Outline

Proposition

Propositional Logic formalises the notion of proposition, that is a statement that can be either true or false.

Proposition

Propositional Logic formalises the notion of proposition, that is a statement that can be either true or false.

There are simple propositions, called atomic. For example:

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

Proposition

Propositional Logic formalises the notion of proposition, that is a statement that can be either true or false.

There are simple propositions, called atomic. For example:

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

More complex propositions are built from simpler ones using a small number of constructs. Examples of more complex propositions:

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

Truth

Each proposition is either true or false.

Truth

Each proposition is either true or false.

The truth value of an atomic proposition, that is, either true or false depends on an interpretation of such propositions.

Truth

Each proposition is either true or false.

The truth value of an atomic proposition, that is, either true or false depends on an interpretation of such propositions.

For example, $1 + 1 = 10$ is false, if we interpret sequences of digits as the decimal notation for numbers and true if we use the binary notation.

Truth

Each proposition is either true or false.

The truth value of an atomic proposition, that is, either true or false depends on an interpretation of such propositions.

For example, $1 + 1 = 10$ is false, if we interpret sequences of digits as the decimal notation for numbers and true if we use the binary notation.

If a complex proposition C is build from simpler propositional $S_1, \ldots, S_n$ using a construct, then the truth value of C is determined by the truth value of $S_1, \ldots, S_n$. More precisely, it is a function of truth values of $S_1, \ldots, S_n$ defined by this construct.

Truth

Each proposition is either true or false.

The truth value of an atomic proposition, that is, either true or false depends on an interpretation of such propositions.

For example, $1 + 1 = 10$ is false, if we interpret sequences of digits as the decimal notation for numbers and true if we use the binary notation.

If a complex proposition C is build from simpler propositional $S_1, \dots, S_n$ using a construct, then the truth value of C is determined by the truth value of $S_1, \dots, S_n$. More precisely, it is a function of truth values of $S_1, \dots, S_n$ defined by this construct.

For example, $1 + 1 = 10$ or $1 + 1 \neq 10$ is true if $1 + 1 \neq 10$ is true.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.
- $\top$ and $\bot$ are formulas.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.
- $\top$ and $\bot$ are formulas.
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $(A_1 \wedge \ldots \wedge A_n)$ and $(A_1 \vee \ldots \vee A_n)$ are formulas.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.
- $\top$ and $\bot$ are formulas.
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $(A_1 \wedge \ldots \wedge A_n)$ and $(A_1 \vee \ldots \vee A_n)$ are formulas.
- If A is a formula, then $(\neg A)$ is a formula.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.
- $\top$ and $\bot$ are formulas.
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $(A_1 \wedge \ldots \wedge A_n)$ and $(A_1 \vee \ldots \vee A_n)$ 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.

Propositional Logic: Syntax

Assume a countable set of boolean variables.
Propositional formula:

- Every boolean variable is a formula, also called atomic formula, or simply atom.
- $\top$ and $\bot$ are formulas.
- If $A_1, \ldots, A_n$ are formulas, where $n \geq 2$, then $(A_1 \wedge \ldots \wedge A_n)$ and $(A_1 \vee \ldots \vee A_n)$ 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.

The symbols $\top, \bot, \wedge, \vee, \neg, \rightarrow, \leftrightarrow$ are called connectives.

Subformula

- Formulas $A_1, \ldots, A_n$ are the immediate subformulas of $(A_1 \wedge \ldots \wedge A_n)$ and $(A_1 \vee \ldots \vee A_n)$.
- Formula A is the immediate subformula of $(\neg A)$.
- Formulas A and B are the immediate subformulas of $(A \rightarrow B)$ and $(A \leftrightarrow B)$.
- Every formula A is a subformula of itself.
- If A is a subformula of B and B is a subformula of C, then A is a subformula of C.

Parsing Formulas

We want to avoid expressions cluttered with parentheses. The standard way to avoid them is to assign precedence to operators and use the precedence to disambiguate expressions.

