Logic for Computer Science

02 – Propositions

Wouter Swierstra

Utrecht University

Organization

- Keep an eye on the website and Teams channels – you can find more information about the quiz, recommended exercises, slides and schedule there.
- There’s a new channel to ask questions about exercises if you are stuck!

Organization

- Keep an eye on the website and Teams channels – you can find more information about the quiz, recommended exercises, slides and schedule there.
- There's a new channel to ask questions about exercises if you are stuck!
- The first exercise session is immediately after this lecture. I'd like to ask everyone in Groups 8 & 9 (Ruppert, Bologna, DDW, …) to spread out over the other groups in the DALTON 500 building. There are *werkcolleges* in rooms: 3.19, 6.27, 7.27, 8.06, 8.08, 8.09, and 8.27.

(I'll repeat all these announcements in Teams too)

Last time

- Organization
- What is logic?
- Why study logic?
- How to reason about programs?

Today

Building on the introduction too boolean logic from *Computerarchitectuur en netwerken*...

- Propositional logic
- Truth tables

Propositions

In the last lecture, I talked about **proofs** – but **what** kind of things can we prove:

Propositions

In the last lecture, I talked about **proofs** – but **what** kind of things can we prove:

A *proposition* is a statement that may be true or not:

- My name is Wouter;
- When it rains, the streets are wet.
- I like to drink coffee or tea with my breakfast;
- 2+3=5
- 2+3=6
- x + y = 9
- this method will return a value greater than 7 on all inputs less than 5;

Not all these propositions are true – but they can be all be checked to hold or not.

Non-propositions

Not *everything* is a proposition:

- Don't you agree that this lecturer is amazing?
- `2+=*34`
- `if x==0 then y = 3 else y = 4`
- Sit up and pay attention!

These are all examples of sentences, programs or symbols that are not true or false.

Propositional logic

Propositional logic studies when such statements are true or false.

We can do this by constructing a *derivation*, showing how a *conclusion* follows from various *assumptions*.

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. The fire alarm goes off.
3. **Conclusion:** We leave the classroom.

Question: Is this conclusion valid?

Propositional logic

Propositional logic studies when such statements are true or false.

We can do this by constructing a *derivation*, showing how a *conclusion* follows from various *assumptions*.

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. The fire alarm goes off.
3. **Conclusion:** We leave the classroom.

Question: Is this conclusion valid?

Yes!

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. The fire alarm does **not** go off.
3. **Conclusion:** Hence we **don't** leave the classroom

Question: Is this conclusion valid?

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. The fire alarm does **not** go off.
3. **Conclusion:** Hence we **don't** leave the classroom

Question: Is this conclusion valid?

No! We could leave the class for any number of reasons: the class may have ended, the break may have started, or some other calamity.

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. We leave the classroom.
3. **Conclusion:** Hence the fire alarm must be going off.

Question: Is this conclusion valid?

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. We leave the classroom.
3. **Conclusion:** Hence the fire alarm must be going off.

Question: Is this conclusion valid?

No! There are plenty of other reasons to leave the class room.

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. We are **not** leaving the classroom.
3. **Conclusion:** Hence the fire alarm is **not** going off.

Question: Is this derivation correct?

Fire alarm

Example

1. When the fire alarm goes off, we must evacuate the classroom.
2. We are **not** leaving the classroom.
3. **Conclusion:** Hence the fire alarm is **not** going off.

Question: Is this derivation correct?

Yes! After all, if the fire alarm did go off, we would have to leave the classroom.

Implication

This example illustrates something called **logical implication**.

This occurs in natural language in many different ways:

- **If** the fire alarm goes off, **then** we must leave the classroom.
- We will leave the classroom, **provided** the fire alarm goes off.
- **Whenever** the fire alarm goes off, we'll leave the classroom.
- ...

Implication

This example illustrates something called **logical implication**.

This occurs in natural language in many different ways:

- **If** the fire alarm goes off, **then** we must leave the classroom.
- We will leave the classroom, **provided** the fire alarm goes off.
- **Whenever** the fire alarm goes off, we'll leave the classroom.
- ...

Logical implication is one of the cornerstones of logic.

I'll tell you more about its meaning shortly...

