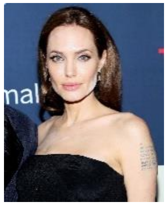
COM1002
Foundations of Computer Science

Dr Paul Watton

Modelling with Prop. Logic: Ex 1

Brad wishes to go to a party tonight and would be happy to go with either **Jen** or **Ang**. However as he is currently dating both Jen and Ang, he doesn't want to go to the party if they both will be there.

Can we formalise Brad's predicament with propositional logic ?



A= Ang goes to party

B = Brad goes to party

J = Jen goes to party

if Ang and Jen both go to the party.
then Brad does not go to the party.

$$(A \wedge J) \Rightarrow \neg B$$

if Ang and Jen both go to the party.
then Brad does not go to the party.

$$(A \wedge J) \Rightarrow \neg B$$

Are there equivalent ways of expressing this statement in propositional logic ?

It is **not** the case that Brad, Ang and Jen all go to the party. $\neg(B \wedge A \wedge J)$

Brad doesn't go to party **or** Ang doesn't go to party **or** Jen doesn't go to party. $\neg B \vee \neg A \vee \neg J$

Brad goes to party *only if* Ang and Jen don't both go to the party. $B \Rightarrow \neg(A \wedge J)$

How do we show that statements are logically equivalent ?

- using truth tables

- using Algebraic laws for logical equivalences

Todays lecture:

- Propositional logic syntax
- Truth tables
- Equivalent propositions
- Algebraic laws of Propositional calculus
- Example: modelling with propositional logic

A logician said to his son…

"If you don't eat your vegetables, you can't have any ice cream."

Upon hearing this, the son choked down a plate of broccoli, and his father, duly impressed, sent him to bed without any ice cream.

LEARNING OUTCOME

By the end of this lecture we should be able to understand this joke using our knowledge of propositional logic…

(*It is not required to find it funny…)*

Negation

For a proposition **p**:

- written **¬ p**,
- usually pronounced “**not p**”,
- meaning of **¬ p** is “**p is false**”.

Laws for negation:

- **¬ true** = **false**, and **¬ false** = **true**;
- for any **p**, **¬ ¬ p = p**
  - *the law of double negation.*

p	**¬ p**
F	**T**
T	**F**

Disjunction

For propositions **p** and **q**:

- written **p** $\vee$ **q**,
- usually pronounced "**p *or* q**",
- is true if either p is true, or q is true, or both,
- **p** and q are often called disjuncts.

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

Law for disjunction: **p** $\vee \neg$ **p** = true

the law of the excluded middle.

Conjunction

For propositions **p** and **q**:

- written $p \wedge q$,
- usually pronounced "**p** and **q**",
- is true if **both** **p** is **true** and **q** is **true**,
- **p** and **q** are often called conjuncts.

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

Law for conjunction:

$p \wedge \neg p =$ **false.**

Implication

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

For propositions **p** and **q**:

- written **$p \Rightarrow q$**,
- p is the premise
- q is the conclusion.

usually pronounced "**p** implies **q**", or "if **p** then **q**",

($p \Rightarrow q$) is **TRUE** if **p** is false, or **q** is true

EQUIVALENTLY

($p \Rightarrow q$) is **FALSE** if, and only, p is **TRUE**, **and** q is **FALSE**,

Law for implication: **$p \Rightarrow q$** is equivalent to **$\neg q \Rightarrow \neg p$**.

Equivalence

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

For propositions **p** and **q**:

written $p \Leftrightarrow q$,

usually pronounced

"**p** is equivalent to **q**", or "**p**, if and only if **q**",

is **true** if **both p and q have the same value**

Law for equivalence:

$p \Leftrightarrow q$ is equivalent to $(p \Rightarrow q) \wedge (q \Rightarrow p)$.

Propositional Logic Syntax

Defines propositional formulae:

Atomic formulae:

- two constants (true, false),
- single variables P, Q, R, …;

Compound formulae:

- formulae linked by the connectives,
- **$\neg$ p, p $\vee$ q, p $\wedge$ q, p $\Rightarrow$ q, p $\Leftrightarrow$ q.**

Propositional Logic Syntax 2

Parentheses and Precedences

Example from ordinary algebra:

- 5 + 3 x 2 is understood as 5 + (3 x 2),
- the brackets specify the order of execution,
- but the operators have a "precedence":

  multiplication has higher precedence than addition.