Parsing Formulas

We want to avoid expressions cluttered with parentheses. The standard way to avoid them is to assign precedence to operators and use the precedence to disambiguate expressions.
For example, in arithmetic we know that the expression

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

is equivalent to

$$(x \cdot y) + (2 \cdot z),$$

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

Connectives and Their Precedences

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

Parsing Formulas

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

Parsing Formulas

Let us parse

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

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

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):

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

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):

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

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):

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

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

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

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

Outside-in (starting with the lowest precedence connectives):

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

Parsing Formulas

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

Let us parse

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

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

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

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):

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

Outside-in (starting with the lowest precedence connectives):

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

Semantics, Interpretation

Consider an arithmetical expression, for example

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

In arithmetic the meaning of expressions with variables is defined as follows.

Semantics, Interpretation

Consider an arithmetical expression, for example

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

In arithmetic the meaning of expressions with variables is defined as follows.
Take a mapping from variables to (integer) values, for example

$$\{x \mapsto 1, y \mapsto 7, z \mapsto -3\}.$$

Semantics, Interpretation

Consider an arithmetical expression, for example

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

In arithmetic the meaning of expressions with variables is defined as follows.
Take a mapping from variables to (integer) values, for example

$$\{x \mapsto 1, y \mapsto 7, z \mapsto -3\}.$$

Then, under this mapping the expression has the value 1. In other words, when we interpret variables as values, we can compute the value of any expression built using these variables.

Semantics, Interpretation

Likewise, the semantics of propositional formulas can be defined by assigning values to variables.

Semantics, Interpretation

Likewise, the semantics of propositional formulas can be defined by assigning values to variables.

- There are two boolean values, also called truth values: true (denoted 1) and false (denoted 0).

Semantics, Interpretation

Likewise, the semantics of propositional formulas can be defined by assigning values to variables.

- There are two boolean values, also called truth values: true (denoted 1) and false (denoted 0).
- An interpretation for a set P of boolean variables is a mapping $I : P \to \{1, 0\}$.

Semantics, Interpretation

Likewise, the semantics of propositional formulas can be defined by assigning values to variables.

- There are two boolean values, also called truth values: true (denoted 1) and false (denoted 0).
- An interpretation for a set P of boolean variables is a mapping $I : P \rightarrow \{1, 0\}$.
- Interpretations are also called truth assignments.

Interpreting Formulas

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

Interpreting Formulas

The truth value of a complex formula is determined by the truth values of its components.
Given an interpretation I, extend I to a mapping from all formulas to truth values as follows.

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

Operation Tables

$I(A_1 \vee A_2) = 1$ if and only if $I(A_1) = 1$ or $I(A_2) = 1$.

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

Operation Tables

$I(A_1 \leftrightarrow A_2) = 1$ if and only if $I(A_1) = I(A_2)$.

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

Operation Tables

$\wedge$	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

Operation Tables

$I(A_1 \vee A_2) = 1$ if and only if $I(A_1) = 1$ or $I(A_2) = 1$.
$I(A_1 \leftrightarrow A_2) = 1$ if and only if $I(A_1) = I(A_2)$.

$\wedge$	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.

Satisfiability, Validity, Equivalence

Let A be a formula.

- If $I(A) = 1$, then we say that the formula A is true in I and that I satisfies A and that I is a model of A, denoted by $I \models A$.

Satisfiability, Validity, Equivalence

Let A be a formula.

- If $I(A) = 1$, then we say that the formula A is true in I and that I satisfies A and that I is a model of A, denoted by $I \models A$.
- If $I(A) = 0$, then we say that the formula A is false in I.

Satisfiability, Validity, Equivalence

Let A be a formula.

- If $I(A) = 1$, then we say that the formula A is true in I and that I satisfies A and that I is a model of A, denoted by $I \models A$.
- If $I(A) = 0$, then we say that the formula A is false in I.
- A is satisfiable if it is true in some interpretation.
- A is valid (or a tautology) if it is true in every interpretation.