Abstraction & logic

Making abstractions

In primary school, you learn to make sums:

- 2 + 2 = 4
- 2 + 3 = 5
- 2 + 12 = 14

Making abstractions

In primary school, you learn to make sums:

- 2 + 2 = 4
- 2 + 3 = 5
- 2 + 12 = 14

In high school, you learn to make an **abstraction**. Instead of working with concrete numbers (like 5 or 21), you learn to manipulate **variables** that might stand for any number:

- 2 + x = 17
- $x = \frac{-b+\sqrt{b^2-4ac}}{2a}$

Instead of plain arithmetic, you learn to solve equations.

Abstractions in logic

I could give endless examples along the lines:

- If the fire alarm goes off, we must leave the class room.

But we would stay stuck at ‘primary school logic’ – we wouldn’t have any *variables* that could range over different propositions.

I want to work *abstractly* with propositions and study their meaning.

Abstractions in logic

I could give endless examples along the lines:

- If the fire alarm goes off, we must leave the class room.

But we would stay stuck at ‘primary school logic’ – we wouldn’t have any *variables* that could range over different propositions.

I want to work *abstractly* with propositions and study their meaning.

That’s why I will use **variables** (such as P or Q) instead of concrete propositions about fire alarms and class rooms.

In the same way as you learn to manipulate formulas and solve equations in high school, we will start studying abstract propositions involving variables.

Variables - I

We'll use variables written with a capital letter, such as `P` and `Q`, to refer to *atomic* propositions.

These are propositions that we assume to indivisible, and not built up from smaller pieces using logical connectives such as implication.

- the fire alarm goes off;
- it is raining;
- `x > 14`

but not:

- if the fire alarm goes off, we must leave the class room.
- it is raining and the sun is shining.
- Logic is fun or my name is Wouter.

Compound propositions

Using these atomic propositions, we can formulate more interesting statements:

We've seen a first example: $P \Rightarrow Q$

(read as P implies Q).

- Where P might stand for 'the fire alarm goes off
- and Q stands for 'we leave the class room'.

Compound propositions

Using these atomic propositions, we can formulate more interesting statements:

We've seen a first example: $P \Rightarrow Q$

(read as P implies Q).

- Where P might stand for 'the fire alarm goes off
- and Q stands for 'we leave the class room'.

In high school, you learned to build up complicated mathematical expressions using operators such as +, ×,÷, etc.

We have now encountered our first logical operator, namely $\Rightarrow$, used to represent implication.

Propositional logic

A **propositional formula** (or proposition for short) is built as follows:

- atomic propositions `P`, `Q`, `R`, ...
- `true` – sometimes written `T`, `1`
- `false` - sometimes written `F`, `0`

Propositional logic

A **propositional formula** (or proposition for short) is built as follows:

- atomic propositions `P`, `Q`, `R`, ...
- `true` – sometimes written T, 1
- `false` - sometimes written F, 0

Or the formula is constructed from smaller pieces. If p and q are propositional formulas, then the following are also propositional formulas:

- $p \Rightarrow q$ - implication (if p then q)
- $p \wedge q$ - conjunction (p and q)
- $p \vee q$ - disjunction (p or q)
- $\neg p$ - negation (p does not hold)

Two kinds of variables

A **propositional formula** (or proposition for short) is built as follows:

- atomic propositions P, Q, R, ...

Or the formula is constructed from smaller pieces. If p and q are propositional formulas, then the following are also propositional formulas:

- $p \Rightarrow q$ - implication (if p then also q)...

Question: What is the difference between p and P?

Two kinds of variables

A **propositional formula** (or proposition for short) is built as follows:

- atomic propositions P, Q, R, ...

Or the formula is constructed from smaller pieces. If p and q are propositional formulas, then the following are also propositional formulas:

- $p \Rightarrow q$ - implication (if p then also q)...

Question: What is the difference between p and P?

We use variables starting with capital letters P to denote *atomic propositions*; lower case variables, such as p, refer to (possible non-atomic) propositions.

Variables such as p are **not** propositional formulas themselves! Instead these are sometimes referred to as *metavariables*.

Syntax and semantics

On a previous slide, I gave a definition of how propositions are built:

Propositional logic

Propositional formulas are constructed as follows:

- atomic propositions P, Q, R,...
- true
- false
- ...

This defines the *structure* of propositions (sometimes also referred to as the **syntax**).

Later on, we'll start to study the *meaning* of propositions also called the **semantics**).

Syntax and semantics

The distinction between syntax and semantics shows up all the time.

This text is a fragment of well formed C# code:

```
int x = 14;
int y = x + 12;
```

But this is not:

```
xxb =[i'3[hxktz ;O;OIDL
```

We only can study the meaning (that is the *semantics*) of code that has a valid *syntax*.

