CS:4350 Logic in Computer Science

Propositional Logic

Cesare Tinelli

Spring 2022

The University of Iowa

Credits

These slides are largely based on slides originally developed by **Andrei Voronkov** at the University of Manchester. Adapted by permission.

Propositional Logic

- **Syntax:** set of formulas built with propositional variables and connectives
- **Semantics:** formulas are assigned a Boolean value (true, false)
- **Inference system:** several

The sentences of the language (formulas) are also called *propositions*

Propositional Logic

- **Syntax:** set of formulas built with propositional variables and connectives
- **Semantics:** formulas are assigned a Boolean value (true, false)
- **Inference system:** several

The sentences of the language (formulas) are also called *propositions*

Propositions in mathematical logic

Formalize natural language statements that can be either true or false (but not both)

Basic propositions are called *atomic*

Examples:

1. $0 < 1$
2. Alan Turing was born in Manchester
3. $1 + 1 = 10$

More complex propositions are built from simpler ones via a small number of constructs

Examples:

1. If $0 < 1$ then Alan Turing was born in Manchester
2. $1 + 1 = 10$ or $1 + 1 \neq 10$

Propositions in mathematical logic

Formalize natural language statements that can be either true or false (but not both)

Basic propositions are called *atomic*

Examples:

1. $0 < 1$
2. Alan Turing was born in Manchester
3. $1 + 1 = 10$

More complex propositions are built from simpler ones via a small number of constructs

Examples:

1. If $0 < 1$ then Alan Turing was born in Manchester
2. $1 + 1 = 10$ or $1 + 1 \neq 10$

Propositions in mathematical logic

Formalize natural language statements that can be either true or false (but not both)

Basic propositions are called *atomic*

Examples:

1. $0 < 1$
2. Alan Turing was born in Manchester
3. $1 + 1 = 10$

More complex propositions are built from simpler ones via a small number of constructs

Examples:

1. If $0 < 1$ then Alan Turing was born in Manchester
2. $1 + 1 = 10$ or $1 + 1 \neq 10$

Propositions in mathematical logic

Formalize natural language statements that can be either true or false (but not both)

Basic propositions are called *atomic*

Examples:

1. $0 < 1$
2. Alan Turing was born in Manchester
3. $1 + 1 = 10$

More complex propositions are built from simpler ones via a small number of constructs

Examples:

1. If $0 < 1$ then Alan Turing was born in Manchester
2. $1 + 1 = 10$ or $1 + 1 \neq 10$

Truth of atomic propositions

Each proposition formalizes a statement that is either true or false

The *truth value* (true or false) of an atomic proposition P depends on P's *interpretation*

Example What is the truth value of the equality $1 + 1 = 10$?

- it is false, if we interpret 1 and 10 as integers in decimal notation (and $+$ as addition)
- it is true, if we interpret 1 and 10 as integers in binary notation (and $+$ as addition)

Truth of atomic propositions

Each proposition formalizes a statement that is either true or false

The *truth value* (true or false) of an atomic proposition P depends on P's *interpretation*

Example What is the truth value of the equality $1 + 1 = 10$?

- it is false, if we interpret 1 and 10 as integers in decimal notation (and $+$ as addition)
- it is true, if we interpret 1 and 10 as integers in binary notation (and $+$ as addition)

Truth of atomic propositions

Each proposition formalizes a statement that is either true or false

The *truth value* (true or false) of an atomic proposition P depends on P's *interpretation*

Example What is the truth value of the equality $1 + 1 = 10$?

- it is false, if we interpret 1 and 10 as integers in decimal notation (and $+$ as addition)
- it is true, if we interpret 1 and 10 as integers in binary notation (and $+$ as addition)

Truth of atomic propositions

Each proposition formalizes a statement that is either true or false

The *truth value* (true or false) of an atomic proposition P depends on P's *interpretation*

Example What is the truth value of the equality $1 + 1 = 10$?

- it is false, if we interpret 1 and 10 as integers in decimal notation (and $+$ as addition)
- it is true, if we interpret 1 and 10 as integers in binary notation (and $+$ as addition)

Truth of atomic propositions

Each proposition formalizes a statement that is either true or false

The *truth value* (true or false) of an atomic proposition P depends on P's *interpretation*

Example What is the truth value of the equality $1 + 1 = 10$?

- it is false, if we interpret 1 and 10 as integers in decimal notation (and $+$ as addition)
- it is true, if we interpret 1 and 10 as integers in binary notation (and $+$ as addition)

Truth of complex propositions

Consider a complex proposition P built
with a construct c from simpler propositions $S_1, \ldots, S_n$

The truth value of P univocally depends on

1. the meaning of c
2. the truth value of $S_1, \ldots, S_n$

More precisely, it is a function (determined by c) of the truth values of $S_1, \ldots, S_n$

Example

is true if at least one of $1 + 1 = 10$, $1 + 1 \neq 10$ is true

Truth of complex propositions

Consider a complex proposition P built
with a construct c from simpler propositions $S_1, \ldots, S_n$

The truth value of P univocally depends on

1. the meaning of c
2. the truth value of $S_1, \ldots, S_n$

More precisely, it is a function (determined by c) of the truth values of $S_1, \ldots, S_n$

Example

is true if at least one of $1 + 1 = 10$, $1 + 1 \neq 10$ is true

Truth of complex propositions

Consider a complex proposition P built
with a construct c from simpler propositions $S_1, \ldots, S_n$

