Logic and Proof notes

Jamie Balfour

10/3/2012

Notes on logic

$2 \mid a$ ← means that 2 divides by a $(2 \mid a)$

$\frac{a}{2}$ is a number

To have a programming language you must:
$\mathbb{N}$ - must use real numbers (natural)

$x := 0$
$x = x + 1$

$x := x - 1$

while $x \neq y$

Lecture 8

A statement is a sentence that is capable of being either True or False.

How do we decide if a statement is true or false?

It is true precisely when we can prove it.

To do this we must have some background assumptions, often called axioms.

A proof of a statement is an argument that makes use of axioms, previously proved statements and the rules of logic that concludes with the

statement we are trying to prove.

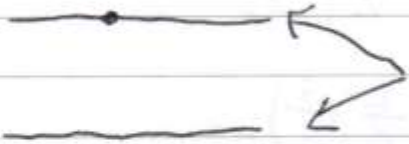
What is an argument?

This is what logic tells us.

proved statements are called theorem, proposition, lemma

If you think something is true, this is called a conjecture.

This approach goes back to Euclid's book, "The Elements" where theorems of geometry were proved by 5 axioms.



Playfair's axiom

Between the axioms and the theorems are the definitions

+	odd	even
odd	even	odd
even	odd	even

×	odd	even
odd	odd	even
even	even	even

1) If x is even, then x^2 is even.

2) If x^2 is even, then x is even.

We know that x is either odd or even.

- We suppose that x is odd.

Then x^2 is odd - contradicting the statement.

We deduce that x must be even, as there are only two paths.

We can write this as follows:

$$\text{If } 2 \mid x^2 \text{ then } 2 \mid x$$

Rational numbers $\frac{p}{q}$ $\left(\frac{1}{2}, \frac{2}{4}, \frac{3}{11}\right)$...

$\frac{1}{2} = \frac{2}{4} = \frac{6}{12}$ etc...

$\frac{6}{12} = \frac{3}{6} = \frac{1}{2}$

Every rational number can be written as a fraction in its lowest terms.

A real number is irrational if it is not rational.

Claim

"root two"

Prove $\sqrt{2}$ is irrational.

Proof

Assume that $\sqrt{2}$ is rational

This means that we can find $a, b \in \mathbb{N}$

$$\sqrt{2} = \frac{a}{b}$$

To get rid of square root so we can have an equation just involving numbers.

$$2 = \frac{a^2}{b^2}$$

Multiply both sides by b^2:

$$2b^2 = a^2$$

We can now see that $2 \mid a^2$ and using a previous theorem, $2 \mid a$, this means

$a = 2\alpha$ for some α α = alpha

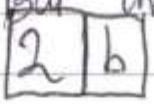
Sub $a = 2\alpha$ in $2b^2 = a^2$

$$2b = (2\alpha)^2 = 4\alpha^2$$

Divide both sides by 2

$$b^2 = 2a^2$$

But this implies $2 \mid b^2$; but in our previous theorem,



This is a contradiction because we assumed $\frac{a}{b}$ is its lowest term; therefore $\sqrt{2}$ is irrational.

Lecture 9

Babylon 1

When will this program terminate?

Two successive guesses need to be equal.

next guess ← x ; current guess ← $\frac{1}{2}\left(x + \frac{a}{x}\right)$

$$x = \frac{1}{2}\left(x + \frac{a}{x}\right)$$

$$(\times 2) \quad 2x = x + \frac{a}{x}$$

$$(\times x) \quad 2x^2 = x^2 + a$$

$$(-x^2) \quad x^2 = a$$

$$\therefore \quad x = \sqrt{a}$$

Reciprocal subtraction

Input $(a, b) \in \mathbb{N}^{\times} \times \mathbb{N}^{\times}$ where $\mathbb{N} = \{1, 2, 3, \ldots\}$

Procedure If $a = b$ output a and stop

If $a > b$

$$(a, b) \rightarrow (a - b, b)$$

If $b > a$

$$\searrow (a, b - a)$$

repeat

Examples

- $(12,6) \to (6,6)$ Stop.
- $(9,2) \to (7,2) \to (5,2) \to (3,2) \to (1,2) \to \underline{(1,1)}$ St

Two basic questions

1) What is this program doing?
2) Will this program terminate?

a, b are $\mathbb{N}$

If $d|a$ and $d|b$ then d is called a common divisor.

The largest common divisor of a and b is called the greatest common divisor (gcd)

Claim Reciprocal subtraction computes gcd (a,b)

When the program terminates it equals (a, a) and the gcd is a.

$a > b$

$(a,b) \longrightarrow (a-b, b)$ (c,c)

Claim gcd does not change $\gcd(a,b) = \gcd(a-b,b)$

$d|a$ $d|b$ then $d|a+b$ and $d|a-b$ $d|b$

$e \mid a-b, e \mid b$ means

$a - b = ex$ for some x

$b = ey$ for some y

$a = (a-b) + b = ex + ey\ (x+y)$

$\therefore e \mid a$

For revision

Counting programs
Transistor
Sets

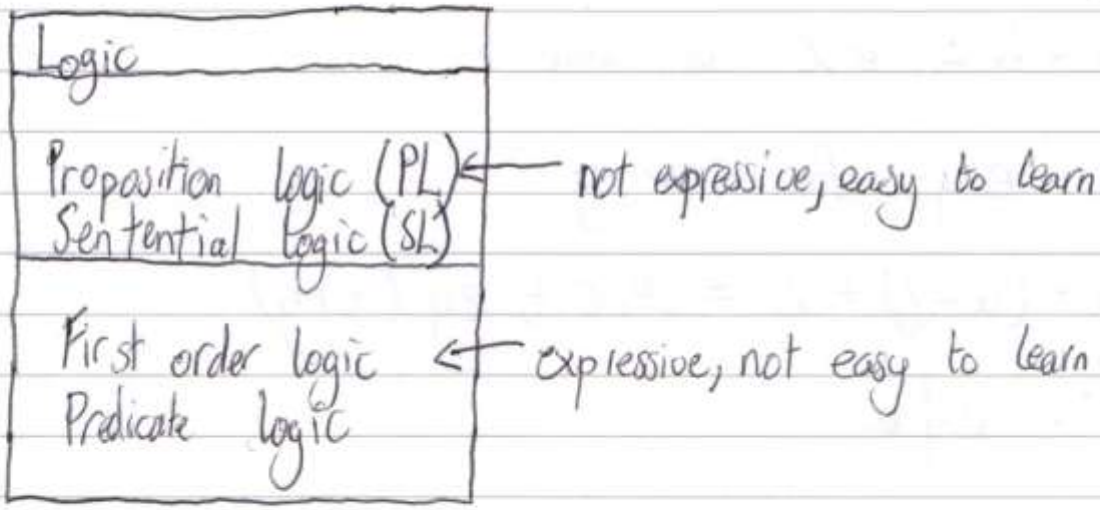
→ Test for Logic and Proof

Lecture 10

Propositional Logic

Language
- natural language (Welsh, Estonian, ...)
  - expressive/ambiguous
- artificial languages
  - set theory (all maths)
  - programming languages (Java)
    (describes algorithms)
    - first-order logic - describing reasons

Both PROLOG and databases use first-order logic.



$P \neq NP$?

Informal description of propositional logic

We are interested only in those sentences which are capable of being either True (T) or False (F).

We call such sentences statements.

We shall now analyse the structure of statements. We shall do this in terms of certain connectives.

"It is not the case that" S = "It is raining"

"It is not the case that it is raining" = "It is not raining"

"Not (It is raining)" Negation

Not (True) = False Not (False) = True

Negation is a unary connective.

"And" - no sense of time

It is raining and it is snowing.

means the same as

It is snowing and it is raining.

"Or" - this is inclusive or

"either p or q or both p and q"

"Implies" - "p implies q" "If p then q"

This has a special meaning - see later.

