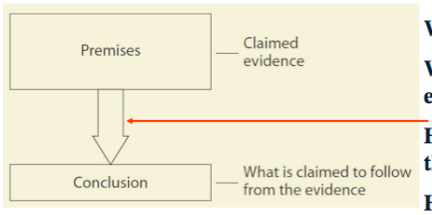
An Introduction to Classical Propositional Calculus

Jingde Cheng

Saitama University

Logic: What Is It?



[From P. J. Hurley, "A Concise Introduction to Logic"]

What entails what?

What follows from what?

Why? What are the evaluation criteria?

How to establish/define the evaluation criteria?

How to evaluate arguments/reasoning?

It is LOGIC to answer these fundamental questions.

The Fundamental Questions

What is (mathematical) logic?
What entails what? What follows from what?
Why? What are the evaluation criteria?
How to establish/define the evaluation criteria?
How to evaluate arguments/reasoning?
What is Classical Propositional Calculus?

Note: This lecture note is an introduction to CPC for CS/IS students from the viewpoint of applications; it includes all of important meta-theorems of CPC but no proof is given.

An Introduction to Classical Propositional Calculus (CPC)

- ♣ Formal (Object) Language (Syntax) of **CPC**
- ♣ Principles of Structural Induction and Structural Recursion
- ♣ Semantics (Model Theory) of **CPC**
- ♣ Semantic (Model-theoretical, Logical) Consequence Relation
- ♣ Normal Forms and Uniform Notation of Formulas
- ♣ Deduction System (Proof Theory) of **CPC**
- ♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
- ♣ Hilbert Style Formal Logic Systems for **CPC**
- ♣ Gentzen's Natural Deduction System for **CPC**
- ♣ Gentzen's Sequent Calculus System for **CPC**
- ♣ Semantic Tableau System for **CPC**
- ♣ Resolution System for **CPC**
- ♣ Forward Deduction and Backward Deduction

Formal (Object) Language (Syntax) of CPC: Alphabet (Symbols)

♣ Alphabet (Symbols)

- ***Alphabet***: $\{\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \top, \bot, p_1, p_2, ..., p_n, ..., (,)\}$ (Note: countable).
- ***Logical (propositional) connectives***: $\rightarrow$ (***material implication***), $\wedge$ (***conjunction***), $\vee$ (***disjunction***), $\leftrightarrow$ (***equivalence***), $\neg$ (***negation***).
- ***Logical constants***: $\top$ (***top***) and $\bot$ (***bottom***).
- ***Propositional variables*** (***letters***): $\mathbf{V} =_{df} \{p_1, p_2, ..., p_n, ..., \}$(Note: countable).
- Punctuation: left and right parentheses '(' and ')'.

♣ Notes

- $A \wedge B =_{df} \neg(A \rightarrow (\neg B))$, $A \vee B =_{df} (\neg A) \rightarrow B$
- $A \rightarrow B =_{df} \neg(A \wedge (\neg B))$ or $(\neg A) \vee B$, $A \leftrightarrow B =_{df} (A \rightarrow B) \wedge (B \rightarrow A)$
- Symbols 'A' and 'B' belong to our meta-language but not the object language of **CPC**.

Formal (Object) Language (Syntax) of CPC: Logical Connectives

♣ ***Boolean operators*** (Logical connectives) [Ben-Ari] ("≡" means equivalence)

negation	$\neg$
disjunction	$\vee$
conjunction	$\wedge$
implication	$\rightarrow$
equivalence	$\leftrightarrow$
exclusive or	$\oplus$
nor	$\downarrow$
nand	$\uparrow$

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

$A \oplus B \equiv \neg(A \rightarrow B) \vee \neg(B \rightarrow A)$
$A \rightarrow B \equiv \neg(A \wedge \neg B)$
$A \wedge B \equiv \neg(\neg A \vee \neg B)$
$A \wedge B \equiv \neg(A \rightarrow \neg B)$

Formal (Object) Language (Syntax) of CPC: Logical Connectives

♣ Boolean operators (Logical connectives) [Ben-Ari]

Absorption of Constants

Let us extend the syntax of Boolean formulas to include the two constant atomic propositions *true* and *false*. (Another notation is $\top$ for *true* and $\bot$ for *false*.) Their semantics are defined by $\mathscr{I}(true) = T$ and $\mathscr{I}(false) = F$ for any interpretation. Do not confuse these symbols in the object language of propositional logic with the truth values T and F used to define interpretations. Alternatively, it is possible to regard *true* and *false* as abbreviations for the formulas $p \vee \neg p$ and $p \wedge \neg p$, respectively.

The appearance of a constant in a formula can collapse the formula so that the binary operator is no longer needed; it can even make a formula become a constant whose truth value no longer depends on the non-constant subformula.

$$\begin{array}{llll} A \vee true & \equiv & true & \qquad A \wedge true \equiv A \\ A \vee false & \equiv & A & \qquad A \wedge false \equiv false \\ A \rightarrow true & \equiv & true & \qquad true \rightarrow A \equiv A \\ A \rightarrow false & \equiv & \neg A & \qquad false \rightarrow A \equiv true \\ A \leftrightarrow true & \equiv & A & \qquad A \oplus true \equiv \neg A \\ A \leftrightarrow false & \equiv & \neg A & \qquad A \oplus false \equiv A \end{array}$$

Formal (Object) Language (Syntax) of CPC: Logical Connectives

♣ Boolean operators (Logical connectives) [Ben-Ari]

Identical Operands

Collapsing can also occur when both operands of an operator are the same or one is the negation of another.

$$\begin{array}{llll} A & \equiv & \neg\neg A & \\ A & \equiv & A \wedge A & \qquad A \equiv A \vee A \\ A \vee \neg A & \equiv & true & \qquad A \wedge \neg A \equiv false \\ A \rightarrow A & \equiv & true & \\ A \leftrightarrow A & \equiv & true & \qquad A \oplus A \equiv false \\ \neg A & \equiv & A \uparrow A \neg A & \equiv A \downarrow A \\ & & & \qquad \neg A \equiv (A \downarrow A) \end{array}$$

Formal (Object) Language (Syntax) of CPC: Logic al Connectives

♣ Boolean operators (Logical connectives) [Ben-Ari]

Commutativity, Associativity and Distributivity

The binary Boolean operators are commutative, except for implication.

$$\begin{array}{llll} A \vee B & \equiv & B \vee A & \qquad A \wedge B \equiv B \wedge A \\ A \leftrightarrow B & \equiv & B \leftrightarrow A & \qquad A \oplus B \equiv B \oplus A \\ A \uparrow B & \equiv & B \uparrow A & \qquad A \downarrow B \equiv B \downarrow A \end{array}$$

If negations are added, the direction of an implication can be reversed:

$$A \rightarrow B \equiv \neg B \rightarrow \neg A$$

The formula $\neg B \rightarrow \neg A$ is the *contrapositive* of $A \rightarrow B$.

Disjunction, conjunction, equivalence and non-equivalence are associative.

$$\begin{array}{llll} A \vee (B \vee C) & \equiv & (A \vee B) \vee C & \qquad A \wedge (B \wedge C) \equiv (A \wedge B) \wedge C \\ A \leftrightarrow (B \leftrightarrow C) & \equiv & (A \leftrightarrow B) \leftrightarrow C & \qquad A \oplus (B \oplus C) \equiv (A \oplus B) \oplus C \end{array}$$

Implication, *nor* and *nand* are not associative.

Disjunction and conjunction distribute over each other.

$$\begin{array}{lll} A \vee (B \wedge C) & \equiv & (A \vee B) \wedge (A \vee C) \\ A \wedge (B \vee C) & \equiv & (A \wedge B) \vee (A \wedge C) \end{array}$$

Formal (Object) Language (Syntax) of CPC: Well-Formed Formulas

♣ ***Formulas*** (***Well-formed formulas***, ***wffs***)

- (1) Every ***propositional variable*** (***letter***), $\top$, or $\bot$ is a ***formula*** (called an ***atomic formula***) (Note: countable);
  (2) If A and B are formulas, then so are $(\neg A)$, $(A \rightarrow B)$, $(A \wedge B)$, $(A \vee B)$, $(A \leftrightarrow B)$ (Note: countable);
  (3) Nothing else are formulas.
- **WFF**$_{\mathbf{CPC}}$: the set of all formulas of **CPC** (**WFF** for short).
- 'A' and 'B' belong to our meta-language but not the object language of **CPC**.
- If we call 'A' and 'B' "***formula letters***", then a logic expression built up from the formula letters can be called a "***formula form***".

♣ Homework

- Try to develop an algorithm to check whether or not a string is a formula of **CPC**.

Formal (Object) Language (Syntax) of CPC: Subformulas

♣ Immediate subformulas

- ***Immediate subformulas*** are defined as follows:
  (1) an atomic formula has no immediate subformula;
  (2) the only immediate subformula of $(\neg A)$ is A;
  (3) for a binary connective *, the immediate subformulas of $(A*B)$ are A and B.

♣ Subformulas

- For any $A \in$ **WFF**, The set of ***subformulas*** of A is the smallest set S that contains A and contains, with each member, the immediate subformulas of that member. A is called an ***improper subformula*** of itself.

♣ Homework

- Try to develop an algorithm to generate the set of subformulas of a given formula of **CPC**.

An Introduction to Classical Propositional Calculus (CPC)

Principles of Structural Induction and Structural Recursion [Fitting]

♣ ***Principle of Structural Induction***

- Every formula in **WFF** has a property, Q, provided:
  Basis step: Every atomic formula has property Q;
  Induction steps: For any $A \in$ **WFF**, if A has property Q so does $(\neg A)$;
  For any $A, B \in$ **WFF**, if A and B have property Q so does $(A * B)$, where $*$ is a binary connective.

♣ ***Principle of Structural Recursion***

- There is one and only one function f defined on **WFF** such that:
  Basis step: The value of f is specified explicitly on atomic formulas;
  Recursion steps: For any $A \in$ **WFF**, the value of f on $(\neg A)$ is specified in terms of the value of f on A;
  For any $A, B \in$ **WFF**, the value of f on $(A * B)$ is specified in terms of the values of f on A and on B, where $*$ is a binary connective.

Structural Induction [Ben-Ari]

Structural induction is used to prove that a property holds for *all* formulas. This form of induction is similar to the familiar numerical induction that is used to prove that a property holds for all natural numbers (Appendix A.6). In numerical induction, the *base case* is to prove the property for 0 and then to prove the *inductive step*: assume that the property holds for arbitrary n and then show that it holds for $n + 1$. By Definition 2.10, a formula is either a leaf labeled by an atom or it is a tree with a principal operator and one or two subtrees. The base case of structural induction is to prove the property for a leaf and the inductive step is to prove the property for the formula obtained by applying the principal operator to the subtrees, assuming that the property holds for the subtrees.

Theorem 2.12 (Structural induction) *To show that a property holds for all formulas* $A \in \mathscr{F}$:

1. *Prove that the property holds all atoms* p.
2. *Assume that the property holds for a formula* A *and prove that the property holds for* $\neg A$.
3. *Assume that the property holds for formulas* A_1 *and* A_2 *and prove that the property holds for* A_1 *op* A_2, *for each of the binary operators.*

An Introduction to Classical Propositional Calculus (CPC)

♣ Formal (Object) Language (Syntax) of **CPC**
♣ Principles of Structural Induction and Structural Recursion
♣ Semantics (Model Theory) of **CPC**
♣ Semantic (Model-theoretical, Logical) Consequence Relation
♣ Normal Forms and Uniform Notation of Formulas
♣ Deduction System (Proof Theory) of **CPC**
♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
♣ Hilbert Style Formal Logic Systems for **CPC**
♣ Gentzen's Natural Deduction System for **CPC**
♣ Gentzen's Sequent Calculus System for **CPC**
♣ Semantic Tableau System for **CPC**
♣ Resolution System for **CPC**
♣ Forward Deduction and Backward Deduction

Semantics (Model Theory) of CPC

♣ The fundamental question
- Why is the ***semantics*** (***model theory***) of **CPC** indispensable?

♣ The answer to the question
- Well-formed formulas of any logic system have meaning only when an interpretation is given for the symbols of that logic.
- The semantics (model theory) of **CPC** gives a ***truth-value (truth-functional) interpretation*** for the symbols/well-formed formulas of **CPC**.
- The semantics (model theory) of a logic system provides a (philosophical and mathematical) fundamental basis for studying and using the logic.