The truth value of P univocally depends on

1. the meaning of c
2. the truth value of $S_1, \ldots, S_n$

More precisely, it is a function (determined by c) of the truth values of $S_1, \ldots, S_n$

Example

is true if at least one of $1 + 1 = 10$, $1 + 1 \neq 10$ is true

Truth of complex propositions

Consider a complex proposition P built
with a construct c from simpler propositions $S_1, \ldots, S_n$

The truth value of P univocally depends on

1. the meaning of c
2. the truth value of $S_1, \ldots, S_n$

More precisely, it is a function (determined by c) of the truth values of $S_1, \ldots, S_n$

Example

$$1 + 1 = 10 \text{ or } 1 + 1 \neq 10$$

is true if at least one of $1 + 1 = 10$, $1 + 1 \neq 10$ is true

Truth of complex propositions

Consider a complex proposition P built
with a construct c from simpler propositions $S_1, \ldots, S_n$

The truth value of P univocally depends on

1. the meaning of c
2. the truth value of $S_1, \ldots, S_n$

More precisely, it is a function (determined by c) of the truth values of $S_1, \ldots, S_n$

Example

$$\underbrace{1 + 1 = 10}_{S_1} \ \underbrace{\text{or}}_{c} \ \underbrace{1 + 1 \neq 10}_{S_2}$$

is true if at least one of $1 + 1 = 10$, $1 + 1 \neq 10$ is true

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \land \cdots \land A_n$ and $A_1 \lor \cdots \lor A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \to B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

The symbols $\top, \bot, \land, \lor, \neg, \to, \leftrightarrow$ are called *(logical) connectives*

Propositional logic: syntax

Assume a countable set of *propositional variables* ($\{p, p_1, p_2, \ldots, q, q_1, q_2, \ldots\}$)

Propositional formula:

- Every propositional variable (aka, *atom*) is a formula
- $\top$ and $\bot$ are formulas
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $A_1 \wedge \cdots \wedge A_n$ and $A_1 \vee \cdots \vee A_n$ are formulas
- If A is a formula, then $\neg A$ is a formula
- If A and B are formulas, then $A \rightarrow B$ and $A \leftrightarrow B$ are formulas
- Nothing else is a formula

The symbols $\top, \bot, \wedge, \vee, \neg, \rightarrow, \leftrightarrow$ are called *(logical) connectives*

Note: Some texts considers also $\oplus$ (exclusive or), $\downarrow$ (nor), and $\uparrow$ (nand)

Parsing expressions

In general, we use parentheses to disambiguate the parsing of expressions

Parenthesis clutter can be reduced by assigning precedence to operators

Example In arithmetic we know that the expression

$$x \cdot y + 2 \cdot z \quad \text{stands for} \quad (x \cdot y) + (2 \cdot z)$$

since $\cdot$ has a higher precedence than $+$

Parsing expressions

In general, we use parentheses to disambiguate the parsing of expressions

Parenthesis clutter can be reduced by assigning precedence to operators

Example In arithmetic we know that the expression

$$x \cdot y + 2 \cdot z \quad \text{stands for} \quad (x \cdot y) + (2 \cdot z)$$

since $\cdot$ has a higher precedence than $+$

Parsing expressions

In general, we use parentheses to disambiguate the parsing of expressions

Parenthesis clutter can be reduced by assigning precedence to operators

Example In arithmetic we know that the expression

$$x \cdot y + 2 \cdot z \quad \text{stands for} \quad (x \cdot y) + (2 \cdot z)$$

since $\cdot$ has a higher precedence than $+$

Propositional connectives and their precedence

Connective	Name	Precedence
$\top$	verum (top)	
$\bot$	falsum (bottom)	
$\neg$	negation	5
$\wedge$	conjunction	4
$\vee$	disjunction	3
$\rightarrow$	implication	2
$\leftrightarrow$	equivalence	1

Implication is right-associative:

$A \rightarrow B \rightarrow C$ is parsed as

$A \rightarrow (B \rightarrow C)$

Propositional connectives and their precedence

Connective	Name	Precedence
$\top$	verum (top)	
$\bot$	falsum (bottom)	
$\neg$	negation	5
$\wedge$	conjunction	4
$\vee$	disjunction	3
$\rightarrow$	implication	2
$\leftrightarrow$	equivalence	1

Implication is right-associative:

$A \rightarrow B \rightarrow C$ is parsed as
$A \rightarrow (B \rightarrow C)$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$\neg A \quad \wedge B \quad \rightarrow \quad C \vee D \quad \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$\neg A \quad \wedge B \quad \rightarrow \quad C \vee D \quad \leftrightarrow E$$

Parsing Formulas

Let us parse

$$\neg A \land B \to C \lor D \leftrightarrow E$$

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\land$	4
$\lor$	3
$\to$	2
$\leftrightarrow$	1

Inside-out, starting with the highest precedence connectives:

$$\neg A \quad \land B \quad \to \quad C \lor D \quad \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$\neg A \quad \land B \quad \to \quad C \lor D \quad \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\land$	4
$\lor$	3
$\to$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \land B \to C \lor D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$(((\neg A) \land B) \to (C \lor D)) \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$\neg A \land B \to C \lor D \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$\neg A \quad \wedge B \quad \rightarrow \quad C \vee D \quad \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$\neg A \quad \wedge B \quad \rightarrow \quad C \vee D \quad \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\to$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \to C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$\neg A \quad \wedge B \quad \to \quad C \vee D \quad \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$(((\neg A) \wedge B) \to (C \vee D)) \leftrightarrow E$$