"If and only if" - usually abbreviated as Iff

"p if and only if q" means

(p implies q) and (q implies p)

Notation

English	Symbol	Name
It is not the case that	$\neg$ $(\sim)$	Negation
And	$\wedge$ $(\&)$	Conjunction
Or	$\vee$	Disjunction
Implies	$\rightarrow$	Implication
Iff	$\leftrightarrow$	Biconditional

"An atomic statement is one that cannot be analysed any futher using the above connectives."

Some examples

1) p = "n is an even number"

   $p \vee \neg p$ = n is either even or odd

2) p = "2 is prime" q = "2 is even"

   $p \wedge q$ = p (2) is prime and q (2) is even

3) p = "x is even" q = "x^2 is even"

Not (True) = False Not (False) = True

Negation is a unary connective.

"And" - no sense of time

It is raining and it is snowing.

means the same as

It is snowing and it is raining.

"Or" - this is inclusive or

"either p or q or both p and q"

"Implies" - "p implies q" "If p then q"

This has a special meaning - see later.

"If and only if" - usually abbreviated as Iff

"p if and only if q" means

(p implies q) and (q implies p)

Notation

English	Symbol	Name
It is not the case that	$\neg$ $(\sim)$	Negation
And	$\wedge$ $(\&)$	Conjunction
Or	$\vee$	Disjunction
Implies	$\longrightarrow$	Implication
Iff	$\longleftrightarrow$	Biconditional

"An atomic statement is one that cannot be analysed any futher using the above connectives."

Some examples

1) p = "n is an even number"

$p \vee \neg p$ = n is either even or odd

2) p = "2 is prime" q = "2 is even"

$p \wedge q$ = p (2) is prime and q (2) is even

3) p = "x is even" q = "x^2 is even"

$p \rightarrow q$ is true "if x is even then x^2 is even"

$q \rightarrow p$ "If x^2 is even then x is even"

$p \leftrightarrow$ "x is even if and only if x^2 is even"

Key principle

The truth or falsity of a compound statement is entirely determined by the truth or falsity of the atoms that make it up and the connectives used.

Compound statement

Is one constructed from atomic statements using the connectives.

Example

$A = p \wedge q$

A is a compound statement
p, q are its atoms

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

Formal presentation of propositional logic

Syntax is the grammar of a sentence.
Semantics is the meaning of a sentence.

Syntax of propositional logic

Atoms/atomic statements: $p, q, r, p_1, p_2, p_3 \ldots$

A well formed formula (WFF) is constructed as follows:

(WFF1) All atoms are WFF
(WFF2) If A and B are WFF so too are

$$(\neg A),\ (A \vee B),\ (A \wedge B),\ (A \rightarrow B),$$

$$(A \leftrightarrow B)$$

(WFF3) All WFF are constructed using (WFF1) and (WFF2) a finite number of times.

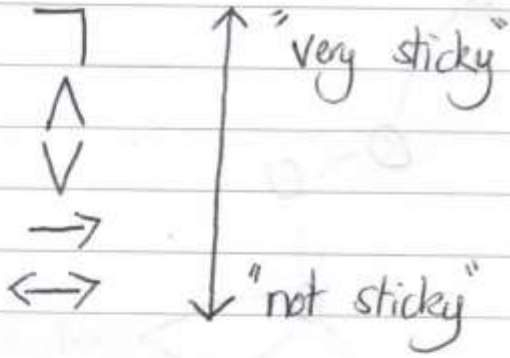
Example

Show that $(\neg((p \vee q) \wedge r))$ is a WFF.

- p, q, r are WFF by (WFF1)

This is the proper definition - we shall usually be much more lax!

- Omit outer brackets
- Omit brackets in the definition of negation.



Examples

1) $(\neg p) \vee q$ ← these are stuck together

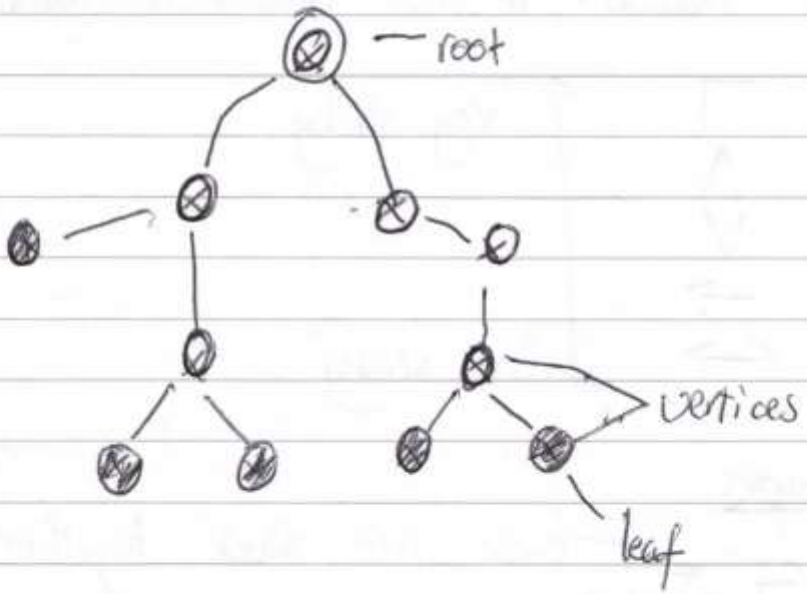
$\neg(p \vee q)$ ← this means that the negation is stuck to the whole formula

2) $\neg p \rightarrow (q \vee r)$

3) $(p \vee q) \leftrightarrow (\neg r \rightarrow (p \leftrightarrow q))$

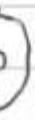
Trees

This is a graphical device useful for representing information. An example is given below.

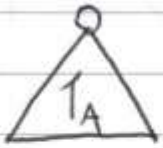


Parse: "show the grammatical structure of..."

Part 1 of parse tree

The parse tree of an atom p is 

Part 2

If A has parse tree 

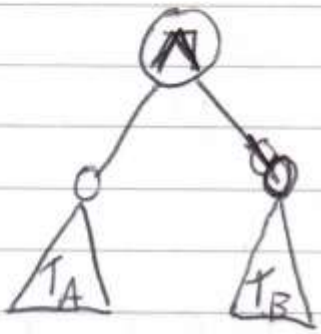
and

B has parse tree 

then

$\neg$ A has parse tree 

$(A \wedge B)$ has parse tree



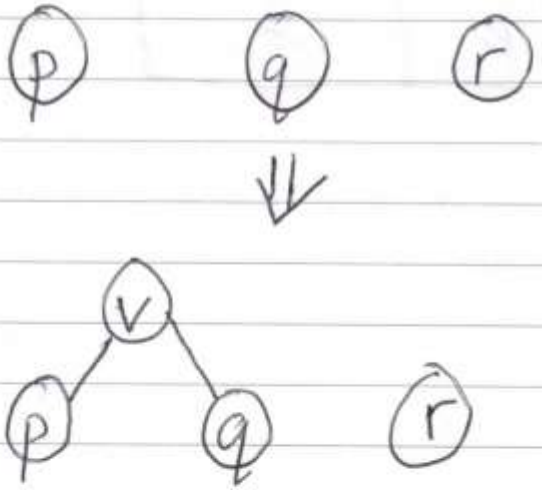
etc...

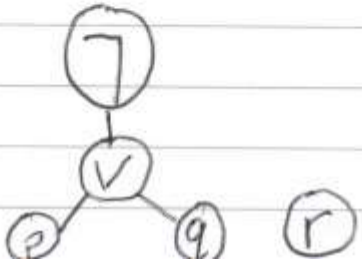
Part 3

Every parse tree is constructed using a finite number of applications.