Semantics (Model Theory) of CPC

♣ Important notes
- The semantics (model theory) of **CPC** is the most intrinsic foundation of **CPC**.
- Without a sound semantics, **CPC** is meaningless.
- The semantics (model theory) of **CPC** is only relative correct/sound/satisfactory, i.e., it is correct/sound/satisfactory only based on those fundamental assumptions/principles underlying **CML** (**C**lassical **M**athematical **L**ogic).
- The above answers to the fundamental question (and the above notes) are true/meaningful for not only **CPC** but also for all logic systems.

Fundamental Assumptions/Principles Underlying CML

♣ **The classical abstraction**
- The only properties of a proposition that matter to logic are its form and its truth-value.

♣ **The Fregean assumption / the principle of extensionality**
- The truth-value of a (composite) proposition depends only on its (composition) form and the truth-values of its constituents, not on their meaning.

♣ **The principle of bivalence**
- There are exactly two truth-values, "***TRUE***" and "***FALSE***". Every proposition has one or other, but not both, of these truth-values.

♣ **The classical account of validity (CAV)**
- An argument is valid if and only if it is impossible for all its premises to be true while its conclusion is false.

Truth-Functions

♣ ***Truth-values***

- We use meta-language symbols '**T**' and '**F**' to represent truth-values "***TRUE***" and "***FALSE***" respectively.

♣ Truth-functions

- A function f: $\{\mathbf{T}, \mathbf{F}\}^n \rightarrow \{\mathbf{T}, \mathbf{F}\}$ is called an ***n-place (n-ary) truth-function***.
- The number of different n-place (n-ary) truth-functions is the 2^nth power of two (2^{2^n}).
- Question: Which fundamental assumption?

♣ ***Zero-place (0-ary) truth-functions***

- $f_{0,1}$: logical constant $\top$
- $f_{0,2}$: logical constant $\bot$

Truth-Functions

♣ ***One-place (unary) truth-functions*** and their truth-tables

- $f_{1,1}$: universally true
- $f_{1,2}$: identity
- $f_{1,3}$: negation
- $f_{1,4}$: universally false

p	$f_{1,1}$	$f_{1,2}$	$f_{1,3}$	$f_{1,4}$
T	**T**	**T**	**F**	**F**
F	**T**	**F**	**T**	**F**

♣ ***Two-place (binary) truth-functions*** and their truth-tables

- $f_{2,1}$: universally true
- $f_{2,2}$: disjunction
- $f_{2,5}$: material implication
- $f_{2,7}$: equivalence
- $f_{2,8}$: conjunction
- $f_{2,9}$: alternative denial (nand)
- $f_{2,15}$: joint denial (nor)
- $f_{2,16}$: universally false

p	q	$f_{2,1}$	$f_{2,2}$	$f_{2,5}$	$f_{2,7}$	$f_{2,8}$	$f_{2,9}$	$f_{2,15}$	$f_{2,16}$
T	**T**	**T**	**T**	**T**	**T**	**T**	**F**	**F**	**F**
T	**F**	**T**	**T**	**F**	**F**	**F**	**T**	**F**	**F**
F	**T**	**T**	**T**	**T**	**F**	**F**	**T**	**F**	**F**
F	**F**	**T**	**F**	**T**	**T**	**F**	**T**	**T**	**F**
		T	$\vee$	$\rightarrow$	$\leftrightarrow$	$\wedge$	$\neg\wedge$	$\neg\vee$	**F**
							$\uparrow$	$\downarrow$	

Truth-Functions [Ben-Ari]

Since there are only two Boolean values T and F, the number of possible n-place operators is 2^{2^n}, because for each of the n arguments we can choose either of the two values T and F and for each of these 2^n n-tuples of arguments we can choose the value of the operator to be either T or F. We will restrict ourselves to one- and two-place operators.

The following table shows the $2^{2^1} = 4$ possible one-place operators, where the first column gives the value of the operand x and the other columns give the value of the nth operator $\circ_n(x)$:

x	$\circ_1$	$\circ_2$	$\circ_3$	$\circ_4$
T	T	T	F	F
F	T	F	T	F

x_1	x_2	$\circ_1$	$\circ_2$	$\circ_3$	$\circ_4$	$\circ_5$	$\circ_6$	$\circ_7$	$\circ_8$
T	T	T	T	T	T	T	T	T	T
T	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F

x_1	x_2	$\circ_9$	$\circ_{10}$	$\circ_{11}$	$\circ_{12}$	$\circ_{13}$	$\circ_{14}$	$\circ_{15}$	$\circ_{16}$
T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F

op	name	symbol	op	name	symbol
$\circ_2$	disjunction	$\vee$	$\circ_{15}$	nor	$\downarrow$
$\circ_8$	conjunction	$\wedge$	$\circ_9$	nand	$\uparrow$
$\circ_5$	implication	$\rightarrow$			
$\circ_7$	equivalence	$\leftrightarrow$	$\circ_{10}$	exclusive or	$\oplus$

Truth-Functions

♣ Expressively complete set of truth-functions

- ***Expressively (functionally) complete set of truth-functions*** is the set of truth-functions such that all truth-functions can be expressed by combining only the truth-functions belonging to this set.

♣ Expressively complete set of truth-functions

- ***Expressively (functionally) independent set of truth-functions*** is the set of truth-functions such that any truth-function of this set cannot be expressed by the combination of other truth-functions belonging to the same set.

♣ Primitive set of truth-functions

- ***Primitive set of truth-functions***: Both expressively complete and expressively independent set of truth-functions.
- $\{\neg, \wedge\}$ ($\{\uparrow\}$, alternative denial, incompatibility), $\{\neg, \vee\}$ ($\{\downarrow\}$, joint denial)
- $\{\bot, \leftarrow\}, \{\bot, \rightarrow\}, \{\top, \neg\rightarrow\}, \{\top, \neg\leftarrow\}$, $\{\neg, \vee\}, \{\neg, \leftarrow\}, \{\neg, \rightarrow\}, \{\neg, \wedge\}, \{\neg, \neg\rightarrow\}, \{\neg, \neg\leftarrow\}$

Truth-functional Connectives (Operations)

♣ ***Truth-functional connectives (operations)***

- A logical connective (operation) is said to be ***truth-functional*** if it is a truth-function.
- Fact: All connectives of **CPC** are truth-functional connectives (operations).

♣ Truth-table of connectives of **CPC** [Fitting]

		Primary								Secondary	
		$\wedge$	$\vee$	$\supset$	$\subset$	$\uparrow$	$\downarrow$	$\not\supset$	$\not\subset$	$\equiv$	$\not\equiv$
t	t	t	t	t	t	f	f	f	f	t	f
t	f	f	t	f	t	t	f	t	f	f	t
f	t	f	t	t	f	t	f	f	t	f	t
f	f	f	f	t	t	t	t	f	f	t	f

TABLE 2.1. Primary and Secondary Connectives

♣ Homework

- Try to prove that all the five connectives '$\neg$','$\rightarrow$','$\wedge$','$\vee$', and '$\leftrightarrow$' can be defined only by a single logical connective '$\uparrow$' ($\neg\wedge$, nand, alternative denial) or '$\downarrow$' ($\neg\vee$, nor, joint denial).

Material Implication as a Truth-functional Connective (Operation): Conditional [Mendelson]

Another important truth-functional operation is the *conditional*: "if A, then B." Ordinary usage is unclear here. Surely, "if A, then B" is false when the *antecedent* A is true and the *consequent* B is false. However, in other cases, there is no well-defined truth value. For example, the following sentences would be considered neither true nor false:

1. If $1 + 1 = 2$, then Paris is the capital of France.
2. If $1 + 1 \neq 2$, then Paris is the capital of France.
3. If $1 + 1 \neq 2$, then Rome is the capital of France.

Their meaning is unclear, since we are accustomed to the assertion of some sort of relationship (usually causal) between the antecedent and the consequent. We shall make the convention that "if A, then B" is false when and only when A is true and B is false. Thus, sentences 1–3 are assumed to be true. Let us denote "if A, then B" by "$A \Rightarrow B$." An expression "$A \Rightarrow B$" is called a *conditional*. Then $\Rightarrow$ has the following truth table:

A	B	$A \Rightarrow B$
T	T	T
F	T	T
T	F	F
F	F	T

This sharpening of the meaning of "if A, then B" involves no conflict with ordinary usage, but rather only an extension of that usage.*

Truth-Table of Formulas: An Example [Mendelson]

For every assignment of truth values T or F to the statement letters that occur in a statement form, there corresponds, by virtue of the truth tables for the propositional connectives, a truth value for the statement form. Thus, each statement form determines a *truth function*, which can be graphically represented by a truth table for the statement form. For example, the statement form $(((\neg A) \vee B) \Rightarrow C)$ has the following truth table:

A	B	C	$(\neg A)$	$((\neg A) \vee B)$	$(((\neg A) \vee B) \Rightarrow C)$
T	T	T	F	T	T
F	T	T	T	T	T
T	F	T	F	F	T
F	F	T	T	T	T
T	T	F	F	T	F
F	T	F	T	T	F
T	F	F	F	F	T
F	F	F	T	T	F

Each row represents an assignment of truth values to the statement letters A, B, and C and the corresponding truth values assumed by the statement forms that appear in the construction of $(((\neg A) \vee B) \Rightarrow C)$.

Truth-Table of Formulas: An Example [Mendelson]

The truth table for $((A \Leftrightarrow B) \Rightarrow ((\neg A) \wedge B))$ is as follows:

A	B	$(A \Leftrightarrow B)$	$(\neg A)$	$((\neg A) \wedge B)$	$((A \Leftrightarrow B) \Rightarrow ((\neg A) \wedge B))$
T	T	T	F	F	F
F	T	F	T	T	T
T	F	F	F	F	T
F	F	T	T	F	F

If there are n distinct letters in a statement form, then there are 2^n possible assignments of truth values to the statement letters and, hence, 2^n rows in the truth table.

Truth-Table of Formulas: An Example [Mendelson]

A truth table can be abbreviated by writing only the full statement form, putting the truth values of the statement letters underneath all occurrences of these letters, and writing, step by step, the truth values of each component statement form under the principal connective of the form.* As an example, for $((A \Leftrightarrow B) \Rightarrow ((\neg A) \wedge B))$, we obtain

$((A$	$\Leftrightarrow$	$B)$	$\Rightarrow$	$((\neg A)$	$\wedge$	$B))$
T	T	T	F	FT	F	T
F	F	T	T	TF	T	T
T	F	F	T	FT	F	F
F	T	F	F	TF	F	F

Semantics (Model Theory) of CPC: Models for CPC

♣ Models for **CPC**

- A ***model*** for **CPC** is an ordered pair $\boldsymbol{M} = (v_a, v_f)$ such that v_a, called a ***truth assignment***, is a function $v_a : \mathbf{V} \rightarrow \{\mathbf{T}, \mathbf{F}\}$, and v_f, called a ***truth valuation***, is a surjective function $v_f : \mathbf{WFF} \rightarrow \{\mathbf{T}, \mathbf{F}\}$ defined as ($A, B \in \mathbf{WFF}$):

(1) for $\top$ and $\bot$, $v_f(\top) = \mathbf{T}$ and $v_f(\bot) = \mathbf{F}$;
(2) $v_f(A) = v_a(A)$ if $A \in \mathbf{V}$;
(3) $v_f(\neg A) = \mathbf{F}$ if $v_f(A) = \mathbf{T}$, and $v_f(\neg A) = \mathbf{T}$ if $v_f(A) = \mathbf{F}$;
(4) $v_f(A \rightarrow B) = \mathbf{F}$ if $v_f(A) = \mathbf{T}$ and $v_f(B) = \mathbf{F}$, and $v_f(A \rightarrow B) = \mathbf{T}$ otherwise;
(5) $v_f(A \wedge B) = \mathbf{T}$ if both $v_f(A) = \mathbf{T}$ and $v_f(B) = \mathbf{T}$, and $v_f(A \wedge B) = \mathbf{F}$ otherwise;
(6) $v_f(A \vee B) = \mathbf{T}$ if $v_f(A) = \mathbf{T}$, $v_f(B) = \mathbf{T}$ or both, and $v_f(A \vee B) = \mathbf{F}$ otherwise;
(7) $v_f(A \leftrightarrow B) = \mathbf{T}$ if $v_f(A) = v_f(B)$, and $v_f(A \leftrightarrow B) = \mathbf{F}$ otherwise.

♣ Notes

- We use meta-language symbols '**T**' and '**F**' to represent truth-values "***TRUE***" and "***FALSE***" respectively.
- There are infinite models for **CPC** (Why?).

Semantics (Model Theory) of CPC: Truth-value of a Formula

♣ Truth-value of a formula in a model

- For any model $\boldsymbol{M} = (v_a, v_f)$ and any $A \in \mathbf{WFF}$, $v_f(A)$ is called the ***truth-value*** of A in $\boldsymbol{M}$.