Parsing Formulas

Connective	Precedence
$\top$	
$\bot$	
$\neg$	5
$\wedge$	4
$\vee$	3
$\rightarrow$	2
$\leftrightarrow$	1

Let us parse

$$\neg A \wedge B \rightarrow C \vee D \leftrightarrow E$$

Inside-out, starting with the highest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Outside-in, starting with the lowest precedence connectives:

$$(((\neg A) \wedge B) \rightarrow (C \vee D)) \leftrightarrow E$$

Semantics, Interpretation

In arithmetic the meaning of expressions with variables like

$$x \cdot y + 2 \cdot z$$

is defined as follows:

1. fix a mapping from variables to (integer) values,
   e.g., $\{ x \mapsto 1, y \mapsto 8, z \mapsto -3 \}$
2. then, under this mapping the expression has value 2

Semantics, Interpretation

In arithmetic the meaning of expressions with variables like

$$x \cdot y + 2 \cdot z$$

is defined as follows:

1. fix a mapping from variables to (integer) values,
   e.g., $\{ x \mapsto 1, y \mapsto 8, z \mapsto -3 \}$
2. then, under this mapping the expression has value 2

Semantics, Interpretation

In arithmetic the meaning of expressions with variables like

$$x \cdot y + 2 \cdot z$$

is defined as follows:

1. fix a mapping from variables to (integer) values,
   e.g., $\{ x \mapsto 1, y \mapsto 8, z \mapsto -3 \}$
2. then, under this mapping the expression has value 2

Semantics, Interpretation

In arithmetic the meaning of expressions with variables like

$$x \cdot y + 2 \cdot z$$

is c

1.

> In other words:
> we can determine the value of an arithmetic expression once we interpret its variables as specific values

2. then, under this mapping the expression has value 2

Semantics, Interpretation

Similarly,

the semantics of propositional formulas can be defined only after assigning values to variables

- There are two *Boolean/truth values*:
  true (denoted by 1) and
  false (denoted by 0)
- An *interpretation* for a set P of propositional variables is a mapping $\mathcal{I} : P \to \mathcal{B}$ where $\mathcal{B} = \{0, 1\}$
- Interpretations are also called *truth assignments*

Semantics, Interpretation

Similarly,

the semantics of propositional formulas can be defined only after assigning values to variables

- There are two *Boolean/truth values*:
  true (denoted by 1) and
  false (denoted by 0)
- An *interpretation* for a set P of propositional variables is a mapping $\mathcal{I} : P \to \mathcal{B}$ where $\mathcal{B} = \{0, 1\}$
- Interpretations are also called *truth assignments*

Semantics, Interpretation

Similarly,

the semantics of propositional formulas can be defined only after assigning values to variables

- There are two *Boolean/truth values*:
  true (denoted by 1) and
  false (denoted by 0)
- An *interpretation* for a set P of propositional variables is a mapping $\mathcal{I} : P \rightarrow \mathcal{B}$ where $\mathcal{B} = \{ 0, 1 \}$
- Interpretations are also called *truth assignments*

Semantics, Interpretation

Similarly,

the semantics of propositional formulas can be defined only after assigning values to variables

- There are two *Boolean/truth values*:
  true (denoted by 1) and
  false (denoted by 0)
- An *interpretation* for a set P of propositional variables is a mapping $\mathcal{I} : P \to \mathcal{B}$ where $\mathcal{B} = \{ 0, 1 \}$
- Interpretations are also called *truth assignments*

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \wedge \cdots \wedge A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \vee \cdots \vee A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \rightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Interpreting formulas

The truth value of a complex formula is determined by the truth values of its components

An interpretation $\mathcal{I}$ extends to a mapping from all formulas to truth values as follows:

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(A_1 \land \cdots \land A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for all i
3. $\mathcal{I}(A_1 \lor \cdots \lor A_n) = 1$ iff $\mathcal{I}(A_i) = 1$ for some i
4. $\mathcal{I}(\neg A) = 1$ iff $\mathcal{I}(A) = 0$
5. $\mathcal{I}(A_1 \to A_2) = 1$ iff $\mathcal{I}(A_1) = 0$ or $\mathcal{I}(A_2) = 1$
6. $\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(A_2)$

Operation tables

$\mathcal{I}(A_1 \vee A_2) = 1$ iff $\mathcal{I}(A_1) = 1$ or $\mathcal{I}(A_2) = 1$

$\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(B_2)$

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

$\neg$	
1	0
0	1

$\rightarrow$	1	0
1	1	0
0	1	1

$\leftrightarrow$	1	0
1	1	0
0	0	1

Therefore, every connective can be considered as a function on truth values ($\neg : \mathcal{B} \rightarrow \mathcal{B},\ \wedge : \mathcal{B}^2 \rightarrow \mathcal{B},\ \ldots$)

Operation tables

$\mathcal{I}(A_1 \vee A_2) = 1$ iff $\mathcal{I}(A_1) = 1$ or $\mathcal{I}(A_2) = 1$

$\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(B_2)$

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

$\neg$	
1	0
0	1

$\rightarrow$	1	0
1	1	0
0	1	1

$\leftrightarrow$	1	0
1	1	0
0	0	1

Therefore, every connective can be considered as a function on truth values ($\neg : \mathcal{B} \rightarrow \mathcal{B}$, $\wedge : \mathcal{B}^2 \rightarrow \mathcal{B}$, $\ldots$)

