COMP9020 Week 5
Term 3, 2019
Logic

What is logic?

Logic is about **formalizing reasoning** and **defining truth**

- Adding rigour
- Removing ambiguity
- Mechanizing the process of reasoning

Summary of topics

Part 1:

- Historical background
- Applications to Computer Science
- Propositional logic, informally
- Propositional logic, formally

Part 2:

- CNF and DNF
- Karnaugh maps
- Boolean algebra
- Other logics

Summary of topics

- Historical background
- Applications to Computer Science
- Propositional logic, informally
- Propositional logic, formally

Loose history of logic

- (Ancient times): Logic exlusive to philosophy
- Mid-19th Century: Logical foundations of Mathematics
- 1910: Russell and Whitehead's Principia Mathematica
- 1928: Hilbert proposes *Entscheidungsproblem*
- 1931: Gödel's Incompleteness Theorem
- 1935: Church's Lambda calculus
- 1936: Turing's Machine-based approach
- 1930s: Circuit logic 电路逻辑
- 1960s: Formal verification; Relational databases

Summary of topics

- Historical background
- Applications to Computer Science
- Propositional logic, informally
- Propositional logic, formally

Applications to Computer Science

$$\text{Computation} \quad = \quad \text{Calculation} + \text{Symbolic manipulation}$$

Applications to Computer Science

$$\text{Computation} \quad = \quad \text{Calculation} + \text{Symbolic manipulation}$$

Computers as logical structures:

- Formal verification
- Proof assistance
- Knowledge Representation and Reasoning
- Automated reasoning
- Databases

Applications to Computer Science

Computation $=$ Calculation $+$ Symbolic manipulation

Logic as 2-valued computation

- Circuit design
- Code optimization
- Boolean algebra

Summary of topics

- Historical background
- Applications to Computer Science
- Propositional logic, informally
- Propositional logic, formally

命题逻辑

Propositions

A **proposition** (or sentence) is a declarative statement; something that is either true or false.

Examples

- Richard Nixon was president of Ecuador.
- A square root of 16 is 4.
- Euclid's program gets stuck in an infinite loop if you input 0.
- Whatever list of numbers you give as input to this program, it outputs the same list but in increasing order.
- $x^n + y^n = z^n$ has no nontrivial integer solutions for $n > 2$.
- 3 divides 24.
- K_5 is planar.

Propositions

Examples

The following are *not* declarative sentences:

- Gubble gimble goo
- For Pete's sake, take out the garbage!
- Did you watch MediaWatch last week?
- Please waive the prerequisites for this subject for me.
- x divides y.
- $x = 3$ and x divides 24.

Propositions

Examples

The following are *not* declarative sentences:

- Gubble gimble goo
- For Pete's sake, take out the garbage!
- Did you watch MediaWatch last week?
- Please waive the prerequisites for this subject for me.
- x divides y. — $R(x, y)$
- $x = 3$ and x divides 24. — $P(x)$

Logical connectives

逻辑连接符

Logical connectives join together propositions to build larger, **compound** propositions.

Examples

- Chef is a bit of a Romeo *and* Kenny is always getting killed.
- Either Bill is a liar *or* Hillary is innocent of Whitewater.
- *It is not the case that* this program always halts.
- *If* it is raining *then* I have an umbrella.

Logical connectives

Common logical connectives:

Symbol	Default	Also known as
$\wedge$	and	but, “;”
$\vee$	or	“either .. or ..”
$\neg$	not	not the case
$\rightarrow$	“if .. then ..”	implies whenever is sufficient for
$\leftrightarrow$	“.. if and only if ..”	bi-implies necessary and sufficient exactly when just in case

Compound propositions

The **truth** of a compound proposition depends on the truth of its components (**atomic propositions**):

Example

P: Chef is a bit of a Romeo and Kenny is always getting killed.