♣ Logical equivalence

- Let $A, B \in \mathbf{WFF}$. If $v_f(A) = v_f(B)$ for any model $\boldsymbol{M} = (v_a, v_f)$, then A is said to be ***logically equivalent*** to B, denoted $A \equiv B$.
- Theorem: $A \equiv B$ IFF $v_f(A \leftrightarrow B) = \mathbf{T}$ for any model $\boldsymbol{M} = (v_a, v_f)$.
- Note: "$\equiv$" is a meta-language symbol.

♣ Replacement theorem

- **Replacement theorem**: Let $F, A, B \in \mathbf{WFF}$, and $F(p)$ mean that p ($p \in \mathbf{V}$) appears in F. If $v_f(A) = v_f(B)$, then $v_f(F(A)) = v_f(F(B))$, where $F(A)$ is the result of substituting A uniformly for p in F, i.e., A replaces every occurrence of p in F.

Logical Equivalence between Formulas [Mendelson]

a.	$A \Rightarrow (B \Rightarrow C)$	$(A \wedge B) \Rightarrow C$	
b.	$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$	(Distributive law)
c.	$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$	(Distributive law)
d.	$(A \wedge B) \vee \neg B$	$A \vee \neg B$	
e.	$(A \vee B) \wedge \neg B$	$A \wedge \neg B$	
f.	$A \Rightarrow B$	$\neg B \Rightarrow \neg A$	(Law of the contrapositive)
g.	$A \Leftrightarrow B$	$B \Leftrightarrow A$	(Biconditional commutativity)
h.	$(A \Leftrightarrow B) \Leftrightarrow C$	$A \Leftrightarrow (B \Leftrightarrow C)$	(Biconditional associativity)
i.	$A \Leftrightarrow B$	$(A \wedge B) \vee (\neg A \wedge \neg B)$	
j.	$\neg(A \Leftrightarrow B)$	$A \Leftrightarrow \neg B$	
k.	$\neg(A \vee B)$	$(\neg A) \wedge (\neg B)$	(De Morgan's law)
l.	$\neg(A \wedge B)$	$(\neg A) \vee (\neg B)$	(De Morgan's law)
m.	$A \vee (A \wedge B)$	A	
n.	$A \wedge (A \vee B)$	A	
o.	$A \wedge B$	$B \wedge A$	(Commutativity of conjunction)
p.	$A \vee B$	$B \vee A$	(Commutativity of disjunction)
q.	$(A \wedge B) \wedge C$	$A \wedge (B \wedge C)$	(Associativity of conjunction)
r.	$(A \vee B) \vee C$	$A \vee (B \vee C)$	(Associativity of disjunction)
s.	$A \oplus B$	$B \oplus A$	(Commutativity of exclusive "or")
t.	$A \oplus B) \oplus C$	$A \oplus (B \oplus C)$	(Associativity of exclusive "or")
u.	$A \wedge (B \oplus C)$	$(A \wedge B) \oplus (A \wedge C)$	(Distributive law)

Logical Equivalence between Formulas [Rautenberg]

all formulas α, β, γ the following equivalences hold:

$$\begin{array}{rcl|rcl|l}
\alpha\wedge(\beta\wedge\gamma) & \equiv & \alpha\wedge\beta\wedge\gamma, & \alpha\vee(\beta\vee\gamma) & \equiv & \alpha\vee\beta\vee\gamma & \text{(associativity);}\\
\alpha\wedge\beta & \equiv & \beta\wedge\alpha, & \alpha\vee\beta & \equiv & \beta\vee\alpha & \text{(commutativity);}\\
\alpha\wedge\alpha & \equiv & \alpha, & \alpha\vee\alpha & \equiv & \alpha & \text{(idempotency);}\\
\alpha\wedge(\alpha\vee\beta) & \equiv & \alpha, & \alpha\vee\alpha\wedge\beta & \equiv & \alpha & \text{(absorption);}\\
\alpha\wedge(\beta\vee\gamma) & \equiv & \alpha\wedge\beta\vee\alpha\wedge\gamma, & & & & (\wedge\text{-distributivity});\\
\alpha\vee\beta\wedge\gamma & \equiv & (\alpha\vee\beta)\wedge(\alpha\vee\gamma) & & & & (\vee\text{-distributivity});\\
\neg(\alpha\wedge\beta) & \equiv & \neg\alpha\vee\neg\beta, & \neg(\alpha\vee\beta) & \equiv & \neg\alpha\wedge\neg\beta & \text{(de Morgan rules).}
\end{array}$$

Furthermore, $\alpha\vee\neg\alpha\equiv\top$, $\alpha\wedge\neg\alpha\equiv\bot$, and $\alpha\wedge\top\equiv\alpha\vee\bot\equiv\alpha$. It is also useful to list certain equivalences for formulas containing $\rightarrow$, for example the frequently used $\alpha\rightarrow\beta\equiv\neg\alpha\vee\beta$ ($\equiv\neg(\alpha\wedge\neg\beta)$), and the important

$$\alpha\rightarrow\beta\rightarrow\gamma\equiv\alpha\wedge\beta\rightarrow\gamma\equiv\beta\rightarrow\alpha\rightarrow\gamma.$$

To generalize: $\alpha_1\rightarrow\cdots\rightarrow\alpha_n\equiv\alpha_1\wedge\cdots\wedge\alpha_{n-1}\rightarrow\alpha_n$. Further, we mention the "left distributivity" of implication with respect to $\wedge$ and $\vee$, namely

$$\alpha\rightarrow\beta\wedge\gamma\equiv(\alpha\rightarrow\beta)\wedge(\alpha\rightarrow\gamma);\quad \alpha\rightarrow\beta\vee\gamma\equiv(\alpha\rightarrow\beta)\vee(\alpha\rightarrow\gamma).$$

Should the symbol $\rightarrow$ lie to the right then the following are valid:

$$\alpha\wedge\beta\rightarrow\gamma\equiv(\alpha\rightarrow\gamma)\vee(\beta\rightarrow\gamma);\quad \alpha\vee\beta\rightarrow\gamma\equiv(\alpha\rightarrow\gamma)\wedge(\beta\rightarrow\gamma).$$

Semantics (Model Theory) of CPC: Satisfiability and Validity of a Formula

♣ ***Satisfiability*** of a formula

- For any model $\boldsymbol{M}=(v_a, v_f)$ and any $A\in$ **WFF**, $\boldsymbol{M}$ ***satisfies*** A or A is ***true*** in $\boldsymbol{M}$, written as $\models_{\boldsymbol{M}} A$, IFF $v_f(A)=\mathbf{T}$; $\boldsymbol{M}$ ***does not satisfy*** A or A is ***false*** in $\boldsymbol{M}$, written as $\not\models_{\boldsymbol{M}} A$, IFF $v_f(A)=\mathbf{F}$.
- For any $A\in$ **WFF**, A is ***satisfiable*** IFF there is some model $\boldsymbol{M}$ such that $\models_{\boldsymbol{M}} A$; A is ***unsatisfiable*** IFF $\not\models_{\boldsymbol{M}} A$ for any model $\boldsymbol{M}$ (Ex.: $(A\wedge\neg A)$).
- Note: '$\models$ ($\not\models$)' is a meta-language(logic) symbol but not an object language symbol of **CPC**.

♣ ***Logical validity*** of a formula (***logical theorem***)

- For any $A\in$ **WFF**, A is ***logically valid*** IFF $\models_{\boldsymbol{M}} A$ for any model $\boldsymbol{M}$ of **CPC**.
- Ex.: $(A\vee\neg A)$
- Decidability theorem for **CPC**: The validity problem for **CPC**, i.e., whether a formula of **CPC** is valid or not, is decidable. (Note: The decidability of **CPC** is an intrinsically important property of **CPC**.)

Semantics (Model Theory) of CPC: Satisfiability and Validity of a Formula

♣ Satisfiability and validity of formulas [Ben-Ari]

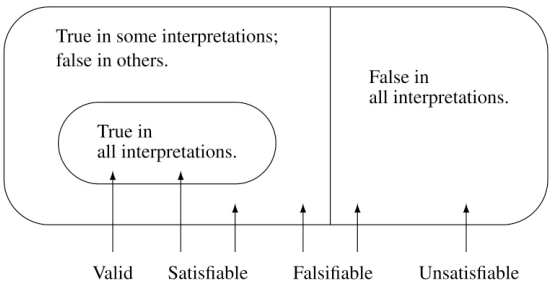


Fig. 2.6 Satisfiability and validity of formulas

Semantics (Model Theory) of CPC: Tautologies, Contradictions, and Contingencies

♣ Tautologies, contradictions, and contingencies

- A formula $A\in$ **WFF** is a ***tautology*** (***logical theorem***) of **CPC**, written as $\models_{\mathbf{CPC}} A$, IFF $\models_{\boldsymbol{M}} A$ for any model $\boldsymbol{M}$ of **CPC**, i.e., A is logically valid;
  A formula $A\in$ **WFF** is a ***contradiction*** of **CPC**, written as $\not\models_{\mathbf{CPC}} A$, IFF $\not\models_{\boldsymbol{M}} A$ for any model $\boldsymbol{M}$ of **CPC**;
  A formula is a ***contingency*** IFF it is neither a tautology nor a contradiction.
- The set of all tautologies (logical theorems) of **CPC** is denoted by **Th(CPC)**.
- For any $A\in$ **WFF**, NOT both $\models_{\mathbf{CPC}} A$ ($A\in$ **Th(CPC)**) and $\models_{\mathbf{CPC}} \neg A$ ($\neg A\in$ **Th(CPC)**), i.e., **CPC** is ***consistent***.

♣ Notes

- A formula must be any one of tautology, contradiction, and contingency.
- The form of a tautology is sufficient to determining its validity.

Semantics (Model Theory) of CPC: Tautologies, Contradictions, and Contingencies

♣ Relationship between tautologies (logical theorems) and contradictions

- **Theorem**: For any $A\in$ **WFF**, A is a tautology (logical theorem) IFF $(\neg A)$ is a contradiction, and A is a contradiction IFF $(\neg A)$ is a tautology (logical theorem). (There is a bijection between tautologies (logical theorems) and contradictions of **CPC**.)

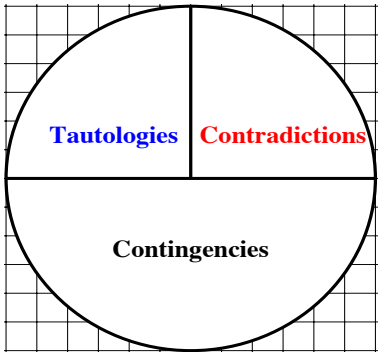
♣ Replacement theorem for tautologies (logical theorems)

- **Replacement theorem**: Let $F, A, B\in$ **WFF**, and $F(p)$ mean that p ($p\in$ **V**) appears in F. If $(A\leftrightarrow B)$ is a tautology (logical theorem), then so is $(F(A)\leftrightarrow F(B))$, where $F(A)$ is the result of substituting A uniformly for p in F, i.e., A replaces every occurrence of p in F.

♣ The semantic validity of Modus Ponens (MP) for material implication

- **Theorem**: Let $A, B\in$ **WFF**. If $(A\rightarrow B)$ and A are tautologies (logical theorems), then so is B.

Formulas of CPC: Tautologies, Contradictions, and Contingencies



Some Tautologies of CPC in Material Implication [Rautenberg]

$p \to p$ (self-implication),
$(p \to q) \to (q \to r) \to (p \to r)$ (chain rule),
$(p \to q \to r) \to (q \to p \to r)$ (exchange of premises),
$p \to q \to p$ (premise charge),
$(p \to q \to r) \to (p \to q) \to (p \to r)$ (Frege's formula),
$((p \to q) \to p) \to p$ (Peirce's formula).

- Homework: Prove the above tautologies (logical theorems).

Some Tautologies (Logical Theorems) of CPC

♣ Paradoxes of material implication
- $A \to (B \to A)$, $B \to (\neg A \vee A)$
- $\neg A \to (A \to B)$, $(\neg A \wedge A) \to B$
- $(A \to B) \vee (\neg A \to B)$, $(A \to B) \vee (A \to \neg B)$
- $(A \to B) \vee (B \to A)$
- $((A \wedge B) \to C) \to ((A \to C) \vee (B \to C))$
- :

♣ Paradoxes of relevant implication
- $(A \wedge B) \to A$, $(A \wedge B) \to B$
- $A \to A \vee B$, $B \to A \vee B$
- $(A \to B) \to ((A \wedge C) \to B)$
- $(A \to B) \to (A \to (B \vee C))$
- :

♣ Homework
- Prove the above tautologies (logical theorems).

