robinson and is considered the most powerful automated proving technique, in particular, applied in the logic programming area (e.g., PROLOG).

Resolution rule, denoted by (*res*), is formulated for clauses as follows:

$$\frac{\alpha \vee L \quad \neg L \vee \beta}{\alpha \vee \beta}$$

where L is a literal and α, β are clauses.
 The position of L and $\neg L$ in clauses does not matter.
 The empty clause is equivalent to FALSE and will be denoted by FALSE.

Resolution rule reflects the transitivity of implication! — why?

Presenting α, β as clauses, the resolution rule takes form:

$$\frac{L_1 \vee \dots \vee L_{k-1} \vee L_k \quad \neg L_k \vee M_1 \vee \dots \vee M_l}{L_1 \vee \dots \vee L_{k-1} \vee M_1 \vee \dots \vee M_l}$$

Examples

1.

1	$John\ sleeps \vee John\ works$	
2	$John\ sleeps \vee John\ doesn't\ work$	
3	$John\ sleeps$	$(res): 1, 2$

2.

1	$P \vee Q$	
2	$\neg Q \vee S \vee T$	
3	$P \vee S \vee T$	$(res): 1, 2$

3.

1	$\neg Q \vee P$	
2	$S \vee T \vee Q$	
3	$P \vee S \vee T$	$(res): 1, 2$

4. obtaining the empty clause is equivalent to (\perp intro):

1	P	
2	$\neg P$	
3	$FALSE$	$(res): 1, 2$

Factorization rule for propositional calculus

Factorization rule, denoted by (*fctr*):
Remove from a clause all repetitions of literals.

Examples

1.

1	$P \vee P \vee Q \vee P \vee Q$	
2	$P \vee Q$	$(fctr): 1$

2.

1	$P \vee P$	
2	$\neg P \vee \neg P$	
3	P	$(fctr): 1$
4	$\neg P$	$(fctr): 2$
5	$FALSE$	$(res): 3, 4$

Resolution method

Resolution method is based on the principle of reduction to absurdity. Assume formula A is to be proved. Then:

1. consider negation of the input formula $\neg A$
2. transform $\neg A$ into the conjunctive normal form
3. try to derive the empty clause $FALSE$ by applying resolution (*res*) and factorization (*fctr*)
4.
 - if the empty clause is obtained then A is a tautology,
 - if the empty clause cannot be obtained no matter how (*res*) and (*fctr*) are applied, then conclude that A is not a tautology.

Example

Prove that formula $[(A \vee B) \wedge (\neg A)] \rightarrow B$ is a tautology:

1. negate: $(A \vee B) \wedge (\neg A) \wedge \neg B$ – this formula is in the CNF
2. obtaining the empty clause $FALSE$:

1	$A \vee B$	the first clause	
2	$\neg A$	the second clause	
3	$\neg B$	the third clause	
4	B		$(res): 1, 2$
5	$FALSE$		$(res): 3, 4$

Example from lecture II, slides 26-27 revisited

Recall that the first task we considered there was to check whether:

$$\left. \begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(tx \wedge \neg ir) \rightarrow \neg un] \end{array} \right\} \rightarrow [(\neg ir) \rightarrow ge]$$

Negating the above formula we obtain:

$$\begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(tx \wedge \neg ir) \rightarrow \neg un] \\ \wedge \neg ir \\ \wedge \neg ge \end{array}$$

Conjunctive normal form:

$$\begin{array}{l} [ir \vee ge \vee un] \\ \wedge [ge \vee tx] \\ \wedge [\neg tx \vee ir \vee \neg un] \\ \wedge \neg ir \\ \wedge \neg ge \end{array}$$

Proof:

1	$ir \vee ge \vee un$	
2	$ge \vee tx$	
3	$\neg tx \vee ir \vee \neg un$	
4	$\neg ir$	
5	$\neg ge$	
6	$\neg tx \vee \neg un$	(res): 3, 4
7	$ge \vee \neg un$	(res): 2, 6
8	$ge \vee un$	(res): 1, 4
9	$ge \vee ge$	(res): 7, 8
10	ge	(fctr): 9
11	FALSE	(res): 5, 10

Thus our implication indeed holds!

The second task

The second task considered there was to check whether:

$$\left. \begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(\neg ir) \rightarrow ge] \end{array} \right\} \rightarrow [(tx \wedge \neg ir) \rightarrow \neg un]$$

We negate it:

$$\begin{array}{l} [(\neg ir) \rightarrow (ge \vee un)] \\ \wedge [(\neg ge) \rightarrow tx] \\ \wedge [(\neg ir) \rightarrow ge] \\ \wedge [tx \wedge \neg ir] \\ \wedge un \end{array}$$

Conjunctive normal form:

$$\begin{array}{l} [ir \vee ge \vee un] \\ \wedge [ge \vee tx] \\ \wedge [ir \vee ge] \\ \wedge tx \\ \wedge \neg ir \\ \wedge un \end{array}$$

No matter how we apply (res) and (fctr) to the above clauses, we cannot infer FALSE.