But I have no idea what the semantics of the second example is!

Syntax

Propositional logic

Propositional formulas are defined as follows:

- atomic propositions `P`, `Q`, `R`
- `true` – truth, T, 1
- `false` - falsity, F, 0
- `p` $\Rightarrow$ `q` - implication (if `p` then `q`)
- `p` $\wedge$ `q` - conjunction (`p` an `q`)
- `p` $\vee$ `q` - disjunction (`p` or `q`)
- $\neg$`p` - negation (`p` does not hold)

Question: Can you give an example of a text that is almost, but not quite, a syntactically valid propositional formula?

Propositional logic – syntactic conventions

Question: What is the result of evaluating `3 * 2 + 1`?

Propositional logic – syntactic conventions

Question: What is the result of evaluating `3 * 2 + 1`?

7 of course!

Propositional logic – syntactic conventions

Question: What is the result of evaluating `3 * 2 + 1`?

7 of course!

But why did you read this as `(3 * 2) + 1` rather than `3 * (2 + 1)`?

In high school you learn that the different operators have a different *precedence*

Meneer van Dalen wacht op antwoord – machtsverheffen, vermenigvuldigen, delen, worteltrekken, optellen, aftrekken – in that order.

Propositional logic – syntactic conventions

Question: What is the result of evaluating `3 * 2 + 1`?

7 of course!

But why did you read this as `(3 * 2) + 1` rather than `3 * (2 + 1)`?

In high school you learn that the different operators have a different *precedence*

Meneer van Dalen wacht op antwoord – machtsverheffen, vermenigvuldigen, delen, worteltrekken, optellen, aftrekken – in that order.

There is something similar going on for propositional logic.

Propositional logic – syntactic conventions

1. Parentheses
2. Negation $\neg$
3. Conjunction $\wedge$
4. Disjunction $\vee$
5. Implication $\Rightarrow$

Question: How is this expressed parenthesized?

$\neg\ \mathtt{P} \vee \mathtt{Q} \Rightarrow \mathtt{Q} \wedge \mathtt{P}$

Propositional logic – syntactic conventions

1. Parentheses
2. Negation $\neg$
3. Conjunction $\wedge$
4. Disjunction $\vee$
5. Implication $\Rightarrow$

Question: How is this expressed parenthesized?

$\neg\ P \vee Q \Rightarrow Q \wedge P$

That should be read:

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

Propositional logic – syntactic conventions

1. Parentheses
2. Negation $\neg$
3. Conjunction $\wedge$
4. Disjunction $\vee$
5. Implication $\Rightarrow$

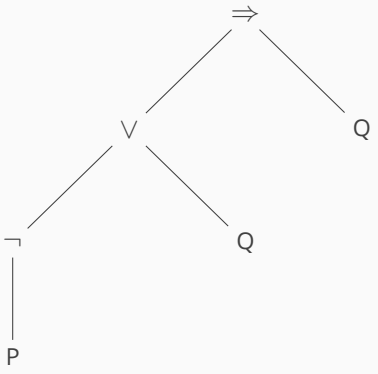
Homework: Come up with a good mnemonic.

Propositional logic – syntax trees

We can make it explicit how to read a proposition such as $\neg\ P\ \vee\ Q\ \Rightarrow\ Q$ by writing unnecessary parentheses:

$((\neg\ P)\ \vee\ Q)\ \Rightarrow\ Q$

But we could also make the structure explicit in a *syntax tree*:



Why trees?

These trees make the syntactic structure explicit.

And you'll see these such tree-like structures over and over again in your degree:

- Later in this course, we'll see how to define tree structures more formally.
- In *Talen en Compilers* you will learn how to read in a string and produce the corresponding *syntax tree*.
- In *Functioneel Programmeren* you'll learn how to represent such trees and compute values from them.
- In courses on Algorithms and Datastructures, trees are indispensable to get sub-linear lookup times.

Trees are everywhere!

Operators and arities

Note that $\vee$ and $\wedge$ are very different from $\neg$.

When p and q are propositions, so are $p \vee q$ and $p \wedge q$.

That is, $\vee$ and $\wedge$ require **two** arguments in order to form a new proposition – hence we refer to them as **binary** operators.

The $\neg$ operator, on the other hand, requires **one** argument: $\neg p$ rather than $q \neg p$. For that reason, we call $\neg$ a **unary** operator.