Some Tautologies (Logical Theorems) of CPC

- $A \to (B \to B)$, $A \to (B \to (B \to A))$, $((B \to C) \to (A \to B)) \to (A \to B)$

- $(A \leftrightarrow B) \leftrightarrow (B \leftrightarrow A)$ {Commutativity}
- $(A \leftrightarrow (B \leftrightarrow C)) \leftrightarrow ((A \leftrightarrow B) \leftrightarrow C)$ {Associativity}
- $(A \leftrightarrow (A \leftrightarrow B)) \to B$

- Homework: Prove the above tautologies (logical theorems).

Some Tautologies (Logical Theorems) of CPC

- $A \leftrightarrow (A \wedge A)$, $(A \vee A) \leftrightarrow A$ {Idempotent law}
- $(A \wedge B) \leftrightarrow (B \wedge A)$, $(A \vee B) \leftrightarrow (B \vee A)$ {Commutativity}
- $(A \wedge (B \wedge C)) \leftrightarrow ((A \wedge B) \wedge C)$, $(A \vee (B \vee C)) \leftrightarrow ((A \vee B) \vee C)$ {Associativity}
- $(A \wedge (B \vee C)) \leftrightarrow ((A \wedge B) \vee (A \wedge C))$, $(A \vee (B \wedge C)) \leftrightarrow ((A \vee B) \wedge (A \vee C))$ {Distributive law}
- $(A \vee (A \wedge B)) \leftrightarrow A$, $(A \wedge (A \vee B)) \leftrightarrow A$ {Absorption law}
- $(A \to (B \to C)) \leftrightarrow ((A \wedge B) \to C)$ {Exportation} , $(A \wedge (A \to B)) \to B$, $A \to (B \to (A \wedge B))$
- $(A \wedge B) \to (A \vee C)$, $(A \wedge B) \to (B \vee C)$, $(A \to B) \to ((A \vee C) \to (B \vee C))$
- $(A \to B) \leftrightarrow (A \leftrightarrow (A \wedge B))$, $(A \wedge B) \leftrightarrow (A \leftrightarrow (A \to B))$, $(A \vee B) \leftrightarrow (B \leftrightarrow (A \to B))$
- $(A \wedge B) \leftrightarrow ((A \vee B) \leftrightarrow (A \to B))$, $(A \vee B) \leftrightarrow ((A \wedge B) \leftrightarrow (A \to B))$, $(A \wedge (B \to (C \vee D))) \to (A \vee B)$
- $((A \to B) \wedge (B \to C)) \to (A \to C)$, $((A \to B) \wedge (B \to C)) \to (A \to (B \wedge C))$
- $(A \to C) \to ((B \to C) \to ((A \vee B) \to C))$, $((A \to C) \wedge ((B \to C)) \to ((A \vee B) \to C))$
- $((A \to C) \wedge (B \wedge C)) \to ((A \vee B) \to C))$, $(((A \wedge B) \to C) \wedge (A \to (B \vee C))) \to (A \to C)$

- Homework: Prove the above tautologies (logical theorems).

Some Tautologies (Logical Theorems) of CPC

- $(A \to B) \leftrightarrow (\neg B \to \neg A)$ {Contraposition}
- $(\neg\neg A) \to A$, $A \to (\neg\neg A)$, $(\neg\neg A) \leftrightarrow A$ {Double negation}
- $(A \leftrightarrow B) \leftrightarrow (\neg A \leftrightarrow \neg B)$, $(\neg(A \leftrightarrow B)) \leftrightarrow (A \leftrightarrow \neg B)$
- $(\neg A) \to (A \to B)$, $(\neg A \to A) \to A$, $(\neg(A \to B)) \to A$, $(\neg(A \to B)) \to (\neg B)$
- $A \to (\neg B \to \neg(A \to B))$, $(A \to B) \to ((\neg A \to B) \to B)$, $(\neg A \to B) \to ((A \to B) \to B)$
- $(\neg A \to \neg B) \to ((\neg A \to B) \to A)$

- Homework: Prove the above tautologies (logical theorems).

Some Tautologies (Logical Theorems) of CPC

- $(\neg(A \wedge B)) \leftrightarrow ((\neg A) \vee (\neg B))$, $(\neg(A \vee B)) \leftrightarrow ((\neg A) \wedge (\neg B))$ {De Morgan's law}
- $(A \to B) \leftrightarrow ((\neg A) \vee B)$, $(A \to B) \leftrightarrow \neg(A \wedge (\neg B))$, $(\neg(A \to B)) \leftrightarrow (A \wedge (\neg B))$
- $(\neg A) \to (\neg(A \wedge B))$, $(\neg B) \to (\neg(A \wedge B))$, $(A \wedge B) \leftrightarrow \neg((\neg A) \vee (\neg B))$
- $(\neg(A \vee B)) \to (\neg A)$, $(\neg(A \vee B)) \to (\neg B)$, $(A \vee B)) \leftrightarrow \neg((\neg A) \wedge (\neg B))$
- $(A \wedge B) \leftrightarrow (\neg(A \to (\neg B)))$, $(A \vee B) \leftrightarrow ((\neg A) \to B)$
- $(A \leftrightarrow (A \wedge B)) \leftrightarrow (B \leftrightarrow (A \vee B))$, $((A \to B) \wedge (B \to A)) \leftrightarrow ((A \wedge B) \vee ((\neg A) \wedge (\neg B)))$
- $(A \leftrightarrow \neg B) \to (A \vee B)$, $((A \to B) \wedge (A \to \neg B)) \to (\neg B)$, $(A \leftrightarrow ((\neg B) \vee C)) \to (\neg A \to B)$
- $(A \to B) \to (\neg(B \vee C) \to \neg(A \vee C))$
- $(A \leftrightarrow B) \leftrightarrow ((A \wedge B) \vee ((\neg A) \wedge (\neg B)))$
- $((A \wedge B) \vee (\neg B)) \leftrightarrow (A \vee (\neg B))$, $((A \vee B) \wedge (\neg B)) \leftrightarrow (A \wedge (\neg B))$

- Homework: Prove the above tautologies (logical theorems).

Semantics (Model Theory) of CPC: Models of Formulas

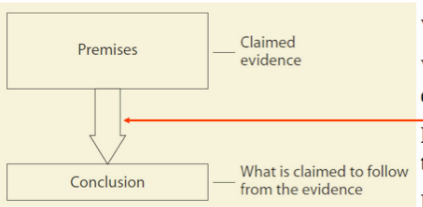
♣ Models of formulas
- For any $\Gamma \subseteq$ **WFF**, model $\boldsymbol{M}$ is called a ***model*** of Γ IFF $\models_M A$ for any $A \in \Gamma$.
- The set of all models of Γ is denoted by $M(\Gamma)$.
- $\boldsymbol{M}(\Delta) \subseteq \boldsymbol{M}(\Gamma)$, if $\Gamma \subseteq \Delta$.

♣ ***Consistence (Satisfiability)*** of a set of formulas
- For any $\Gamma \subseteq$ **WFF**, Γ is ***semantically (model-theoretically, logically) consistent (satisfiable)*** IFF it has at least one model; Γ is ***semantically (model-theoretically, logically) inconsistent (unsatisfiable)*** IFF it has no model.
- Ex.: $\{A, \neg A, \ldots\}$ is semantically (model-theoretically, logically) inconsistent.

An Introduction to Classical Propositional Calculus (CPC)

♣ Formal (Object) Language (Syntax) of **CPC**
♣ Principles of Structural Induction and Structural Recursion
♣ Semantics (Model Theory) of **CPC**
♣ Semantic (Model-theoretical, Logical) Consequence Relation
♣ Normal Forms and Uniform Notation of Formulas
♣ Deduction System (Proof Theory) of **CPC**
♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
♣ Hilbert Style Formal Logic Systems for **CPC**
♣ Gentzen's Natural Deduction System for **CPC**
♣ Gentzen's Sequent Calculus System for **CPC**
♣ Semantic Tableau System for **CPC**
♣ Resolution System for **CPC**
♣ Forward Deduction and Backward Deduction

Logic: What Is It?



[From P. J. Hurley, "A Concise Introduction to Logic"]

What entails what?

What follows from what?

Why? What are the evaluation criteria?

How to establish/define the evaluation criteria?

How to evaluate arguments/reasoning?

It is LOGIC to answer these fundamental questions.

Semantic (Model-theoretical, Logical) Consequence Relation

♣ Semantic (Model-theoretical, Logical) consequence relation
- For any $\Gamma \subseteq$ **WFF** and any $A \in$ **WFF**,
  Γ ***semantically (model-theoretically, logically) entails*** A, or
  A ***semantically (model-theoretically, logically) follows from*** Γ, or
  A is a ***semantic (model-theoretical, logical) consequence*** of Γ,
  written as $\Gamma \models_{\mathbf{CPC}} A$, IFF $\models_M A$ for any model $\boldsymbol{M}$ of Γ.
- $\phi \models_{\mathbf{CPC}} A =_{df} \models_{\mathbf{CPC}} A$ and it means that $\models_M A$ for any model $\boldsymbol{M}$ of **CPC**, i.e., A is a tautology (logical theorem) of **CPC**, $A \in$ **Th**(**CPC**).

♣ All semantic (model-theoretical, logical) consequences of premises
- The set of all semantic (model-theoretical, logical) consequences of Γ is denoted by $C_{sem}(\Gamma)$ (Note: A **CPC**-theory with Γ).

♣ Note
- The semantic (model-theoretical, logical) consequence relation of **CPC** is a semantic (model-theoretical) formalization of the notion that one proposition "***logically follows from***" another or others.

Semantic (Model-theoretical, Logical) Consequence Relation: Examples

- Using truth-table to determine whether semantic consequence relations hold.

A	B	C	$A\to B$	$B\to C$	$A\to C$	$\neg A$	$\neg B$	$\neg C$	$A\wedge B$	$A\vee B$	$B\vee C$	$A\vee C$
T	T	T	T	T	T	F	F	F	T	T	T	T
T	T	F	T	F	F	F	F	T	T	T	T	T
T	F	T	F	T	T	F	T	F	F	T	T	T
T	F	F	F	T	F	F	T	T	F	T	F	T
F	T	T	T	T	T	T	F	F	F	T	T	T
F	T	F	T	F	T	T	F	T	F	T	T	F
F	F	T	T	T	T	T	T	F	F	F	T	T
F	F	F	T	T	T	T	T	T	F	F	F	F

- $\{ A\to B , A \} \models_{\mathbf{CPC}} B$, because when both $A\to B$ and A are true, B is also true.
- Similarly, $\{ A\to B , \neg B \} \models_{\mathbf{CPC}} \neg A$, $\{ A\to B , B\to C \} \models_{\mathbf{CPC}} A\to C$, ……
- $\{ A\to B , B \} \not\models_{\mathbf{CPC}} A$, because when both $A\to B$ and B are true, A may be false.
- Similarly, $\{ A\to B , \neg A \} \not\models_{\mathbf{CPC}} \neg B$, $\{ A\vee B , B\vee C \} \not\models_{\mathbf{CPC}} A\vee C$, ……
- Homework: Using the above truth-table to determine more examples.

Semantic (Model-theoretical, Logical) Equivalence Relation

♣ Semantic (model-theoretical, logical) equivalence relation
- For any $A, B \in$ **WFF**, A is ***semantically (model-theoretically, logically) equivalent*** to B in **CPC** IFF both $\{A\} \models_{\mathbf{CPC}} B$ and $\{B\} \models_{\mathbf{CPC}} A$.
- **Theorem**: A is semantically (model-theoretically, logically) equivalent to B IFF $(A\leftrightarrow B)$ is a tautology.

Properties of the Semantic (Model-theoretical, Logical) Consequence Relation

♣ Properties of the semantic consequence relation

- If $\models_{\mathbf{CPC}} A$, then $\Gamma \models_{\mathbf{CPC}} A$. (A tautology is a consequence of any premise; **Th(CPC)** is a proper subset of $\mathbf{C}_{sem}(\Gamma)$ for any Γ.) (Note: A **CPC**-theory with Γ)
- $\{A\} \models_{\mathbf{CPC}} A$.
- If $A \in \Gamma$, then $\Gamma \models_{\mathbf{CPC}} A$. (***Reflexivity***)
- $\Gamma \subseteq \mathbf{C}_{sem}(\Gamma)$.
- $\mathbf{C}_{sem}(\Gamma) = \mathbf{C}_{sem}(\mathbf{C}_{sem}(\Gamma))$.
- If $\Gamma \models_{\mathbf{CPC}} A$ and $\Gamma \subseteq \Delta$, then $\Delta \models_{\mathbf{CPC}} A$. (***Monotonicity***)
- $\mathbf{C}_{sem}(\Gamma) \subseteq \mathbf{C}_{sem}(\Delta)$, if $\Gamma \subseteq \Delta$. (***Monotonicity***)
- The monotonicity is an intrinsically important property of **CPC**.