Satisfiability, Validity, Equivalence

Let A be a formula.

- If $I(A) = 1$, then we say that the formula A is true in I and that I satisfies A and that I is a model of A, denoted by $I \models A$.
- If $I(A) = 0$, then we say that the formula A is false in I.
- A is satisfiable if it is true in some interpretation.
- A is valid (or a tautology) if it is true in every interpretation.
- Two formulas A and B are called equivalent, denoted by $A \equiv B$ if they have the same models.

Examples

$A \rightarrow A$ and $A \vee \neg A$ are valid for all formulas A.

Examples

$A \to A$ and $A \vee \neg A$ are valid for all formulas A.

Evidently, every valid formula is also satisfiable.

Examples

$A \to A$ and $A \vee \neg A$ are valid for all formulas A.

Evidently, every valid formula is also satisfiable.

$A \wedge \neg A$ is unsatisfiable for all formulas A.

Examples

$A \to A$ and $A \vee \neg A$ are valid for all formulas A.

Evidently, every valid formula is also satisfiable.

$A \wedge \neg A$ is unsatisfiable for all formulas A.

Formula p, where p is a boolean variable, is satisfiable but not valid.

Examples: Equivalences

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

$$
\begin{align}
A \to \bot &\equiv \neg A; \tag{1}\\
\top \to A &\equiv A; \tag{2}\\
A \to B &\equiv \neg(A \land \neg B); \tag{3}\\
A \land B &\equiv \neg(\neg A \lor \neg B); \tag{4}\\
A \lor B &\equiv \neg A \to B. \tag{5}
\end{align}
$$

Connections Between These Notions

1. A formula A is valid if and only if $\neg A$ is unsatisfiable.
2. A formula A is satisfiable if and only if $\neg A$ is not valid.

Connections Between These Notions

3. A formula A is valid if and only if A is equivalent to $\top$.
4. Formulas A and B are equivalent if and only if the formula $A \leftrightarrow B$ is valid.

Connections Between These Notions

5. Formulas A and B are equivalent if and only if the formula $\neg(A \leftrightarrow B)$ is unsatisfiable.
6. A formula A is satisfiable if and only if A is not equivalent to $\bot$.

Connections Between These Notions

1. A formula A is valid if and only if $\neg A$ is unsatisfiable.
2. A formula A is satisfiable if and only if $\neg A$ is not valid.
3. A formula A is valid if and only if A is equivalent to $\top$.
4. Formulas A and B are equivalent if and only if the formula $A \leftrightarrow B$ is valid.
5. Formulas A and B are equivalent if and only if the formula $\neg(A \leftrightarrow B)$ is unsatisfiable.
6. A formula A is satisfiable if and only if A is not equivalent to $\bot$.

How to Evaluate a Formula?

Let's evaluate the formula

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

in the interpretation

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}.$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

Evaluating a Formula

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

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

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$$

So the formula is true in this interpretation.

End of Lecture 2

Slides for lecture 2 end here . . .

Equivalent replacement

Lemma (Equivalent Replacement)

Let A_1 be a subformula of B_1 and $I \models A_1 \leftrightarrow A_2$. Suppose that B_2 is obtained from B_1 by replacing one or more occurrences or A_1 by A_2. Then $I \models B_1 \leftrightarrow B_2$.

Equivalent replacement

Lemma (Equivalent Replacement)

Let A_1 be a subformula of B_1 and $I \models A_1 \leftrightarrow A_2$. Suppose that B_2 is obtained from B_1 by replacing one or more occurrences or A_1 by A_2. Then $I \models B_1 \leftrightarrow B_2$.

Theorem (Equivalent Replacement)

Let A_1 be a subformula of B_1 and $A_1 \equiv A_2$. Suppose that B_2 is obtained from B_1 by replacing one or more occurrences or A_1 by A_2. Then $B_1 \equiv B_2$.

In other words, replacing, in a formula B_1, a subformula A_1 by an equivalent formula A_2 gives an equivalent formula.