Thus the implication we consider is not valid.

Generalizing to quantifiers

Consider the following two formulas:

$$\begin{array}{l} \forall x [rare(x) \rightarrow expensive(x)] \\ \forall y [expensive(y) \rightarrow guarded(y)] \end{array}$$

Clause form:

$$\begin{array}{l} \neg rare(x) \vee expensive(x) \\ \neg expensive(y) \vee guarded(y) \end{array}$$

Can we apply (res)?

NO!

Because “*expensive(x)*” and “*expensive(y)*” are not the same!
On the other hand, $\forall y$ could equivalently be replaced by $\forall x$ and resolution could be applied.

Consider:

$$\begin{array}{l} \neg rare(x) \vee expensive(x) \\ rare(VermeerPainting) \end{array}$$

We would like to conclude *expensive(VermeerPainting)*.

But again (res) cannot be applied

We need a general method to solve such problems. It is offered by unification.

Unification

Given two expressions, *unification* depends on substituting variables by expressions so that both input expressions become identical.
If this is possible, the given expressions are said to be *unifiable*.

Examples

1. In order to unify expressions

$$father(x) \text{ and } father(mother(John)),$$

we have to substitute variable *x* by expression *mother(John)*.

2. In order to unify expressions

$$(x + f(y)) \text{ and } (\sqrt{2 * z} + f(3)),$$

we can substitute *x* by $\sqrt{2 * z}$ and *y* by 3

3. How to unify expressions:

$$(x + f(y)) \text{ and } (\sqrt{2 * z} + z)$$

4. The following expressions cannot be unified:

$$father(x) \text{ and } mother(father(John))$$

— why?

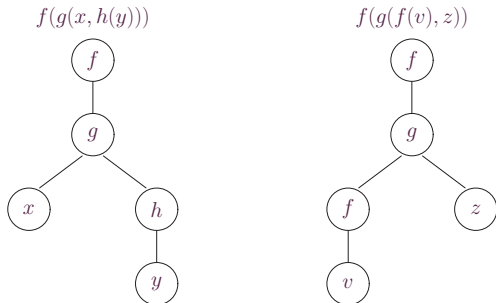
Expression trees

Recall from discrete mathematics, lecture III, that expressions can be represented as trees.

For example, expressions

$$f(g(x, h(y))) \text{ and } f(g(f(v), z))$$

can be represented as:



In order to unify expressions we have to detect nonidentical sub-trees and try to make them identical.

The unification algorithm

The *unification algorithm*:

Input: expressions e, e'

Output: substitution of variables which makes e and e' identical or, if such a substitution does not exist, a suitable information

1. traverse trees corresponding to expressions e, e'
2. if the trees are identical then stop, otherwise proceed with the next steps
3. let t and t' be subtrees that have to be identical, but are not
 - if t and t' are function symbols then conclude that the substitutions do not exist and stop
 - otherwise t or t' is a variable; let t be a variable, then substitute all occurrences of t by the expression represented by t' assuming that t does not occur in t' (if it occurs then conclude that the substitutions do not exist and stop)
4. change the trees, according to the substitution determined in the previous step and repeat from step 2.

Resolution rule for the first-order case

Resolution rule, denoted by (*res*), is formulated for first-order clauses as follows:

$$\left\{ \begin{array}{l} L_1(\bar{t}_1) \vee \dots \vee L_{k-1}(\bar{t}_{k-1}) \vee L_k(\bar{t}_k) \\ \neg L_k(\bar{t}'_k) \vee M_1(\bar{s}_1) \vee \dots \vee M_l(\bar{s}_l) \\ \vdots \\ L_1(\bar{t}'_1) \vee \dots \vee L_{k-1}(\bar{t}'_{k-1}) \vee M_1(\bar{s}'_1) \vee \dots \vee M_l(\bar{s}'_l) \end{array} \right.$$

where:

- $L_1, \dots, L_k, M_1, \dots, M_l$ are literals
- t_k and t'_k are unifiable
- primed expressions are obtained from non-primed expressions by applying substitutions unifying t_k and t'_k .

Example

$$\begin{array}{l|l} 1 & \neg \text{parent}(x, y) \vee [x = \text{father}(y) \vee x = \text{mother}(y)] \\ 2 & \text{parent}(\text{John}, \text{Mary}) \\ \hline 3 & \text{John} = \text{father}(\text{Mary}) \vee \text{John} = \text{mother}(\text{Mary}) \\ 4 & \text{(res): 1, 2 with } x = \text{John}, y = \text{Mary} \end{array}$$

Factorization rule for the first-order case

Factorization rule, denoted by (*fctr*):

Unify some terms in a clause and remove from the clause all repetitions of literals.