♣ Homework

- Try to prove the above properties of the semantic (model-theoretical, logical) consequence relation according to definition.

Properties of the Semantic (Model-theoretical, Logical) Consequence Relation

♣ Properties of the semantic consequence relation

- ***Transitivity*** (***the Cut rule***):
  If $\Gamma \models_{\mathbf{CPC}} A$ and $\Delta \cup \{A\} \models_{\mathbf{CPC}} B$, then $\Gamma \cup \Delta \models_{\mathbf{CPC}} B$.
- If $\Gamma \cup \{A_1, ..., A_n\} \models_{\mathbf{CPC}} B$ and $\Gamma \models_{\mathbf{CPC}} A_i$ for $i = 1, ..., n$, then $\Gamma \models_{\mathbf{CPC}} B$.
- ***Substitution***: If $\models_{\mathbf{CPC}} F(p)$, then $\models_{\mathbf{CPC}} F(A)$, where $F(p)$ means that p ($p \in \mathbf{V}$) appears in F ($F \in \mathbf{WFF}$) and $F(A)$ is the result of substituting A uniformly for p in F, i.e., A replaces every occurrence of p in F.
- ***Compactness (Finiteness)***: $\Gamma \models_{\mathbf{CPC}} A$ IFF there is a finite set $\Delta \subseteq \Gamma$ such that $\Delta \models_{\mathbf{CPC}} A$; Γ has a model IFF every finite subset of Γ has a model.

♣ Homework

- Try to prove the above properties of the semantic (model-theoretical, logical) consequence relation according to definition.

Semantic Deduction Theorems

♣ Semantic deduction theorems

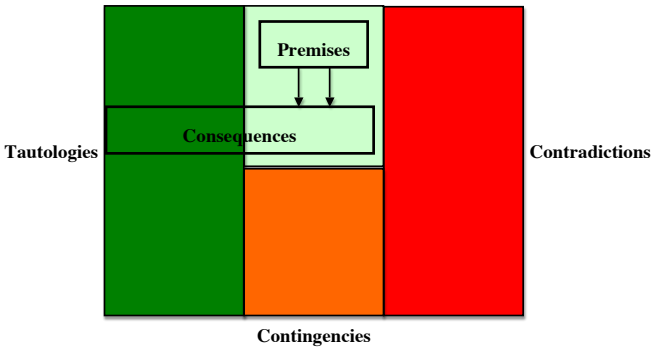
- ***Semantic (model-theoretical, logical) deduction theorem for CPC***: For any $A, B \in \mathbf{WFF}$ and any $\Gamma \subseteq \mathbf{WFF}$,
  $\Gamma \cup \{A\} \models_{\mathbf{CPC}} B$ IFF $\Gamma \models_{\mathbf{CPC}} (A \rightarrow B)$;
  $\{A\} \models_{\mathbf{CPC}} B$ IFF $\models_{\mathbf{CPC}} (A \rightarrow B)$.
- ***Semantic (model-theoretical, logical) deduction theorem for CPC for finite consequences***: For any $A_1, ..., A_{n-1}, A_n, B \in \mathbf{WFF}$ and any $\Gamma \subseteq \mathbf{WFF}$,
  $\Gamma \cup \{A_1, ..., A_{n-1}, A_n\} \models_{\mathbf{CPC}} B$ IFF $\Gamma \models_{\mathbf{CPC}} (A_1 \rightarrow (...(A_{n-1} \rightarrow (A_n \rightarrow B))...))$;
  $\Gamma \cup \{A_1, ..., A_{n-1}, A_n\} \models_{\mathbf{CPC}} B$ IFF $\Gamma \models_{\mathbf{CPC}} ((A_1 \wedge (...(A_{n-1} \wedge A_n)...)) \rightarrow B)$.
- The semantic deduction theorems are intrinsically important meta-theorems of **CPC**.

Semantic Deduction Theorems

♣ Notes

- As a special case of the above deduction theorems, $\{A\} \models_{\mathbf{CPC}} B$ IFF $\models_{\mathbf{CPC}} (A \rightarrow B)$, i.e., A semantically (model-theoretically, logically) entails B IFF $(A \rightarrow B)$ is a tautology.
- In the framework of **CPC**, the semantic (model-theoretical, logical) consequence relation, which is a representation of the notion of entailment in the sense of meta-logic, is "equivalent" to the notion of material implication (denoted by '→' in **CPC**).
- However, in semantics, the notion of material implication is NOT an accurate representation of the notion of entailment.

Formulas of CPC: Tautologies, Contradictions, Contingencies, Premises, and Consequences



An Introduction to Classical Propositional Calculus (CPC)

♣ Formal (Object) Language (Syntax) of **CPC**
♣ Principles of Structural Induction and Structural Recursion
♣ Semantics (Model Theory) of **CPC**
♣ Semantic (Model-theoretical, Logical) Consequence Relation
♣ Normal Forms and Uniform Notation of Formulas
♣ Deduction System (Proof Theory) of **CPC**
♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
♣ Hilbert Style Formal Logic Systems for **CPC**
♣ Gentzen's Natural Deduction System for **CPC**
♣ Gentzen's Sequent Calculus System for **CPC**
♣ Semantic Tableau System for **CPC**
♣ Resolution System for **CPC**
♣ Forward Deduction and Backward Deduction

Normal Forms of Formulas: Literal and Clause

♣ Generalized disjunction and conjunction

Let $X_1, X_2, ..., X_n$ be a list of formulas.

- ***Generalized disjunction***: $[X_1, X_2, ..., X_n] =_{df} X_1 \vee X_2 \vee ... \vee X_n$.
- ***Generalized conjunction***: $\langle X_1, X_2, ..., X_n \rangle =_{df} X_1 \wedge X_2 \wedge ... \wedge X_n$.

♣ Literal

- A formula is called a ***literal***, if it is a propositional variable or the negation of a propositional variable, or a constant, $\top$ or $\bot$.

♣ Clause

- A ***clause*** is a generalized disjunction $[X_1, X_2, ..., X_n]$ in which each member is a literal.
- A ***dual clause*** is a generalized conjunction $\langle X_1, X_2, ..., X_n \rangle$ in which each member is a literal.

Normal Forms of Formulas: CNF and DNF

♣ Conjunctive normal form

- A formula is called a ***formula in conjunctive normal form*** or a ***formula in clause form*** or a ***clause set***, if it is a generalized conjunction $\langle C_1, C_2, ..., C_n \rangle$ in which each member is a clause.
- A ***conjunctive normal form*** (***CNF***) for a formula A is a formula B in conjunctive normal form such that B contains exactly the same propositional variables in A and is semantically equivalent to A.

♣ Disjunctive normal form

- A formula is called a ***formula in disjunctive normal form*** or a ***formula in dual clause form*** or a ***dual clause set***, it is a generalized disjunction $[D_1, D_2, ..., D_n]$ in which each member is a dual clause.
- A ***disjunctive normal form*** (***DNF***) for a formula A is a formula B in disjunctive normal form such that B contains exactly the same propositional variables in A and is semantically equivalent to A.

Normal Forms of Formulas: The Normal Form Theorems

♣ ***The normal form theorem for CPC***

- Theorem: There are algorithms for converting any ordinary formula into its conjunctive normal form (CNF) and its disjunctive normal form (DNF).

♣ Homework

- Try to develop the algorithms mentioned in the above theorem.

♣ ***The normal form representation theorem for CPC***

- Theorem: Every truth-function can be represented by a formula in CNF or a formula in DNF.

♣ The important fact based on the above two theorems

- It is enough in **CPC** that we only deal with formulas in CNF/DNF.

Uniform Notation of Formulas

♣ Uniform notation of formulas [R. M. Smullyan, 1968]

- Classify all formulas of the forms $(A*B)$ and $(\neg(A*B))$, where $*$ is a binary connective, into two categories, i.e., ***α-formulas*** which act conjunctively, and ***β-formulas*** which act disjunctively.
- For each α formula, we define two components, which we denote α_1 and α_2. For each β-formula, we define two components, which we denote β_1 and β_2.

♣ β formulas of "$\leftrightarrow$"

- If a β-formula is of form $(A \leftrightarrow B)$, then β_1 denotes $(A \wedge B)$ and β_2 denotes $((\neg A) \wedge (\neg B))$
- If a β-formula is of form $(\neg(A \leftrightarrow B))$, then β_1 denotes $((\neg A) \wedge B)$ and β_2 denotes $(A \wedge (\neg B))$.

Uniform Notation of Formulas

♣ **Theorems**

- For any model, an α-formula is true IFF both α_1 AND α_2 true; a β-formula is true IFF β_1 OR β_2 is true.
- For any α and β, $(\alpha \leftrightarrow (\alpha_1 \wedge \alpha_2))$ and $(\beta \leftrightarrow (\beta_1 \vee \beta_2))$ are tautologies.

♣ ***Principle of Structural Induction*** [Fitting]

- Every formula in **WFF** has a property, Q, provided:

  Basis step: Every atomic formula and its negation has property Q;

  Induction steps: For any $A \in$ **WFF**, if A has property Q, so does $(\neg(\neg A))$;
  For any α_1 and $\alpha_2 \in$ **WFF**, if α_1 and α_2 have property Q, so does α;
  For any β_1 and $\beta_2 \in$ **WFF**, if β_1 and β_2 have property Q, so does β.

Uniform Notation of Formulas

♣ ***Principle of Structural Recursion*** [Fitting]

- There is one and only one function f defined on **WFF** such that:

  Basis step: The value of f is specified explicitly on atomic formulas and their negations;

  Recursion steps: For any $A \in$ **WFF**, the value of f on $(\neg(\neg A))$ is specified in terms of the value of f on A;
  For any α_1 and $\alpha_2 \in$ **WFF**, the value of f on α is specified in terms of the values of f on α_1 and α_2;
  For any β_1 and $\beta_2 \in$ **WFF**, the value of f on β is specified in terms of the values of f on β_1 and β_2.

α-Formulas and β-Formulas and Their Components [Fitting]

Conjunctive			Disjunctive		
α	α_1	α_2	β	β_1	β_2
$X \wedge Y$	X	Y	$\neg(X \wedge Y)$	$\neg X$	$\neg Y$
$\neg(X \vee Y)$	$\neg X$	$\neg Y$	$X \vee Y$	X	Y
$\neg(X \rightarrow Y)$	X	$\neg Y$	$X \rightarrow Y$	$\neg X$	Y
$\neg(X \leftarrow Y)$	$\neg X$	Y	$X \leftarrow Y$	X	$\neg Y$
$\neg(X \neg \wedge Y)$	X	Y	$X \neg \wedge Y$	$\neg X$	$\neg Y$
$X \neg \vee Y$	$\neg X$	$\neg Y$	$\neg(X \neg \vee Y)$	X	Y
$X \neg \rightarrow Y$	X	$\neg Y$	$\neg(X \neg \rightarrow Y)$	$\neg X$	Y
$X \neg \leftarrow Y$	$\neg X$	Y	$\neg(X \neg \leftarrow Y)$	X	$\neg Y$

α-Formulas and β-Formulas and Their Components [Ben-Ari]

α	α_1	α_2	β	β_1	β_2
$\neg\neg A_1$	A_1				
$A_1 \wedge A_2$	A_1	A_2	$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$	$B_1 \vee B_2$	B_1	B_2
$\neg(A_1 \rightarrow A_2)$	A_1	$\neg A_2$	$B_1 \rightarrow B_2$	$\neg B_1$	B_2
$\neg(A_1 \uparrow A_2)$	A_1	A_2	$B_1 \uparrow B_2$	$\neg B_1$	$\neg B_2$
$A_1 \downarrow A_2$	$\neg A_1$	$\neg A_2$	$\neg(B_1 \downarrow B_2)$	B_1	B_2
$A_1 \leftrightarrow A_2$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$	$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$
$\neg(A_1 \oplus A_2)$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$	$B_1 \oplus B_2$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$

Fig. 2.8 Classification of α- and β-formulas

- α-formulas are conjunctive and are satisfiable only if both subformulas α_1 and α_2 are satisfied, while β-formulas are disjunctive and are satisfied even if only one of the subformulas β_1 or β_2 is satisfiable.