Example

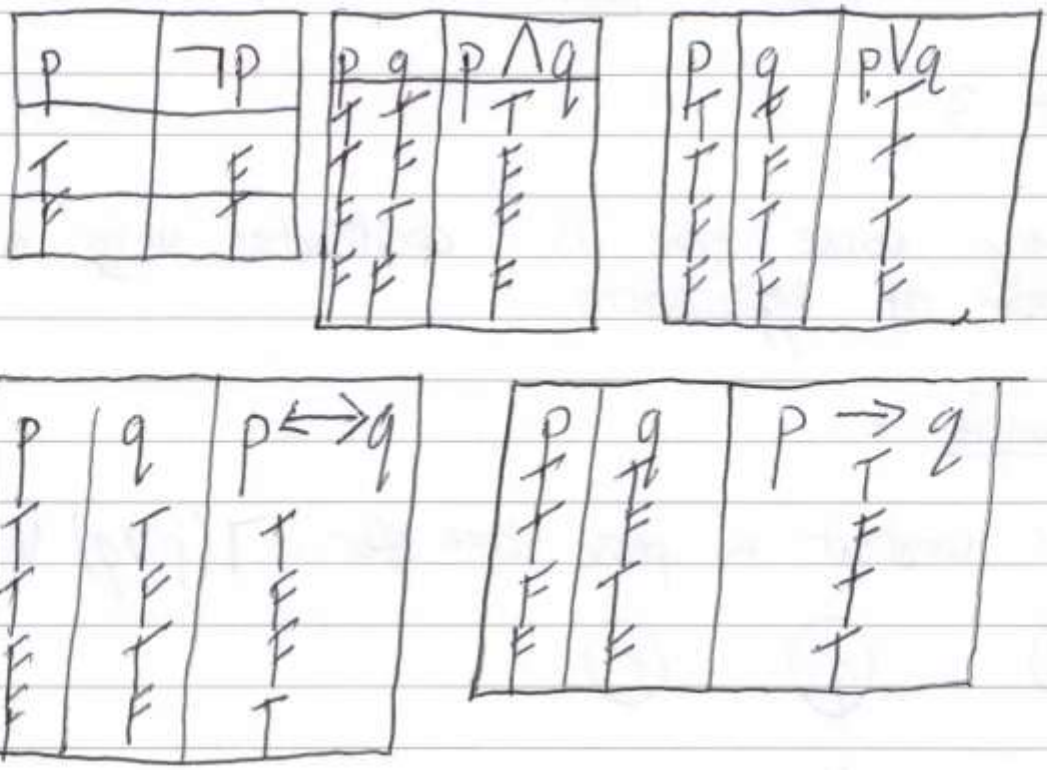
We construct a parse tree for $\neg(p \vee q) \vee r$



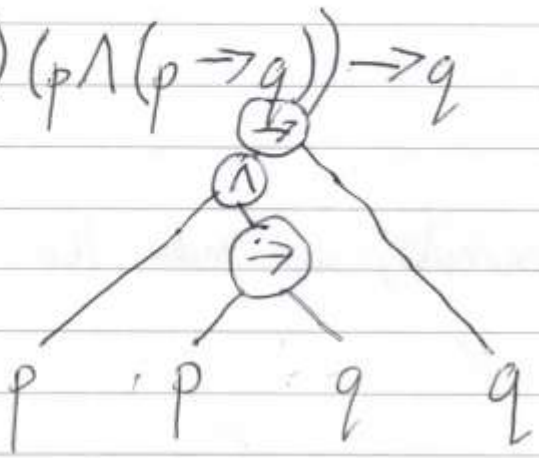


The Boolean connective corresponding to the root of the tree is called the principle connective.

Semantics - the truth tables



ii) $(p \wedge (p \rightarrow q)) \rightarrow q$



Truth tables - number of atoms to the power of two gives the row count.

i)

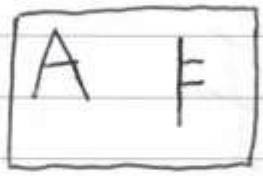
p	q	$p \vee q$	$p \rightarrow (p \vee q)$	A
T	T	T	T	T
T	F	T	F	F
F	T	T	T	F
F	F	F	T	F

2) $B = (p$

On the basis of the examples, we make the following definitions:

A is a WFF

- A is called a <u>contradiction</u> if it takes a False value for all truth assignments in its atoms.



Says A is a contradiction.

$\vdash$ - semantic

- A is called a <u>tautology</u> if it takes the value of True for all truth assignments in its atoms.

- A is called a contingency if it takes both the values true and false.

<u>The satisfyability problem</u>

- Given a WFF, determine whether there is some assignment of the truth table values which

makes it the value true. If there is, we say it is satisfiable.

The tautology problem

Given a WFF decide whether it is a tautology or not.

Simplifying WFF

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

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

p and $p \wedge (q \vee \neg q)$ have the same meaning.

We write $p \equiv p \wedge (q \vee \neg q)$, we can say that they are logically equivalent.

p and $p \wedge (q \vee \neg q)$ do not have the same atoms - so strictly speaking do not have the same truth tables.

There is a better way of putting it

We say $\boxed{A \equiv B}$ precisely when $A \leftrightarrow B$ is tatology

ie $\boxed{\vdash A \leftrightarrow B}$

Examples (Proofs in Ex4)

1) $p \to q \equiv \neg p \vee q$

$[\vdash (p \to q) \leftrightarrow (\neg p \vee q)]$

2) $p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$

$[\vdash (p \leftrightarrow q) \leftrightarrow [(p \to q) \wedge (q \to p)]]$

3) $\neg \neg p \equiv p$ (double negation)

4) $p \wedge p \equiv p$; $p \vee p \equiv p$ (idempotence)

5) $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

This means we can just write $p \wedge q \wedge r$

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

6) $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

With the distributary laws, what is true with And is true with Or.

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

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

These are known as De Morgan's Laws.

Exercises (to get started with)

1. What does $\neg (p \leftrightarrow q)$ mean?

$\neg p \leftrightarrow \neg q$

2. Determine whether $\models p \rightarrow (q \rightarrow p)$, Meaning=semantics

3. What happens when? $\not\models A$ Answer: $\neg \not\models A$

1.

p	q	$p \leftrightarrow q$	$\neg(p \leftrightarrow q)$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	T	F

This is the truth table for exclusive - xor

2. Draw a truth table

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

Truth functions

A WFF atoms $P_1, \ldots, P_n$

P_1	P_2	P_3	...	P_n

The truth table of A is actually a function:

$$\text{domain} = \{T, F\}^n = \{(V_1, V_2, \ldots V_n) : V_i \in \{T, F\}$$

$$\text{codomain} = \{T, F\}$$

I shall denote their function by f_n

$$f_n : \{T, F\}^n \to \{T, F\}$$

Key definition

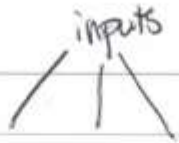
ANY function $f : \{T, F\}^n \to \{T, F\}$ is called a truth function.

Example

An arbitary truth function f

$$f : \{T, F\}^3 \to \{T, F\}$$

Draw a table



x	y	z	f(x,y,z) ← rule
T	T	T	T
T	T	F	F
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

Question: Can we find a WFF A with atoms p, q, r whose truth table is equal to table for f?

Normal forms and adequate sets of connectives

Will answer two questions:

1) The seemingly arbitrary choice of primitive logical connectives.

2) Given a truth function find a WFF whose truth table is equal to that function.

Negation Normal Form (NNF)

We have already proved the following:

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

We may use these results to show that every WFF is equivalent ($\equiv$) to one using only $\neg, \wedge, \vee$

We shall also use de Morgan's laws.

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

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

Example

$$\neg[p \rightarrow \neg(p \wedge q)] \equiv \neg[\neg p \vee \neg(p \wedge q)] \quad \begin{matrix} x \to \neg x \\ y \to \neg y \end{matrix}$$

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

$$\equiv p \wedge (p \wedge q)$$

$$\equiv (p \wedge p) \wedge q \quad \text{associability}$$

$$\equiv p \wedge q$$

Terminology A Literal is either an atom or the negation of an atom.

A WFF is in NNF if it is a WFF constructed using only $\wedge$, $\vee$ and literals.

This illustrates the proof of the following.