Operation tables

$\mathcal{I}(A_1 \vee A_2) = 1$ iff $\mathcal{I}(A_1) = 1$ or $\mathcal{I}(A_2) = 1$

$\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(B_2)$

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

$\neg$	
1	0
0	1

$\rightarrow$	1	0
1	1	0
0	1	1

$\leftrightarrow$	1	0
1	1	0
0	0	1

Therefore, every connective can be considered as a function on truth values ($\neg : \mathcal{B} \rightarrow \mathcal{B}$, $\wedge : \mathcal{B}^2 \rightarrow \mathcal{B}$, $\ldots$)

Operation tables

$\mathcal{I}(A_1 \vee A_2) = 1$ iff $\mathcal{I}(A_1) = 1$ or $\mathcal{I}(A_2) = 1$

$\mathcal{I}(A_1 \leftrightarrow A_2) = 1$ iff $\mathcal{I}(A_1) = \mathcal{I}(B_2)$

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

$\neg$	
1	0
0	1

$\rightarrow$	1	0
1	1	0
0	1	1

$\leftrightarrow$	1	0
1	1	0
0	0	1

Therefore, every connective can be considered as a function on truth values ($\neg : \mathcal{B} \rightarrow \mathcal{B}$, $\wedge : \mathcal{B}^2 \rightarrow \mathcal{B}$, $\ldots$)

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A

- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A

- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise

- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise

- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Satisfiability, validity, equivalence

Let A and B be two formulas and $\mathcal{I}$ an interpretation

- If $\mathcal{I}(A) = 1$, we write $\mathcal{I} \models A$ and say, equivalently, that A is *true* in $\mathcal{I}$, $\mathcal{I}$ *satisfies* A, or $\mathcal{I}$ is a *model* of A
- If $\mathcal{I}(A) = 0$, we write $\mathcal{I} \not\models A$ and say, equivalently, that A is *false* in $\mathcal{I}$, $\mathcal{I}$ *falsifies* A, or $\mathcal{I}$ is *not a model* of A
- A is *satisfiable* if it is true in some interpretation, and is *unsatisfiable* otherwise
- A is *valid*, or a *tautology*, if it is true in every interpretation, and is *invalid*, or *falsifiable*, otherwise
- A and B are *equivalent*, written $A \equiv B$, if they have exactly the same models

Examples

p, q propositional variables A, B propositional formulas

- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all satisfiable
- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all falsifiable
- $A \to A,\ A \vee \neg A,\ A \to (B \to A)$ are all valid

Examples

p, q propositional variables $\qquad A, B$ propositional formulas

- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all satisfiable
- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all falsifiable
- $A \to A,\ A \vee \neg A,\ A \to (B \to A)$ are all valid

Examples

p, q propositional variables $\qquad A, B$ propositional formulas

- $p,\ p \rightarrow q,\ p \vee \neg q,\ (p \rightarrow q) \rightarrow p$ are all satisfiable
- $p,\ p \rightarrow q,\ p \vee \neg q,\ (p \rightarrow q) \rightarrow p$ are all falsifiable
- $A \rightarrow A,\ A \vee \neg A,\ A \rightarrow (B \rightarrow A)$ are all valid

Examples

p, q propositional variables $\quad$ A, B propositional formulas

- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all satisfiable
- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all falsifiable
- $A \to A,\ A \vee \neg A,\ A \to (B \to A)$ are all valid

Examples

p, q propositional variables $\quad A, B$ propositional formulas

- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all satisfiable
- $p,\ p \to q,\ p \vee \neg q,\ (p \to q) \to p$ are all falsifiable
- $A \to A,\ A \vee \neg A,\ A \to (B \to A)$ are all valid

Note:

- $\top$ is valid
- $\bot$ is unsatisfiable
- Every valid formula is satisfiable
- Every unsatisfiable formula is falsifiable

Examples: equivalences

For all formulas A and B, the following equivalences hold:

$$
\begin{aligned}
A \to \bot &\equiv \neg A && (1)\\
\top \to A &\equiv A && (2)\\
A \to B &\equiv \neg A \lor B && (3)\\
&\equiv \neg(A \land \neg B) && (4)\\
A \land B &\equiv \neg(\neg A \lor \neg B) && (5)\\
A \lor B &\equiv \neg A \to B && (6)\\
A \to A &\equiv \top && (7)\\
A \land \neg A &\equiv \bot && (8)
\end{aligned}
$$

Connections between these notions

For all formulas A and B,

1. A is valid iff $\neg A$ is unsatisfiable
2. A is satisfiable iff $\neg A$ is falsifiable
3. A is valid iff A is equivalent to $\top$
4. A and B are equivalent iff $A \leftrightarrow B$ is valid
5. A and B are equivalent iff $\neg(A \leftrightarrow B)$ is unsatisfiable
6. A is satisfiable iff A is not equivalent to $\bot$

Connections between these notions

For all formulas A and B,

1. A is valid iff $\neg A$ is unsatisfiable
2. A is satisfiable iff $\neg A$ is falsifiable

3. A is valid iff A is equivalent to $\top$
4. A and B are equivalent iff $A \leftrightarrow B$ is valid

5. A and B are equivalent iff $\neg(A \leftrightarrow B)$ is unsatisfiable
6. A is satisfiable iff A is not equivalent to $\bot$

