Discrete Mathematics

Alexander Pasko, Evgenii Maltsev, Dmitry Popov

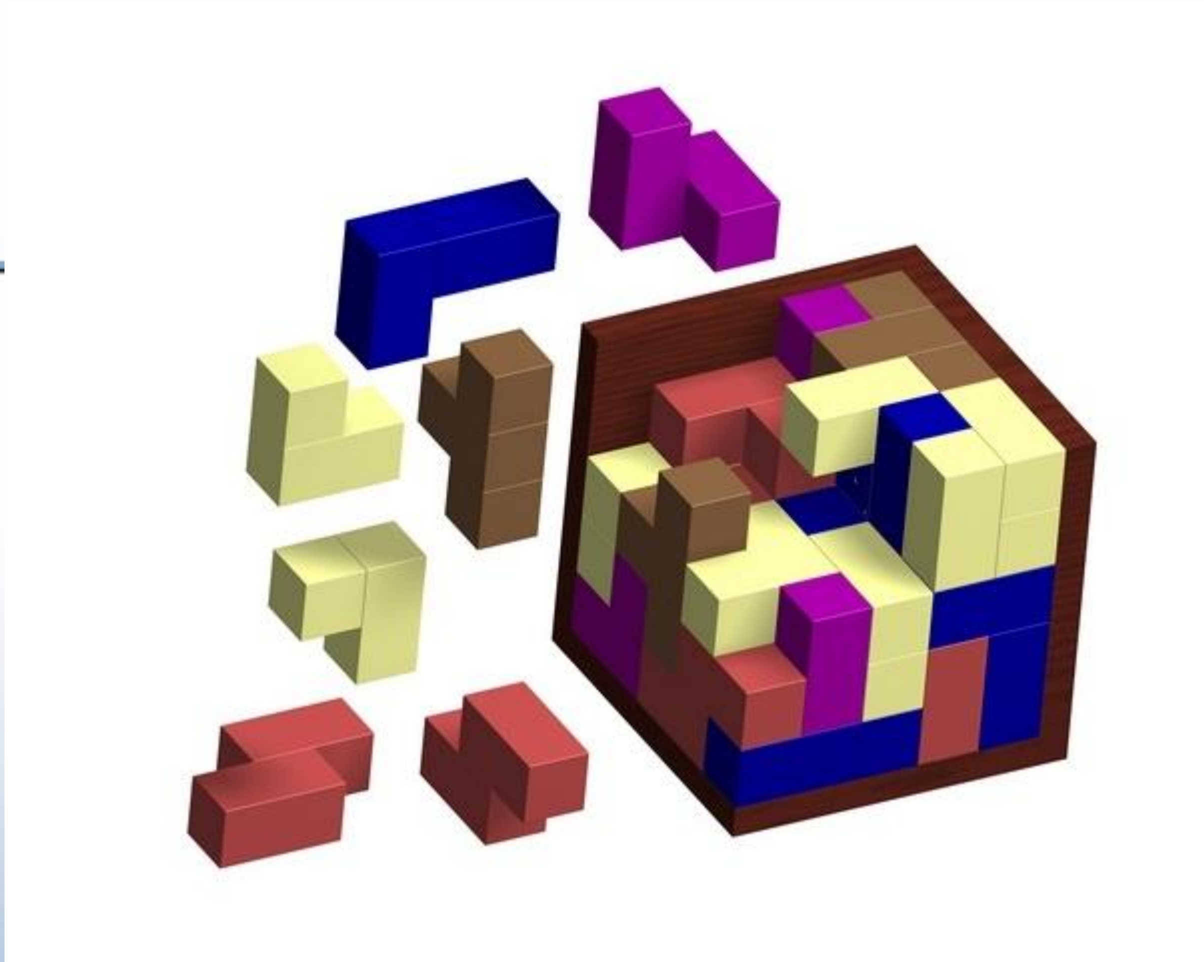
Unit materials

- Lecture notes
- Seminar handouts

are available at

http://gm.softalliance.net/

Advice: download and print lecture notes before the next lecture



http://www.craftsmanspace.com

Propositional Logic

Contents

Notion of Logic

The word *logic* is inherited from classical Greek λόγος logos; meaning word, thought, idea, argument, account, reason, or principle.

In common sense, logic is a tool for distinguishing between the true and the false (Averroes).

Logic is the study of the principles and criteria of valid inference and demonstration.

Inference means making a conclusion based solely on what one already knows.