Proposition Every WFF is logically equivalent to a WFF in an NFF.

A set of logical connective is said to be adequate if every WFF is logically equivalent to one using only those connectives in the set.

Corollary $\{\neg, \vee, \wedge\}$ is an adequate set of connectives.

Observe that

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

by de Morgan

Corollary

i) $\{\neg, \wedge\}$ is an adequate set of connectives.

ii) $\{\neg, \vee\}$

Are there binary logical connectives which are adequate on their own?

$$p \downarrow q \overset{\text{def}}{=} \neg(p \vee q)$$

$\downarrow$ often called nor, Sheffer stroke

Proposition $\{\downarrow\}$ is an adequate set of connectives.

Proof $\neg p \equiv p \downarrow p$

$p \wedge q \equiv (p \downarrow p) \downarrow (q \downarrow q)$

There is another:

$p \mid q \overset{\text{def}}{=} \neg(p \wedge q)$

$\mid$ is called nand

Proposition show that $\{\mid\}$ is an adequate set of connectives.

Lecture 15

Disjunctive normal form (DNF)

A WFF is in DNF if it is a disjunction of a conjunction of literals.

Example

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

To convert a WFF to DNF, first covert NNF and then you may have to use the distributary laws:

$(a \vee b) \wedge c \equiv ($

Example write $(p \vee \neg\neg q) \wedge (\neg r \rightarrow s)$ in DNF

$(p \vee \neg\neg q) \wedge (\neg r \rightarrow s) \equiv (p \vee \neg\neg q) \wedge (\neg\neg r \vee s)$

$\equiv (p \vee q) \wedge (r \vee s)$

$\equiv ((p \vee q) \wedge r) \vee ((p \vee q) \wedge s)$

$\equiv (p \wedge r) \vee (q \wedge r) \vee (p \wedge s) \vee (q \wedge s)$

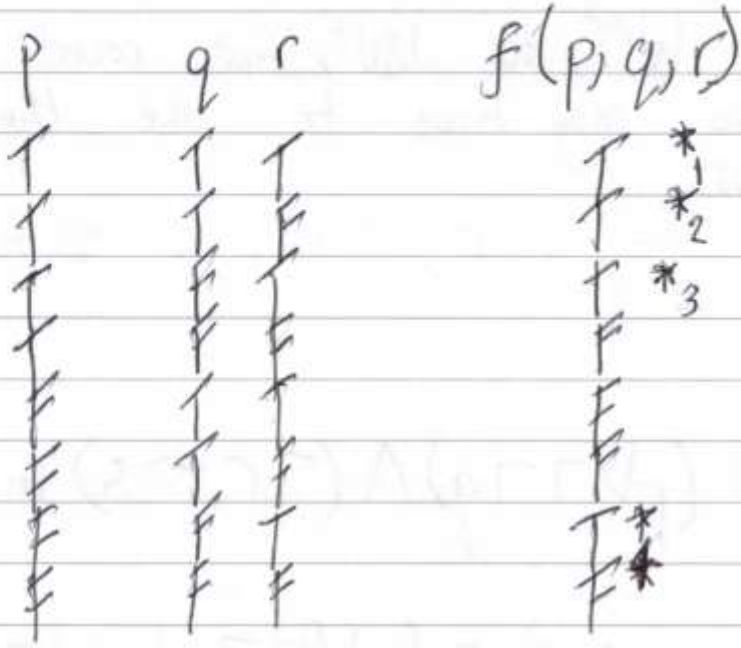
IMPORTANT

Given a truth function

$$f: \{T, F\}^n \to \{T, F\}$$

Find a WFF; X, whose atoms are n, where the truth table is equivalent to f.

Example Below is a truth table



① $p \wedge q \wedge r$ - True if p, q, r are true

② $p \wedge q \wedge \neg r$ - True if p, q are true and r = false

③ $p \wedge \neg q \wedge r$ - True if q is false and p, r are tru

④ $\neg p \wedge \neg q \wedge r$ - True when p, q are false and r is true

Overall

$$w = (p \wedge q \wedge r) \vee (p \wedge q \wedge \neg r) \vee (p \wedge \neg q \wedge r) \vee (\neg p \wedge \neg q \wedge r)$$

Special case

What if $f: \{T, F\}^n \to \{T, F\}$ is always false

$(x_1 \wedge \neg x_1) \wedge x_2 \wedge \dots \wedge x_n$

Proposition

Let $f: \{T, F\}^n \to \{T, F\}$ be any truth function. This is a WFF in DNF where truth table is equivalent to f.

NB. This is the basis of circuit design.

Conjunctive Normal Form (CNF) (important)

A WFF is in CNF if it is a conjunction of disconjunctions of literals.

Example

p	q	r	(A)

Same truth table as before

p	q	r	¬A
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	F

DNF for not A - ¬A.

Note to self
Learn to use ¬ instead of "not."

$$\neg A \equiv (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge q \wedge r) \vee (\neg p \wedge q \wedge \neg r) \vee (\neg p \wedge \neg q \wedge \neg r)$$

$$A \equiv \neg(\neg A) \equiv (\neg p \vee q \vee r) \wedge$$

$$p \vee \neg$$

Valid argument

We can now formalise within propositional logic what we mean by a valid argument.

We say that the statement B is a consequence of the statements $A_1, \ldots, A_n$ written

$$A_1, \ldots, A_n \models B$$

If and only if any truth assignment making all of $A_1, \ldots, A_n$ are true also makes B true. Therefore B is a consequence of those statements in $A_1, \ldots, A_n$.

$A_1, \ldots, A_n$ hypotheses/premeses

B is the conclusion.

Examples

1. ① $p, p \to q \vdash q$ We show it is a valid argument.

② We create a truth table.

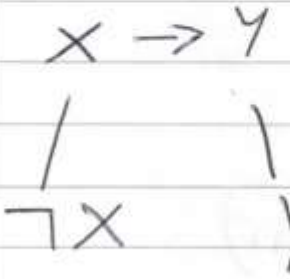
p	q	$p \to q$
T	T	T
F	F	F
F	T	T
F	F	T

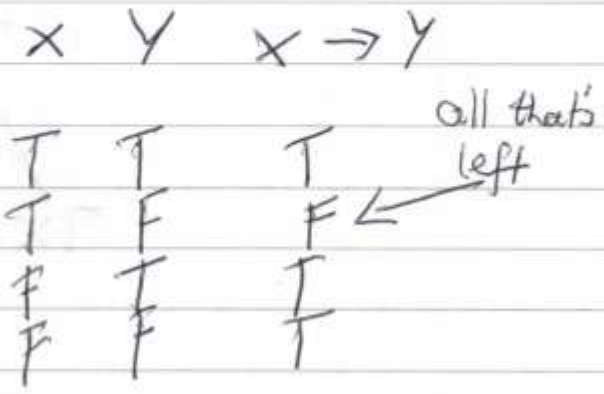
③ We are only interested in the cases where p is true and p implies ($\to$) q is true.

2. $p \to q, \neg q \vdash \neg p$ (modus tollens)

We show this is a valid argument.

p	·	q	¬p	¬q	p → q
T		T	F	F	T
T		F	F	T	F
F		T	T	F	T
F		F	T	T	T ← this line





Truth table examples

A truth tree is a data structure that encodes the same information as a truth table but in a more compact form.