Connections between these notions

For all formulas A and B,

1. A is valid iff $\neg A$ is unsatisfiable
2. A is satisfiable iff $\neg A$ is falsifiable

3. A is valid iff A is equivalent to $\top$
4. A and B are equivalent iff $A \leftrightarrow B$ is valid

5. A and B are equivalent iff $\neg(A \leftrightarrow B)$ is unsatisfiable
6. A is satisfiable iff A is not equivalent to $\bot$

Connections between these notions

For all formulas A and B,

1. A is valid iff $\neg A$ is unsatisfiable
2. A is satisfiable iff $\neg A$ is falsifiable

3. A is valid iff A is equivalent to $\top$
4. A and B are equivalent iff $A \leftrightarrow B$ is valid

5. A and B are equivalent iff $\neg(A \leftrightarrow B)$ is unsatisfiable
6. A is satisfiable iff A is not equivalent to $\bot$

Syntactic vs. semantic symbols

For all formulas A and B,

- A is valid iff $A \equiv \top$
- $A \leftrightarrow B$ is valid iff $(A \leftrightarrow B) \equiv \top$

So, what is the difference between $\equiv$ and $\leftrightarrow$?

$\leftrightarrow$ is a connective in the language of propositional logic

$\equiv$ is mathematical notation to express formula equivalence

Syntactic vs. semantic symbols

For all formulas A and B,

- A is valid iff $A \equiv \top$
- $A \leftrightarrow B$ is valid iff $(A \leftrightarrow B) \equiv \top$

So, what is the difference between $\equiv$ and $\leftrightarrow$?

$\leftrightarrow$ is a connective in the language of propositional logic

$\equiv$ is mathematical notation to express formula equivalence

Syntactic vs. semantic symbols

For all formulas A and B,

- A is valid iff $A \equiv \top$
- $A \leftrightarrow B$ is valid iff $(A \leftrightarrow B) \equiv \top$

So, what is the difference between $\equiv$ and $\leftrightarrow$?

$\leftrightarrow$ is a connective in the language of propositional logic

$\equiv$ is mathematical notation to express formula equivalence

Syntactic vs. semantic symbols

For all formulas A and B,

- A is valid iff $A \equiv \top$
- $A \leftrightarrow B$ is valid iff $(A \leftrightarrow B) \equiv \top$

So, what is the difference between $\equiv$ and $\leftrightarrow$?

$\leftrightarrow$ is a connective in the language of propositional logic

$A \leftrightarrow B$ is formula of the logic

$\equiv$ is mathematical notation to express formula equivalence

$A \equiv B$ is a shorthand for a statement about the interpretations of A and B

How to evaluate a formula?

Let's evaluate the formula

$$A = (p \to q) \land (p \land q \to r) \to (p \to r)$$

in the interpretation

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r)$	1
$p \rightarrow r$	1
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r)$	0
$p \wedge q \rightarrow r$	1
$p \rightarrow q$	0
$p \wedge q$	0
p	1
q	0
r	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
p p p	1
q q	0
r r	1

$$\mathcal{I} = \{\, p \mapsto 1, q \mapsto 0, r \mapsto 1 \,\}$$

Evaluating a formula

formula	value
$(p \to q) \wedge (p \wedge q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \wedge (p \wedge q \to r)$	0
$p \wedge q \to r$	1
$p \to q$	0
$p \wedge q$	0
p	1
q	0
r	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
p	1
q	0
r	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{\, p \mapsto 1, q \mapsto 0, r \mapsto 1 \,\}$$

Evaluating a formula

formula	value
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r)$	1
$p \rightarrow r$	1
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r)$	0
$p \wedge q \rightarrow r$	1
$p \rightarrow q$	0
$p \wedge q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{\, p \mapsto 1, q \mapsto 0, r \mapsto 1 \,\}$$

Evaluating a formula

formula	value
$(p \to q) \wedge (p \wedge q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \wedge (p \wedge q \to r)$	0
$p \wedge q \to r$	1
$p \to q$	0
$p \wedge q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r)$	1
$p \rightarrow r$	1
$(p \rightarrow q) \wedge (p \wedge q \rightarrow r)$	0
$p \wedge q \rightarrow r$	1
$p \rightarrow q$	0
$p \wedge q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
p p p	1
q q	0
r r	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Evaluating a formula

formula	value
$(p \to q) \land (p \land q \to r) \to (p \to r)$	1
$p \to r$	1
$(p \to q) \land (p \land q \to r)$	0
$p \land q \to r$	1
$p \to q$	0
$p \land q$	0
$p \quad p \quad p$	1
$q \quad q$	0
$r \quad r$	1

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

So the formula is true in interpretation $\mathcal{I}$

Equivalent replacement

Let $B[A]$ denote a formula B with a fixed occurrence of a subformula A

Let $B[A']$ then denote the formula obtained from B by replacing that occurrence of A by A'

Equivalent replacement

Let $B[A]$ denote a formula B with a fixed occurrence of a subformula A

Let $B[A']$ then denote the formula obtained from B by replacing that occurrence of A by A'

Example

$$B = (p_1 \wedge p_2) \vee (p_1 \wedge p_3) \tag{9}$$

$$A = p_1 \wedge p_3 \tag{10}$$

$$A' = p_1 \vee \neg p_4 \tag{11}$$

$$B[A'] = (p_1 \wedge p_2) \vee (p_1 \vee \neg p_4) \tag{12}$$

Equivalent replacement

Let $B[A]$ denote a formula B with a fixed occurrence of a subformula A

Let $B[A']$ then denote the formula obtained from B by replacing that occurrence of A by A'

Lemma 1 (Equivalent Replacement)
Let $\mathcal{I}$ be an interpretation and $\mathcal{I} \models A_1 \leftrightarrow A_2$. Then $\mathcal{I} \models B[A_1] \leftrightarrow B[A_2]$.