Examples

1.
$$\begin{array}{l|l} 1 & \text{parent}(x, y) \vee \text{parent}(\text{Jack}, \text{mother}(\text{Eve})) \\ 2 & \text{parent}(\text{Jack}, \text{mother}(\text{Eve})) \\ \hline 3 & \text{(fctr): 1 with } x = \text{Jack}, y = \text{mother}(\text{Eve}) \end{array}$$
2.
$$\begin{array}{l|l} 1 & P(x, y) \vee S(y, z, u) \vee P(z, u) \\ 2 & P(x, y) \vee S(y, x, y) \\ \hline 3 & \text{(fctr): 1 with } z = x, u = y \end{array}$$

Examples of proof by resolution in the first-order case

Consider the following formulas, where *med* stands for “medicine student” (recall that clauses are assumed to be implicitly universally quantified):

$$\Delta = \begin{cases} [med(x) \wedge attends(x, y)] \rightarrow [likes(x, y) \vee obliged(x, y)] \\ likes(x, math) \rightarrow \neg med(x) \\ med(John) \\ attends(John, math) \end{cases}$$

Do the above formulas imply that *John* is obliged to take math?

We consider implication:

$$\Delta \rightarrow obliged(John, math)$$

To apply the resolution method we negate this formula and obtain

$$\Delta \wedge \neg obliged(John, math),$$

i.e.,

$$[med(x) \wedge attends(x, y)] \rightarrow [likes(x, y) \vee obliged(x, y)]$$

$$likes(x, math) \rightarrow \neg med(x)$$

$$med(John)$$

$$attends(John, math)$$

$$\neg obliged(John, math).$$

Proof:

1	$\neg med(x) \vee \neg attends(x, y) \vee likes(x, y) \vee obliged(x, y)$	
2	$\neg likes(x, math) \vee \neg med(x)$	
3	$med(John)$	
4	$attends(John, math)$	
5	$\neg obliged(John, math)$	
6	$\neg med(x) \vee \neg attends(x, math) \vee \neg med(x) \vee obliged(x, math)$	
		(res): 1, 2 with $y = math$
7	$\neg med(x) \vee \neg attends(x, math) \vee obliged(x, math)$	(fctr) : 6
8	$\neg attends(John, math) \vee obliged(John, math)$	
		(res): 3, 7 with $x = John$
9	$obliged(John, math)$	(res) : 4, 8
10	FALSE	(res) : 5, 9

Check whether 1-3, 5 imply that *John* does not attend *math*.

What you should have learnt from Lecture VII?

- what is negation normal form?
- what is conjunctive normal form?
- what is disjunctive normal form?
- what is prenex normal form?
- how to transform formulas into NNF, CNF, DNF, PNF?
- what is resolution in propositional (zero-order) logic?
- what is factorization in propositional (zero-order) logic?
- what is unification?
- what is resolution in quantified (first-order) logic?
- what is factorization in quantified (first-order) logic?

What is a database?

A *database* is any finite collection of related data.

Examples

1. a figure illustrating dependencies between shapes on a plane
2. a phone book
3. a Swedish-to-English dictionary
4. a library
5. the Mendeleyev periodic table
6. an electronic mailbox
7. a personnel database of a department
8. an accounting database
9. a credit card database
10. a flight reservation database
11. ...

A closer look at databases

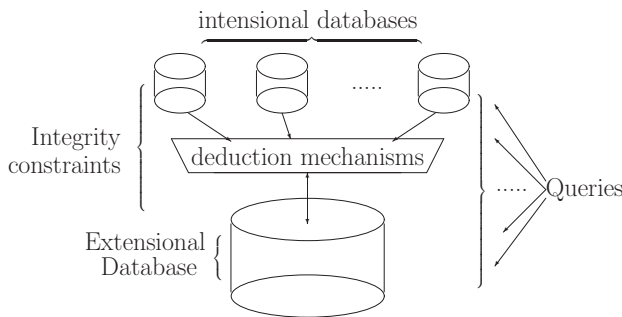
Database	Representation in logic
an atomic data item = number, letter, etc.	an element of a domain (universe)
a record = a tuple of atomic data items	a <i>fact</i>
a table = a collection of "compatible" records	a relation
a relational database = a collection of tables + integrity constraints	an <i>extensional database</i> = a collection of relations + formulas
a deductive database = relational database + deduction mechanism	a <i>deductive database</i> extensional database + <i>intensional database</i> (rules)
a database query a linguistic tool for selecting pieces of information	a formula

An *integrity constraint* of a database is a constraint that has to be satisfied by any instance of the database.