Chef is a bit of a Romeo	Kenny is always getting killed	P
True	True	True
False	True	False
True	False	False
False	False	False

Compound propositions

A	B	$A \wedge B$	$A \vee B$	$\neg A$	$A \rightarrow B$	$A \leftrightarrow B$
True	True	True	True	False	True	True
False	True	False	True	True	True	False
True	False	False	True	False	False	False
False	False	False	False	True	True	True

Vacuous truth

How to interpret $A \to B$ when A is false?

$A \to B$ If A (premise) then B (conclusion)

Material implication is false *only when* the premise holds and the conclusion does not.

If the premise is false, the implication is true no matter how absurd the conclusion is.

Both the following statements are true:

- If February has 30 days then March has 31 days.
- If February has 30 days then March has 42 days.

Exercises

Exercises

$\boxed{\text{LLM: Problem 3.2}}$

$$
\begin{aligned}
p &= \text{“you get an HD on your final exam”} \\
q &= \text{“you do every exercise in the book”} \\
r &= \text{“you get an HD in the course”}
\end{aligned}
$$

Translate into logical notation:

(a) You get an HD in the course although you do not do every exercise in the book.

(c) To get an HD in the course, you must get an HD on the exam.

(d) You get an HD on your exam, but you don't do every exercise in this book; nevertheless, you get an HD in this course.

Exercises

Exercises

$\boxed{\text{LLM: Problem 3.2}}$

p	$=$	"you get an HD on your final exam"
q	$=$	"you do every exercise in the book"
r	$=$	"you get an HD in the course"

Translate into logical notation:

(a) You get an HD in the course although you do not do every exercise in the book.

Exercises

Exercises

$\boxed{\text{LLM: Problem 3.2}}$

p	$=$	"you get an HD on your final exam"
q	$=$	"you do every exercise in the book"
r	$=$	"you get an HD in the course"

Translate into logical notation:

(a) You get an HD in the course although you do not do every exercise in the book. $r \wedge \neg q$

(c) To get an HD in the course, you must get an HD on the exam.

Exercises

Exercises

$\boxed{\text{LLM: Problem 3.2}}$

p	$=$	"you get an HD on your final exam"
q	$=$	"you do every exercise in the book"
r	$=$	"you get an HD in the course"

Translate into logical notation:

(a) You get an HD in the course although you do not do every exercise in the book. $r \wedge \neg q$

(c) To get an HD in the course, you must get an HD on the exam. $r \rightarrow p$

(d) You get an HD on your exam, but you don't do every exercise in this book; nevertheless, you get an HD in this course.

Exercises

Exercises

$\boxed{\text{LLM: Problem 3.2}}$

p	$=$	"you get an HD on your final exam"
q	$=$	"you do every exercise in the book"
r	$=$	"you get an HD in the course"

Translate into logical notation:

(a) You get an HD in the course although you do not do every exercise in the book. $r \wedge \neg q$

(c) To get an HD in the course, you must get an HD on the exam. $r \rightarrow p$

(d) You get an HD on your exam, but you don't do every exercise in this book; nevertheless, you get an HD in this course. $p \wedge \neg q \wedge r$

Tautologies, Contradictions and Contingencies

Definition

A proposition is:

- a **tautology** if it is always true,
- a **contradiction** if it is always false,
- a **contingency** if it is neither a tautology or a contradiction,
- **satsfiable** if it is not a contradiction.

Example

- Contingency: It is raining
- Tautology: It is raining or it is not raining
- Contradiction: It is raining and it is not raining

Applications I: Constraint Satisfaction Problems

These are problems such as timetabling, activity planning, etc. Many can be understood as showing that a formula is satisfiable.

Example

You are planning a party, but your friends are a bit touchy about who will be there.

1. If John comes, he will get very hostile if Sarah is there.
2. Sarah will only come if Kim will be there also.
3. Kim says she will not come unless John does.

Who can you invite without making someone unhappy?

