cse371/mat371
LOGIC

Professor Anita Wasilewska

LECTURE 2

Chapter 2

Introduction to Classical Logic Languages and Semantics

Chapter 2
Introduction to Classical Logic Languages and Semantics

Lecture 2

Part 1: Classical Logic Model

Part 2: Propositional Language

Part 3: Propositional Semantics

Part 4: Examples of Propositional Tautologies

Lecture 2a

Part 5: Predicate Language

Part 6: Predicate Tautologies- Laws for Quantifiers

Very Short History

Logic Origins: Stoic school of philosophy (3rd century B.C.), with the most eminent representative was Chryssipus.

Modern Origins: Mid-19th century

English mathematician G. Boole, who is sometimes regarded as the founder of mathematical logic

First Axiomatic System: 1879 by German logician G. Frege.

Chapter 2
Introduction to Classical Logic Languages and Semantics

Part 1: **Classical Logic Model**

Logic

Logic builds symbolic **models** of our world

Logic builds the **models** in order to describe **formally** the ways we reason in and about our world

Logic also poses questions about **correctness** of such **models** and develops tools to answer them

Classical Model Assumptions

Assumption 1

Classical logic **model** admits only two logical values

Why two logical values only?

Classical logic was created to model the **reasoning principles** of mathematics

We expect from mathematical theorems to be always either true or false and the reasonings leading to them should guarantee this without any ambiguity

Classical Model Assumptions

Assumption 2

1. The language in which we **reason** uses **sentences**

2. The sentences are build up from basic assertions about the world using **special** words or phrases:

"not", "not true" "and", "or", " implies", "if ..... then", "from the fact that .... we can deduce", " if and only if", "equivalent", "every", "for all", "any", "some"," exists"

3. We use **symbols** do denote basic assertions and **special** words or phrases

Hence the name **symbolic logic**

Logic

Logic studies the **behavior** of the special words and phrases

Special words and phrases have accepted intuitive meanings

Logic builds **models** to formalize these **intuitive meanings**

To do so we first **define** formal **symbolic languages** and then define a formal meaning of their symbols

The formal meaning is called **semantics**

Propositional Connectives

The **symbols** for he special words and phrases are called **propositional connectives**

There are different choices of **symbols** for the propositional connectives; we adopt the following:

$\neg$ for "not", "not true"

$\cap$ for "and"

$\cup$ for "or"

$\Rightarrow$ for " implies" , "if ..... then", "from the fact that... we can deduce"

$\Leftrightarrow$ for " if and only if", "equivalent"

The **names** for the **propositional connectives** are:

$\neg$ negation

$\cap$ conjunction, $\cup$ disjunction

$\Rightarrow$ implication and $\Leftrightarrow$ equivalence.

Propositional Logic

Restricting our attention to the role of **propositional connectives** yields to what is called **propositional logic**

The basic components of the **propositional logic** are a propositional language and a propositional semantics

The **propositional logic** is a quite simple model to **justify**, **describe** and **develop**

We will devote first few chapters to it

We do it both for its own sake and because it provides a good background for developing and understanding more difficult logics to follow

Quantifiers and Predicate Logic

We use **symbols**:

$\forall$ for "every", "any", "all"

$\exists$ for "some" ," exists", "there is"

The **symbols** $\forall$, $\exists$ are called **quantifiers**

Consideration and study of the **role** of propositional connectives and quantifiers leads to what is called a **predicate logic**

The **basic components** of the **predicate logic** are predicate language and predicate semantics

The **predicate logic** is a much more complicated model

We **develop** and **study** it in full formality in chapters following the introduction and examination of the **propositional logic** model

Chapter 2

Introduction to Classical Logic Languages and Semantics

Part 2: **Propositional Language**

Propositional Language

Propositional language is a quite simple, symbolic language into which we can **translate (represent)** sentences of a natural language

Example

Consider natural language sentence
" If $2 + 2 = 5$, then $2 + 2 = 4$"

We translate it into the **propositional language** as follows

We **denote** the **basic assertion** (proposition) "$2 + 2 = 5$" by a variable, let's say a, and the proposition "$2 + 2 = 4$" by a variable b

We write a connective $\Rightarrow$ for "if ..... then"

As a result we obtain a propositional language **formula**

$$(a \Rightarrow b)$$

Propositional Translation

Exercise

Translate a natural language sentence **S**
"The fact that it is not true that at the same time 2+2 = 4 and 2+2 = 5 implies that 2+2 = 4"

into a corresponding propositional language **formula**

We carry the translation as follows

1. We **identify** all words and phrases representing the **logical connectives** and we re-write the sentence **S** in a simpler form introducing parenthesis to better express its meaning

Propositional Translation

The sentence **S** becomes:

" If not $(2 + 2 = 4$ and $2 + 2 = 5)$ then $2 + 2 = 4$"

2.