Equivalent replacement

Lemma (Equivalent Replacement)

Let A_1 be a subformula of B_1 and $I \models A_1 \leftrightarrow A_2$. Suppose that B_2 is obtained from B_1 by replacing one or more occurrences or A_1 by A_2. Then $I \models B_1 \leftrightarrow B_2$.

Theorem (Equivalent Replacement)

Let A_1 be a subformula of B_1 and $A_1 \equiv A_2$. Suppose that B_2 is obtained from B_1 by replacing one or more occurrences or A_1 by A_2. Then $B_1 \equiv B_2$.

In other words, replacing, in a formula B_1, a subformula A_1 by an equivalent formula A_2 gives an equivalent formula.

(thanks to compositionality!)

A purely syntactic algorithm

If $I \models p$, then $I \models p \leftrightarrow \top$;
If $I \not\models p$, then $I \models p \leftrightarrow \bot$;

A purely syntactic algorithm

If $I \models p$, then $I \models p \leftrightarrow \top$;
If $I \not\models p$, then $I \models p \leftrightarrow \bot$;

Since we can replace a subformula by a formula with the same value, we can replace every variable p by either $\top$ or $\bot$, depending on the value of p in I.

Rewrite rules for evaluating a formula

Suppose that we have a formula consisting only of $\perp$ and $\top$.
One can note that every formula of this form different from $\perp$ and $\top$ can be "simplified" to a smaller equivalent formula.

Rewrite rules for evaluating a formula

Suppose that we have a formula consisting only of $\bot$ and $\top$.
One can note that every formula of this form different from $\bot$ and $\top$ can be "simplified" to a smaller equivalent formula.
For example, every formula of the form $A \to \top$ is equivalent to a simpler formula $\top$.

Rewrite rules for evaluating a formula

Suppose that we have a formula consisting only of $\bot$ and $\top$.
One can note that every formula of this form different from $\bot$ and $\top$ can be "simplified" to a smaller equivalent formula.
For example, every formula of the form $A \to \top$ is equivalent to a simpler formula $\top$.
This simplification process can be formalised as a rewrite rule system:

$$\boxed{\begin{array}{c} \top \land \ldots \land \top \;\Rightarrow\; \top \\ \bot \land A_1 \land \ldots \land A_n \;\Rightarrow\; \bot \end{array}} \quad \boxed{\begin{array}{c} A_1 \lor \ldots \lor \top \lor \ldots \lor A_n \;\Rightarrow\; \top \\ \bot \lor \ldots \lor \bot \;\Rightarrow\; \bot \end{array}}$$

$$\boxed{\begin{array}{lcl} \neg\top & \Rightarrow & \bot \\ \neg\bot & \Rightarrow & \top \end{array}} \quad \boxed{\begin{array}{lcl} A \to \top & \Rightarrow & \top \\ \bot \to A & \Rightarrow & \top \\ \top \to \bot & \Rightarrow & \bot \end{array}} \quad \boxed{\begin{array}{lcl} \top \leftrightarrow \top & \Rightarrow & \top \\ \top \leftrightarrow \bot & \Rightarrow & \bot \\ \bot \leftrightarrow \top & \Rightarrow & \bot \\ \bot \leftrightarrow \bot & \Rightarrow & \top \end{array}}$$

Algorithm for evaluating a formula

We can define a purely syntax algorithm for evaluating a formula using the rewrite rule system.

procedure *evaluate*(G, I)
input: formula G, interpretation I
output: the boolean value $I(G)$

Algorithm for evaluating a formula

We can define a purely syntax algorithm for evaluating a formula using the rewrite rule system.

procedure $evaluate(G, I)$
input: formula G, interpretation I
output: the boolean value $I(G)$
begin
  forall variables p occurring in G
    if $I \models p$
      then replace all occurrences of p in G by $\top$;
      else replace all occurrences of p in G by $\perp$;

end

Algorithm for evaluating a formula

We can define a purely syntax algorithm for evaluating a formula using the rewrite rule system.