Translation to logic: let J, S, K represent "John (Sarah, Kim) comes to the party". Then the constraints are:

1. $J \to \neg S$
2. $S \to K$
3. $K \to J$

Thus, for a successful party to be possible, we want the formula $\phi = (J \to \neg S) \wedge (S \to K) \wedge (K \to J)$ to be satisfiable.
Truth values for J, S, K making this true are called *satisfying assignments*, or *models*.

We figure out where the conjuncts are false, below. (so blank = T)

J	K	S	$J \to \neg S$	$S \to K$	$K \to J$	ϕ
F	F	F				
F	F	T		F		F
F	T	F			F	F
F	T	T			F	F
T	F	F				
T	F	T	F	F		F
T	T	F				
T	T	T	F			F

Conclusion: a party satisfying the constraints can be held. Invite nobody, or invite John only, or invite Kim and John.

Logical equivalence

Definition

Two propositions are **logically equivalent** if they are true for the same truth values of their atomic propositions.

Example

$$A : \text{ “It is raining”}$$

is logically equivalent to ‘

$$\neg(\neg A) : \text{ “It is not the case that it is not raining”}$$

A	$\neg A$	$\neg(\neg A)$
True	False	True
False	True	False

Applications II: Program Logic

Example

```
if x > 0 or (x <= 0 and y > 100):
```

Let $p \overset{\text{def}}{=}$ `(x > 0)` and $q \overset{\text{def}}{=}$ `(y > 100)`

$p \lor (\neg p \land q)$

p	q	$\neg p$	$\neg p \land q$	$p \lor (\neg p \land q)$
F	F	T	F	F
F	T	T	T	T
T	F	F	F	T
T	T	F	F	T

This is equivalent to $p \lor q$. Hence the code can be simplified to

```
if x > 0 or y > 100:
```

Entailment and Validity

An *argument* consists of a set of propositionss called *premises* and a declarative sentence called the *conclusion*.

Example

Premises:	Frank took the Ford or the Toyota.
前提	If Frank took the Ford he will be late.
	Frank is not late.
Conclusion:	Frank took the Toyota

Entailment and Validity

An argument is *valid* if the conclusions are true *whenever* all the premises are true. Thus: if we believe the premises, we should also believe the conclusion.
(Note: we don't care what happens when one of the premises is false.)
Other ways of saying the same thing:

- The conclusion *logically follows* from the premises.
- The conclusion is a *logical consequence* of the premises.
- The premises **entail** the conclusion.

Entailment and Validity

The argument above is valid. The following is invalid:

Example

Premises:	Frank took the Ford or the Toyota.
	If Frank took the Ford he will be late.
	Frank is late.
Conclusion:	Frank took the Ford.

Applications III: Reasoning About Requirements/Specifications

Suppose a set of English language requirements R for a software/hardware system can be formalised by a set of formulae $\{\phi_1, \ldots \phi_n\}$.
Suppose C is a statement formalised by a formula ψ. Then

1. The requirements cannot be implemented if $\phi_1 \wedge \ldots \wedge \phi_n$ is not satisfiable.
2. If $\phi_1, \ldots \phi_n \models \psi$ then every correct implementation of the requirements R will be such that C is always true in the resulting system.
3. If $\phi_1, \ldots \phi_{n-1} \models \phi_n$, then the condition ϕ_n of the specification is redundant and need not be stated in the specification.

Example

Requirements R: A burglar alarm system for a house is to operate as follows. The alarm should not sound unless the system has been armed or there is a fire. If the system has been armed and a door is disturbed, the alarm should ring. Irrespective of whether the system has been armed, the alarm should go off when there is a fire.
Conclusion C: If the alarm is ringing and there is no fire, then the system must have been armed.

Questions

1. Will every system correctly implementing requirements R satisfy C?
2. Is the final sentence of the requirements redundant?