Semantics

Propositional logic – semantics?

So far, we've nailed down the *syntax* of propositional logic;

- atomic propositions `P`, `Q`, `R`, etc.
- `true`
- `false`
- $p \Rightarrow q$ - implication (if p then q)
- $p \wedge q$ - conjunction (p and q)
- $p \vee q$ - disjunction (p or q)
- $\neg p$ - negation (p does not hold)

But what does these symbols **mean**? In other words, what is the **semantics** of propositional formulas?

Semantics

We want to determine for a given propositional formula: is it **true** or **false**?

For atomic propositions, this is clear.

But what is the precise meaning of the other propositional operators.

Semantics

We want to determine for a given propositional formula: is it **true** or **false**?

For atomic propositions, this is clear.

But what is the precise meaning of the other propositional operators.

That is exactly what we specify in our semantics.

Semantics of propositional operators

Our propositional formulas are built from the following operators:

- $\neg p$ - negation (p does not hold)
- $p \wedge q$ - conjunction (p and q)
- $p \vee q$ - disjunction (p or q)
- $p \Rightarrow q$ - implication (if p then q)

In the next slides, we will define the value of these operators for **all** possible values of p and q.

Negation

- If p is true, then $\neg p$ is false;
- If p is false, then is $\neg p$ true.

Negation

- If p is true, then $\neg p$ is false;
- If p is false, then is $\neg p$ true.

Explaining this in English can get old very quickly.

Negation

We can summarize this information more succinctly in a **truth table**:

p	$\neg p$
F	**T**
T	**F**

We can read of the value of $\neg p$ for the two possible values of p.

Conjunction

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

The conjunction $p \wedge q$ is true precisely when both p and q are true;

otherwise, the conjunction is false.

Disjunction

p	q	$p \vee q$
F	**F**	?
F	**T**	?
T	**F**	?
T	**T**	?

Question: When is the disjunction $p \vee q$ true?

Fill in this truth table.

Disjunction

p	q	$p \vee q$
F	F	F
F	T	T
T	F	T
T	T	T

The disjunction $p \vee q$ is true precisely when p is true, or q is true, or both p and q are true.

Or not?

When someone asks, ‘do you want coffee or tea?’, it is a bit rude to ask for both.

But a logical disjunction, $p \vee q$, is true when both p and q are true.

In logic, we also have a separate operator $\oplus$ – the *exclusive or* or *xor* – that is false when both p and q are true.

This shows how important truth tables are to specify semantics precisely.

Implication - I

p	q	$p \Rightarrow q$
F	**F**	**T**
F	**T**	**T**
T	**F**	**F**
T	**T**	**T**

Logical implication, $p \Rightarrow q$, can be interpreted as: whenever p is true, then q must also be true.

An implication is *always* true if p (also known as the *antecedent*) is false.

Implication - II

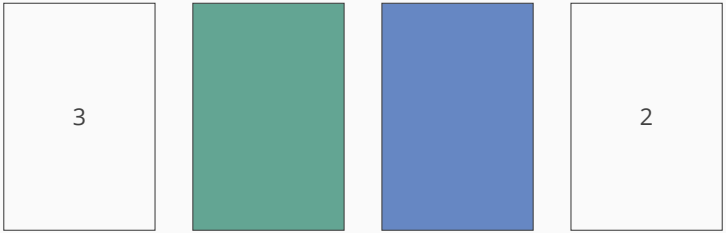
p	q	$p \Rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

- if I win the lottery tomorrow, you will all get 1000 euros;
- if the moon is made of green cheese, everyone will get a 10 for their midterm.

These are examples of propositional formulas that hold – but don't say anything interesting.

Yet logical implication is one of the most fundamental constructs in formal reasoning.

Exercise

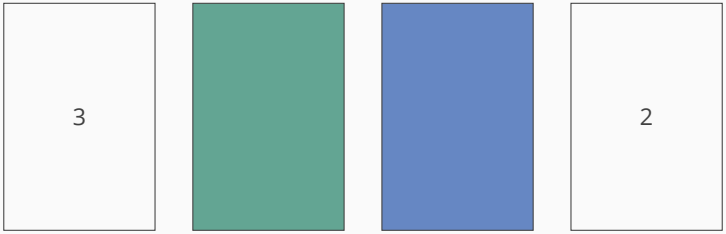


There are four cards in front of you. Each card has a number one side and a color on the other.