Inference is the process of deriving logical conclusions from premises known or assumed to be true.

Example

Flipper is a dolphin	- premise
Every dolphin is a mammal	- premise
Flipper is a mammal	- conclusion

Branches of Logic

- Informal logic is the study of natural language arguments.
- Argument is a set of sentences known as the premises, and another sentence known as the conclusion in which it is assumed that the truth of the conclusion follows from the premises.

Branches of Logic

- Formal logic is the study of inference with purely abstract content not related to any particular thing or property.

- Mathematical logic is the study of formal features of logical inference: mathematical study of logic and the application of this study to other areas of mathematics.

Formal Logic

- Subject: definition of rules of proper inference (when correctly applied to true premises, lead to true conclusions).
- The first rules of formal logic were written by Aristotle in his Organon.

Aristotle
(384-322 BC)

Syllogisms

- Aristotle defined a number of syllogisms, correct three-part inferences, that can be used as building blocks for more complex reasoning.

Example: most famous syllogism

All men are mortal. Socrates is a man.

Therefore Socrates is mortal.

Two premises and conclusion here are true.

- Structure of a syllogism
- Major premise - a general statement
- Minor premise - a specific statement
- Conclusion - based on the two premises

Each part is a proposition in the form:
"All A are B," "Some A are B",
"No A are B" or "Some A are not B"

There are 24 types of logically distinct valid syllogisms.

Does the truth of the conclusion follow from truth of the premises?

Example: valid inference with false premises

- All apples are blue. (False)
- A banana is an apple. (False)
- Therefore, a banana is blue. (False)

For the conclusion to be necessarily true, the premises need to be true.

Example: invalid inference

- All A are B.
- C is a B.
- Therefore, C is an A.

This is invalid form of inference, because from true premises it can lead to a false conclusion:

- All apples are fruit. (True)
- Bananas are fruit. (True)
- Therefore, bananas are apples. (False)

Properties of inference:

- For the conclusion to be necessarily true, the premises need to be true.
- An inference can be valid even if the parts are false, and can be invalid even if the parts are true.
- A valid inference does not depend on the truth of the premises and conclusion, but on the rules of inference studied in formal logic.
- A valid form of inference with true premises will always have a true conclusion.

Mathematical Logic

- Logic is the basis for all mathematical reasoning. To be able to understand and construct our own correct mathematical arguments we must understand logic.
- Mathematical Logic studies formal features of logical inference using symbolic abstractions.
- Mathematical Logic is

  - a tool for working with compound statements built from simpler statements.

  - the foundation for expressing formal proofs in all branches of mathematics.

Mathematical Logic

- Mathematical Logic includes:

  - a formal language for expressing compound statements

  - a concise notation for writing them

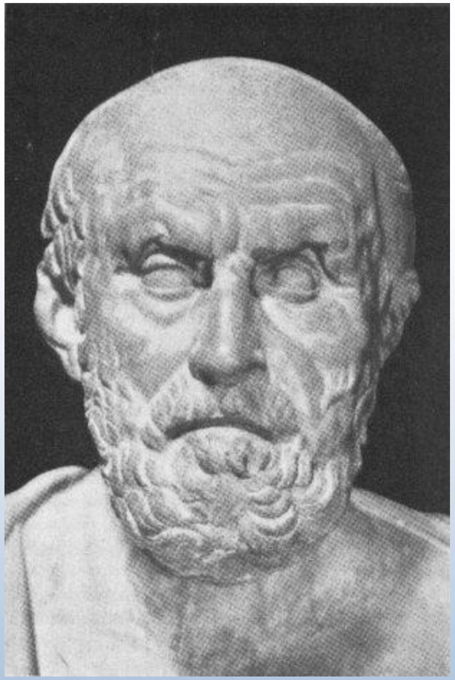
  - a methodology for objectively reasoning about their truth or falsity.

- Divided into two branches: *propositional logic* and *predicate logic*.

Propositional Logic

Aristotle developed a detailed system of logic and Chrysippus of Soli introduced propositional logic centered around logical operations.

It is the logic of compound statements built from simpler statements using logic operators (NOT, AND, OR).



Chrysippus of Soli
(ca. 281 B.C. – 205 B.C.)

Some applications in computer science:

- Design of digital electronic circuits
- Construction of computer programs
- Verification of the correctness of programs
- Queries to databases & search engines
- Constructive geometric modeling

Definition of a Proposition

- A proposition is:

❖ a *statement* (a declarative sentence)

  – with some definite meaning (not vague or ambiguous)

❖ having a truth value that is either true or false (under interpretation)

  – it is never both, neither, or somewhere "in between"

  – however, you might not know the actual truth value and the truth value might depend on the situation or context.