<u>**procedure**</u> *evaluate*(G, I)
<u>**input**</u>: formula G, interpretation I
<u>**output**</u>: the boolean value $I(G)$
<u>**begin**</u>
  <u>**forall**</u> variables p occurring in G
    <u>**if**</u> $I \models p$
      <u>**then**</u> replace all occurrences of p in G by $\top$;
      <u>**else**</u> replace all occurrences of p in G by $\bot$;
  rewrite G into a normal form using the rewrite rules

<u>**end**</u>

Algorithm for evaluating a formula

We can define a purely syntax algorithm for evaluating a formula using the rewrite rule system.

```
procedure evaluate(G, I)
input: formula G, interpretation I
output: the boolean value I(G)
begin
  forall variables p occurring in G
    if I ⊨ p
      then replace all occurrences of p in G by ⊤;
      else replace all occurrences of p in G by ⊥;
  rewrite G into a normal form using the rewrite rules
  if G = ⊤ then return 1 else return 0
end
```

Example

Let us evaluate the formula

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

in the interpretation

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}.$$

Example

Let us evaluate the formula

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

in the interpretation

$$\{p \mapsto 1, q \mapsto 0, r \mapsto 1\}.$$

The value of this formula 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:

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

$$A \land \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$A \land \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{aligned}
A \wedge \bot &\Rightarrow \bot \\
\top \to \bot &\Rightarrow \bot \\
A \to \top &\Rightarrow \top
\end{aligned}
$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{aligned}
A \wedge \bot &\Rightarrow \bot \\
\top \to \bot &\Rightarrow \bot \\
A \to \top &\Rightarrow \top
\end{aligned}
$$

Apply rewrite rules

Inside-out, left-to-right:

$$
\begin{aligned}
&(\top \to \bot) \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\quad \bot \wedge (\top \wedge \bot \to \top) \to (\top \to \top) \Rightarrow \\
&\qquad \bot \wedge (\bot \to \top) \to (\top \to \top) \Rightarrow \\
&\qquad\quad \bot \wedge \top \to (\top \to \top)
\end{aligned}
$$

$$
\begin{aligned}
A \wedge \bot &\Rightarrow \bot \\
\top \to \bot &\Rightarrow \bot \\
A \to \top &\Rightarrow \top
\end{aligned}
$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{aligned}
A \wedge \bot &\Rightarrow \bot \\
\top \to \bot &\Rightarrow \bot \\
A \to \top &\Rightarrow \top
\end{aligned}
$$

Apply rewrite rules

Inside-out, left-to-right:

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

$A \land \bot \Rightarrow \bot$
$\top \to \bot \Rightarrow \bot$
$A \to \top \Rightarrow \top$

Apply rewrite rules

Inside-out, left-to-right:

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

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

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{array}{l}
A \wedge \bot \Rightarrow \bot \\
\top \to \bot \Rightarrow \bot \\
A \to \top \Rightarrow \top
\end{array}
$$

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{array}{l}
A \land \bot \Rightarrow \bot \\
\top \to \bot \Rightarrow \bot \\
A \to \top \Rightarrow \top
\end{array}
$$

Apply rewrite rules

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Outside-in, right-to-left:

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

Apply rewrite rules

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Outside-in, right-to-left:

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

Apply rewrite rules

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Outside-in, right-to-left:

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

Apply rewrite rules

$$A \land \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Outside-in, right-to-left:

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

Apply rewrite rules

$$A \wedge \bot \Rightarrow \bot$$
$$\top \to \bot \Rightarrow \bot$$
$$A \to \top \Rightarrow \top$$

Outside-in, right-to-left:

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

Apply rewrite rules

Inside-out, left-to-right:

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

$$
\begin{array}{l}
A \wedge \bot \Rightarrow \bot \\
\top \to \bot \Rightarrow \bot \\
A \to \top \Rightarrow \top
\end{array}
$$

Outside-in, right-to-left:

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

The result will always be the same independently of the order of rewrites