An Introduction to Classical Propositional Calculus (CPC)

- ♣ Formal (Object) Language (Syntax) of **CPC**
- ♣ Principles of Structural Induction and Structural Recursion
- ♣ Semantics (Model Theory) of **CPC**
- ♣ Semantic (Model-theoretical, Logical) Consequence Relation
- ♣ Normal Forms and Uniform Notation of Formulas
- ♣ Deduction System (Proof Theory) of **CPC**
- ♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
- ♣ Hilbert Style Formal Logic Systems for **CPC**
- ♣ Gentzen's Natural Deduction System for **CPC**
- ♣ Gentzen's Sequent Calculus System for **CPC**
- ♣ Semantic Tableau System for **CPC**
- ♣ Resolution System for **CPC**
- ♣ Forward Deduction and Backward Deduction

Deduction System (Proof Theory) of CPC

- ♣ The fundamental question
  - Why is the deduction system (proof theory) of **CPC** indispensable?
- ♣ The answer to the question
  - It is often difficult to decide whether or not a formula is a theorem of a logic system only by the semantic way (some logics have no decision procedures).
  - The proof/deduction of a formula clearly shows which axioms, theorems and inference rules are used and for what purposes.
  - Once a theorem has been proved, it can be used in proofs/deductions like an axiom.
  - A proof/deduction can be mechanically checked, i.e., given a sequence of formulas, a syntax-based algorithm can easily check whether or not the sequence is a proof/deduction.
  - The deduction system (proof theory) of a logic system provides the foundation for automatic theorem finding/proving in the logic.

Deduction System (Proof Theory) of CPC

- ♣ Important notes
  - The deduction system (proof theory) of **CPC** is the most important ways and/or tools to study and use **CPC**.
  - Without a deduction system, **CPC** is difficult to study and use.
  - The deduction system (proof theory) of **CPC** is completely based on (underlain by) the semantics (model theory) of **CPC**, it is the semantics (model theory) that is the most intrinsic foundation of **CPC**.
  - The above answers to the fundamental question (and the above notes) are true/meaningful for not only **CPC** but also for all logic systems.

Formal Logic (Deductive) Systems, Proofs, and Theorems

- ♣ Formal logic (deductive) system
  - A ***formal logic (deductive) system*** has the following components:
    (1) ***alphabet***: a non-empty set of symbols,
    (2) ***grammar***: a finite set of rules for forming formulas,
    (3) ***axioms***: a set of formulas as start points for deduction, and
    (4) ***deduction (inference) rules***: a finite set of rules for generating a new formula (the consequence) from some old (the premises and/or hypotheses).
- ♣ Formal proof and theorem
  - A ***formal proof*** of f_n in a formal logic system **FLS** is a FINITE sequence of formulas $f_1, ..., f_n$ such that, for all i ($i \leq n$), (1) f_i is an axiom, or (2) there are some members $f_{j1}, ..., f_{jm}$ ($j1, ..., jm < i$) of the sequence, which have f_i as the result of applying one of the deduction rules to $f_{j1}, ..., f_{jm}$.
  - If $f_1, ..., f_n$ is a formal proof in a formal logic system **FLS**, then f_n is called a ***logical theorem*** of **FLS** and said to be ***provable*** in **FLS**, denoted as $\vdash_{FLS} f_n$.
  - The set of all logical theorems of **FLS** is denoted by **Th(FLS)**.

Deduction

- Formal deduction
  - Let P be a set of formulas in a formal logic system **FLS** and f_n be a formula of **FLS**. A ***formal deduction (proof) of f_n from P*** in **FLS**, denoted as $P \vdash_{FLS} f_n$, is a FINITE sequence of formulas $f_1, ..., f_n$ such that, for all i ($i \le n$), (1) f_i is an axiom, or (2) $f_i \in P$, or (3) there are some members $f_{j1}, ..., f_{jm}$ ($j1, ..., jm < i$) of the sequence, which have f_i as the result of applying one of the deduction rules to $f_{j1}, ..., f_{jm}$.
  - Note: '$\vdash$' is a meta-language(logic) symbol but not an object language symbol of **CPC**.
- Premises and consequence
  - If $f_1, ..., f_n$ is a formal deduction (proof) of f_n from P in a formal logic system **FLS**, then P is called the ***premises*** of the deduction and f_n is called the ***consequence*** and said to be ***deducible from*** P in **FLS**.
  - A logical theorem of a formal logic system is deducible from the empty set of premises in the formal logic system.

Formal Logic Systems (Formal Theory) [Mendelson]

A formal theory $\mathscr{F}$ is defined when the following conditions are satisfied:

1. A countable set of symbols is given as the symbols of $\mathscr{F}$.* A finite sequence of symbols of $\mathscr{F}$ is called an *expression* of $\mathscr{F}$.
2. There is a subset of the set of expressions of $\mathscr{F}$ called the set of *well-formed formulas* (wfs) of $\mathscr{F}$. There is usually an effective procedure to determine whether a given expression is a wf.
3. There is a set of wfs called the set of *axioms* of $\mathscr{F}$. Most often, one can effectively decide whether a given wf is an axiom; in such a case, $\mathscr{F}$ is called an *axiomatic* theory.
4. There is a finite set $R_1, \ldots, R_n$ of relations among wfs, called *rules of inference*. For each R_j, there is a unique positive integer j such that, for every set of j wfs and each wf $\mathscr{B}$, one can effectively decide whether the given j wfs are in the relation R_j to $\mathscr{B}$, and, if so, $\mathscr{B}$ is said to *follow from* or to be a *direct consequence of* the given wfs by virtue of R_j.†

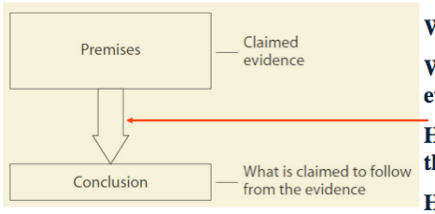
A *proof* in $\mathscr{F}$ is a sequence $\mathscr{B}_1, \ldots, \mathscr{B}_k$ of wfs such that, for each i, either $\mathscr{B}_i$ is an axiom of $\mathscr{F}$ or $\mathscr{B}_i$ is a direct consequence of some of the preceding wfs in the sequence by virtue of one of the rules of inference of $\mathscr{F}$.

A *theorem* of $\mathscr{F}$ is a wf $\mathscr{B}$ of $\mathscr{F}$ such that $\mathscr{B}$ is the last wf of some proof in $\mathscr{F}$. Such a proof is called a *proof of* $\mathscr{B}$ *in* $\mathscr{F}$.

An Introduction to Classical Propositional Calculus (CPC)

- Formal (Object) Language (Syntax) of **CPC**
- Principles of Structural Induction and Structural Recursion
- Semantics (Model Theory) of **CPC**
- Semantic (Model-theoretical, Logical) Consequence Relation
- Normal Forms and Uniform Notation of Formulas
- Deduction System (Proof Theory) of **CPC**
- Syntactic (Proof-theoretical, Deductive) Consequence Relation
- Hilbert Style Formal Logic Systems for **CPC**
- Gentzen's Natural Deduction System for **CPC**
- Gentzen's Sequent Calculus System for **CPC**
- Semantic Tableau System for **CPC**
- Resolution System for **CPC**
- Forward Deduction and Backward Deduction

Logic: What Is It?



[From P. J. Hurley, "A Concise Introduction to Logic"]

What entails what?

What follows from what?

Why? What are the evaluation criteria?

How to establish/define the evaluation criteria?

How to evaluate arguments/reasoning?

It is LOGIC to answer these fundamental questions.

Syntactic (Proof-theoretical, Deductive) Consequence Relation

- Syntactic (Proof-theoretical, Deductive) consequence relation
  - Let **FLS** be a formal logic system. For a set Γ of formulas and a formula A of **FLS**,
    Γ ***syntactically (proof-theoretically, deductively) entails*** A, or
    A ***syntactically (proof-theoretically, deductively) follows from*** Γ, or
    A is a ***syntactic (proof-theoretical, deductive) consequence*** of Γ,
    denoted as $\Gamma \vdash_{FLS} A$ ('$\vdash$' is read as 'the turnstile with subscript **FLS**'),
    IFF A is deducible from Γ in **FLS**.
  - The set of all syntactic (proof-theoretical, deductive) consequences of Γ is denoted by $\mathbf{C}_{\text{syn-FLS}}(\Gamma)$ (Note: A **FLS**-theory with Γ).
  - $\vdash_{FLS} A =_{df} \phi \vdash_{FLS} A$ and it means that A is a logical theorem of **FLS**.
- Note
  - The syntactic (proof-theoretical, deductive) consequence relation is a syntactic (proof-theoretical) formalization of the notion that one proposition ***follows from*** another or others.

Syntactic (Proof-theoretical, Deductive) Equivalence Relation

- Syntactic (proof-theoretical, deductive) equivalence relation
  - For any two formulas A and B, A is ***syntactically (proof-theoretically, deductively) equivalent*** to B in **FLS** IFF both $\{A\} \vdash_{FLS} B$ and $\{B\} \vdash_{FLS} A$.

Properties of the Syntactic (Proof-theoretical, Deductive) Consequence Relation

♣ Properties of the syntactic consequence relation

- If $\vdash_{\mathbf{FLS}} A$, then $\Gamma \vdash_{\mathbf{FLS}} A$. (A theorem is a consequence of any premise; **Th(FLS)** is a proper subset of $\mathbf{C}_{\text{syn-FLS}}(\Gamma)$ for any Γ.) (Note: A **FLS**-theory with Γ)
- $\{A\} \vdash_{\mathbf{FLS}} A$.
- If $A \in \Gamma$, then $\Gamma \vdash_{\mathbf{FLS}} A$. (***Reflexivity***)
- If $\Gamma \vdash_{\mathbf{FLS}} A$ and $\Gamma \subseteq \Delta$, then $\Delta \vdash_{\mathbf{FLS}} A$. (***Monotonicity***)
- ***Transitivity*** (***the Cut rule***): If $\Gamma \vdash_{\mathbf{FLS}} A$ and $\Delta \cup \{A\} \vdash_{\mathbf{FLS}} B$, then $\Gamma \cup \Delta \vdash_{\mathbf{FLS}} B$.
- If $\Gamma \cup \{A_1, ..., A_n\} \vdash_{\mathbf{FLS}} B$ and $\Gamma \vdash_{\mathbf{FLS}} A_i$ for $i = 1, ..., n$, then $\Gamma \vdash_{\mathbf{FLS}} B$.
- ***Compactness (Finiteness)***: $\Gamma \vdash_{\mathbf{FLS}} A$ IFF there is a finite set $\Delta \subseteq \Gamma$ such that $\Delta \vdash_{\mathbf{FLS}} A$.

♣ Homework

- Try to prove the above properties of the syntactic consequence relation according to definition.

Properties of the Syntactic (Proof-theoretical, Deductive) Consequence Relation

♣ Post-consistence and Post-completeness

Let **FLS** be a formal logic system and Γ be a set of formulas.

- Γ is ***Post-consistent*** (***syntactically consistent***) in **FLS** IFF there is some formula A such that $\Gamma \vdash_{\mathbf{FLS}} A$ does not hold.
- Γ is ***Post-complete*** (***syntactically complete***) in **FLS** IFF for every formula B not in Γ and for every C, $\Gamma \cup \{B\} \vdash_{\mathbf{FLS}} C$.

♣ **Theorems**

- If Γ is Post-consistent in **FLS** and $\Delta \subseteq \Gamma$ then Δ is Post-consistent in **FLS**.
- Γ is Post-consistent in **FLS** IFF every finite subset of Γ is Post-consistent in **FLS**.
- If Γ is both Post-complete and Post-consistent in **FLS**, then Γ is a formal theory.

Properties of the Syntactic (Proof-theoretical, Deductive) Consequence Relation

♣ Classical-consistence and Classical-completeness

Let **FLS** be a formal system and Γ be a set of formulas.

- Γ is ***classically consistent*** in **FLS** IFF for every formula A, NOT both $\Gamma \vdash_{\mathbf{FLS}} A$ and $\Gamma \vdash_{\mathbf{FLS}} (\neg A)$.
- Γ is ***classically complete*** **FLS** IFF for every formula A, at least one of A and $(\neg A)$ in Γ.

♣ **Theorems**

- Γ is classically consistent in **FLS** IFF Γ is Post-consistent in **FLS**.
- If Γ is a theory, then Γ is classically complete in **FLS** IFF Γ is Post-complete in **FLS**.

Closures

♣ Closure

- Let **FLS** be a formal logic system. A set Σ of formulas of **FLS** is ***closed under*** a deduction (inference) rule of **FLS** if whenever the premises of the rule are in Σ, then so is the consequence.
- Let R be a deduction (inference) rule in a formal logic system **FLS**, the ***closure of a set Σ of formulas under R***, denoted by $\mathbf{C}_{\text{syn-FLS}}(\Sigma, \text{R})$, is the smallest set such that for any $\Delta \supseteq \Sigma$ and Δ is closed under R, $\Delta \supseteq \mathbf{C}_{\text{syn-FLS}}(\Sigma, \text{R})$.