Propositions in Natural Language

<u>Propositions:</u>

- "It is raining."
- "London is the capital of China."
- 1 + 2 = 2

<u>**NOT** propositions:</u>

- "Who is there?" (interrogative: no truth value)
- "1 + 2" (term: no truth value)
- "kudliva bokra" (no definite meaning in known languages)
- 7 – z = 77 (neither true nor false, since variables are not assigned with their values)
- "Go to the town!" (imperative: no truth value)

Q: Determine whether the following statements are propositions or not and explain your answers:

a) The Earth is flat.

b) Is it raining?

c) Stop and give way to pedestrians.

e) 2 + 8 =10

d) $2x - 7 = 8$

Formally not a proposition because it is neither true nor false. Note that it can be turned into a proposition if we assign a value to the variable x.

Propositional Variables

- Propositional variables (statement variables, atoms) are simply letters which represent propositions: p, q, r, s, t *…*

  Correspond to simple (English) sentences. Examples:

  Proposition p is “I had salad for lunch“

  Proposition q is “Today is Monday”

- Proposition s is ….
- Proposition t is …

Truth Value

- Proposition is either true or false but not both
- Each proposition has a Truth Value.
- If a proposition is
  - true, we denote that by **T**
  - false, we denote that by **F**

T :≡ True; F :≡ False

“ :≡ ” means “ is defined as ”

❖ Proposition p is ”Sun is a planet”. It is ….

Contents

- Notions of logic
- Branches of logic
- Propositional logic
- Boolean operators
- Logic and computing
- Equivalence rules and derivations

Compound Propositions

Compound statements are built from simpler statements using logic operators or Boolean connectives.

These methods were discussed by the English mathematician George Boole in 1854 in his book “The Laws of Thought”.

George Boole (1815-1864)

Compound Propositions

- To create new propositions we may combine one, two or more propositions into complex (compound) propositions.
- Compound propositions are built up from atoms using operators such as NOT, AND, OR. Correspond to compound English sentences ("I had salad for lunch AND I had a steak for dinner.")

Operators / Connectives

- Operator or connective combines n operands (expressions) into a larger expression, *e.g.*, “+” in numeric expression.
- Unary operators take 1 operand (*e.g.*, -3) Binary operators take 2 operands (*e.g.*, 3×4)
- Propositional or Boolean operators (connectives) operate on propositions instead of numbers.

Some Boolean Operators

Formal Name	Name	Arity	Symbol
Negation operator	NOT	Unary	¬
Conjunction operator	AND	Binary	∧
Disjunction operator	OR	Binary	∨
Exclusive-OR operator	XOR	Binary	⊕

Negation Operator

- The unary negation operator “¬” (logical *NOT*)

transforms a proposition into its negation.

- The proposition ¬p is read "not p."
- The truth value of ¬p, is the opposite of the truth value of p.

Example: If p = “I have brown hair.”

then ¬p = “I do **not** have brown hair.”

Truth Table for Negation Operator

- A truth table displays the relationships between the truth values of propositions, when applying an operator to them.

Truth table for proposition and it’s negation:

Operand column

Result column

p	$\neg p$
T	F
F	T

Negation Exercise

Find the negation of the proposition

p="At least 10 inches of rain fell today in Poole."

and express this in simple English.

Solution: the negation is

¬p ="It is not the case that at least 10 inches of rain fell today in Poole."

This negation can be more simply expressed by

¬p = "Less than 10 inches of rain fell today in Poole."

Conjunction Operator

Binary conjunction operator "$\wedge$" (logical *AND*) combines two propositions to form their logical conjunction.

The conjunction $p \wedge q$ is true when both p and q are true and is false otherwise.

Example:

If p = "I had salad for lunch." and
q = "I had a steak for dinner.",
then $p \wedge q$ = "I had salad for lunch **and** I had a steak for dinner."

Conjunction Truth Table

- Note:
  a conjunction
  $p_1 \wedge p_2 \wedge \ldots \wedge p_n$
  of n propositions
  will have 2^n rows
  in its truth table.

Operand Columns		**Result Column**
p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

- Note: ¬ NOT and ∧ AND operations together are sufficient to express any Boolean truth table!

Conjunction Exercise

For the two given propositions

p="Today is Tuesday."

q="It is raining today."

when the conjunction is false?