Example: a mobile phone book

Database	Representation in logic
atomic data: = first&last name, phone number	domains: NAMES, NUMBERS
a record: {Eve Lin, 0701 334567}	a fact: $PN('Eve Lin', 0701 334567)$
a table: {Eve Lin, 0701 334567} {John Smith, 0701 334555} ...	a relation: $PN('Eve Lin', 0701 334567)$ $PN('JohnSmith', 0701 334555)$...
a relational database: the above table + constraints e.g., "different persons have different phone numbers"	the extensional database: the above relation + $\forall x, y \in \text{NAMES } \forall n, m \in \text{NUMBERS}$ $[PN(x, n) \wedge PN(y, m) \wedge x \neq y] \rightarrow$ $n \neq m$
a database query: "find phone numbers of Eve Lin"	formula: $PN('Eve Lin', X)$

The architecture of deductive databases



Extensional databases, integrity constraints and queries are present in traditional database management systems. Deductive databases offer means to store deductive rules (intensional databases) and provide a deduction machinery. The basic language for deductive databases is *Datalog*.

Common assumptions

- **complexity**: database queries are computable (in a reasonable time)
- if c_1, \dots, c_n are all constant symbols appearing in the database, then it is assumed that
 - *UNA (Unique Names Axiom)*:
for all $1 \leq i \neq j \leq n$ we have that $c_i \neq c_j$
 - *DCA (Domain Closure Axiom)*:
 $\forall x [x = c_1 \vee \dots \vee x = c_n]$
- *CWA (Closed World Assumption)*:
if $R(\bar{a}_1), \dots, R(\bar{a}_k)$ are all facts about R stored in the database then
 $\forall \bar{x} [R(\bar{x}) \rightarrow (\bar{x} = \bar{a}_1 \vee \dots \vee \bar{x} = \bar{a}_k)]$
holds.

Facts in Datalog

Facts in Datalog are represented in the form of relations (see also the Discrete Mathematics course, lecture II):

$$\text{name}(arg_1, \dots, arg_k),$$

where *name* is a name of a relation and arg_1, \dots, arg_k are constants.

Examples

- $\text{address}(\text{John}, \text{'Kungsgatan 12'})$
- $\text{likes}(\text{John}, \text{Marc})$.

Atomic queries about facts

Atomic queries are of the form $\text{name}(arg_1, \dots, arg_k)$, where arg_1, \dots, arg_k are constants or variables, where ‘ $_$ ’ is also a variable (without a name, “do-not-care” variable).

Examples

- $\text{likes}(\text{John}, \text{Marc})$ – does John like Marc?
- $\text{likes}(X, \text{Marc})$ – who likes Marc?
(compute X ’s satisfying $\text{likes}(X, \text{Marc})$).

Rules in Datalog

Rules in Datalog are expressed in the form of (a syntactic variant of) Horn clauses:

$$R(\bar{Z}): \neg R_1(\bar{Z}_1), \dots, R_k(\bar{Z}_k)$$

where $\bar{Z}, \bar{Z}_1, \dots, \bar{Z}_k$ are vectors of variable or constant symbols such that any variable appearing on the lefthand side of ‘ \neg ’ (called the *head* of the rule) appears also on the righthand side of the rule (called the *body* of the rule).

The intended meaning of the rule is that

$$[R_1(\bar{Z}_1) \wedge \dots \wedge R_k(\bar{Z}_k)] \rightarrow R(\bar{Z}),$$

where all variables that appear both in the rule’s head and body are universally quantified, while those appearing only in the rule’s body, are existentially quantified.

Example

Rule:

$$R(X, c): \neg Q(X, Z), S(Z, X)$$

denotes implication:

$$\forall X \{ \exists Z [Q(X, Z) \wedge S(Z, X)] \rightarrow R(X, c) \}.$$

Examples of Datalog facts and rules

Facts:

```
is_a(zebra, mammal).
is_a(whale, mammal).
is_a(herring, fish).
is_a(shark, fish).
```

```
lives(zebra, on_land).
lives(whale, in_water).
lives(frog, on_land).
lives(frog, in_water).
lives(shark, in_water).
```

Rules:

```
can_swim(X) :- is_a(X, fish).
can_swim(X) :- lives(X, in_water).
```

Based on the above facts and rules, can one derive that:

- a whale can swim?
- a frog can swim?
- a salmon can swim?

Datalog queries

A *Datalog query* is a conjunction of literals (conjunction is denoted by comma).

Assumptions UNA, DCA and CWA allow one to formally define the meaning of queries as minimal relations which make queries logical consequences of facts and rules stored in the database.

Intuitively,

- if a query does not have free variables, then the answer is either TRUE or FALSE dependently on the database contents
- if a query contains variables, then the result consists of all assignments of values to variables, making the query TRUE.

Examples

- $\text{is_a}(X, \text{mammal})$
returns all X ’s that are listed in the database as mammals
- $\text{is_a}(X, \text{mammal}), \text{can_swim}(X)$
returns all mammals listed in the database that can swim.

Example: authors database

Consider a database containing information about book authors.