Parse tree - syntactic information

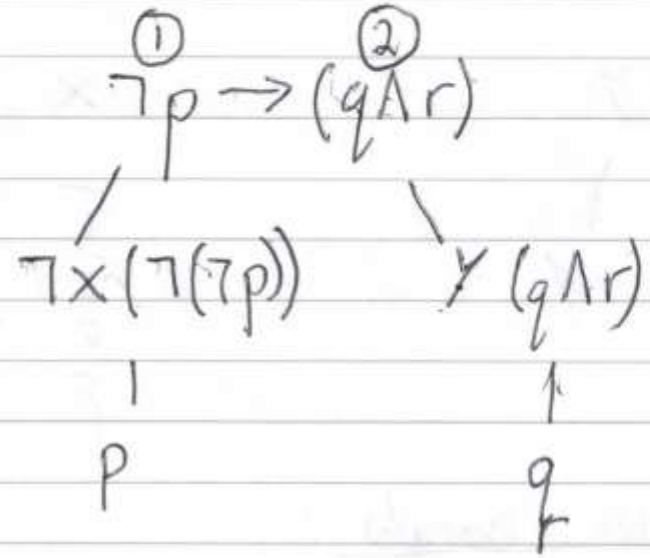
truth tree - semantic information

Truth trees can be used to find all the truth assignments to the atoms of a WFF that make it true.

What is left will be false.

Example 1 - Find all the truth table assignments that make the WFF true.

$\neg p \rightarrow (q \wedge r)$



The WFF at the root of the tree is True iff (if and only if)

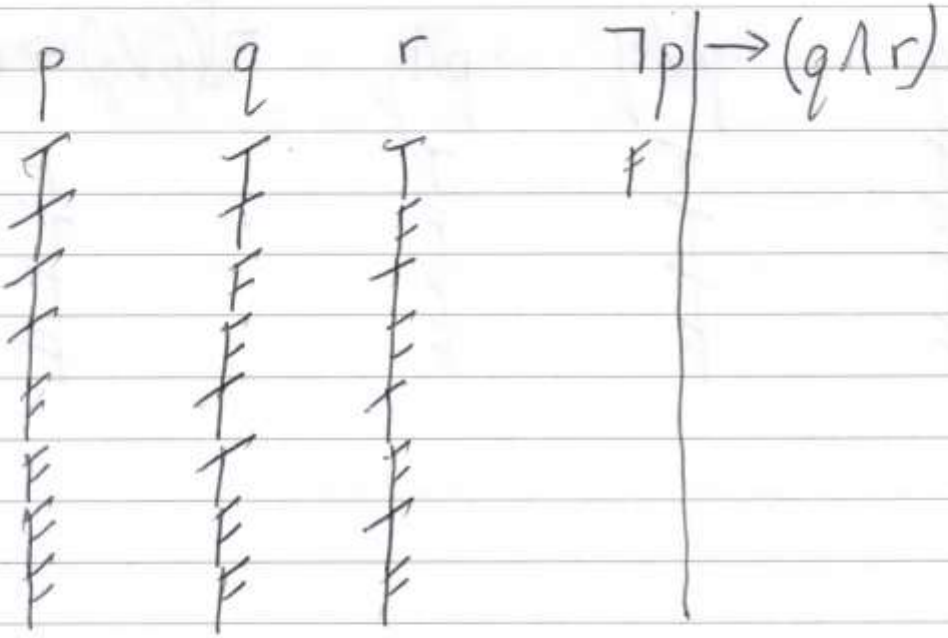
① p is true

or

② q and r is true

p	q	r	¬p → (q ∧ r)
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	T
F	T	T	T
F	T	F	
F	F	T	
F	F	F	

As soon a p is false, the rest of the row is false.



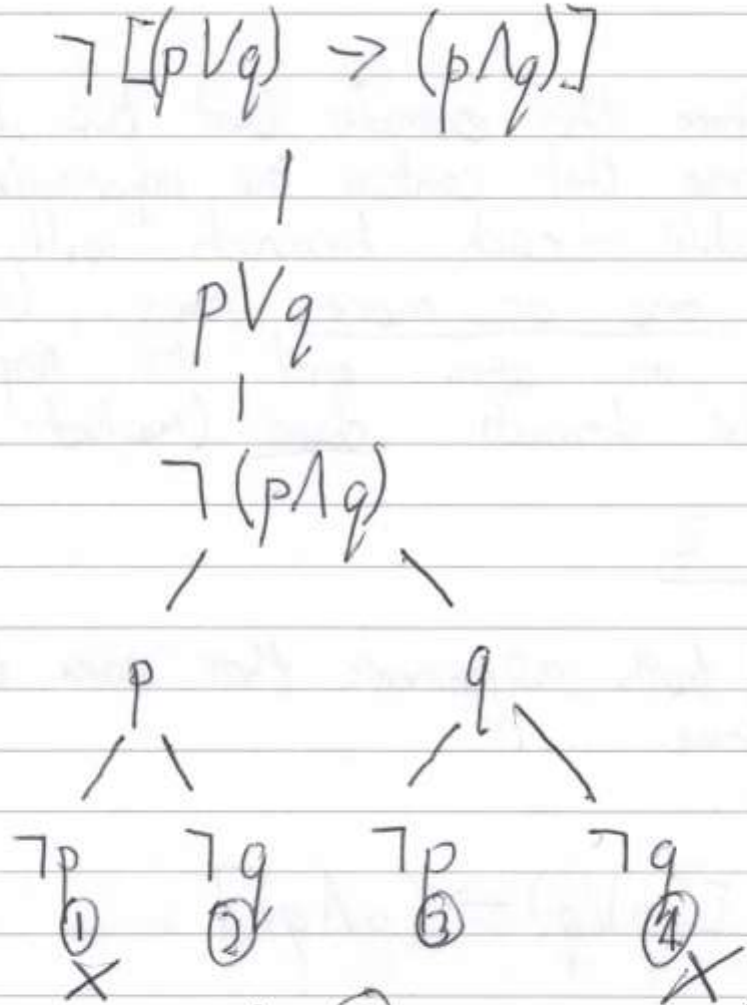
We see from this example that the branches of truth tree that contain the information about the truth table - each branch will contain information about one or more rows UNLESS it contains an atom and its negation - in which case the branch closes (marked with an X)

Example 2

Find all truth assignments that make the following WFF true.

$$\neg [(p \vee q) \rightarrow (p \wedge q)]$$

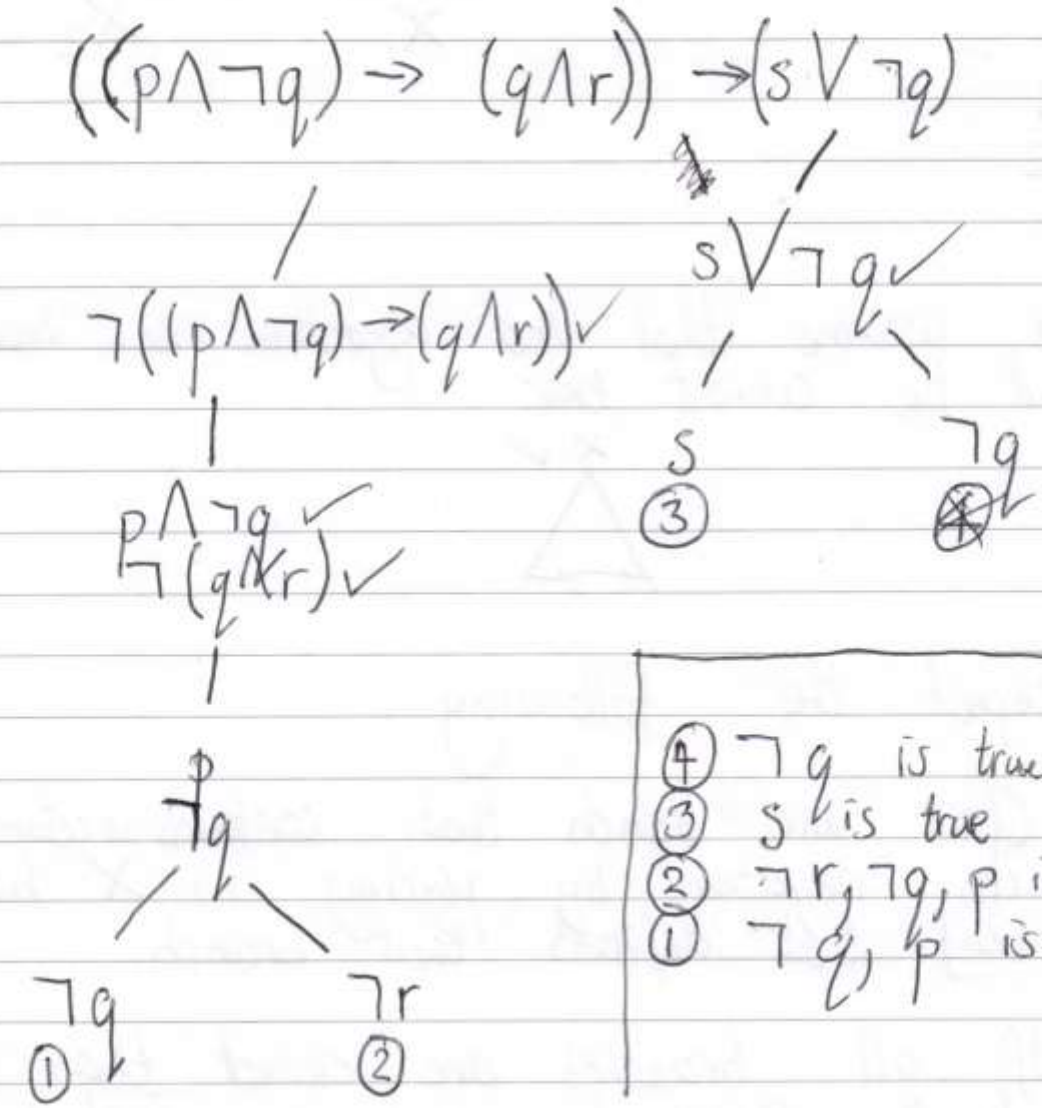
p	q	$p \vee q$	$p \wedge q$	$\neg[(p \vee q) \rightarrow (p \wedge q)]$
T	T	T	T	F
T	F	T	F	T
F	T	T	F	T
F	F	F	F	F