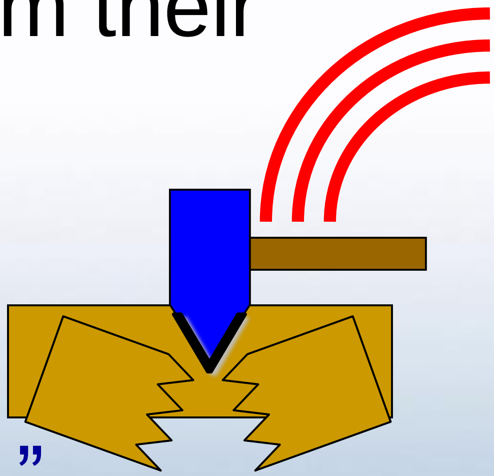
Solution: the conjunction

$p \wedge q$ ="Today is Tuesday and it is raining today."

is false on any day that is not a Tuesday and on Tuesdays when it does not rain.

Disjunction Operator

- Binary disjunction operator "∨" (logical *OR*) combines two propositions to form their logical disjunction.
- Example:
- *p*="My car has a bad engine."
- *q*="My car has a bad carburettor."
- *p*∨*q*="Either my car has a bad engine, **or** my car has a bad carburettor."



After the downward-pointing "axe" of "∨" splits the wood, you can take 1 piece OR the other, or both.

Disjunction Truth Table

- Note that $p \lor q$ means that p is true, or q is true, or both are true!
- This operation is also called inclusive or, because it includes the possibility that both p and q are true.
- Note: "¬" NOT and "∨" OR together are also universal.

p	q	$p \lor q$
F	F	F
F	T	**T**
T	F	**T**
T	T	T

Note difference from AND

Disjunction Exercise

For the two given propositions

p="Today is Tuesday."

q="It is raining today."

when the disjunction is false?

Solution: the disjunction

$p \vee q$ ="Today is Tuesday or it is raining today."

is only false on days that are not Tuesdays when it also does not rain.

Nested Propositional Expressions

- Use parentheses to group *sub-expressions*:
  "I just saw my old friend, and either he's grown or I've shrunk." $:\equiv f \wedge (g \vee s)$
  - $(f \wedge g) \vee s$ would mean something different
  - $f \wedge g \vee s$ would be ambiguous
- By convention, "$\neg$" NOT takes precedence over both "$\wedge$" and "$\vee$".
  - $\neg s \wedge f$ means $(\neg s) \wedge f$, **not** $\neg(s \wedge f)$

Simple Exercise

p = “It rained last night”,
q = “The sprinklers came on last night,”
r = “The lawn was wet this morning.”

Translate each of the following into English:

- $\neg p$ = “It didn’t rain last night.”
- $r \wedge \neg p$ = “The lawn was wet this morning, and it didn’t rain last night.”

❖ $r \wedge (p \vee q) =$

Exclusive-Or Operator

- Binary exclusive-or operator “$\oplus$” (logical *XOR*) combines two propositions to form their logical “exclusive or” (exjunction).

Example:

- p = “I will earn an A in this course,”
- q = “I will drop this course,”
- $p \oplus q$ = “I will either earn an A in this course, or I will drop it (but not both!) ”

Exclusive-Or Truth Table

- Note that $p \oplus q$ means that p is true, or q is true, but not both!
- This operation is called exclusive or, because it excludes the possibility that both p and q are true.

p	q	$p \oplus q$
F	F	F
F	T	T
T	F	T
T	T	**F**

Note difference from OR.

Exclusive-Or Exercise

For the two given propositions

p="Today is Tuesday."

q="It is raining today."

when the exclusive-or is false?

Solution: the exclusive-or

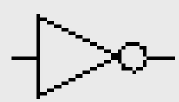
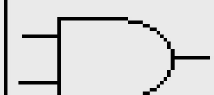
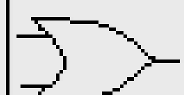
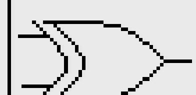
$p \oplus q$ ="Today is either Tuesday or it is raining today, but not both."

is false on rainy Tuesdays and on any other weekday, when it does not rain.

