Mathematics for Computer Science

Logic and Discrete Structures

Daniel Paulusma

Department of Computing Science

Fundamentals of Propositional Logic

Durham University

The rudiments of propositional logic

- Propositional logic
  - the most fundamental logic, lying at the heart of many other logics
  - formalises day-to-day, common-sense reasoning.
- Key to propositional logic are propositions
  - declarative sentences that can be either *true* or *false* (but not both).
- Propositions are represented by propositional variables (Boolean variables, atoms)
  - usually letters such as x, Y, a, … or subscripted letters such as x_2, Y_0, a_1, …
  - which can take a truth value T (*true*) or F (*false*).
- Syntax
  - new propositions called formulae or Boolean formulae or propositional formulae or compound propositions are formed from propositional variables and formulae by repeated use of the logical operators

$\wedge$	conjunction (and)
$\vee$	disjunction (or)
$\neg$	negation (not)
$\Rightarrow$	implies
$\Leftrightarrow$	if and only if (iff).

Some formulae

- Construction
  - the operators $\wedge$, $\vee$, $\Rightarrow$, and $\Leftrightarrow$ take two propositional formulae φ and ψ and yield a new one
    - $\varphi \wedge \psi$ $\varphi \vee \psi$ $\varphi \Rightarrow \psi$ $\varphi \Leftrightarrow \psi$
  - the operator $\neg$ takes one propositional formula φ and yields a new one
    - $\neg\varphi$.
- Use of parentheses
  - $(\varphi \wedge \psi) \vee \chi$ means first build $\varphi \wedge \psi$ and then build $(\varphi \wedge \psi) \vee \chi$
  - $\varphi \wedge (\psi \vee \chi)$ means first build $(\psi \vee \chi)$ and then build $\varphi \wedge (\psi \vee \chi)$.
- Some typical well-formed formulae (where a, b, c and d are propositional variables)
  - $\neg((\neg b \wedge a) \Rightarrow (c \vee \neg d))$
  - $((a \wedge \neg a) \vee ((b \vee c) \vee d)) \Leftrightarrow d$
  - $(((a \Rightarrow b) \Rightarrow c) \Rightarrow d)$.

Semantics of propositional logic

- Semantics: all propositional variables take the value T (*true*) or F (*false*)
  - the value of a formula under some truth assignment is ascertained by using the truth tables for the above logical connectives.
- The truth tables for our logical connectives are as follows

p	q	$p \wedge q$	$p \vee q$	$\neg p$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	T	T	F	T	T
T	F	F	T	F	F	F
F	T	F	T	T	T	F
F	F	F	F	T	T	T

definitions

- In order to build the truth table of a formula
  - we decompose the formula into sub-formulae, e.g.,

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

Semantics of propositional logic

- Semantics: all propositional variables take the value T (*true*) or F (*false*)
  - the value of a formula under some truth assignment is ascertained by using the truth tables for the above logical connectives.
- The truth tables for our logical connectives are as follows

p	q	$p \wedge q$	$p \vee q$	$\neg p$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	T	T	F	T	T
T	F	F	T	F	F	F
F	T	F	T	T	T	F
F	F	F	F	T	T	T

definitions

- In order to build the truth table of a formula
  - we decompose the formula into sub-formulae, e.g.,

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

the parse tree can be viewed as a "circuit" and the truth values as the "inputs"

this is what the formula evaluates to

Some basic notation

- If we have a propositional formula $\varphi(x_1, x_2, \ldots, x_n)$ then
  - we call an assignment f of either T or F to each $x_1, x_2, \ldots, x_n$, i.e., a function

    $$f : \{x_1, x_2, \ldots, x_n\} \rightarrow \{T, F\}$$

  a truth assignment (interpretation, valuation) for φ

- We say that φ evaluates to T (resp. F) under f
  - if the row of the truth table for φ corresponding to f evaluates to T (resp. F).

- If f evaluates φ to T then
  - f satisfies φ or is a satisfying truth assignment of φ or a model of φ.