We identify the **basic assertions** (propositions) and **assign** propositional variables to them:

$$a: \ "2+2=4" \quad \text{and} \qquad b: \ "2+2=5"$$

Step 3

We write the propositional language **formula**:

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

Syntax

A formal description of symbols and the definition of the set of formulas is called a **syntax** of a symbolic language

We use the word syntax to stress that the formulas do not carry neither formal meaning nor a logical value

We **assign** the meaning and logical value to syntactically defined formulas in a **separate step**

This next, separate step is called a **semantics** of the given symbolic language

A given symbolic language can have different semantics and the different semantics can define different logics

Natural Languages

One can think about a **natural language** as a set $\mathcal{W}$ of all words and sentences based on a given alphabet $\mathcal{A}$

This leads to a simple, abstract **model** of a **natural language** NL as a pair

$$NL = (\mathcal{A}, \mathcal{W})$$

Some natural languages share the same alphabet, some have different alphabets.

All of them face serious problems with a proper recognition and definitions of accepted words and complex sentences

Symbolic Languages

We do not want the symbolic languages to share the difficulties of the natural languages

We **define** their components **precisely** and in such a way that their recognition and correctness will be **easily decided**

We call their words and sentences formulas and denote the **set of all formulas** by $\mathcal{F}$

We **define** a **symbolic language** as a pair

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

Symbolic Languages Categories

We distinguish two categories of symbolic languages:

propositional and **predicate**

We define first the propositional language

The definition of the predicate language, with its much more complicated structure will follow

Propositional Language Definition

Definition

By a **propositional language** $\mathcal{L}$ we understand a pair

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

where $\mathcal{A}$ is called propositional **alphabet**

$\mathcal{F}$ is called a set of all well formed **formulas**

Language Components: Alphabet

1. Alphabet $\mathcal{A}$

The alphabet $\mathcal{A}$ consists of

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

a finite set of **propositional connectives**, and

a set of two **parenthesis**

We denote the propositional variables by letters

$$a,\ b,\ c,\ p,\ q,\ r,\ ....$$

with indices if necessary. It means that we can also use

$$a_1, a_2, ..., b_1, b_2, ...$$

as symbols for propositional variables

Language Components: Alphabet

Propositional connectives are:

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

The connectives have well established names

The connectives names are:

negation, conjunction, disjunction, implication, and **equivalence (biconditional)**

for the connectives $\neg$, $\cap$, $\cup$, $\Rightarrow$, and $\Leftrightarrow$, respectively

Parenthesis are symbols (and)

Language Components: Formulas

Formulas are expressions build by means of elements of the alphabet $\mathcal{A}$. We denote formulas by capital letters $A,\ B,\ C,\ D, ...$, with indices, if necessary.
The set $\mathcal{F}$ of **all formulas** of the propositional language $\mathcal{L}$ is **defined recursively** as follows

1. Base step: all propositional variables are are **formulas**

They are called **atomic formulas**

2. Recursive step: for any already defined **formulas** A, B, the expressions

$$\neg A,\ (A \cap B),\ (A \cup B),\ (A \Rightarrow B),\ (A \Leftrightarrow B)$$

are also **formulas**

3. Only those expressions are **formulas** that are determined to be so by means of conditions **1.** and **2.**

Formulas Example

By the definition, any propositional variable is a **formula**. Let's take two variables a and b.