Example

Expressing the requirements as formulas of propositional logic, with

- S = the alarm sounds = the alarm rings
- A = the system is armed
- D = a door is disturbed
- F = there is a fire

we get

Requirements:

1. $S \rightarrow (A \vee F)$
2. $(A \wedge D) \rightarrow S$
3. $F \rightarrow S$

Conclusion: $(S \wedge \neg F) \rightarrow A$

Example

Our two questions then correspond to

1. Does $S \to (A \vee F)$, $(A \wedge D) \to S$, $F \to S$ entail $(S \wedge \neg F) \to A$?
2. Does $S \to (A \vee F)$, $(A \wedge D) \to S$ entail $F \to S$?

Summary of topics

- Historical background
- Applications to Computer Science
- Propositional logic, informally
- Propositional logic, formally

Syntax vs Semantics

The first step in the formal definition of logic is the separation of **syntax** and **semantics**

- Syntax is how things are written: what *defines* a formula
- Semantics is what things mean: what does it mean for a formula to be "true"?

Example

"Rabbit" and "Bunny" are syntactically different, but semantically the same.

Syntax: Well-formed formulas

Let $\textsc{Prop} = \{p, q, r, \ldots\}$ be a set of propositional letters. Consider the alphabet

$$\Sigma = \textsc{Prop} \cup \{\top, \bot, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,)\}.$$

The **well-formed formulas** (wffs) over $\textsc{Prop}$ is the smallest set of words over Σ such that:

- $\top$, $\bot$ and all elements of $\textsc{Prop}$ are wffs
- If φ is a wff then $\neg\varphi$ is a wff
- If φ and ψ are wffs then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, and $(\varphi \leftrightarrow \psi)$ are wffs.

Examples

The following are well-formed formulas:

- $(p \wedge \neg\top)$
- $\neg(p \wedge \neg\top)$
- $\neg\neg(p \wedge \neg\top)$

The following are **not** well-formed formulas:

- $p \wedge \wedge$
- $p \wedge \neg\top$
- $(p \wedge q \wedge r)$
- $\neg(\neg p)$

Syntax: Conventions

To aid readability some conventions and binding rules can and will be used.

- Parentheses omitted if there is no ambiguity (e.g. $p \wedge q$)
- $\neg$ binds more tightly than $\wedge$ and $\vee$, which bind more tightly than $\rightarrow$ and $\leftrightarrow$ (e.g. $p \wedge q \rightarrow r$ instead of $((p \wedge q) \rightarrow r)$

Syntax: Conventions

To aid readability some conventions and binding rules can and will be used.

- Parentheses omitted if there is no ambiguity (e.g. $p \wedge q$)
- $\neg$ binds more tightly than $\wedge$ and $\vee$, which bind more tightly than $\rightarrow$ and $\leftrightarrow$ (e.g. $p \wedge q \rightarrow r$ instead of $((p \wedge q) \rightarrow r)$

Other conventions (rarely used/assumed in this course):

- $'$ or $\bar{\cdot}$ for $\neg$
- $+$ for $\vee$
- $\cdot$ or juxtaposition for $\wedge$
- $\wedge$ binds more tightly than $\vee$
- $\wedge$ and $\vee$ associate to the left: $p \vee q \vee r$ instead of $((p \vee q) \vee r)$
- $\rightarrow$ and $\leftrightarrow$ associate to the right: $p \rightarrow q \rightarrow r$ instead of $(p \rightarrow (q \rightarrow r))$

Syntax: Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

Example

$$((P \land \neg Q) \lor \neg(Q \to P))$$

$$\lor$$

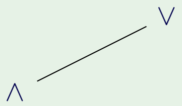
there is always one way to interpret a Syntax