- If φ evaluates to T for every f then φ is a tautology.
- If φ evaluates to F for every f then φ is a contradiction.

- A literal is either a propositional variable, say x, or the negation of a propositional variable, say $\neg x$.

Logical equivalence

- Steps in a mathematical proof are often just the replacement of one statement by another (equivalent) statement (which says the same thing), e.g.

  "*If I don't explain this clearly then the students won't understand.*"

  is the same thing "*Either I explain this clearly or the students won't understand*".
- To see this, denote the sub-statement "*I don't explain this clearly*" as X and
  denote the sub-statement "*the students won't understand*" as Y.
- The former statement is thus $X \Rightarrow Y$ and the latter

X	Y	$X \Rightarrow Y$	$\neg X \vee Y$
T	T	T **T** T	F T **T** T
T	F	T **F** F	F T **F** F
F	T	F **T** T	T F **T** T
F	F	F **T** F	T F **T** F

- We say that two propositional formulae are (logically) equivalent if they have identical truth tables
  - if φ and ψ are equivalent then we write $\varphi \equiv \psi$

A spot of practice

- The exclusive-OR, written $X \oplus Y$, is *true* iff exactly one of X and Y is *true*.
- Prove that $X \oplus Y$ is logically equivalent to both $(X \wedge \neg Y) \vee (\neg X \wedge Y)$ and $\neg(X \Leftrightarrow Y)$.

X	Y	$X \oplus Y$	$(X \wedge \neg Y) \vee (\neg X \wedge Y)$	$\neg(X \Leftrightarrow Y)$
T	T	T **F** T	T F F T **F** F T F T	**F** T T T
T	F	T **T** F	T T T F **T** F T F F	**T** T F F
F	T	F **T** T	F F F T **T** T F T T	**T** F F T
F	F	F **F** F	F F T F **F** T F F F	**F** F T F

De Morgan's Laws

- There are two extremely useful logical equivalences known as De Morgan's Laws.
- De Morgan's Laws are

$$\neg(X \wedge Y) \equiv \neg X \vee \neg Y$$
$$\neg(X \vee Y) \equiv \neg X \wedge \neg Y$$

- These formulae are indeed equivalences

X	Y	$\neg(X \wedge Y)$	$\neg X \vee \neg Y$	$\neg(X \vee Y)$	$\neg X \wedge \neg Y$
T	T	**F** T T T	FT **F** FT	**F** T T T	FT **F** FT
T	F	**T** T F F	FT **T** TF	**F** T T F	FT **F** TF
F	T	**T** F F T	TF **T** FT	**F** F T T	TF **F** FT
F	F	**T** F F F	TF **T** TF	**T** F F F	TF **T** TF

- De Morgan's Laws can be applied not just to variables but to *formulae* φ and ψ.
- De Morgan's Laws are often used to simplify formulae with regard to negations.

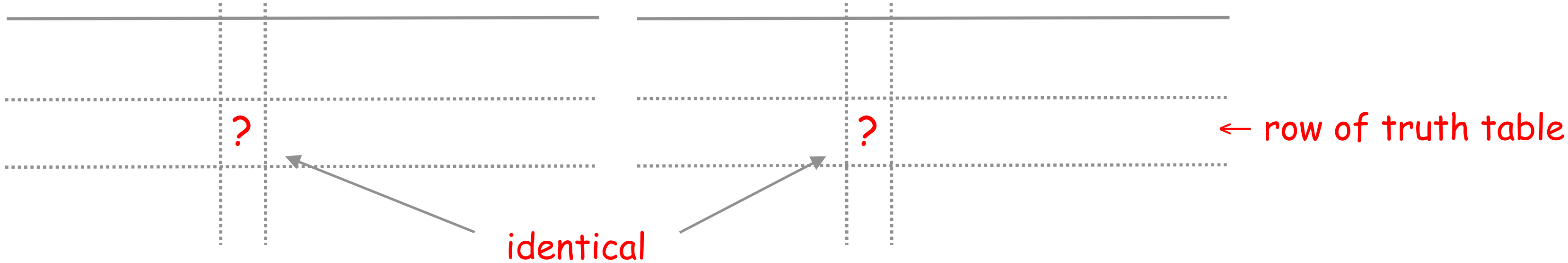
Applying De Morgan's Laws

- In fact, not only can De Morgan's Laws be applied to formulae
  - they can be applied to *sub-formulae* within a formula.
- Consider the propositional formula $\neg(p \vee \neg(q \wedge \neg p)) \wedge \neg(p \Rightarrow q)$
  - take the sub-formula $\neg(q \wedge \neg p)$.
- By De Morgan's Laws