It is reasonable to store authors in one relation, say A , and book titles in another relation, say T . The intended meaning of relations is the following:

- $A(N, Id)$ means that the author whose name is N has its identification number Id
- $T(Id, Ttl)$ means that the author whose identification number is Id is a (co-) author of the book entitled Ttl .

Now, for example, one can define a new relation $Co(N_1, N_2)$, meaning that authors N_1 and N_2 co-authored a book:

```
Co(N_1, N_2) :-
  A(N_1, Id_1),
  T(Id_1, B),
  A(N_2, Id_2),
  T(Id_2, B),
  N_1 != N_2.
```

Note that the above rule reflects the intended meaning provided that authors have unique Id numbers, i.e., that the following integrity constraint holds:

$$\forall n, m, i, j$$

$$[(A(n, i) \wedge A(n, j)) \rightarrow i = j] \wedge$$

$$[(A(n, i) \wedge A(m, i)) \rightarrow n = m]$$

Querying the author database

In the authors database:

- $Co('Widom', 'Ullman')$ returns TRUE iff the database contains a book co-authored by Widom and Ullman
- $Co(N, 'Ullman')$ returns a unary relation consisting of all co-authors of Ullman.

In fact, query language can easily be made more general by allowing all first-order formulas to be database queries.

Example

Query

$$\forall X, Y [T(X, Y) \rightarrow \exists Z.(T(X, Z) \wedge Z \neq Y)]$$

returns TRUE iff any person in the database is an author of at least two books.

Exercises

Design a database and formulate sample queries:

1. for storing information about your favorite movies
2. for storing information about bus/train connections.

Example: a murder case study

Consider the following story, formalized in logic, and try to discover who was the killer.

Assume we have the following relations, with the obvious meaning:

```
person(name, age, sex, occupation)
knew(name, name)
killed_with(name, object)
killed(name)
killer(name)
motive(vice)
smeared_in(name, substance)
owns(name, object)
operates_identically(object, object)
owns_probably(name, object)
suspect(name)
```

Facts about persons:

```
person(bert, 55, m, carpenter) .
person(allan, 25, m, football_player) .
person(allan, 25, m, butcher) .
person(john, 25, m, pickpocket) .
```

```
knew(barbara, john) .
knew(barbara, bert) .
knew(susan, john) .
```

Facts about the murder:

```
killed_with(susan, club) .
killed(susan) .
```

```
motive(money) .
motive(jealousy) .
motive(revenge) .
```

```
smeared_in(bert, blood) .
smeared_in(susan, blood) .
smeared_in(allan, mud) .
smeared_in(john, chocolate) .
smeared_in(barbara, chocolate) .
```

```
owns(bert, wooden_leg) .
owns(john, pistol) .
```

Background knowledge:

```
operates_identically(wooden_leg, club).
operates_identically(bar, club).
operates_identically(pair_of_scissors, knife).
operates_identically(football_boot, club).
```

```
owns_probably(X,football_boot):-
    person(X,_,_,football_player).
owns_probably(X,pair_of_scissors):-
    person(X,_,_,hairdresser).
owns_probably(X,Object):- owns(X,Object).
```

Suspect all those who own a weapon with which Susan could have been killed:

```
suspect(X):-
    killed_with(susan,Weapon) ,
    operates_identically(Object,Weapon) ,
    owns_probably(X,Object).
```

Suspect males who knew Susan and might have been jealous about her:

```
suspect(X):-
    motive(jealousy),
    person(X,_,m,_),
    knew(susan,X).
```

Suspect females who might have been jealous about Susan:

```
suspect(X):-
    motive(jealousy),
    person(X,_,f,_),
    knew(X,Man),
    knew(susan,Man).
```

Suspect pickpockets whose motive could be money:

```
suspect(X):-
    motive(money),
    person(X,_,_,pickpocket).
```

Define a killer:

```
killer(Killer):-
    person(Killer,_,_,_),
    killed(Killed),
    Killed <> Killer, /* It is not a suicide */
    suspect(Killer),
    smeared_in(Killer,Something),
    smeared_in(Killed,Something).
```

What you should have learnt from Lecture VIII?

- what is a logical (deductive) database?
- what are extensional and intensional databases?
- what are integrity constraints?
- what are UNA, DCA and CWA?
- what is Datalog?
- what are Datalog facts and rules?
- how to query a deductive database?

Modal logics

Intensional relations (see Lecture V, slides 88-90) give rise to so-called *modal operators* (*modalities*, in short). Logics allowing such operators are called *modal logics* and sometimes *intensional logics*.

In the simplest case one deals with a single one-argument modality \Box and its dual \Diamond , i.e.,

$$\Diamond A \stackrel{\text{def}}{\Leftrightarrow} \neg \Box \neg A$$

Example

Suppose that the intended meaning of $\Box A$ is “A is sure”.