This is truth iff (2) is true or (3) is true.

Example Find all assignments that make the following WFF true.

$$((p \wedge \neg q) \rightarrow (q \wedge r)) \rightarrow (s \vee \neg q)$$



④ ¬q is true
③ s is true
② ¬r, ¬q, p is true
① ¬q, p is true

Lecture 18

Truth tree method

input wff x

Initialisation place x at the root of the tree (top). Apply α or β rule depending on its style and place a tick (✓) against x to show it has been used.

α rule

X ✓

|

X_1

X_2

β rule

X ✓

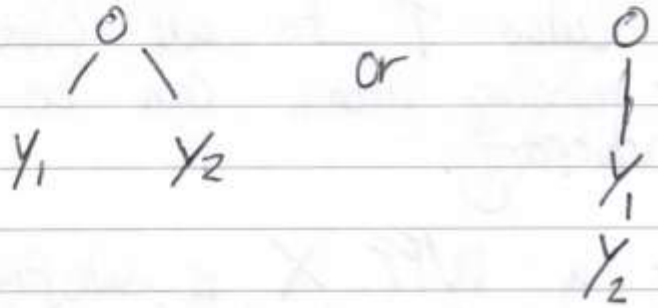
X_1 X_2

We assume that the algorithm has been running and the current tree:

Repeat the following

- Close any branch that contains an atom and its negation by placing an X beneath the leaf that defines that branch.
- If all branches are closed then the algorithm stops, the tree is finished and closed.
- If not all branches are closed but the tree contains only literals or used WFFs, stop. The tree is finished and open.
- If the tree is not finished, choose any unused wff, Y, which is not a literal. Now do the following.

For each open branch that contains Y, append to the leaf that defines that branch either as



as appropriate.

Output

If the tree is finished and closed then X is not satisfiable (ie. $X \not\models$)

If the tree is finished and open then X is satisfiable (ie $\models X$)

[We may find all truth assignments making X the following: for each branch in the finished tree assign the value T to all literals that appear (this makes X true). Unmentioned atoms can be assigned arbitary values]

We may find all truth assignments making X true as follows:

For each branch in the finished tree, for X assign the value T to all literals on that branch - missing atoms can be assigned truth values arbitrarily.

1) Deciding whether a WFF X is satisfiable or not: Construct the truth tree for X if when it is open then X is satisfiable else X is a contradiction and so not satisfiable.

2) We may use truth trees to rewrite a WFF in DNF.

Example $A = ((p \wedge \neg q) \rightarrow (q \wedge r)) \rightarrow (s \vee \neg q)$

① $p, \neg q$
or
② $p, \neg q, \neg r$
or
③ s
or
④ $\neg q$

} all true

$\therefore A \equiv (p \wedge \neg q) \vee (p \wedge \neg q \wedge$
$\vee s \vee \neg q$

The two applications that cause problems (but not this time)

3) Deciding whether $\models X$

4) $A_1, A_2, \ldots, A_n \models B$

:-3) To use truth trees to prove that $\models X$ (tautology) show that the truth tree is $\boxed{\neg X}$ is closed. This means that $\neg X$ is a contradiction. Therefore $\neg\neg X$ is a tautology. But $\neg\neg X \equiv X$. Therefore X is a tautology.

This shows that

$$\neg X \models \Rightarrow \models X$$

:-4) To check $A_1, \ldots, A_n \models B$

Using truth ~~tables~~ trees to decide if $A_1, \ldots, A_n \models B$ is a valid argument iff

$$\models (A_1 \wedge \ldots \wedge A_n) \rightarrow B$$

We start the truth tree with

$$\neg[(A_1 \wedge \ldots \wedge A_n) \rightarrow B]$$

$$\neg[(A_1 \wedge \ldots \wedge A_n) \rightarrow B] \equiv \neg(X \rightarrow B)$$

$\equiv \neg(\neg X \vee B)$

A_1
$\vdots$
A_n $\equiv \neg\neg X \wedge \neg B$

$\equiv X \wedge \neg B$

$\neg B$ $\equiv A_1 \wedge \dots \wedge A_n \wedge \neg B$

Example Show that the following is valid

$p \to q,\ q \to r \models p \to r$

$p \to q$ ✓
$q \to r$ ✓
$\neg(p \to r)$ ✓
|
p
$\neg r$
/ \
q
|
$\neg p$ $\neg q$
×

Lecture 20

Boolean algebras

$\{\neg, \wedge, \vee\}$ is an adequate set of connectives.

$\underline{0}$ means any contradiction

$\underline{1}$ means any tautology (always True)

$\vee$ (or) $\rightarrow +$

$\wedge$ (and) $\rightarrow \cdot$

$\neg$ (not) $\rightarrow \bar{x}$ - place a line above

$\underline{1} \rightarrow 1$

$\underline{0} \rightarrow 0$

$(B, +, \cdot, \bar{x}, 0, 1)$

($\bar{x}$: unary operator; $+, \cdot$: Binary operator; $0, 1$: constants)

Example X set (finite)

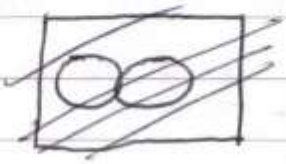
$P(X)$ = set of all subsets of X

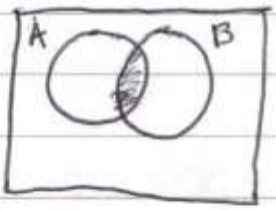
$X = \{a, b, c\}$ $P(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}$