Equivalent replacement

Let $B[A]$ denote a formula B with a fixed occurrence of a subformula A

Let $B[A']$ then denote the formula obtained from B by replacing that occurrence of A by A'

Lemma 1 (Equivalent Replacement)
Let $\mathcal{I}$ be an interpretation and $\mathcal{I} \models A_1 \leftrightarrow A_2$. Then $\mathcal{I} \models B[A_1] \leftrightarrow B[A_2]$.

Theorem 2 (Equivalent Replacement)
Let $A_1 \equiv A_2$. Then $B[A_1] \equiv B[A_2]$.

Equivalent replacement

Let $B[A]$ denote a formula B with a fixed occurrence of a subformula A

Let $B[A']$ then denote the formula obtained from B by replacing that occurrence of A by A'

Lemma 1 (Equivalent Replacement)

Let $\mathcal{I}$ be an interpretation and $\mathcal{I} \models A_1 \leftrightarrow A_2$. Then $\mathcal{I} \models B[A_1] \leftrightarrow B[A_2]$.

Theorem 2 (Equivalent Replacement)

Let $A_1 \equiv A_2$. Then $B[A_1] \equiv B[A_2]$.

Thanks to compositionality!

A purely syntactic formula evaluation algorithm

Let $\mathcal{I}$ be an interpretation

Note:

- If $\mathcal{I} \models p$ then $\mathcal{I} \models p \leftrightarrow \top$
- If $\mathcal{I} \not\models p$ then $\mathcal{I} \models p \leftrightarrow \bot$

By the previous lemma, we can replace a subformula by a formula with the same value

Hence, we can replace every atom p by either $\top$ or $\bot$, depending on the value of p in $\mathcal{I}$

A purely syntactic formula evaluation algorithm

Let $\mathcal{I}$ be an interpretation

Note:

- If $\mathcal{I} \models p$ then $\mathcal{I} \models p \leftrightarrow \top$
- If $\mathcal{I} \not\models p$ then $\mathcal{I} \models p \leftrightarrow \bot$

By the previous lemma, we can replace a subformula by a formula with the same value

Hence, we can replace every atom p by either $\top$ or $\bot$, depending on the value of p in $\mathcal{I}$

A purely syntactic formula evaluation algorithm

Let $\mathcal{I}$ be an interpretation

Note:

- If $\mathcal{I} \models p$ then $\mathcal{I} \models p \leftrightarrow \top$
- If $\mathcal{I} \not\models p$ then $\mathcal{I} \models p \leftrightarrow \bot$

By the previous lemma, we can replace a subformula by a formula with the same value

Hence, we can replace every atom p by either $\top$ or $\bot$, depending on the value of p in $\mathcal{I}$

Rewrite rules for evaluating a formula

Consider a formula whose atoms consist only of $\bot$ and $\top$

Any such formula, other than $\bot$ and $\top$, can be rewritten to a smaller, equivalent formula

Examples

- $A \to \top$ is equivalent to $\top$
- $A \lor \bot$ is equivalent to A

This simplification process can be formalized as a rewrite rule system

Rewrite rules for evaluating a formula

Consider a formula whose atoms consist only of $\bot$ and $\top$

Any such formula, other than $\bot$ and $\top$, can be rewritten to a smaller, equivalent formula

Examples

- $A \to \top$ is equivalent to $\top$
- $A \lor \bot$ is equivalent to A

This simplification process can be formalized as a rewrite rule system

Rewrite rules for evaluating a formula

Consider a formula whose atoms consist only of $\bot$ and $\top$

Any such formula, other than $\bot$ and $\top$, can be rewritten to a smaller, equivalent formula

Examples

- $A \to \top$ is equivalent to $\top$
- $A \lor \bot$ is equivalent to A

This simplification process can be formalized as a rewrite rule system

Rewrite rules for evaluating a formula

Consider a formula whose atoms consist only of $\bot$ and $\top$

Any such formula, other than $\bot$ and $\top$, can be rewritten to a smaller, equivalent formula

Examples

- $A \to \top$ is equivalent to $\top$
- $A \lor \bot$ is equivalent to A

This simplification process can be formalized as a rewrite rule system

Rewrite system for formula evaluation

$$\begin{array}{rcl} \top \wedge \cdots \wedge \top & \Rightarrow & \top \\ \bot \vee \cdots \vee \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{rcl} A_1 \wedge \cdots \wedge \bot \wedge \cdots \wedge A_n & \Rightarrow & \bot \\ A_1 \vee \cdots \vee \top \vee \cdots \vee A_n & \Rightarrow & \top \end{array}$$

$$\begin{array}{rcl} \neg\top & \Rightarrow & \bot \\ \neg\bot & \Rightarrow & \top \end{array}$$

$$\begin{array}{rcl} A \rightarrow \top & \Rightarrow & \top \\ \bot \rightarrow A & \Rightarrow & \top \\ \top \rightarrow \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{rcl} \top \leftrightarrow \top & \Rightarrow & \top \\ \top \leftrightarrow \bot & \Rightarrow & \bot \\ \bot \leftrightarrow \top & \Rightarrow & \bot \\ \bot \leftrightarrow \bot & \Rightarrow & \top \end{array}$$