Syntax: Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

Example

$$((P \land \neg Q) \lor \neg(Q \to P))$$

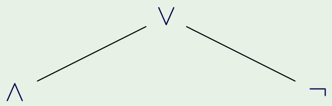


Syntax: Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

Example

$$((P \land \neg Q) \lor \neg(Q \to P))$$

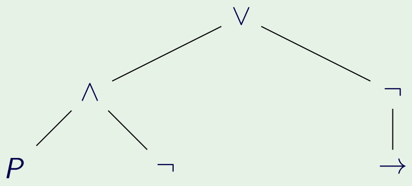


Syntax: Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

Example

$$((P \wedge \neg Q) \vee \neg(Q \rightarrow P))$$

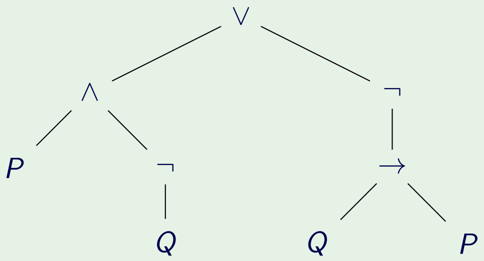


Syntax: Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

Example

$$((P \land \neg Q) \lor \neg(Q \to P))$$



Syntax: Parse trees formally

Formally, we can define a parse tree as follows:
A parse tree is either:

- (B) A node containing $\top$;
- (B) A node containing $\bot$;
- (B) A node containing a propositional variable;
- (R) A node containing $\neg$ with a single parse tree child;
- (R) A node containing $\wedge$ with two parse tree children;
- (R) A node containing $\vee$ with two parse tree children;
- (R) A node containing $\rightarrow$ with two parse tree children; or
- (R) A node containing $\leftrightarrow$ with two parse tree children.

Semantics: Boolean Algebra I

Let $\mathbb{B} = \{\texttt{true}, \texttt{false}\} = \{T, F\} = \{1, 0\}$

bang

Define !, &&, ||, $\rightsquigarrow$, $\leftrightsquigarrow$ as *operations* on $\mathbb{B}$ (as per slide 17)

- $!\texttt{true} = \texttt{false}$; $!\texttt{false} = \texttt{true}$,
- $\texttt{true}\ \&\&\ \texttt{true} = \texttt{true}$; ...
- $\texttt{true} \parallel \texttt{true} = \texttt{true}$; ...
- $\texttt{true} \rightsquigarrow \texttt{true} = \texttt{true}$; ...
- $\texttt{true} \leftrightsquigarrow \texttt{true} = \texttt{true}$; ...

Semantics: Truth valuations

A *truth assignment* is a function $v : \mathit{Prop} \to \mathbb{B}$.

We can extend a truth valuation, v, to all wffs of propositional logic as follows:

- $v(\top) = \texttt{true}$,
- $v(\bot) = \texttt{false}$,
- $v(\neg\phi) = !v(\phi)$,
- $v(\phi \wedge \psi) = v(\phi) \mathbin{\&\&} v(\psi)$
- $v(\phi \vee \psi) = v(\psi) \parallel v(\psi)$
- $v(\phi \to \psi) = v(\phi) \rightsquigarrow v(\psi)$
- $v(\phi \leftrightarrow \psi) = v(\phi) \leftrightsquigarrow v(\psi)$

Semantics: Exercises

Exercises

Evaluate the following formulae with the truth assignment $v(p) = v(q) = \texttt{false}$

- $p \to q$
- $(p \to q) \to (p \to q)$
- $\neg\neg p$
- $\top \wedge \neg\bot \to p$

Semantics: Truth tables

- Row for every **truth assignment** — assignment of T/F to elements of *Prop*
- Columns for subformulae

p	q	$\neg p$	$\neg p \wedge q$	$p \vee (\neg p \wedge q)$
F	F	T	F	F
F	T	T	T	T
T	F	F	F	T
T	T	F	F	T

Satisfiability, Validity and Equivalence

A formula φ is

真值指派

- **satisfiable** if $v(\varphi) = \texttt{true}$ for some truth assignment v (v **satisfies** φ)
- a **tautology** if $v(\varphi) = \texttt{true}$ for all truth assignments v
- **unsatisfiable** or a **contradiction** if $v(\varphi) = \texttt{false}$ for all truth assignments v

Exercise

$\boxed{\text{2.7.14 (supp)}}$

Which of the following formulae are *always* true?