$A \subseteq X$ $\{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$

Boolean operations on $P(X)$

$A, B \subseteq X$





X ← venn diagrams

$A \cap B = \{x \in X : (x \in A) \wedge (x \in B)\}$

intersection

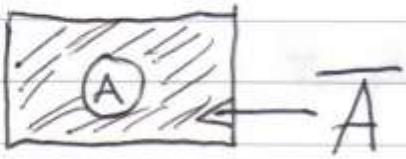
$A \cup B = \{x \in X : (x \in A) \vee (x \in B)\}$

union

$\overline{A} = \{x \in X : \neg(x \in A)\}$

$x \notin A$

complementation



Claim

This is Boolean algebra

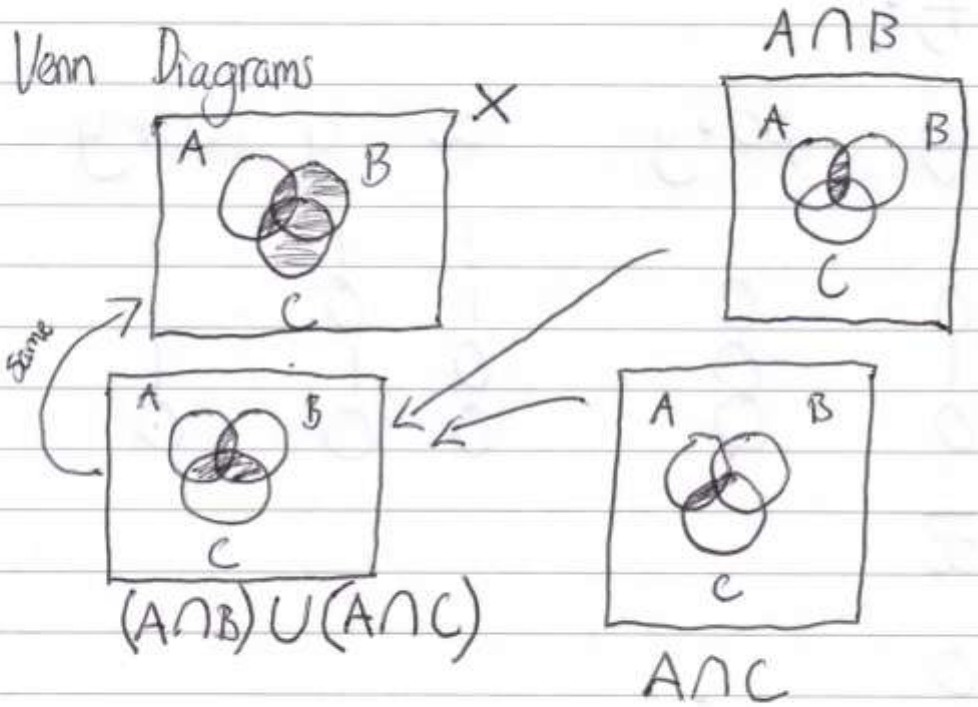
$(P(X), \cup, \cap, \overline{\ }, \emptyset, X)$

Check $x \cdot (y+z) = x \cdot y + x \cdot z$

$A \cap (B \cup C) = (A \cap B) \vee (A \cap \frac{1}{2} C)$

Method 1

Venn Diagrams



Method 2
Proof by Propositional Logic equivelants

$a = x \in A$ $\quad a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

$b = x \in B$ $\quad$ is true iff $\quad$ is true iff

$c = x \in C$ $\quad x \in A \cap (B \cup C)$ $\quad x \in (A \cap B) \cup (A \cap C)$

Key Boolean algebra in circuit design

$$\dot{B} = (\{0,1\}, +, \cdot, \bar{}, 0, 1)$$

Define $+, \cdot, \bar{}$

x	y	$x \cdot y$
1	1	1
1	0	0
0	1	0
0	0	0

x	y	$x+y$
1	1	1
1	0	1
0	1	1
0	0	0

x	$\bar{x}$
1	0
0	1

Lecture 22 - Circuit design

$\mathbb{B} = \{0, 1\}, \cdot, +, -$

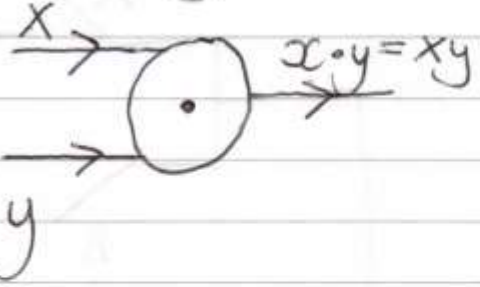
$\cdot = \wedge$

$+ = \vee$

$- = \neg$

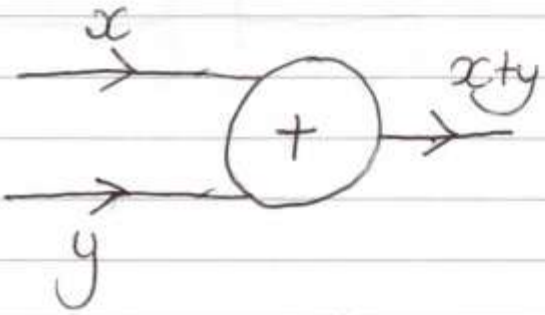
x	y	$x \cdot y$
T	T	1
T	F	0
F	T	0
F	F	0

And gate

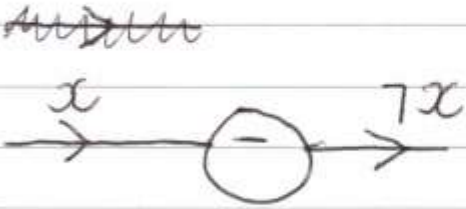


x	y	x+y
1	1	1
1	0	1
0	1	1
0	0	0

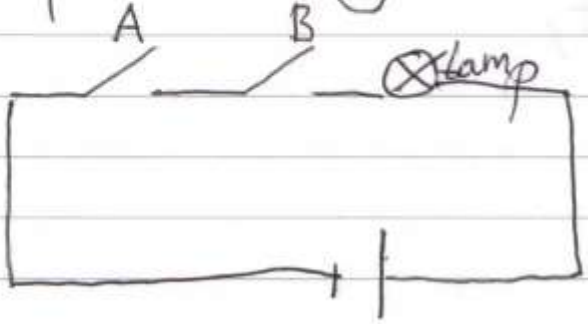
Or gate



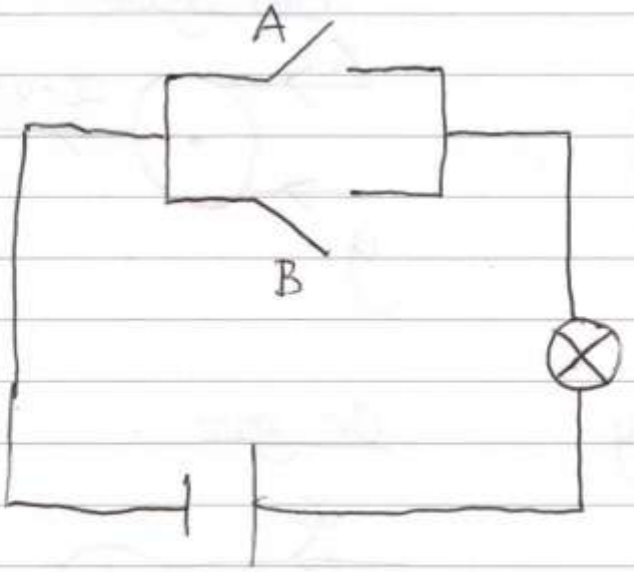
x	y(¬x)
0	1
1	0
0	1
1	0



Examples (motivating)

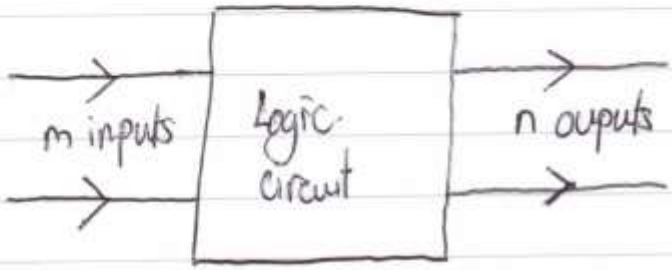


and circuit



Combinatinal circuits

- No internal memory

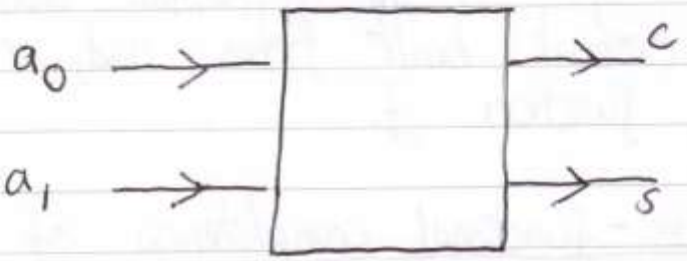


- Modelled by a function.

$$\{0,1\}^m \xrightarrow{f} \{0,1\}^n$$

$$x \in \{0,1\}$$

Example



$$[f: \{0,1\}^2 \rightarrow \{0,1\}^2]$$

inputs			
a_1	a_0	c	s
1	1	1	0
1	0	0	1
0	1	0	1
0	0	0	0

To build our circuit; we handle each output separately.

a_1	a_2	s
1	1	0
1	0	1
0	1	1
0	0	0

In general, we need only construct circuits for functions $f: \{0,1\}^m \rightarrow \{0,1\}$

Our goal is an algorithm that will construct a circuit built from and, or and not gates for any function f.