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

- So

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



- Indeed, we can always replace *any sub-formula* of some propositional formula
  - with an *equivalent formula* without affecting the truth (table) of the original.

A spot of practice

- Consider $\neg(p \vee \neg(q \wedge \neg p)) \wedge \neg(p \Rightarrow q)$.
- Can we manipulate it so as to simplify it?

$\neg(p \vee \neg(q \wedge \neg p)) \wedge \neg(p \Rightarrow q)$	apply De Morgan's Laws
$\equiv \neg(p \vee (\neg q \vee \neg\neg p)) \wedge \neg(p \Rightarrow q)$	remove double-negation
$\equiv \neg(p \vee (\neg q \vee p)) \wedge \neg(p \Rightarrow q)$	apply De Morgan's Laws
$\equiv (\neg p \wedge \neg(\neg q \vee p)) \wedge \neg(p \Rightarrow q)$	apply De Morgan's Laws
$\equiv (\neg p \wedge (\neg\neg q \wedge \neg p)) \wedge \neg(p \Rightarrow q)$	remove double-negation
$\equiv (\neg p \wedge (q \wedge \neg p)) \wedge \neg(p \Rightarrow q)$	$\Rightarrow$ using $\vee$, $\neg$
$\equiv (\neg p \wedge (q \wedge \neg p)) \wedge \neg(\neg p \vee q)$	apply De Morgan's Laws
$\equiv (\neg p \wedge (q \wedge \neg p)) \wedge (\neg\neg p \wedge \neg q)$	remove double-negation
$\equiv (\neg p \wedge (q \wedge \neg p)) \wedge (p \wedge \neg q)$	associativity of $\wedge$
$\equiv (\neg p \wedge q \wedge \neg p) \wedge (p \wedge \neg q)$	associativity of $\wedge$
$\equiv \neg p \wedge q \wedge \neg p \wedge p \wedge \neg q$	commutativity of $\wedge$
$\equiv \neg p \wedge \neg p \wedge p \wedge q \wedge \neg q$	$X \wedge \neg X \equiv \mathrm{F}$
$\equiv \neg p \wedge \mathrm{F} \wedge q \wedge \neg q$	$\mathrm{F} \wedge \varphi \equiv \mathrm{F}$
$\equiv \mathrm{F}$	

Generalised De Morgan's Laws

- We can actually generalise De Morgan's Laws so that
  - negations can be "pushed inside" conjunctions/disjunctions of *more* than two literals (or formulae).
- To do this
  - we apply De Morgan's Laws to sub-formulae of a formula.
- Consider $\neg(X \vee Y \vee Z)$
  - Rewrite this formula as $\neg(X \vee (Y \vee Z))$ and denote $Y \vee Z$ by φ.
  - Applying De Morgan's Laws to $\neg(X \vee \varphi)$ yields an equivalent formula $\neg X \wedge \neg\varphi$
    - i.e., the formula $\neg X \wedge \neg(Y \vee Z)$.
  - Applying De Morgan's Laws again yields the equivalent formula $\neg X \wedge \neg Y \wedge \neg Z$.

- Similar arguments yield the generalised De Morgan's Laws

$$\neg(X_1 \vee X_2 \vee \ldots \vee X_n) \equiv \neg X_1 \wedge \neg X_2 \wedge \ldots \wedge \neg X_n$$

$$\neg(X_1 \wedge X_2 \wedge \ldots \wedge X_n) \equiv \neg X_1 \vee \neg X_2 \vee \ldots \vee \neg X_n$$