By the recursive step we get that

$$(a \cap b),\ (a \cup b),\ (a \Rightarrow b),\ (a \Leftrightarrow b),\ \neg a,\ \neg b$$

are **formulas**

The recursive step applied again produces for example **formulas** :

$$\neg(a \cap b),\ \ ((a \Leftrightarrow b) \cup \neg b),\ \ \neg\neg a,\ \ \neg\neg(a \cap b)$$

Formulas

Observe that we listed only few formulas obtained in the first recursive step

As as the recursive process **continue** we obtain a set of well formed of **formulas**

The set of all formulas is countably infinite

Formulas

Remark that we put parenthesis within the **formulas** in a way to avoid ambiguity

The expression

$$a \cap b \cup a$$

is ambiguous

We don't know whether it represents a formula

$$(a \cap b) \cup a \quad \text{or a formula} \quad a \cap (b \cup a)$$

Observe that **neither** of formulas $a \cap b \cup a$, $(a \cap b) \cup a$ or $a \cap (b \cup a)$ is a **well formed formula**

Exercises

Exercise

Consider a following set

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

1. Determine which of the elements of $\mathcal{S}$ are, and which are not well formed formulas of $\mathcal{L} = (\mathcal{A}, \mathcal{F})$

2. For any $A \notin \mathcal{F}$ re-write it as a **correct** formula and write what it says in the natural language

Exercises

Solution

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

A corrected formula is $(\neg a \Rightarrow (a \cup b))$

It says: "If a is not true , then we have a or b "

Another corrected formula in is $\neg(a \Rightarrow (a \cup b))$

It says: "It is not true that a implies a or b "

Exercises

Solution

The formula $((\neg a) \Rightarrow (a \cup b))$ is **not correct** because $(\neg a) \notin \mathcal{F}$

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

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

The formula $\neg(a \rightarrow a) \notin \mathcal{F}$ is **not correct**

The connective $\rightarrow$ does not belong to the language $\mathcal{L}$

$\neg(a \rightarrow a)$ is a correct formula of **another propositional language**; the one that uses a symbol $\rightarrow$ for implication

Exercise

Write following natural language statement:

"One likes to play bridge or from the fact that the weather is good we conclude the following: one does not like to play bridge or one likes swimming"

as a formula of the propositional language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$

Solution

First we identify the needed components of the alphabet $\mathcal{A}$:

propositional variables: a, b, c

a denotes statement: one likes to play bridge, b denotes a statement: the weather is good, c denotes a statement: one likes swimming

Connectives: $\cup$, $\Rightarrow$, $\cup$. $\neg$

The corresponding **formula** of $\mathcal{L}$ is

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

Symbols for Connectives

The connectives symbols we use are not the only one used in mathematical, logical, or computer science literature

Some Other Symbols

Negation	Disjunction	Conjunction	Implication	Equivalence
$-A$	$(A \cup B)$	$(A \cap B)$	$(A \Rightarrow B)$	$(A \Leftrightarrow B)$
NA	DAB	CAB	IAB	EAB
$\overline{A}$	$(A \vee B)$	$(A \,\&\, B)$	$(A \rightarrow B)$	$(A \leftrightarrow B)$
$\sim A$	$(A \vee B)$	$(A \cdot B)$	$(A \supset B)$	$(A \equiv B)$
A'	$(A + B)$	$(A \cdot B)$	$(A \rightarrow B)$	$(A \equiv B)$

The first notation is the closest to ours and is drawn mainly from the algebra of sets and lattice theory

The second comes from the Polish logician **J. Łukasiewicz** and is called the Polish notation

The third was used by **D. Hilbert.**

The fourth comes from **Peano** and **Russell**

The fifth goes back to **Schröder** and **Pierce**

Chapter 2
Introduction to Classical Logic Languages and Semantics

Part 3: **Propositional Semantics**

Propositional Semantics

We present now **definitions** of propositional connectives in terms of **two logical values** true or false and discuss their **motivations**

The resulting definitions are called a **semantics** for the **classical** propositional connectives

The **semantics** presented here is fairly **informal**

The **formal definition** of **classical** propositional semantics is presented in **chapter 4**

Conjunction: Motivation and Definition

Conjunction

A **conjunction** $(A \cap B)$ is a **true** formula if both A and B are **true** formulas

If one of the formulas, or both, are **false**, then the **conjunction** is a **false** formula

Let's denote statement: "formula A is **false** " by $A = F$ and a statement: "formula A is **true** " by $A = T$

Conjunction: Definition

Conjunction

The logical value of a **conjunction** depends on the logical values of its factors in a way which is express in the form of the following table (truth table)

Conjunction Table

A	B	$(A \cap B)$
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction

The word or is used in natural language in two different senses.