$\Rightarrow$ is a *rewrite relation*

Rewrite system for formula evaluation

$$\top \wedge \cdots \wedge \top \Rightarrow \top$$
$$\bot \vee \cdots \vee \bot \Rightarrow \bot$$

$$A_1 \wedge \cdots \wedge \bot \wedge \cdots \wedge A_n \Rightarrow \bot$$
$$A_1 \vee \cdots \vee \top \vee \cdots \vee A_n \Rightarrow \top$$

$$\neg\top \Rightarrow \bot$$
$$\neg\bot \Rightarrow \top$$

$$A \rightarrow \top \Rightarrow \top$$
$$\bot \rightarrow A \Rightarrow \top$$
$$\top \rightarrow \bot \Rightarrow \bot$$

$$\top \leftrightarrow \top \Rightarrow \top$$
$$\top \leftrightarrow \bot \Rightarrow \bot$$
$$\bot \leftrightarrow \top \Rightarrow \bot$$
$$\bot \leftrightarrow \bot \Rightarrow \top$$

$\Rightarrow$ is a *rewrite relation*

Rewrite system for formula evaluation

$$\begin{array}{lcl} \top \wedge \cdots \wedge \top & \Rightarrow & \top \\ \bot \vee \cdots \vee \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{lcl} A_1 \wedge \cdots \wedge \bot \wedge \cdots \wedge A_n & \Rightarrow & \bot \\ A_1 \vee \cdots \vee \top \vee \cdots \vee A_n & \Rightarrow & \top \end{array}$$

$$\begin{array}{lcl} \neg\top & \Rightarrow & \bot \\ \neg\bot & \Rightarrow & \top \end{array}$$

$$\begin{array}{lcl} A \rightarrow \top & \Rightarrow & \top \\ \bot \rightarrow A & \Rightarrow & \top \\ \top \rightarrow \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{lcl} \top \leftrightarrow \top & \Rightarrow & \top \\ \top \leftrightarrow \bot & \Rightarrow & \bot \\ \bot \leftrightarrow \top & \Rightarrow & \bot \\ \bot \leftrightarrow \bot & \Rightarrow & \top \end{array}$$

$\Rightarrow$ is a *rewrite relation*

Writing $B \Rightarrow B'$ means that B can be rewritten to B' in one step using one of the rules above

Rewrite system for formula evaluation

$$\begin{array}{lcl} \top \wedge \cdots \wedge \top & \Rightarrow & \top \\ \bot \vee \cdots \vee \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{lcl} A_1 \wedge \cdots \wedge \bot \wedge \cdots \wedge A_n & \Rightarrow & \bot \\ A_1 \vee \cdots \vee \top \vee \cdots \vee A_n & \Rightarrow & \top \end{array}$$

$$\begin{array}{lcl} \neg\top & \Rightarrow & \bot \\ \neg\bot & \Rightarrow & \top \end{array}$$

$$\begin{array}{lcl} A \rightarrow \top & \Rightarrow & \top \\ \bot \rightarrow A & \Rightarrow & \top \\ \top \rightarrow \bot & \Rightarrow & \bot \end{array}$$

$$\begin{array}{lcl} \top \leftrightarrow \top & \Rightarrow & \top \\ \top \leftrightarrow \bot & \Rightarrow & \bot \\ \bot \leftrightarrow \top & \Rightarrow & \bot \\ \bot \leftrightarrow \bot & \Rightarrow & \top \end{array}$$

$\Rightarrow$ is a *rewrite relation*

A formula A is in *normal form* (wrt $\Rightarrow$) if it cannot be rewritten by any of the rules above

A syntactic evaluation algorithm

evaluate evaluates any formula G in any interpretation $\mathcal{I}$ using the previous rewrite system

procedure *evaluate*$(G, \mathcal{I})$
input: formula G, interpretation $\mathcal{I}$
output: the boolean value $\mathcal{I}(G)$
begin
  forall atoms p occurring in G
    if $\mathcal{I} \models p$
      then replace all occurrences of p in G by $\top$
      else replace all occurrences of p in G by $\bot$
  rewrite G into a normal form using the rewrite rules
  if $G = \top$ **then return** 1 **else return** 0
end

A syntactic evaluation algorithm

evaluate evaluates any formula G in any interpretation $\mathcal{I}$ using the previous rewrite system

procedure *evaluate*$(G, \mathcal{I})$
input: formula G, interpretation $\mathcal{I}$
output: the boolean value $\mathcal{I}(G)$
begin
  forall atoms p occurring in G
    if $\mathcal{I} \models p$
      then replace all occurrences of p in G by $\top$
      else replace all occurrences of p in G by $\bot$
  rewrite G into a normal form using the rewrite rules
  if $G = \top$ **then return** 1 **else return** 0
end

A syntactic evaluation algorithm

evaluate evaluates any formula G in any interpretation $\mathcal{I}$ using the previous rewrite system

procedure $\mathit{evaluate}(G, \mathcal{I})$
input: formula G, interpretation $\mathcal{I}$
output: the boolean value $\mathcal{I}(G)$
begin
 forall atoms p occurring in G
  if $\mathcal{I} \models p$
   then replace all occurrences of p in G by $\top$
   else replace all occurrences of p in G by $\perp$
 rewrite G into a normal form using the rewrite rules
 if $G = \top$ **then return** 1 **else return** 0
end