Then its dual, $\Diamond A$, is defined to be:

$$\begin{aligned} \Diamond A &\stackrel{\text{def}}{\Leftrightarrow} \neg \Box \neg A \\ &\Leftrightarrow \neg \text{“}(\neg A) \text{ is sure”} \\ &\Leftrightarrow \text{“}(\neg A) \text{ is not sure”} \\ &\Leftrightarrow \text{“}(\neg A) \text{ is doubtful”} \\ &\Leftrightarrow \text{“}A \text{ is doubtful”} \end{aligned}$$

Readings of modalities

Modalities \Box and \Diamond have many possible readings, dependent on a particular application. The following table summarizes the most frequent readings of modalities.

a possible reading of $\Box A$	a possible reading of $\Diamond A$
A is necessary	A is possible
A is obligatory	A is allowed
always A	sometimes A
in the next time moment A	in the next time moment A
A is known	A is believed
all program states satisfy A	some program state satisfies A
A is provable	A is not provable

Types of modal logics

The types of modal logics reflect the possible readings of modalities. The following table summarizes types of logics corresponding to the possible readings of modalities.

Reading of modalities	Type of modal logic
necessary, possible	<i>alethic logics</i>
obligatory, allowed	<i>deontic logics</i>
always, sometimes, in the next time	<i>temporal logics</i>
known, believed	<i>epistemic logics</i>
all (some) states of a program satisfy	<i>logics of programs</i>
(un)provable	<i>provability logics</i>

How to chose/design a logic for a particular application

Recall (see Lecture I), that there are two approaches to defining logics, the semantical and syntactical approach. In consequence there are two approaches to choosing/defining modal logics suitable for a given application:

1. syntactical approach, which depends on providing a set of axioms describing the desired properties of modal operators
2. semantical approach, which starts with suitable models and then develops syntactic characterization of modalities.

The applications might, e.g., be:

- temporal reasoning
- understanding (fragments of) natural language
- commonsense reasoning
- specification and verification of software
- understanding law and legal reasoning
- etc.

Example: syntactical approach

Let us first focus on the first approach. Suppose we are interested in formalizing knowledge operator. One can postulate, or instance, the following properties:

1. $\Box A \rightarrow \Diamond A$
— if A is known then A is believed
2. $\Box A \rightarrow A$
— if A is known then it actually holds
3. $A \rightarrow \Diamond A$
— if A holds then it is believed.

Of course there are many important questions at this point, including the following:

- are the properties consistent?
- do the properties express all the desired phenomena?
- is a property a consequence of another property?

Usually answers to such questions are given by semantic investigations.

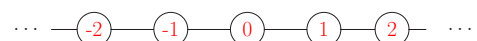
Example: semantical approach

Consider temporal logic, where $\Box A$ means “always A ” and $\Diamond A$ means “sometimes A ”.

In order to define meaning of \Box and \Diamond we first have to decide what is the time structure, e.g.,

- consisting of points or intervals?
- continuous or discrete?
- linear or branching?
- is there the earliest moment (starting point) or there is an unbounded past?
- is there the latest moment (end point) or there is an unbounded future?

Assume that time is discrete and linear, with time points labelled by integers, as in th figure below.



Now “always A ” and “sometimes A ” can be more or less be defined as follows:

$$\begin{aligned} \text{“always } A\text{”} &\stackrel{\text{def}}{\Leftrightarrow} \forall i [A \text{ is satisfied in time point } i] \\ \text{“sometimes } A\text{”} &\stackrel{\text{def}}{\Leftrightarrow} \exists i [A \text{ is satisfied in time point } i] \end{aligned}$$

Lemmon's classification of modal logics

Lemmon introduced a widely accepted classification of modal logics, defined below.

The basis for Lemmon's classification is the logic **K**.
K is defined to extend the classical propositional logic by adding rules:

1. rule (K):

$$\frac{\begin{array}{l} \Box(A \rightarrow B) \\ \vdots \\ \Box A \rightarrow \Box B \end{array}}{}{}$$

2. rule (Gen):

$$\frac{A}{\Box A}$$

Other normal modal logics are defined by additional axioms expressing the desired properties of modalities.

Bellow we present some of the most important axioms.

D	$\stackrel{\text{def}}{\Leftrightarrow} \Box A \rightarrow \Diamond A$
T	$\stackrel{\text{def}}{\Leftrightarrow} \Box A \rightarrow A$
4	$\stackrel{\text{def}}{\Leftrightarrow} \Box A \rightarrow \Box \Box A$
E	$\stackrel{\text{def}}{\Leftrightarrow} \Diamond A \rightarrow \Box \Diamond A$
B	$\stackrel{\text{def}}{\Leftrightarrow} A \rightarrow \Box \Diamond A$
Tr	$\stackrel{\text{def}}{\Leftrightarrow} \Box A \leftrightarrow A$
M	$\stackrel{\text{def}}{\Leftrightarrow} \Box \Diamond A \rightarrow \Diamond \Box A$
G	$\stackrel{\text{def}}{\Leftrightarrow} \Diamond \Box A \rightarrow \Box \Diamond A$
H	$\stackrel{\text{def}}{\Leftrightarrow} (\Diamond A \wedge \Diamond B) \rightarrow [\Diamond(A \wedge B) \vee \Diamond(A \wedge \Diamond B) \vee \Diamond(B \wedge \Diamond A)]$
Grz	$\stackrel{\text{def}}{\Leftrightarrow} \Box(\Box(A \rightarrow \Box A) \rightarrow A) \rightarrow A$
Dum	$\stackrel{\text{def}}{\Leftrightarrow} \Box(\Box(A \rightarrow \Box A) \rightarrow A) \rightarrow (\Diamond \Box A \rightarrow A)$
W	$\stackrel{\text{def}}{\Leftrightarrow} \Box(\Box A \rightarrow A) \rightarrow \Box A.$