(a) $(p \wedge (p \to q)) \to q$

(b) $((p \vee q) \wedge \neg p) \to \neg q$

(e) $((p \to q) \vee (q \to r)) \to (p \to r)$

(f) $(p \wedge q) \to q$

Exercise

$\boxed{\text{2.7.14 (supp)}}$

Which of the following formulae are *always* true?

(a) $(p \wedge (p \rightarrow q)) \rightarrow q$ — always true

(b) $((p \vee q) \wedge \neg p) \rightarrow \neg q$ — not always true

(e) $((p \rightarrow q) \vee (q \rightarrow r)) \rightarrow (p \rightarrow r)$ — not always true

(f) $(p \wedge q) \rightarrow q$ — always true

Logical equivalence

Definition

Two formulas, φ and ψ, are **logically equivalent**, $\varphi \equiv \psi$, if $v(\varphi) = v(\psi)$ for all truth assignments v.

Fact

$\equiv$ *is an equivalence relation.*

Logical equivalence

Definition

Two formulas, φ and ψ, are **logically equivalent**, $\varphi \equiv \psi$, if $v(\varphi) = v(\psi)$ for all truth assignments v.

Fact

$\equiv$ *is an equivalence relation.*

Example

- Commutativity: $(p \vee q) \equiv (q \vee p)$
- Double negation: $\neg\neg p \equiv p$
- Contrapositive: $(p \to q) \equiv (\neg q \to \neg p)$
- De Morgan's: $(p \vee q)' \equiv p' \wedge q'$

Logical equivalence

Fact

$\varphi \equiv \psi$ *if, and only if,* $(\varphi \leftrightarrow \psi)$ *is a tautology.*

Examples

$\boxed{2.2.18}$ Prove or disprove:
(a) $p \to (q \to r)$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $p \rightarrow (q \rightarrow r) \quad \equiv$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $p \to (q \to r) \equiv (p \to q) \to (p \to r)$

(c) $(p \to q) \to r$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $p \to (q \to r) \;\equiv\; (p \to q) \to (p \to r)$

(c) $(p \to q) \to r \;\equiv$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $p \to (q \to r) \equiv (p \to q) \to (p \to r)$

(c) $(p \to q) \to r \equiv p \to (q \to r)$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \rightarrow q) \rightarrow (p \rightarrow r)$

$\equiv$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\equiv \neg(p \to q) \vee (\neg p \vee r)$$

$$\equiv$$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\begin{aligned} &\equiv \neg(p \to q) \vee (\neg p \vee r) \\ &\equiv (p \wedge \neg q) \vee \neg p \vee r \\ &\equiv \end{aligned}$$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\begin{aligned}
&\equiv \neg(p \to q) \vee (\neg p \vee r) \\
&\equiv (p \wedge \neg q) \vee \neg p \vee r \\
&\equiv (p \vee \neg p \vee r) \wedge (\neg q \vee \neg p \vee r) \\
&\equiv
\end{aligned}$$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\begin{aligned}
&\equiv \neg(p \to q) \vee (\neg p \vee r) \\
&\equiv (p \wedge \neg q) \vee \neg p \vee r \\
&\equiv (p \vee \neg p \vee r) \wedge (\neg q \vee \neg p \vee r) \\
&\equiv \top \wedge (\neg p \vee \neg q \vee r) \\
&\equiv
\end{aligned}$$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\begin{aligned}
&\equiv \neg(p \to q) \vee (\neg p \vee r) \\
&\equiv (p \wedge \neg q) \vee \neg p \vee r \\
&\equiv (p \vee \neg p \vee r) \wedge (\neg q \vee \neg p \vee r) \\
&\equiv \top \wedge (\neg p \vee \neg q \vee r) \\
&\equiv p \to (\neg q \vee r) \\
&\equiv
\end{aligned}$$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \to q) \to (p \to r)$

$$\begin{aligned}
&\equiv \neg(p \to q) \vee (\neg p \vee r) \\
&\equiv (p \wedge \neg q) \vee \neg p \vee r \\
&\equiv (p \vee \neg p \vee r) \wedge (\neg q \vee \neg p \vee r) \\
&\equiv \top \wedge (\neg p \vee \neg q \vee r) \\
&\equiv p \to (\neg q \vee r) \\
&\equiv p \to (q \to r)
\end{aligned}$$