Boolean Operators Summary

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$
F	F	T	F	F	F
F	T	T	F	T	T
T	F	F	F	T	T
T	T	F	T	T	F

Some Alternative Notations

Name:	not	and	or	xor
Propositional logic:	$\neg$	$\wedge$	$\vee$	$\oplus$
Boolean algebra:	$\overline{p}$	pq	$+$	$\oplus$
C/C++/Java (wordwise):	!	&&	||	!=
C/C++/Java (bitwise):	~	&	|	^
Logic gates:				

Implication

- Implication (conditional statement) $p \rightarrow q$ is false when p is true and q is false, and true otherwise.
- p is hypothesis (or antecedent or premise) and q is conclusion (or consequence).
- Note: The implication is false only when P is true and Q is false!

Implication

- Equivalent forms:
  - If p, then q
  - p implies q
  - If p, q
  - p only if q
  - p is a sufficient condition for q
  - q if p
  - q whenever p
  - q is a necessary condition for p

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

Implication Example

Professor's promise:

$p \rightarrow q$: "If you get 100% on the final, then you will get an A."

- If you manage to get a 100% on the final, then you would expect to receive an A: promise is kept
- If you do not get 100% you may or may not receive an A depending on other factors: promise can be kept
- However, if you do get 100%, but the professor does not give you an A, you will feel cheated: false promise

Conditional Statement in Programming

Let a program statement take FALSE value, if there is an error (division by zero, etc.).

Conditional statement $A \rightarrow B$ in the form

if(A) then B;

triggers an error event (FALSE value), only if

the condition A is TRUE and

B is FALSE,

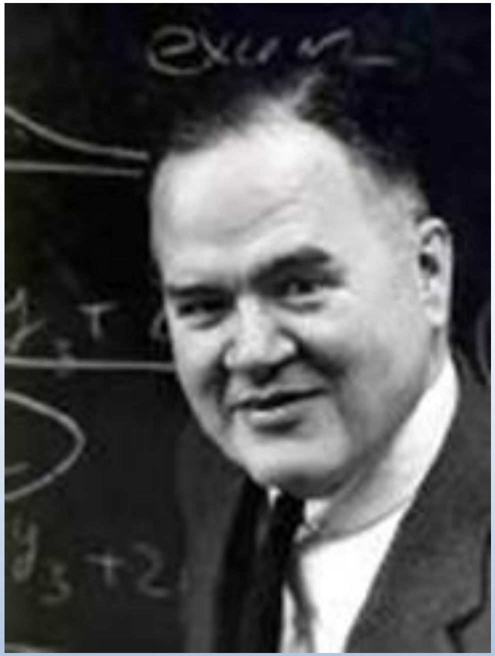
meaning an error happens when executing B.

Contents

- Notions of logic
- Branches of logic
- Propositional logic
- Boolean operators
- Logic and computing
- Equivalence rules and derivations

BIT – BInary digiT

- *Bit* is a binary (base 2) digit: 0 or 1.
- Bits may be used to represent truth values.
- By convention:
  0 represents "False";
  1 represents "True".

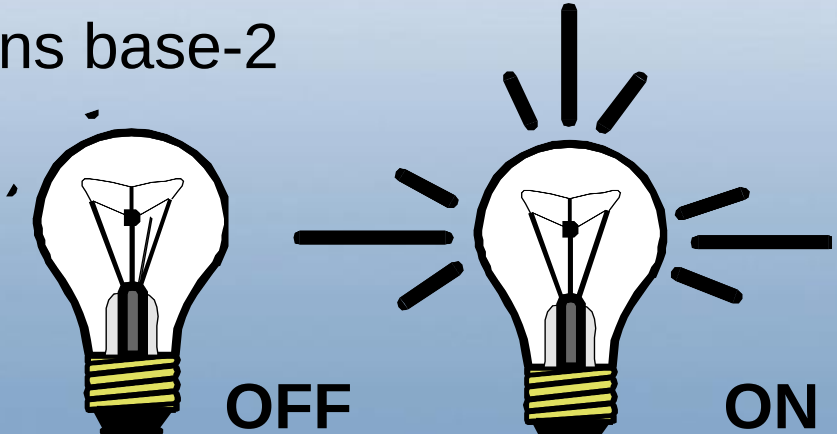


John Tukey
(1915-2000)

This terminology was introduced by statistician John Tukey in 1946.

BIT – BInary digiT

- Computers are made of a series of switches
- Each switch has two states: ON or OFF
- **Bit** (Binary Digit) = Basic unit of information, representing one of two discrete states.
  The smallest unit of information within the computer.
- The only thing a computer understands.
- Bit has one of two values: **1** (ON) or **0** (OFF)
- Binary means base-2



Tables for Bit Operations

- Boolean algebra is like ordinary algebra except that variables stand for bits; + means “or”; multiplication means “and”.
- A variable is called Boolean variable if its value is either true or false.