I claim that *if a card has an even number, the other side of the card is green*.

Question: Which cards should I turn over to check this?

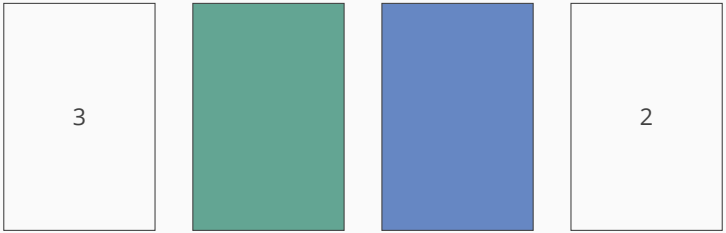
Exercise



Claim: if a card has an even number, the other side of the card is green.

I don't need to turn over the card with 3 on it – my claim doesn't say anything about cards with odd numbers on them.

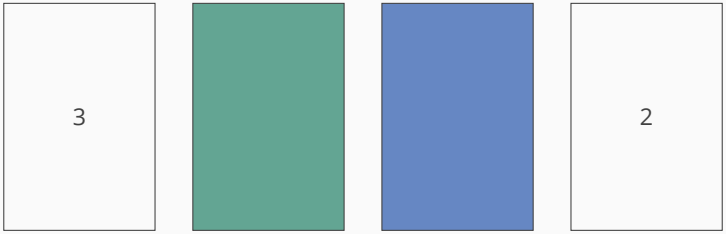
Exercise



Claim: if a card has an even number, the other side of the card is green.

I do need to turn over the card with 2 on it – I should check if the other side is indeed green.

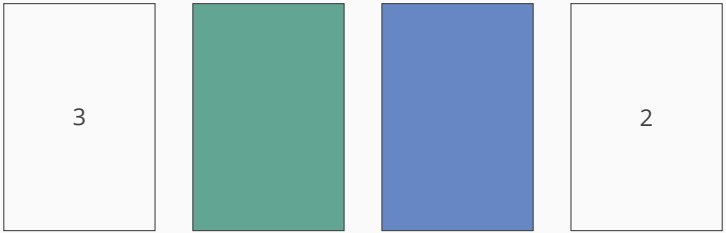
Exercise



Claim: if a card has an even number, the other side of the card is green.

I don't need to turn over the green card. It doesn't matter what is on the other side: the statement cannot be falsified.

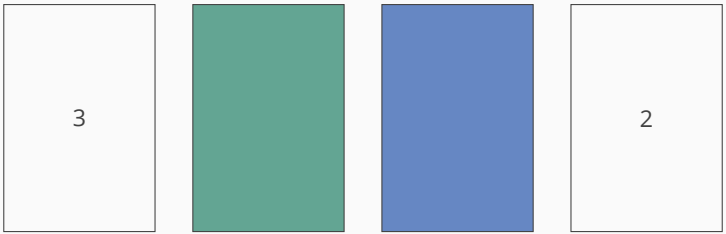
Exercise



Claim: if a card has an even number, the other side of the card is green.

I do need to turn over the blue card – I should check that the number on the other side is odd. If the number is even, my claim is false.

Exercise



This example illustrates how logical implication can be counter intuitive at times...

Be careful: the direction of the implication arrow is very important: $(p \Rightarrow q)$ and $(q \Rightarrow p)$ usually mean very different things!

Equivalence

Sometimes we will write $p \Leftrightarrow q$ – pronounced *p* if and only if *q*.

p	q	$p \Leftrightarrow q$
F	**F**	**T**
F	**T**	**F**
T	**F**	**F**
T	**T**	**T**

If $p \Leftrightarrow q$ holds, we say that *p* and *q* are *logically equivalent*.

This truth table is the same as $(p \Rightarrow q) \wedge (q \Rightarrow p)$ – so adding the equivalence operator doesn't add anything new to the logic that we couldn't express before.

Proving equivalences

We can claim that for all p and q the propositions $p \Rightarrow q$ and $\neg p \vee q$ are equivalent.

Proving equivalences

We can claim that for all p and q the propositions $p \Rightarrow q$ and $\neg p \vee q$ are equivalent.

But how can we verify such a statement?

Proving equivalences

We can claim that for all p and q the propositions $p \Rightarrow q$ and $\neg p \vee q$ are equivalent.

But how can we verify such a statement?

With our semantics of course!

p	q	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	?
F	T	?
T	F	?
T	T	?