Precedence in propositional logic:

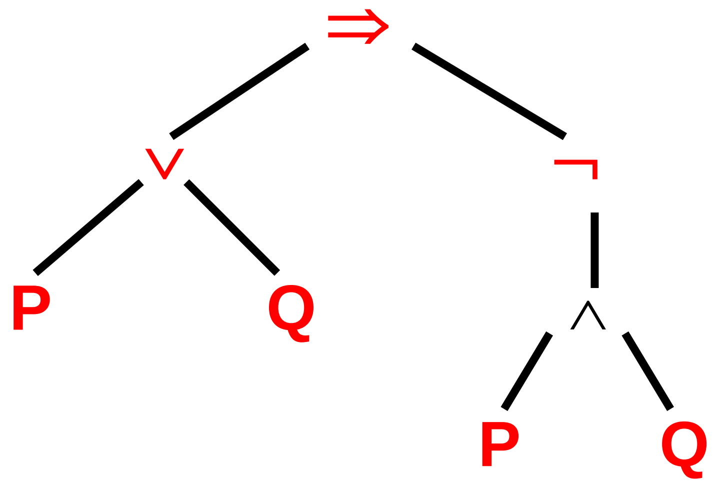
- highest is $\neg$, then $\wedge$**,** $\vee$**,** $\Rightarrow$, lowest is $\Leftrightarrow$,
- there is also a right-to-left rule,
- *but if in doubt*: **use brackets**.

Syntax Trees

Diagrammatic representations of the structures of formulae:

Eg the formula $(\mathbf{P} \vee \mathbf{Q}) \Rightarrow \neg\, (\mathbf{P} \wedge \mathbf{Q})$

The tree is:



Try: *exercise 1.23*

Exercise 1.23

Construct truth tables for the following formulae:

$$\neg(P \Leftrightarrow \neg Q)$$

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

$$(P \wedge Q) \Rightarrow (\neg R \wedge S)$$

Equivalent Propositions

Different propositions may be equivalent:

e.g. in the earlier example:

$$B \Rightarrow \neg(A \wedge J) \qquad\qquad (A \wedge J) \Rightarrow \neg B$$

How do we show they are logically equivalent ?

Equivalent Propositions

If **p** and **q** are logically equivalent **propositional statements**, then:

their truth tables must be the same,

and

the proposition p $\Leftrightarrow$ q must be **true**;

A proposition that **must be true** is called a **tautology**:

A proposition that **must be false** is called a **contradiction**.

Try: *EXERCISE 1.25*

Exercise 1.25

Construct truth tables for each of the following formulae to determine which are tautologies and which are contradictions.

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

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

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

$$(p \Rightarrow q) \Rightarrow p$$

$$p \Rightarrow (q \Rightarrow p)$$

Modelling with Prop. Logic: Ex 2

An Airplane Controller Example:

Jimbo

Some current code:

if (*CabinPressure* < *MinPressure*)
then *PrepareForLanding*;
if (*FlightHeight* < *MinHeight*)
then *PrepareForLanding*;

Gloria Gatwick

Amanda baggage

A programmer proposes optimisation:

if ((*CabinPressure* < *MinPressure*)
and (*FlightHeight* < *MinHeight*))
then *PrepareForLanding*;

Sammy steps

Tommy Tow-Truck

i) Is this a valid optimisation?
ii) How do we verify whether this is a valid optimisation ?
iii) Clearly – serious implications, e.g.:
:

Will Jimbo land successfully and be reunited with his friends ?

Solution:

Define appropriate variables:

let `P` be '`CabinPressure < MinPressure`'

let `H` be '`FlightHeight < MinHeight`'

let `L` be the action '`PrepareForLanding`'

Model the current behaviour: $(P \to L) \wedge (H \to L)$

Consider two optimisations: $(P \wedge H) \to L$

$$(P \vee H) \to L$$

Which of these formulae are correct ?

(use Truth Tables)

Solution (using truth tables)

P	H	L	$(P \vee H)$	$(P \wedge H)$	$P \to L$	$H \to L$	$(P \vee H) \to L$	$(P \wedge H) \to L$	$(P \to L) \wedge (H \to L)$
T	T	T	T	T	T	T	**T**	T	**T**
T	T	F	T	T	F	F	**F**	F	**F**
T	F	T	T	F	T	T	**T**	T	**T**
T	F	F	T	F	F	T	**F**	T	**F**
F	T	T	T	F	T	T	**T**	T	**T**
F	T	F	T	F	T	F	**F**	T	**F**
F	F	T	F	F	T	T	**T**	T	**T**
F	F	F	F	F	T	T	**T**	T	**T**