NOT	0	1
	1	0

OR	0	1
0	0	1
1	1	1

AND	0	1
0	0	0
1	0	1

XOR	0	1
0	0	1
1	1	0

Bit Strings

- Bit string of length n is an ordered sequence (series, tuple) of $n \geq 0$ bits.
- By convention, bit strings are (sometimes) written left to right:
- the "first" bit of the bit string of length ten "1001101010" is 1.
- When a bit string represents a base-2 number, by convention, the first (leftmost) bit is the *most significant* bit.
- Example: $1101_2=8+4+1=13$.

Bitwise Operations

- Boolean operations can be extended to operate on bit strings as well as single bits.

❖ E.g.:

01 1011 0110
<u>11 0001 1101</u>
11 1011 1111 Bit-wise OR
01 0001 0100 Bit-wise AND
10 1010 1011 Bit-wise XOR

Contents

- Notions of logic
- Branches of logic
- Propositional logic
- Boolean operators
- Logic and computing
- Equivalence rules and derivations

Tautologies

Tautology is a compound proposition that is true no matter what the truth values of its atomic propositions are! When every row of the truth table gives T.

Example: $p \vee \neg p$

- What is its truth table?

Contradiction and Contingency

Contradiction is a compound proposition that is false no matter what!

When every row of the truth table gives F

Example: $p \wedge \neg p$

- Truth table?

A proposition that is neither a tautology nor a contradiction is called a contingency.

Logical Equivalence

- Compound proposition p is logically equivalent to compound proposition q, written $p \Leftrightarrow q$, if p and q contain the same truth values in <u>all</u> rows of their truth tables
- They express the same truth function (the same function from values for atoms to values for the whole formula):
  “p if and only if q” or
  “if p then q and conversely”

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

Proving Equivalence via Truth Tables

Example: prove that $p \lor q \Leftrightarrow \neg(\neg p \land \neg q)$.

p	q	$p \lor q$	$\neg p$	$\neg q$	$\neg p \land \neg q$	$\neg(\neg p \land \neg q)$
F	F	F	T	T	T	F
F	T	T	T	F	F	T
T	F	T	F	T	F	T
T	T	T	F	F	F	T

Equivalence Laws

- Equivalence rules provide a pattern or template that can be used to match all or part of a much more complicated proposition and to find an equivalence for it.
- These rules are similar to the arithmetic identities you may have learnt in algebra, but for propositional equivalences instead.

Equivalence Laws

- *Identity*: $p \wedge \mathbf{T} \Leftrightarrow p \qquad p \vee \mathbf{F} \Leftrightarrow p$
- *Domination*: $p \vee \mathbf{T} \Leftrightarrow \mathbf{T} \qquad p \wedge \mathbf{F} \Leftrightarrow \mathbf{F}$
- *Idempotent*: $p \vee p \Leftrightarrow p \qquad p \wedge p \Leftrightarrow p$
- *Double negation:* $\neg\neg p \Leftrightarrow p$
- *Commutative:* $p \vee q \Leftrightarrow q \vee p \qquad p \wedge q \Leftrightarrow q \wedge p$
- *Associative:* $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

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

Equivalence Laws

Augustus De Morgan
(1806-1871)

- *Distributive*:

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

- *De Morgan's laws*:

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

Derivation

Derivation is a finite sequence of propositions each of which follows from the preceding one in the sequence following equivalence laws:

$$p \Leftrightarrow q \Leftrightarrow r \Leftrightarrow \ldots$$

Example:

$(w \vee x) \vee (w \vee z) \Leftrightarrow$ associative

$((w \vee x) \vee w) \vee z \Leftrightarrow$ commutative

$((x \vee w) \vee w) \vee z \Leftrightarrow$ associative

$((x \vee (w \vee w)) \vee z) \Leftrightarrow$ idempotent

$(x \vee w) \vee z$

Review: Propositional Logic

- Atomic propositions: p, q, r, …
- Boolean operators: $\neg \wedge \vee \oplus$ …
- Compound propositions: $s :\equiv (p \wedge \neg q) \vee r$
- Equivalences: $p \wedge \neg q \Leftrightarrow \neg(p \rightarrow q)$
- Proving equivalences using:
  - Truth tables.
  - Symbolic derivations: $p \Leftrightarrow q \Leftrightarrow r$ …

Questions?