We would like to fill in this truth table. If we can fill in **T** for all the question marks, our proposition always holds, regardless of the values of p and q.

Proofs

p	q	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	?
F	T	?
T	F	?
T	T	?

We would like to fill in this truth table. If we can fill in **T** for all the question marks, our proposition always holds, regardless of the values of p and q.

But this looks a bit tricky...

p	q	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	?
F	T	?
T	F	?
T	T	?

We would like to fill in this truth table. If we can fill in **T** for all the question marks, our proposition always holds, regardless of the values of p and q.

But this looks a bit tricky...

Fortunately, we can break our problem into smaller pieces.

We know what the behaviour is of logical implication.

To check if $(\neg p \vee q) \Rightarrow (p \Rightarrow q)$ is true, we need to establish whether or not $(\neg p \vee q)$ and $(p \Rightarrow q)$ are true or not.

Let's add two new columns to our truth table:

Proofs

p	q	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	?	?	?
F	T	?	?	?
T	F	?	?	?
T	T	?	?	?

Once we know the values of our new columns, we can fill in the last column.

We can repeat this trick, and split $\neg p \vee q$ into two pieces, $\neg p$ and q.

Because we already have a column for q, we add one new column for $\neg p$.

p	q	$\neg p$	$\neg p \lor q$	$p \Rightarrow q$	$(\neg p \lor q) \Leftrightarrow (p \Rightarrow q)$
F	F	?	?	?	?
F	T	?	?	?	?
T	F	?	?	?	?
T	T	?	?	?	?

Now let's start filling in our truth table.

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	**T**	?	?	?
F	T	?	?	?	?
T	F	?	?	?	?
T	T	?	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	?	?	?
F	T	**T**	?	?	?
T	F	?	?	?	?
T	T	?	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	?	?	?
F	T	T	?	?	?
T	F	**F**	?	?	?
T	T	?	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	?	?	?
F	T	T	?	?	?
T	F	F	?	?	?
T	T	**F**	?	?	?

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	?	?	?
F	T	T	?	?	?
T	F	F	?	?	?
T	T	**F**	?	?	?

Now let's look at the next column in our table...

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	**T**	?	?
F	T	T	?	?	?
T	F	F	?	?	?
T	T	F	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	T	?	?
F	T	T	**T**	?	?
T	F	F	?	?	?
T	T	F	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	T	?	?
F	T	T	T	?	?
T	F	F	**F**	?	?
T	T	F	?	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	T	?	?
F	T	T	T	?	?
T	F	F	F	?	?
T	T	F	**T**	?	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	T	**T**	?
F	T	T	T	**T**	?
T	F	F	F	**F**	?
T	T	F	T	**T**	?

Proofs

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$(\neg p \vee q) \Leftrightarrow (p \Rightarrow q)$
F	F	T	T	T	**T**
F	T	T	T	T	**T**
T	F	F	F	F	**T**
T	T	F	T	T	**T**

p	q	$(\neg p$	$\vee$	$q)$	$\Leftrightarrow$	$(p \Rightarrow q)$
F	F	T	T	F	**T**	T
F	T	T	T	T	**T**	T
T	F	F	F	F	**T**	F
T	T	F	T	T	**T**	T

Instead of adding lots of columns, we can space out the truth table a bit, writing the value of each subexpression:

- Start with $\neg p$ en q;
- Then $\neg p \vee q$;
- Then $(p \Rightarrow q)$;
- And finally, the entire implication.