♣ Questions

- $\mathbf{C}_{\text{syn-FLS}}(\Sigma) = \bigcup_{\text{R} \in \mathbf{FLS}} \mathbf{C}_{\text{syn-FLS}}(\Sigma, \text{R})$?
- $\mathbf{C}_{\text{syn-FLS}}(\Sigma) = \mathbf{C}_{\text{syn-FLS}}(\Sigma, \text{R})$ if **FLS** has R as the single deduction (inference) rule?

Formal Theories

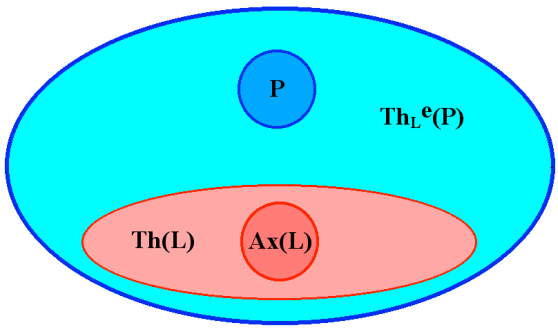
♣ Formal theories with premises

- Let **FLS** be a formal logic system. For a set Σ of formulas (as premises), the set of all syntactic consequences of Σ is called the ***formal theory of Σ based on FLS*** (or the ***formal theory with premises Σ based on FLS***, the ***FLS-theory with premises Σ***), denoted $T_{\mathbf{FLS}}(\Sigma)$, i.e., $T_{\mathbf{FLS}}(\Sigma) =_{\text{df}} \{A \mid \Sigma \vdash_{\mathbf{FLS}} A\}$.
- $T_{\mathbf{FLS}}(\Sigma) =_{\text{df}} \mathbf{C}_{\text{syn-FLS}}(\Sigma)$
- $T_{\mathbf{FLS}}(\Sigma) =_{\text{df}} Th(\mathbf{FLS}) \cup Th_{\mathbf{FLS}}^{e}(\Sigma)$ where $Th_{\mathbf{FLS}}^{e}(\Sigma) =_{\text{df}} \{ et \mid \Sigma \vdash_{\mathbf{FLS}} et \text{ and } et \notin Th(\mathbf{FLS})\}$

♣ Formal theories

- Let **FLS** be a formal logic system. A set Σ of formulas is called a ***formal theory based on FLS*** (or a ***formal theory based on FLS***, a ***FLS-theory***), if it is closed under the relation of syntactic consequence, i.e., $T_{\mathbf{FLS}}(\Sigma) = \Sigma$.

Formal Theories



An Introduction to Classical Propositional Calculus (CPC)

- ♣ Formal (Object) Language (Syntax) of **CPC**
- ♣ Principles of Structural Induction and Structural Recursion
- ♣ Semantics (Model Theory) of **CPC**
- ♣ Semantic (Model-theoretical, Logical) Consequence Relation
- ♣ Normal Forms and Uniform Notation of Formulas
- ♣ Deduction System (Proof Theory) of **CPC**
- ♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
- ♣ Hilbert Style Formal Logic Systems for **CPC**
- ♣ Gentzen's Natural Deduction System for **CPC**
- ♣ Gentzen's Sequent Calculus System for **CPC**
- ♣ Semantic Tableau System for **CPC**
- ♣ Resolution System for **CPC**
- ♣ Forward Deduction and Backward Deduction

Hilbert Style Formal Logic Systems

- ♣ Hilbert style formal logic systems
  - The most classical (historical) style of formal systems, which was first given by G. Frege in 1879.
  - The mechanism of a Hilbert style formal system works in the forward deduction principle.
  - For certain philosophical logics, only Hilbert style formulations are known to exist.
  - Hilbert style formal systems are widespread, and should be familiar to everyone who uses formal logic.
- ♣ ***Forward deduction principle***
  - To prove a formula in a formal logic system, one starts with some formulas as premises which are known facts or assumed hypotheses, derives immediate consequences, immediate consequences of the immediate consequences, and so on by applying inference rules, until the desired formula is reached.

L: A Hilbert Style Formal System for CPC

- ♣ Alphabet
  - $\{\neg, \rightarrow, p_1, p_2, ..., p_n, ..., (,) \}$ (Using negation '$\neg$' and material implication '$\rightarrow$' as the primitive connectives)
- ♣ Formulas (Well-formed formulas) of **L**: $\mathbf{WFF_L}$
  - (1) $p_1, p_2, ..., p_n, ...$ are (atomic) formulas;
  - (2) if A and B are formulas, then so are $(\neg A)$ and $(A \rightarrow B)$;
  - (3) Nothing else are formulas.
- ♣ Axiom schemata of **L** [**L**ukasiewicz, 1930] For $A, B, C \in \mathbf{WFF_L}$:
  - AS1 $(A \rightarrow (B \rightarrow A))$
  - AS2 $((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)))$
  - AS3 $(((\neg A) \rightarrow (\neg B)) \rightarrow (B \rightarrow A))$ [or $(((\neg A) \rightarrow (\neg B)) \rightarrow ((\neg A) \rightarrow B) \rightarrow A))$]
- ♣ Deduction (inference) rule of **L** ($A, B \in \mathbf{WFF_L}$)
  - ***Modus Ponens (MP) for material implication (the detachment rule)***: From $(A \rightarrow B)$ and A, to infer B.

L: A Hilbert Style Formal System for CPC

- ♣ Notes
  - Other logical connectives (conjunction '$\wedge$', disjunction '$\vee$', and equivalence '$\leftrightarrow$') can be defined by negation '$\neg$' and material implication '$\rightarrow$'. Therefore, **L** does not lose generality.
  - Each axiom schema can have many (infinite) instances (replace meta-symbols 'A', 'B', and 'C' by some formulas. Therefore, **L** has infinite axioms.
  - **L** has only one deduction (inference) rule: MP.
  - **L** is just one of many possible Hilbert style formal systems for **CPC**.

Syntactic (proof-theoretical) deduction theorems for L

- ♣ ***Syntactic (proof-theoretical) deduction theorems for L*** [Herbrand, 1930]
  - For any $A, B \in \mathbf{WFF_L}$ and any $\Gamma \subseteq \mathbf{WFF_L}$, $\Gamma \cup \{A\} \vdash_{\mathbf{L}} B$ IFF $\Gamma \vdash_{\mathbf{L}} A \rightarrow B$.
  - For any $A_1, ..., A_{n-1}, A_n, B \in \mathbf{WFF_L}$ and any $\Gamma \subseteq \mathbf{WFF_L}$, $\Gamma \cup \{A_1, ..., A_{n-1}, A_n\} \vdash_{\mathbf{L}} B$ IFF $\Gamma \vdash_{\mathbf{L}} (A_1 \rightarrow (...(A_{n-1} \rightarrow (A_n \rightarrow B))...))$; $\Gamma \cup \{A_1, ..., A_{n-1}, A_n\} \vdash_{\mathbf{L}} B$ IFF $\Gamma \vdash_{\mathbf{L}} ((A_1 \wedge (...(A_{n-1} \wedge A_n)...)) \rightarrow B)$.
- ♣ Notes
  - $\{A\} \vdash_{\mathbf{L}} B$ IFF $\vdash_{\mathbf{L}} A \rightarrow B$, i.e., B syntactically (proof-theoretically, deductively) entails A IFF $A \rightarrow B$ is a theorem. This means that in the framework of **CPC**, the syntactic (proof-theoretical, deductive) consequence relation, which is the notion of conditional in the sense of meta-logic, is "equivalent" to the notion of material implication.
  - The above deduction theorems are also true for any Hilbert style formal logic system with at least axiom schemata "$(A \rightarrow (B \rightarrow A))$" and "$((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)))$" and with Modus Ponens for material implication as the only inference rule.

An Example of Theorem Proof in L

- ♣ $\vdash_{\mathbf{L}} X \rightarrow X$? (Omit the outermost brackets)

1. $(X \rightarrow ((X \rightarrow X) \rightarrow X)) \rightarrow ((X \rightarrow (X \rightarrow X)) \rightarrow (X \rightarrow X))$
   $\{$AS2 $((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)))$, X/A, $X \rightarrow X/B$, X/C $\}$
2. $X \rightarrow ((X \rightarrow X) \rightarrow X)$ $\{$ AS1 $(A \rightarrow (B \rightarrow A))$, X/A, $X \rightarrow X/B$ $\}$
3. $(X \rightarrow (X \rightarrow X)) \rightarrow (X \rightarrow X)$ $\{$ Follow from 1 and 2 by MP $\}$
4. $X \rightarrow (X \rightarrow X)$ $\{$ AS1 $(A \rightarrow (B \rightarrow A))$, X/A, X/B $\}$
5. $X \rightarrow X$ $\{$ Follow from 3 and 4 by MP $\}$

An Example of Theorem Proof in L

♣ $\vdash_{\mathbf{L}} (X\rightarrow(Y\rightarrow Z))\rightarrow(Y\rightarrow(X\rightarrow Z))$? (Omit the outermost brackets)

- Let us show $\{ (X\rightarrow(Y\rightarrow Z)), X, Y \} \vdash_{\mathbf{L}} Z$ (An example of deduction in **L**)

1. $X\rightarrow(Y\rightarrow Z)$ { Premise }
2. X { Premise }
3. $Y\rightarrow Z$ { Follow from 1 and 2 by MP }
4. Y { Premise }
5. Z { Follow from 3 and 4 by MP }

- Using deduction theorem, it follows that $\{ (X\rightarrow(Y\rightarrow Z)), Y \} \vdash_{\mathbf{L}} X\rightarrow Z$.
- Using deduction theorem again, it follows that $\{ (X\rightarrow(Y\rightarrow Z)) \} \vdash_{\mathbf{L}} Y\rightarrow(X\rightarrow Z)$.
- Finally, using deduction theorem again, it follows that $\vdash_{\mathbf{L}} (X\rightarrow(Y\rightarrow Z))\rightarrow(Y\rightarrow(X\rightarrow Z))$.

Some Derived Rules in L (CPC) [Mendelson]

- Negation elimination: $\neg\neg B \vdash_{\mathbf{L}} B$
- Negation introduction: $B \vdash_{\mathbf{L}} \neg\neg B$
- Conjunction elimination: $B\wedge C \vdash_{\mathbf{L}} B$; $B\wedge C \vdash_{\mathbf{L}} C$; $\neg(B\wedge C) \vdash_{\mathbf{L}} \neg B\vee\neg C$
- Conjunction introduction: $B, C \vdash_{\mathbf{L}} B\wedge C$
- Disjunction elimination: $B\vee C, \neg B \vdash_{\mathbf{L}} C$; $B\vee C, \neg C \vdash_{\mathbf{L}} B$; $\neg(B\vee C) \vdash_{\mathbf{L}} \neg B\wedge\neg C$; $B\rightarrow D, C\rightarrow D, B\vee C \vdash_{\mathbf{L}} D$
- Disjunction introduction: $B \vdash_{\mathbf{L}} B\vee C$; $C \vdash_{\mathbf{L}} B\vee C$
- Conditional elimination: $B\rightarrow C, \neg C \vdash_{\mathbf{L}} \neg B$; $B\rightarrow\neg C, C \vdash_{\mathbf{L}} \neg B$; $\neg B\rightarrow C, \neg C \vdash_{\mathbf{L}} B$; $\neg B\rightarrow\neg C, C \vdash_{\mathbf{L}} B$; $\neg(B\rightarrow C) \vdash_{\mathbf{L}} B$; $\neg(B\rightarrow C) \vdash_{\mathbf{L}} \neg C$
- Conditional introduction: $B, \neg C \vdash_{\mathbf{L}} \neg(B\rightarrow C)$
- Conditional contrapositive: $B\rightarrow C \vdash_{\mathbf{L}} \neg C\rightarrow\neg B$; $\neg C\rightarrow\neg B \vdash_{\mathbf{L}} B\rightarrow C$

Some Derived Rules in L (CPC) [Mendelson]

- Biconditional elimination: $B\leftrightarrow C, B \vdash_{\mathbf{L}} C$; $B\leftrightarrow C, \neg B \vdash_{\mathbf{L}} \neg C$; $B\leftrightarrow C, C \vdash_{\mathbf{L}} B$; $B\leftrightarrow C, \neg C \vdash_{\mathbf{L}} \neg B$; $B\leftrightarrow C \vdash_{\mathbf{L}} B\rightarrow C$; $B\leftrightarrow C \vdash_{\mathbf{L}} C\rightarrow B$
- Biconditional introduction: $B\rightarrow C, C\rightarrow B \vdash_{\mathbf{L}} B\leftrightarrow C$
- Biconditional negation: $B\leftrightarrow C \vdash_{\mathbf{L}} \neg B\leftrightarrow\neg C$; $\neg B\leftrightarrow\neg C \vdash_{\mathbf{L}} B\leftrightarrow C$

- Proof by contradiction: If $\Gamma, \neg B \vdash_{\mathbf{L}} C\wedge\neg C$, then $\Gamma \vdash_{\mathbf{L}} B$. Similarly, one obtains $\Gamma \vdash_{\mathbf{L}} \neg B$ from $\Gamma, B \vdash_{\mathbf{L}} C\wedge\neg C$.