In the formulas given above:

- **D** comes from *deontic*
- **T** is a traditional name of the axiom (after Feys)
- **4** is characteristic for logic **S4** of Lewis
- **E** comes from *Euclidean* (this axiom is often denoted by **5**)
- **B** comes after *Brouwer*
- **Tr** abbreviates *trivial*
- **M** comes after *McKinsey*
- **G** comes after *Geach*
- **H** comes after *Hintikka*
- **Grz** comes after *Grzegorzcyk*
- **Dum** comes after *Dummett*
- **W** comes from *reverse well founded* (it is also known as the Löb axiom).

In Lemmon's classification **KX₀...X_m** denotes the logic extending **K** in which formulas **X₀, ..., X_m** are accepted as axioms.

The following logics are frequently used and applied.

KT	=	T	=	logic of Gödel/Feys/Von Wright
KT4	=	S4		
KT4B	=	KT4E = S5		
K4E	=	K45		
KD	=	deontic T		
KD4	=	deontic S4		
KD4B	=	deontic S5		
KTb	=	Brouwer logic		
KT4M	=	S4.1		
KT4G	=	S4.2		
KT4H	=	S4.3		
KT4Dum	=	D	=	Prior logic
KT4Grz	=	KGrz	=	Grzegorzcyk logic
K4W	=	KW	=	Löb logic
KTr	=	KT4BM	=	trivial logic.

Kripke semantics for modal logics

There are many semantics for modal logics. Here we shall follow the Kripke-like style of defining semantics for modal logics (in fact, similar semantics was a couple of years earlier defined by Kanger. Then, in the same year, the similar semantics was given by Hintikka and also Guillaume).

The basis for the Kripke semantics is the universe of *possible worlds*. Formulas are evaluated in the so-called *actual worlds*. The other worlds, alternatives to the actual world correspond to possible situations.

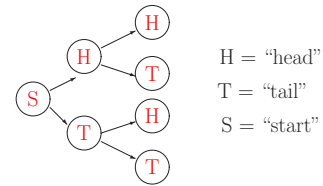
Modalities \Box and \Diamond have the following intuitive meaning:

- formula $\Box A$ holds in a given world w , if A holds in all worlds alternative for w
- formula $\Diamond A$ holds in a given world w , if A holds in some world alternative for w .

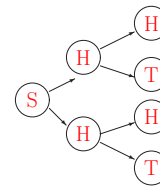


Example

Consider the situation where one drops a coin twice. The following Kripke structure describes all possible situations.



1. does $\Diamond T$ hold?
2. does $\Box H$ hold?
3. does $\Diamond \Diamond H$ hold?



What are answers to the same questions for the above structure?

Exercises

1. Design a logic for formalizing the reasoning of perception, where $\Box A$ is intended to mean that “an event satisfying formula A is being observed”.
2. Propose a deontic logic (for “obligatory” – “allowed” modalities)
3. Characterize a commonsense implication.

Multimodal logics

In many applications one needs to introduce many modalities. In such a case we have *multimodal logics*.

Examples

What modalities appear in the following sentences:

1. “When it rains one ought to take umbrella”.
2. “It is necessary that anyone should be allowed to have right to vote”.
3. “If you know what I believe to, then you might be wiser than myself”.

What you should have learnt from Lecture IX?

- what is modal operator (modality)?
- what are possible readings of modalities?
- what types of modal logics are frequently considered?
- what is Lemmon’s classification of modal logics?
- what are multi-modal logics?