Exercise

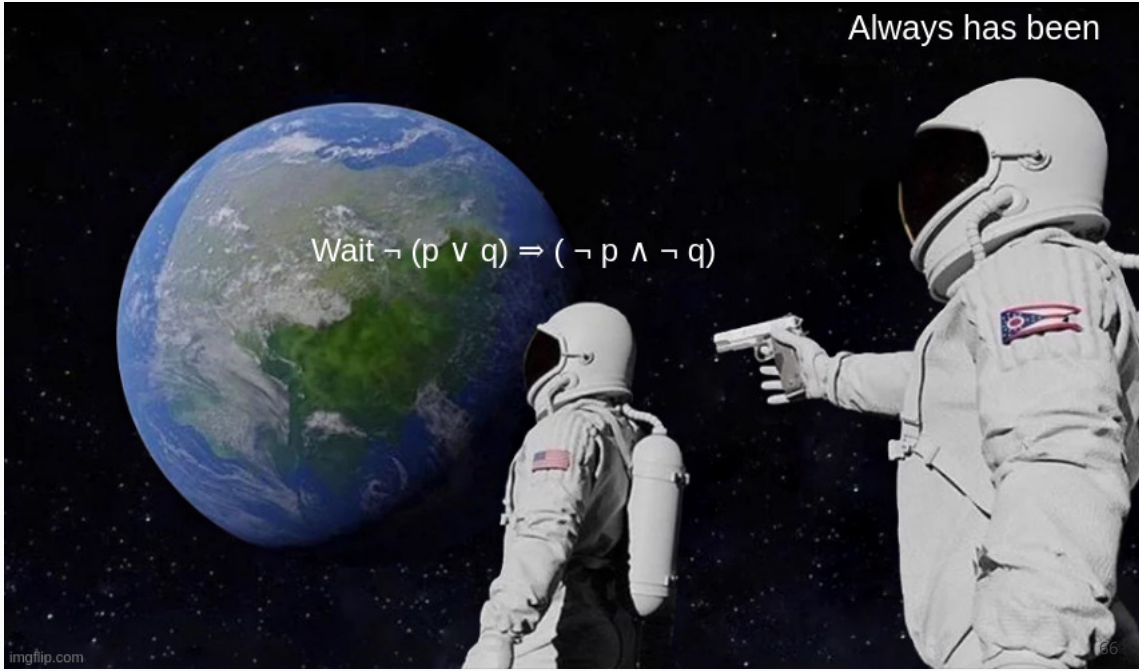
Question: Complete the following truth table for the propositional formula

$\neg(p \vee q) \Rightarrow (\neg p \wedge \neg q)$

p	q	$\neg$	$(p$	$\vee$	$q)$	$\Rightarrow$	$(\neg p$	$\wedge$	$\neg q)$

Solution

p	q	¬	(p	∨	q)	⇒	(¬p	∧	¬q)
F	F	T	F	F	F	**T**	T	T	T
F	T	F	F	T	T	**T**	T	F	F
T	F	F	T	T	F	**T**	F	F	T
T	T	F	T	T	T	**T**	F	F	F


Always has been
Wait ¬ (p ∨ q) ⇒ (¬ p ∧ ¬ q)
imgflip.com

Tautologies and contradictions

Using truth tables, we can check when a given proposition is always true or not.

A propositional formula that always holds is known as a **tautology**:

- $p \vee \neg p$
- $p \Rightarrow p$
- $(p \Rightarrow q) \Rightarrow (\neg q \Rightarrow \neg p)$
- ... and many others

A proposition that is always false is called a **contradiction**.

Counter examples

Not every propositional formula is true.

Question: Why are $(p \Rightarrow q)$ and $(q \Rightarrow p)$ not equivalent?

Counter examples

Not every propositional formula is true.

Question: Why are $(p \Rightarrow q)$ and $(q \Rightarrow p)$ not equivalent?

Let's fill in a truth table

p	q	$(p \Rightarrow q)$	$(q \Rightarrow p)$
F	F	T	T
F	T	**T**	**F**
T	F	**F**	**T**
T	T	T	T

Now we can read of which choice of p and q distinguishes these two formulas.

Beyond truth tables

Proofs

Using truth tables, we can establish when a given formula holds or not.

Is that the only way to verify a propositional formula?

Proofs

Using truth tables, we can establish when a given formula holds or not.

Is that the only way to verify a propositional formula?

No! There are many other techniques

- algebraic proofs;
- natural deduction;
- semantic tableaux;
- …

Truth tables are relatively straightforward and don't require any mathematically mature techniques – making them perfect for us to start!

Alternative: using algebraic laws

In high school, you may have manipulated equations such as:

$$
\begin{aligned}
&3x + (x^2 - 1 - 3x) \\
&\quad = (x^2 - 1) \\
&\quad = (x - 1)\,(x + 1)
\end{aligned}
$$

Similarly, you might simplify fractions, solve equations, etc.

Alternative: using algebraic laws

In high school, you may have manipulated equations such as:

$$
\begin{aligned}
&3x + (x^2 - 1 - 3x) \\
&\quad = (x^2 - 1) \\
&\quad = (x - 1)\,(x + 1)
\end{aligned}
$$

Similarly, you might simplify fractions, solve equations, etc.

Doing so relies on fundamental properties of addition, multiplication, subtraction and division:

- $x + y - y = x$
- $x + y = y + x$
- $1 \times y = y$
- $0 + y = y$
- ...