Properties of L: Consistency

♣ ***Consistency of* L**

- For any $A \in \mathbf{WFF}_{\mathbf{L}}$, NOT both $\vdash_{\mathbf{L}} A$ and $\vdash_{\mathbf{L}} \neg A$, i.e., **L** is ***consistent***.

Properties of L: Soundness and Completeness

♣ A fundamental question about **L** and **CPC**

- What is the relationship between **Th**(**L**) and **Th**(**CPC**)?

♣ ***Soundness theorems for L***

- Theorem (***soundness***): If $\vdash_{\mathbf{L}} A$ then $\models_{\mathbf{CPC}} A$, for any $A \in \mathbf{WFF}_{\mathbf{L}}$, i.e., for any $A \in \mathbf{WFF}_{\mathbf{L}}$, if $A \in \mathbf{Th}(\mathbf{L})$ then $A \in \mathbf{Th}(\mathbf{CPC})$, $\mathbf{Th}(\mathbf{L}) \subseteq \mathbf{Th}(\mathbf{CPC})$.
- Theorem (***strong soundness***): If $\Gamma \vdash_{\mathbf{L}} A$ then $\Gamma \models_{\mathbf{CPC}} A$, for any $A \in \mathbf{WFF}_{\mathbf{L}}$ and any $\Gamma \subseteq \mathbf{WFF}_{\mathbf{L}}$.

♣ ***Completeness theorems for L***

- Theorem (***completeness***): If $\models_{\mathbf{CPC}} A$ then $\vdash_{\mathbf{L}} A$, for any $A \in \mathbf{WFF}_{\mathbf{L}}$, i.e., for any $A \in \mathbf{WFF}_{\mathbf{L}}$, if $A \in \mathbf{Th}(\mathbf{CPC})$ then $A \in \mathbf{Th}(\mathbf{L})$, $\mathbf{Th}(\mathbf{CPC}) \subseteq \mathbf{Th}(\mathbf{L})$.
- Theorem (***strong completeness***): If $\Gamma \models_{\mathbf{CPC}} A$ then $\Gamma \vdash_{\mathbf{L}} A$, for any $A \in \mathbf{WFF}_{\mathbf{L}}$ and any $\Gamma \subseteq \mathbf{WFF}_{\mathbf{L}}$.

♣ **L** as a formal logic system of **CPC**

- **Th**(**L**) = **Th**(**CPC**)

Other Hilbert Style Axiomatizations for CPC

♣ Frege's system (the first axiomatization for **CPC**) [G. Frege, 1879]

- '$\rightarrow$' and '$\neg$' are the primitive connectives.
- $(A\rightarrow(B\rightarrow A))$
- $((A\rightarrow(B\rightarrow C))\rightarrow((A\rightarrow B)\rightarrow(A\rightarrow C)))$
- $((A\rightarrow(B\rightarrow C))\rightarrow(B\rightarrow(A\rightarrow C)))$ (this is deducible from the above two)
- $((A\rightarrow B)\rightarrow((\neg B)\rightarrow(\neg A)))$
- $((\neg(\neg A))\rightarrow A)$
- $(A\rightarrow(\neg(\neg A)))$ (the last three can be replaced by $(((\neg A)\rightarrow(\neg B))\rightarrow(B\rightarrow A))$
- The only rule of inference is Modus Ponens.

♣ Notes

- **L** [Lukasiewicz, 1930] is a simplification of Frege's system.
- A variant of **L**, **L'**, can be obtained by replacing the third axiom scheme of **L** '$(((\neg A)\rightarrow(\neg B))\rightarrow(B\rightarrow A))$' by '$(((\neg A)\rightarrow(\neg B))\rightarrow((\neg A)\rightarrow B)\rightarrow A))$'.

Other Hilbert Style Axiomatizations for CPC

♣ Rusell and Whitehead' system [B.A.W. Rusell and A.N. Whitehead, PM, 1910]

- Using '¬' and '∨' as the primitive connectives. $A \rightarrow B$ is an abbreviation for $\neg A \vee B$.
- $((A \vee A) \rightarrow A)$
- $(A \rightarrow (A \vee B))$
- $((A \vee B) \rightarrow (B \vee A))$
- $((A \rightarrow B) \rightarrow ((C \vee A) \rightarrow (C \vee B)))$
- $((A \vee (B \vee C)) \rightarrow ((A \vee B) \vee C))$
- The only rule of inference is Modus Ponens.
- Note: In fact, the last axiom scheme was redundant – it was derivable from the other four axiom schemata.

Other Hilbert Style Axiomatizations for CPC

♣ Hilbert and Bernays' system [D. Hilbert and P. Bernays, 1934]

- Using '→', '∧', '∨', '↔', and '¬' as the (not primitive!) connectives.
- $(A \rightarrow (B \rightarrow A))$
  $((A \rightarrow (A \rightarrow B)) \rightarrow (A \rightarrow B))$
  $((A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C)))$
- $((A \wedge B) \rightarrow A)$, $((A \wedge B) \rightarrow B)$
  $((A \rightarrow B) \rightarrow ((A \rightarrow C) \rightarrow (A \rightarrow (B \wedge C))))$
- $(A \rightarrow (A \vee B))$, $(B \rightarrow (A \vee B))$
  $((A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C)))$
- $((A \leftrightarrow B) \rightarrow (A \rightarrow B))$, $((A \leftrightarrow B) \rightarrow (B \rightarrow A))$
  $((A \rightarrow B) \rightarrow ((B \rightarrow A) \rightarrow (A \leftrightarrow B)))$
- $((A \rightarrow B) \rightarrow ((\neg B) \rightarrow (\neg A)))$
  $(A \rightarrow (\neg(\neg A)))$, $((\neg(\neg A)) \rightarrow A)$
- The only rule of inference is Modus Ponens.

Examples of Deduction/Proof in Hilbert and Bernays' System (HB)

♣ $\{ X, Y \} \vdash_{HB} (X \wedge Y)$?

1. X { Premise }
2. Y { Premise }
3. $(X \rightarrow (X \rightarrow X))$ { AS $(A \rightarrow (B \rightarrow A)), X=A, X=B$ }
4. $(X \rightarrow X)$ { Follow from 1 and 3 by MP }
5. $((X \rightarrow X) \rightarrow ((X \rightarrow Y) \rightarrow (X \rightarrow (X \wedge Y))))$
   { AS $((A \rightarrow B) \rightarrow ((A \rightarrow C) \rightarrow (A \rightarrow (B \wedge C)))), X=A, X=B, Y=C$ }
6. $((X \rightarrow Y) \rightarrow (X \rightarrow (X \wedge Y)))$ { Follow from 4 and 5 by MP }
7. $(Y \rightarrow (X \rightarrow Y))$ { AS $(A \rightarrow (B \rightarrow A)), Y=A, X=B$ }
8. $(X \rightarrow Y)$ { Follow from 1 and 7 by MP }
9. $(X \rightarrow (X \wedge Y))$ { Follow from 6 and 8 by MP }
10. $(X \wedge Y)$ { Follow from 1 and 9 by MP }

♣ $\vdash_{HB} (A \rightarrow (B \rightarrow (A \wedge B)))$?

By $\{A, B\} \vdash_{HB} (A \wedge B)$ and the deduction theorem, we can directly have $\vdash_{HB} (A \rightarrow (B \rightarrow (A \wedge B)))$.

Other Hilbert Style Axiomatizations for CPC

♣ Hilbert and Ackermann' system [D. Hilbert and W. Ackermann, 1950]

- '∨' and '¬' are the primitive connectives. $A \rightarrow B$ is an abbreviation for $\neg A \vee B$.
- $((A \vee A) \rightarrow A)$
- $(A \rightarrow (A \vee B))$
- $((A \vee B) \rightarrow (B \vee A))$
- $((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow (A \vee C)))$
- The only rule of inference is Modus Ponens.

Other Hilbert Style Axiomatizations for CPC

♣ Kleene's system [S. C. Kleene, 1952]

- Using '→', '∧', '∨', and '¬' as the (not primitive!) connectives.
- $(A \rightarrow (B \rightarrow A))$
  $((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)))$
  [or $((A \rightarrow B) \rightarrow ((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))))$]
- $((A \wedge B) \rightarrow A)$
  $((A \wedge B) \rightarrow B)$
  $(A \rightarrow (B \rightarrow (A \wedge B)))$
- $(A \rightarrow (A \vee B))$
  $(B \rightarrow (A \vee B))$
  $((A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C)))$
- $((A \rightarrow B) \rightarrow ((A \rightarrow (\neg B)) \rightarrow (\neg A)))$
  $((\neg(\neg A)) \rightarrow A)$
- The only rule of inference is Modus Ponens.

Other Hilbert Style Axiomatizations for CPC

♣ Rosser's system [J.B. Rosser, 1953]

- '∧' and '¬' are the primitive connectives. $A \rightarrow B$ is an abbreviation for $\neg(A \wedge \neg B)$.
- $A \rightarrow (A \wedge A)$
- $(A \wedge B) \rightarrow A$
- $(A \rightarrow B) \rightarrow (\neg(B \wedge C) \rightarrow \neg(C \wedge A))$
- The only rule of inference is Modus Ponens.

♣ Meredith's system [Meredith, 1953]

- '→' and '¬' are the primitive connectives.
- $\{[(((A \rightarrow B) \rightarrow ((\neg C) \rightarrow (\neg B))) \rightarrow C) \rightarrow D] \rightarrow [(D \rightarrow A) \rightarrow (B \rightarrow A)]\}$
- The only rule of inference is Modus Ponens.

Other Hilbert Style Axiomatizations for CPC

♣ Church's system [A. Church, 1956]

- $A \to B$ is the primitive connective. $\neg A$ is an abbreviation for $A \to \bot$.
- $(A \to (B \to A))$
- $((A \to (B \to C)) \to ((A \to B) \to (A \to C)))$
- $((A \to \bot) \to \bot) \to A)$
- The only rule of inference is Modus Ponens.

Other Hilbert Style Axiomatizations for CPC

♣ The connective '↑' (***incompatibility***, ***alternative denial***, ***Sheffer stroke***)

- $A \uparrow B =_{df} \neg(A \wedge B)$; $A \uparrow B =_{df} (\neg A) \vee (\neg B)$
- The truth table of connective '↑' (***incompatibility***, ***alternative denial***, ***Sheffer stroke***) is as follows:

A	B	$A \uparrow B$
T	**T**	**F**
F	**T**	**T**
T	**F**	**T**
F	**F**	**T**

- $(\neg A) \leftrightarrow (A \uparrow A)$, $(A \vee B) \leftrightarrow ((A \uparrow A) \uparrow (B \uparrow B))$

♣ Nicod's system [Nicod, 1917]

- '↑' is the only connective.
- $(A \uparrow (B \uparrow C)) \uparrow \{[B \uparrow (B \uparrow B)] \mid [(D \uparrow B) \mid ((A \uparrow D) \mid (A \uparrow D))]\}$
- The only rule of inference is: from $A \uparrow (B \uparrow C)$ and A, to infer C.

An Introduction to Classical Propositional Calculus (CPC)

♣ Formal (Object) Language (Syntax) of **CPC**
♣ Principles of Structural Induction and Structural Recursion
♣ Semantics (Model Theory) of **CPC**
♣ Semantic (Model-theoretical, Logical) Consequence Relation
♣ Normal Forms and Uniform Notation of Formulas
♣ Deduction System (Proof Theory) of **CPC**
♣ Syntactic (Proof-theoretical, Deductive) Consequence Relation
♣ Hilbert Style Formal Logic Systems for **CPC**
♣ Gentzen's Natural Deduction System for **CPC**
♣ Gentzen's Sequent Calculus System for **CPC**
♣ Semantic Tableau System for **CPC**
♣ Resolution System for **CPC**
♣ Forward Deduction and Backward Deduction