Index

0-1 method, 32
actual world, 161
affirming the consequent, 48
alethic logic, 153
analytic tableaux, 41
Aristotelian form, 81
Aristotle, 81
assumption, 42, 44
atomic
 query, 141
auxiliary symbol, 6
axiom, 15, 42
 4, 158
 5, 158
 B, 158
 D, 158
 Dum, 158
 E, 158
 G, 158
 Grz, 158
 H, 158
 M, 158
 T, 158
 Tr, 158
 W, 158
Beth, 41
body of a rule, 142
bound variable, 80
Brouwer, 159, 160
clause, 111
closed world assumption, 140
CNF, 111
complete proof system, 19
completeness, 19
conclusion, 42, 44
conjunction, 22
 elimination, 65
 introduction, 66
conjunctive normal form, 111
connective, 6
 and, 22
 equivalent to, 24
 implies, 25
 not, 21
 or, 23
constant, 6
correct
 argument, 2
 proof system, 19
 reasoning, 2
correctness, 19
counter-tautology, 32
counterexample, 14

CWA, 140
database, 136
Datalog, 139
 query, 144
DCA, 140
deductive database, 137
definite description, 107
DeMorgan, 36
 laws for quantifiers, 91
DeMorgan's laws, 36
denying the antecedent, 48
deontic logic, 153
deontic variant of S4, 160
deontic variant of S5, 160
deontic variant of T, 160
derivation, 43
 rule, 42
disjunction, 23
 elimination, 67
 introduction, 68
disjunctive normal form, 115
DNF, 115
domain
 closure axiom, 140
dual modality, 152
Dummett, 159
epistemic logic, 153
equivalence, 24
 elimination, 74
 introduction, 75
existential
 elimination, 100
 introduction, 101
 quantifier, 77
extensional
 database, 137
 relation, 88
fact, 137
 in Datalog, 141
factorization, 122
 rule, 122, 132
false
 elimination, 61
 introduction, 61
Feys, 159, 160
first-order
 logic, 78
 quantifier, 78
Fitch, 44
 format, 44
 notation, 44
formal proof, 43
formula, 5, 9
free variable, 80
function symbol, 6
Gödel, 160
Geach, 159
general conditional proof, 98

generalized universal introduction, interpretation, 11
 98
Gentzen, 41
ground
 formula, 9
 term, 8
Grzegorzcyk, 160
Guillaume, 161
head of a rule, 142
Hilbert, 41
Hintikka, 159, 161
Horn, 111
 clause, 111
identity
 elimination, 57
 introduction, 57
implication, 25
 elimination, 71
 introduction, 72
 laws, 37
individual, 6
individual constant, 6
inference, 43
 rule, 42
integrity constraint, 137
intensional
 database, 137
 logics, 152
 relation, 88
justification, 44
Kanger, 161
Kripke, 161
 semantics, 161
Löb, 159, 160
language, 5
Lemmon, 157
Lewis, 159
literal, 109
 negative, 109
 positive, 109
logic, 2, 17
 alethic, 153
 deontic, 153
 epistemic, 153
 K, 157
 K45, 160
 KD, 160
 KD4, 160
 of Brouwer, 160
 of Grzegorzcyk, 160
 of Löb, 160
 of Prior, 160
 of programs, 153
 of provability, 153
 S4, 160
 S4.1, 160

S4.2, 160
S4.3, 160
S5, 160
semantical definition, 4
syntactical definition, 4
T, 160
temporal, 153
logical
 constant, 6
 language, 4, 5
 operators, 6
logics of programs, 153
McKinsey, 159
meaning, 11
meta-property, 19
modal
 logic, 152
 operators, 152
modality, 152
modus
 ponens, 47, 50
 tollens, 47, 50
multimodal logics, 163
natural deduction, 41
necessary condition, 70
negation, 21
 elimination, 62
 introduction, 62
 normal form, 109
negative literal, 109
NNF, 109
numerical quantifiers, 105
PNF, 118
positive literal, 109
possible world, 161
premise, 42, 44
prenex normal form, 118
Prior, 160
proof, 43
 by cases, 51
 by contradiction, 54
 rule, 15, 42
 system, 15, 41, 42
provability logic, 153
provable formula, 43
quantifier, 77
reduction to absurdity, 54
relation symbol, 6
resolution, 41, 120
 method, 123
 rule, 120, 131
restricted
 existential quantification, 81
 quantifier, 81
 universal quantification, 81
Robinson, 41, 120
rule, 15

for equality, 57
for identity, 57
in Datalog, 142
Russell, 107
satisfiable formula, 32
scope of a quantifier, 80
second-order
 logic, 78
 quantifier, 78
semantical approach, 11, 12, 154
Smullyan, 41
sound
 argument, 2
 proof system, 19
soundness, 19
subproof, 49
sufficient condition, 70
symmetry of identity, 58, 59
syntactical approach, 11, 15, 154
tautology, 32
temporal logic, 153
term, 8, 115
third-order
 logic, 78
 quantifier, 78
transitivity of identity, 58, 59
trivial modal logic, 160
truth
 table, 21
 for a formula, 32
 for conjunction, 22
 for disjunction, 23
 for equivalence, 24
 for implication, 25
 for negation, 21
 value, 17
UNA, 140
unifiable expressions, 128
unification, 128
 algorithm, 130
unique names axiom, 140
universal
 elimination, 95
 introduction, 96
 quantifier, 77
variable, 6
Von Wright, 160
zero-order logic, 78