First: A or B is true if at least one of the statements A, B is true

Second: A or B is true if one of the statements A and B is true and the other is false

In mathematics and hence in logic, the word or is used in the **first sense**

Disjunction: Definition

Disjunction

We adopt the convention that a **disjunction** $(A \cup B)$ is true if at least one of the formulas A , B is true

Disjunction Table

A	B	$(A \cup B)$
T	T	T
T	F	T
F	T	T
F	F	F

Negation: Definition

Negation

The **negation** of a true formula is a false formula, and the negation of a false formula is a true formula

Negation Table

A	$\neg A$
T	F
F	T

Implication: Motivation and Definition

The semantics of the statements in the form

if A, then B

needs a little bit more discussion.

In everyday language a statement *if A, then B* is interpreted to mean that B can be **inferred** from A.

In mathematics its interpretation differs from that in natural language

Implication: Motivation and Definition

Consider the following

Theorem

For every natural number n,

if 6 DIVIDES n, then 3 DIVIDES n

The theorem is **true** for any natural number, hence in particular, it is **true** for numbers 2, 3, 6

Consider number 2

The following proposition is **true**

if 6 DIVIDES 2, then 3 DIVIDES 2

It means an implication $(A \Rightarrow B)$ in which A and B are **false** is interpreted as a **true** statement

Implication: Motivation and Definition

Consider now a number 3

The following proposition **is true**

if 6 DIVIDES 3, then 3 DIVIDES 3,

It means that an implication $(A \Rightarrow B)$ in which A is **false** and B is **true** is interpreted as a **true statement**

Consider now a number 6

The following proposition is **true**

if 6 DIVIDES 6, then 3 DIVIDES 6.

It means that an implication $(A \Rightarrow B)$ in which A and B are **true** is interpreted as a **true statement**

Implication: Motivation and Definition

One more case.

What happens when in the implication $(A \Rightarrow B)$ the formula A is **true** and the formula B is **false**

Consider a sentence

if 6 DIVIDES 12, then 6 DIVIDES 5.

Obviously, this is a **false statement**

Implication: Definition

Implication

The above examples **justify** adopting the following definition of a semantics for the implication $(A \Rightarrow B)$

Implication Table

A	B	$(A \Rightarrow B)$
T	T	T
T	F	F
F	T	T
F	F	T

Equivalence Definition

Equivalence

An equivalence $(A \Leftrightarrow B)$ is **true** if both formulas A and B have the same logical value

Equivalence Table

A	B	$(A \Leftrightarrow B)$
T	T	T
T	F	F
F	T	F
F	F	T

Truth Tables Semantics

We summarize the tables for propositional connectives in the following one table.

We call it a **truth table definition** of propositional; connectives and hence we call the semantics defined here a **truth tables semantics**.

A	B	$\neg A$	$(A \cap B)$	$(A \cup B)$	$(A \Rightarrow B)$	$(A \Leftrightarrow B)$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Truth Tables Semantics

The truth tables indicate that the logical value of of propositional connectives **independent** of the formulas A, B

We write the connectives in a **"formula independent"** form as a set of of the following **equations**

$$\neg T = F, \quad \neg F = T;$$

$$T \cap T = T, \quad T \cap F = F, \quad F \cap T = F, \quad F \cap F = F;$$

$$T \cup T = T, \quad T \cup F = T, \quad F \cup T = T, \quad F \cup F = F;$$

$$T \Rightarrow T = T, \quad T \Rightarrow F = F, \quad F \Rightarrow T = T, \quad F \Rightarrow F = T;$$

$$T \Leftrightarrow T = T, \quad T \Leftrightarrow F = F, \quad F \Leftrightarrow T = F, \quad T \Leftrightarrow T = T$$

We use the above **set of connectives equations** to evaluate **logical values** of formulas

Exercise

Show that $(A \Rightarrow (\neg A \cap B)) = F$ for the following **logical values** of its **basic components**: A=T and B=F