A syntactic evaluation algorithm

evaluate evaluates any formula G in any interpretation $\mathcal{I}$ using the previous rewrite system

procedure *evaluate*$(G, \mathcal{I})$
input: formula G, interpretation $\mathcal{I}$
output: the boolean value $\mathcal{I}(G)$
begin
  forall atoms p occurring in G
    if $\mathcal{I} \models p$
      then replace all occurrences of p in G by $\top$
      else replace all occurrences of p in G by $\bot$
  rewrite G into a normal form using the rewrite rules
  if $G = \top$ **then return** 1 **else return** 0
end

Example

Let us evaluate the formula

$$G = (p \to q) \land (p \land q \to r) \to (p \to r)$$

in the interpretation

$$\mathcal{I} = \{\, p \mapsto 1, q \mapsto 0, r \mapsto 1 \,\}$$

Its value is equal to the value of

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

Example

Let us evaluate the formula

$$G = (p \to q) \land (p \land q \to r) \to (p \to r)$$

in the interpretation

$$\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$$

Its value is equal to the value of

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \wedge \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land (\top \land \bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land (\bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land \top \to (\top \to \top) &\Rightarrow \\
\bot \to (\top \to \top) &\Rightarrow \\
\bot \to \top &\Rightarrow \\
\top &
\end{aligned}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
&(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land \top \to (\top \to \top) \Rightarrow \\
&\bot \to (\top \to \top) \Rightarrow \\
&\bot \to \top \Rightarrow \\
&\top
\end{aligned}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
&(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land \top \to (\top \to \top) \Rightarrow \\
&\bot \to (\top \to \top) \Rightarrow \\
&\bot \to \top \Rightarrow \\
&\top
\end{aligned}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land (\top \land \bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land (\bot \to \top) \to (\top \to \top) &\Rightarrow \\
\bot \land \top \to (\top \to \top) &\Rightarrow \\
\bot \to (\top \to \top) &\Rightarrow \\
\bot \to \top &\Rightarrow \\
\top &
\end{aligned}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{gathered}
(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \wedge (\bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \wedge \top \to (\top \to \top) \Rightarrow \\
\bot \to (\top \to \top) \Rightarrow \\
\bot \to \top \Rightarrow \\
\top
\end{gathered}
$$

1. $A \wedge \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
&(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow \\
&\bot \land \top \to (\top \to \top) \Rightarrow \\
&\bot \to (\top \to \top) \Rightarrow \\
&\bot \to \top \Rightarrow \\
&\top
\end{aligned}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{array}{c}
(\top \rightarrow \bot) \wedge (\top \wedge \bot \rightarrow \top) \rightarrow (\top \rightarrow \top) \Rightarrow \\
\bot \wedge (\top \wedge \bot \rightarrow \top) \rightarrow (\top \rightarrow \top) \Rightarrow \\
\bot \wedge (\bot \rightarrow \top) \rightarrow (\top \rightarrow \top) \Rightarrow \\
\bot \wedge \top \rightarrow (\top \rightarrow \top) \Rightarrow \\
\bot \rightarrow (\top \rightarrow \top) \Rightarrow \\
\bot \rightarrow \top \Rightarrow \\
\top
\end{array}
$$

1. $A \wedge \bot \Rightarrow \bot$
2. $\top \rightarrow \bot \Rightarrow \bot$
3. $A \rightarrow \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \rightarrow \bot) \wedge (\top \wedge \bot \rightarrow \top) \rightarrow (\top \rightarrow \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top)$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \wedge \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to \top \Rightarrow$$
$$\top$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$(\top \to \bot) \land (\top \land \bot \to \top) \to \top \Rightarrow$$
$$\top$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Inside-out, left-to-right:

$$
\begin{gathered}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land \top \to (\top \to \top) \Rightarrow \\
\bot \to (\top \to \top) \Rightarrow \\
\bot \to \top \Rightarrow \\
\top
\end{gathered}
$$

Outside-in, right-to-left:

$$
\begin{gathered}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
(\top \to \bot) \land (\top \land \bot \to \top) \to \top \Rightarrow \\
\top
\end{gathered}
$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \wedge \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \wedge \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to \top \Rightarrow$$
$$\top$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{array}{c}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow \\
\bot \land \top \to (\top \to \top) \Rightarrow \\
\bot \to (\top \to \top) \Rightarrow \\
\bot \to \top \Rightarrow \\
\top
\end{array}
$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$
\begin{array}{c}
(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow \\
(\top \to \bot) \land (\top \land \bot \to \top) \to \top \Rightarrow \\
\top
\end{array}
$$

The result will always be the same independently of the order of rewriting!

Apply rewrite rules

Inside-out, left-to-right:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land (\bot \to \top) \to (\top \to \top) \Rightarrow$$
$$\bot \land \top \to (\top \to \top) \Rightarrow$$
$$\bot \to (\top \to \top) \Rightarrow$$
$$\bot \to \top \Rightarrow$$
$$\top$$

1. $A \land \bot \Rightarrow \bot$
2. $\top \to \bot \Rightarrow \bot$
3. $A \to \top \Rightarrow \top$

Outside-in, right-to-left:

$$(\top \to \bot) \land (\top \land \bot \to \top) \to (\top \to \top) \Rightarrow$$
$$(\top \to \bot) \land (\top \land \bot \to \top) \to \top \Rightarrow$$
$$\top$$

The result will always be the same independently of the order of rewriting!