(c) $(p \to q) \to r$

Examples

$\boxed{2.2.18}$ Prove or disprove:

(a) $(p \rightarrow q) \rightarrow (p \rightarrow r)$

$$\begin{aligned}
&\equiv \neg(p \rightarrow q) \vee (\neg p \vee r) \\
&\equiv (p \wedge \neg q) \vee \neg p \vee r \\
&\equiv (p \vee \neg p \vee r) \wedge (\neg q \vee \neg p \vee r) \\
&\equiv \top \wedge (\neg p \vee \neg q \vee r) \\
&\equiv p \rightarrow (\neg q \vee r) \\
&\equiv p \rightarrow (q \rightarrow r)
\end{aligned}$$

(c) $(p \rightarrow q) \rightarrow r \;\equiv\; p \rightarrow (q \rightarrow r)$

Counterexample:

p	q	r	$(p \rightarrow q) \rightarrow r$	$p \rightarrow (q \rightarrow r)$
F	T	F	F	T

Theories and entailment

A set of formulas is a **theory**

A truth assignment v *satisfies* a theory T if $v(\varphi) = \texttt{true}$ for all $\varphi \in T$

A theory T **entails** a formula φ, $T \models \varphi$, if $v(\varphi) = \texttt{true}$ for all truth assignments v which satisfy T

Entailment example

For arguments in propositional logic, we can capture validity as follows:
Let $\phi_1, \ldots, \phi_n$ and ϕ be formulae of propositional logic. Draw a truth table with columns for each of $\phi_1, \ldots, \phi_n$ and ϕ.
The argument with premises $\phi_1, \ldots, \phi_n$ and conclusion ϕ is valid, denoted

$$\phi_1, \ldots, \phi_n \models \phi$$

if in every row of the truth table where $\phi_1, \ldots, \phi_n$ are all true, ϕ is true also.

Entailment example

Example

Premises:	Frank took the Ford or the Toyota.
	If Frank took the Ford he will be late.
	Frank is not late.
Conclusion:	Frank took the Toyota

Entailment example

We mark only true locations (blank = F)

Frd	*Tyta*	*Late*	$Frd \vee Tyta$	$Frd \rightarrow Late$	$\neg Late$	*Tyta*
F	F	F		T	T	
F	F	T		T		
F	T	F	T	T	T	T
F	T	T	T	T		T
T	F	F	T		T	
T	F	T	T	T		
T	T	F	T		T	T
T	T	T	T	T		T

This shows $Frd \vee Tyta,\ Frd \rightarrow Late,\ \neg Late \models Tyta$

The following row shows $Frd \vee Tyta$, $Frd \to Late$, $Late \not\models Frd$

Frd	$Tyta$	$Late$	$Frd \vee Tyta$	$Frd \to Late$	$Late$	Frd
F	T	T	T	T	T	F

Relating Entailment, Tautology and Logical equivalence

Theorem

The following are equivalent:

- $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$
- $\emptyset \models ((\varphi_1 \wedge \varphi_2) \wedge \ldots \varphi_n) \rightarrow \psi$
- $((\varphi_1 \wedge \varphi_2) \wedge \ldots \varphi_n) \rightarrow \psi$ *is a tautology*
- $\emptyset \models \varphi_1 \rightarrow (\varphi_2 \rightarrow (\ldots \rightarrow \varphi_n) \rightarrow \psi)) \ldots)$