Solution

We **calculate** the **logical value** of the formula

$$(A \Rightarrow (\neg A \cap B))$$

by **substituting** the respective logical values T, F for the component formulas A, B and applying the set of **connectives equations** as follows

$$T \Rightarrow (\neg T \cap F) = T \Rightarrow (F \cap F) = T \Rightarrow F = F$$

Extensional Connectives

Extensional connectives are the connectives that have the following property:
the logical value of the formulas form by means of these connectives and certain given formulas depends only on the logical value(s) of the given formulas

All classical **propositional connectives**

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

are **extensional**

Propositional Connectives

Remark

In everyday language there are expressions such as

"I believe that", "it is possible that", " certainly", etc....

They are represented by some **propositional connectives** which **are not extensional**

They do not play any role in **mathematics** and so are not discussed in **classical logic**, they belong to **non-classical logics**

All Extensional Two Valued Connectives

There are many **other binary** (two valued) **extensional** propositional connectives

Here is a table of **all unary** connectives

A	$\nabla_1 A$	$\nabla_2 A$	$\neg A$	$\nabla_4 A$
T	F	T	F	T
F	F	F	T	T

All Extensional Binary Connectives

Here is a table of **all binary connectives**

A	B	$(A \circ_1 B)$	$(A \cap B)$	$(A \circ_3 B)$	$(A \circ_4 B)$
T	T	F	T	F	F
T	F	F	F	T	F
F	T	F	F	F	T
F	F	F	F	F	F

A	B	$(A \downarrow B)$	$(A \circ_6 B)$	$(A \circ_7 B)$	$(A \Leftrightarrow B)$
T	T	F	T	T	T
T	F	F	T	F	F
F	T	F	F	T	F
F	F	T	F	F	T

A	B	$(A \circ_9 B)$	$(A \circ_{10} B)$	$(A \circ_{11} B)$	$(A \cup B)$
T	T	F	F	F	T
T	F	T	T	F	T
F	T	T	F	T	T
F	F	F	T	T	F

A	B	$(A \circ_{13} B)$	$(A \Rightarrow B)$	$(A \uparrow B)$	$(A \circ_{16} B)$
T	T	T	T	F	T
T	F	T	F	T	T
F	T	F	T	T	T
F	F	T	T	T	T

Functional Dependency Definition

Definition

Functional dependency of connectives is the ability of defining **some connectives** in terms of some **others**

All classical propositional connectives can be defined in terms of disjunction and negation

Two binary connectives: $\downarrow$ and $\uparrow$ suffice, each of them separately, to define **all classical connectives**, whether unary or binary

Functional Dependency

The connective $\uparrow$ was discovered in 1913 by **H.M. Sheffer**, who called it **alternative negation**

Now it is often called a Sheffer's connective

The formula

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

Negation $\neg A$ is defined as $A \uparrow A$.

Disjunction $(A \cup B)$ is defined as $(A \uparrow A) \uparrow (B \uparrow B)$

Functional Dependency

The connective $\downarrow$ was discovered by **J. Łukasiewicz** and is called a **joint negation**

The formula

$A \downarrow B$ **reads**: neither A nor B.

It was proved in 1925 by **E. Żyliński** that **no propositional connective** other than $\uparrow$ and $\downarrow$ **suffices** to define all the remaining **classical connectives**

Chapter 2

Introduction to Classical Logic Languages and Semantics

Part 4: Propositional Tautologies

Propositional Tautologies

Now we connect **syntax** (formulas of a given language $\mathcal{L}$) with **semantics** (assignment of truth values to the formulas of the language $\mathcal{L}$)

In **logic** we are interested in those propositional **formulas** that must be **always true** because of their syntactical structure **without reference** to the natural language meaning of the propositions they **represent**

Such formulas are called **propositional tautologies**

Example

Example

Given a formula $(A \Rightarrow A)$

We evaluate the logical value of our formula for **all possible** logical values of its basic component A

We put our **calculation** in a form of a table, called a **truth table** below

A	$(A \Rightarrow A)$ computation	$(A \Rightarrow A)$
T	$T \Rightarrow T = T$	**T**
F	$F \Rightarrow F = T$	**T**

The **logical value** of the formula $(A \Rightarrow A)$ is **always** T

This means that it is a **propositional tautology**.