Theorem - functional completeness of $\mathbb{B}$

- Every function can be constructed from and, or and not gates. The method is the same for DNF BVI using Boolean

Example s

a_1	a_0	s
1	1	0
1	0	1
0	1	1
0	0	0

$a_1 \bar{a}_0$

$\bar{a}_1 a_0$

So the previous example has the function

$a_1 \overline{a_0} + \overline{a_1} a_0$

Example c

a_1	a_0	c
1	1	1
1	0	0
0	1	0
0	0	0

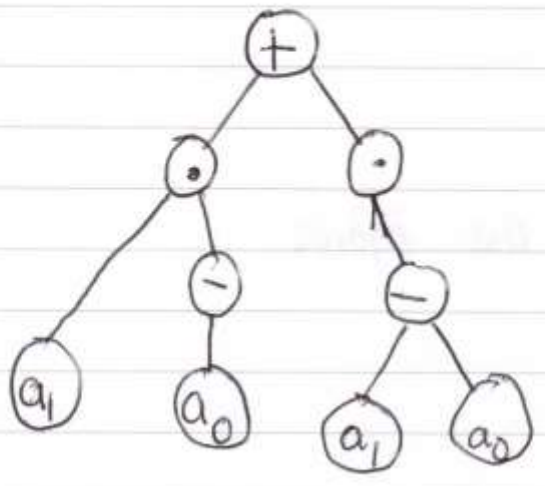
$c = a_1 [x] a_0$

Now convert into circuits

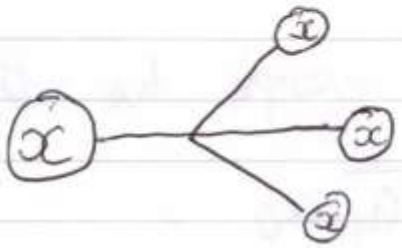
$S = a_1 \overline{a_0} + \overline{a_1} a_0$

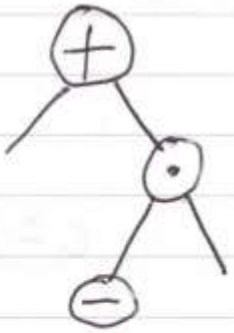
Parse tree

$(a_1 \wedge \neg a_0) \vee (\neg a_1 \wedge a_0)$



Fan out





The role of transistors

We want to show that and, or and not gates can be built from transistors.

x	y	x * y
1	1	0
1	0	1
0	1	0
0	0	0

* is called "stellar"

In fact, we have seen this before:

$$x * y = \overline{x \to y}$$
$$= \overline{(\bar{x} + y)}$$
$$= x \cdot \bar{y}$$

Observe

$$1 * x = 1 \cdot \overline{x} = \overline{x}$$

$$(1 * (1 * x)) * (1 * y)) = (1 * \overline{x}) * \overline{y}$$

$$= \overline{\overline{x}} * \overline{y}$$
$$= x * \overline{y}$$
$$= x \cdot \overline{y}$$
$$= x \cdot y$$

It follows that as long as we have access to an input line that is always 1, we have the following:

Theorem

Every combinational circuit can be constructed from transistors.

Lecture 23

Venn diagram

$$A \vee (B \wedge C) \vee (\overline{A} \wedge B) \vee (A \wedge \overline{B} \wedge C) = A \vee B$$

$$a + b \cdot c + \overline{a} \cdot b + a \cdot \overline{b} \cdot c = a + b$$

$$p \wedge p \equiv p \qquad p^2 = p \cdot p = p$$

Ex 7 - Homework

In any Boolean algebra, we always have $b^2 = b \cdot b = b$

Proof

$b = b1$ (by axiom B6)

$= b(b + \bar{b})$ (by axiom B9)

$= bb + b\bar{b}$ (by axiom B7)

$= bb + 0$ (by axiom B10)

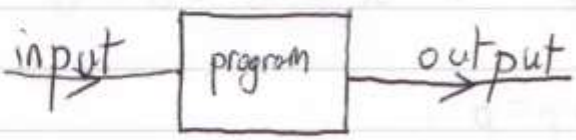
$= bb$ (by B3)

P = NP

(SAT) Satisfiability problem The following decision problem:

input: WFF in CNF
output: "Yes" if satisfiable or "No" if not

Fast algorithm



$n \longmapsto f(n)$ complexity function

n is the size of input

$f(n)$ = Worst "time" to compute output for inputs of length N.

Examples

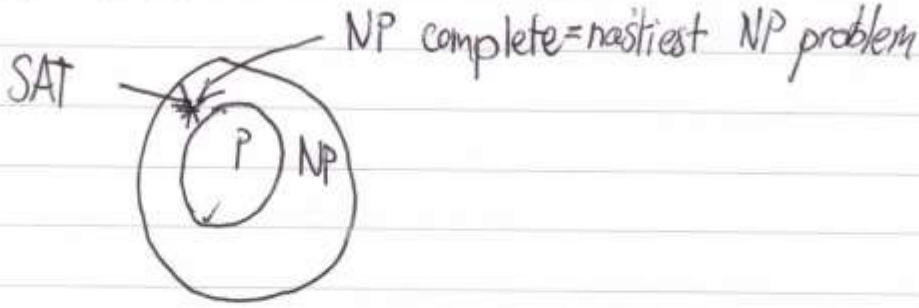
$n \longmapsto 10n$
$n \longmapsto 15 \cdot 2n^2$
$n \longmapsto 25n^3$
} all good "polynomial time"

Truth tables $n \longmapsto 2^n$ — bad "exponential time"
"goes to"

P is the class of all problems that can be solved in polynomial time.

NP is the class of all problems where the solution can be checked in polynomial time.

So how does P=NP?



SAT is NP-complete

① $(\forall x)\ (H(x) \rightarrow M(x))$

② $H(\text{Socrates})$

③ $\therefore M(\text{Socrates})$

$\forall x$ is "for all x"
$\exists y$ is "there exists a y"

Need to prove this is a valid argument.

Because ① this is true, it must be the case that $H(a) \rightarrow M(a)$ for any named object a.

$\therefore$ $H(\text{Socrates}) \rightarrow M(\text{Socrates})$ is true

$p \rightarrow q$

$H(\text{Socrates}) = p$ is true

By PL, we can know that $M(\text{Socrates})$ is true.

Key idea

Arguments using quantifier in First Order Logic will convert into arguments in PL using names.

Tree rules

α-rules	β-rules
$X \wedge Y$ X Y	$X \vee Y$ X | Y
$\neg(X \vee Y)$ $\neg X$ $\neg Y$	$\neg(X \wedge Y)$ $\neg X$ | $\neg Y$
$\neg(X \rightarrow Y)$ X $\neg Y$	$X \rightarrow Y$ $\neg X$ | Y
$\neg\neg X$ X	$X \leftrightarrow Y$ X, Y | $\neg X$, $\neg Y$
You will have to learn these	$\neg(X \leftrightarrow Y)$ X, $\neg Y$ | $\neg X$, Y