Alternative: algebraic laws

Similar laws hold for propositional formulas

Commutativity

- $p \wedge q \Leftrightarrow q \wedge p$
- $p \vee q \Leftrightarrow q \vee p$

Associativity

- $p \wedge (q \wedge r) \Leftrightarrow (p \wedge q) \wedge r$
- $p \vee (q \vee r) \Leftrightarrow (p \vee q) \vee r$

Idempotence

- $p \wedge p \Leftrightarrow p$
- $p \vee p \Leftrightarrow p$

Alternative: algebraic laws

De Morgan

- $\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$
- $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

Properties of T en F

- $p \vee F \Leftrightarrow p$
- $p \wedge F \Leftrightarrow F$
- $q \vee T \Leftrightarrow T$
- $q \wedge T \Leftrightarrow q$

Double negation

- $p \Leftrightarrow \neg (\neg p)$

Alternative: algebraic laws

Tertium non datur

- $p \vee \neg p \Leftrightarrow \mathsf{T}$

Implication

- $(p \Rightarrow q) \Leftrightarrow (\neg p \vee q)$

Contraposition

- $(p \Rightarrow q) \Leftrightarrow (\neg q \Rightarrow \neg p)$

And many more…

(By the way, how do you prove such laws?)

Example

Instead of filling out a truth table, prove $(p \Rightarrow q) \vee (q \Rightarrow p)$ directly using these laws.

Example

Instead of filling out a truth table, prove $(p \Rightarrow q) \lor (q \Rightarrow p)$ directly using these laws.

$(p \Rightarrow q) \lor (q \Rightarrow p)$	(implication)
$\Leftrightarrow (\neg p \lor q) \lor (\neg q \lor p)$	(associativity)
$\Leftrightarrow \neg p \lor ((q \lor \neg q) \lor p)$	(tertium non datur)
$\Leftrightarrow \neg p \lor (\mathsf{T} \lor p)$	(property of T)
$\Leftrightarrow \neg p \lor \mathsf{T}$	(property of T)
$\Leftrightarrow \mathsf{T}$	(property of T)

Example

Instead of filling out a truth table, prove $(p \Rightarrow q) \lor (q \Rightarrow p)$ directly using these laws.

$(p \Rightarrow q) \lor (q \Rightarrow p)$	(implication)
$\Leftrightarrow (\neg p \lor q) \lor (\neg q \lor p)$	(associativity)
$\Leftrightarrow \neg p \lor ((q \lor \neg q) \lor p)$	(tertium non datur)
$\Leftrightarrow \neg p \lor (\mathsf{T} \lor p)$	(property of T)
$\Leftrightarrow \neg p \lor \mathsf{T}$	(property of T)
$\Leftrightarrow \mathsf{T}$	(property of T)

We can compute with propositions!

Taking a step back

What have we achieved?

Taking a step back

What have we achieved?

- We have defined what propositional formulas are;
- We have given meaning to these formulas using truth tables;
- Using truth tables, we can establish that a given formula is true (or find a counter example if it is not);

This gives us the ‘mental toolbox’ to precisely describe all kinds of situations when developing software.

Why do all this?

I may not spend much time in my research filling out truth tables

Why do all this?

I may not spend much time in my research filling out truth tables

I reason with propositional operators every day!

Why do all this?

I may not spend much time in my research filling out truth tables

I reason with propositional operators every day!

And the only way to come to grips with the meaning of these operators is **practice** – in particular by filling in truth tables.

Other logics

Propositional logic is one of the simplest logics that exist.

You can’t formulate very interesting properties…

But it’s an excellent first step!

Other logics

Propositional logic is one of the simplest logics that exist.

You can't formulate very interesting properties…

But it's an excellent first step!

Throughout the rest of this course and the remainder of your degree, you may run into much richer logics:

- Predicate logic
- Higher-order logic
- Constructive logic
- Modal logic
- …

Next time

Propositional logic lets us write and prove interesting statements.

In the next lecture, we will study *sets* – that allows us to model interesting data.

And as it will turn out, these sets and propositions share some very similar structure…

Next time

Propositional logic lets us write and prove interesting statements.

In the next lecture, we will study *sets* – that allows us to model interesting data.

And as it will turn out, these sets and propositions share some very similar structure…

Don’t forget the first exercise session after this lecture!

Literature

- *Modelling Computer Systems: Mathematics for Computer Science* - Chapter 1