Example

Example

Here is a **truth table** for a formula $(A \Rightarrow B)$

A	B	$(A \Rightarrow B)$ computation	$(A \Rightarrow B)$
T	T	$T \Rightarrow T = T$	**T**
T	F	$T \Rightarrow F = F$	**F**
F	T	$F \Rightarrow T = T$	**T**
F	F	$F \Rightarrow F = T$	**T**

The **logical value** of the formula $(A \Rightarrow B)$ is F for A = T and B = F what means that **it is not** a **propositional tautology**

Tautology Definition

Definition

For any formula $A \in \mathcal{F}$ of a propositional language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$, we say that A is a propositional **tautology**

if and only if the **logical value** of A is T (we write it A = T) for **all possible logical values** of its **basic components**

We write

$$\models A$$

to denote that A is a **tautology**

Classical Tautologies

Here is a **list of some** of the most known classical **notions** and **tautologies**

Modus Ponens known to the Stoics (3rd century B.C)

$$\models ((A \cap (A \Rightarrow B)) \Rightarrow B)$$

Detachment

$$\models ((A \cap (A \Leftrightarrow B)) \Rightarrow B)$$

$$\models ((B \cap (A \Leftrightarrow B)) \Rightarrow A)$$

Sufficient and Necessary

Sufficient: Given an implication $(A \Rightarrow B)$,

A is called a sufficient condition for B to hold.

Necessary : Given an implication $(A \Rightarrow B)$,

B is called a necessary condition for A to hold.

Implication Names

Simple:

$(A \Rightarrow B)$ is called a simple implication

Converse:

$(B \Rightarrow A)$ is called a converse implication to $(A \Rightarrow B)$

Opposite:

$(\neg B \Rightarrow \neg A)$ is called an opposite implication to $(A \Rightarrow B)$

Contrary:

$(\neg A \Rightarrow \neg B)$ is called a contrary implication to $(A \Rightarrow B)$

Laws of Contraposition

$$\models ((A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)),$$

$$\models ((B \Rightarrow A) \Leftrightarrow (\neg A \Rightarrow \neg B)).$$

These Laws make it possible to **replace**, in any deductive argument, a sentence of the form $(A \Rightarrow B)$ by $(\neg B \Rightarrow \neg A)$, and conversely

Necessary and sufficient

We read the formula $(A \Leftrightarrow B)$ as

"B **is necessary and sufficient for** A"

because of the following tautology

$$\models ((A \Leftrightarrow B)) \Leftrightarrow ((A \Rightarrow B) \cap (B \Rightarrow A)))$$

Stoics, 3rd century B.C.

Hypothetical Syllogism

$$\models (((A \Rightarrow B) \cap (B \Rightarrow C)) \Rightarrow (A \Rightarrow C)),$$

$$\models ((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))),$$

$$\models ((B \Rightarrow C) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))).$$

Modus Tollendo Ponens

$$\models (((A \cup B) \cap \neg A) \Rightarrow B),$$

$$\models (((A \cup B) \cap \neg B) \Rightarrow A)$$

12 to 19 Century

Duns Scotus 12/13 century

$$\models (\neg A \Rightarrow (A \Rightarrow B))$$

Clavius 16th century

$$\models ((\neg A \Rightarrow A) \Rightarrow A)$$

Frege 1879

$$\models (((A \Rightarrow (B \Rightarrow C)) \cap (A \Rightarrow B)) \Rightarrow (A \Rightarrow C)),$$

$$\models ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$$

Frege gave the the **first formulation** of the classical propositional logic as a formalized axiomatic system

Apagogic Proofs

Apagogic Proofs: means proofs by reductio ad absurdum

Reductio ad absurdum: to prove A to be true,

we assume $\neg A$

If we get a contradiction, it means that we have proved A to be true

Correctness of this reasoning is guarantee by the following tautology

$$\models ((\neg A \Rightarrow (B \cap \neg B)) \Rightarrow A)$$

Chapter 2 Classical Tautologies

Chapter 2 contains a very extensive list of **classical propositional tautologies**

Read, prove , and **memorize** as many as you can

We will use them freely in later Chapters assuming that you are really familiar with all of them