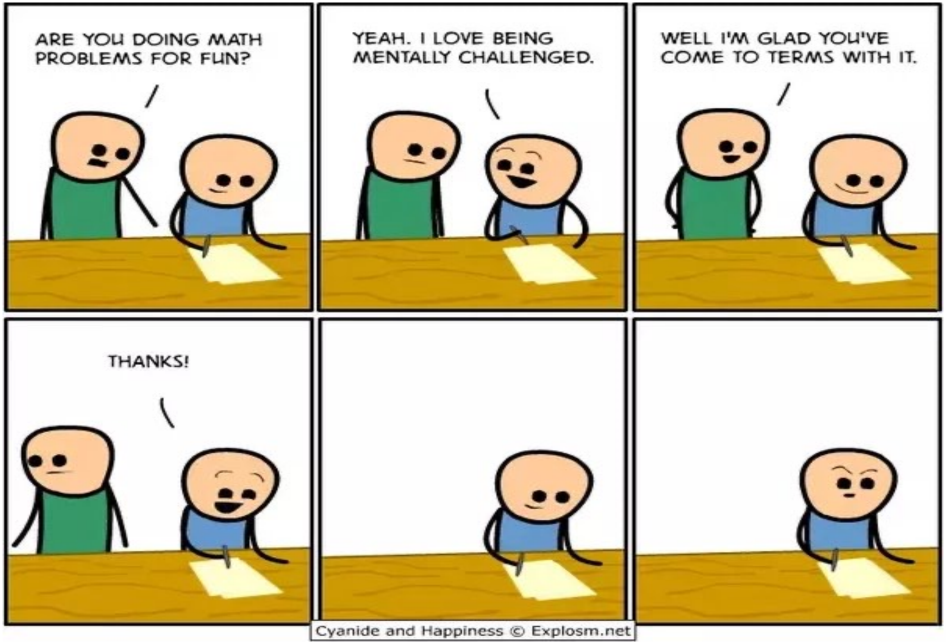
Welcome!

- Glad to see everyone!
- These are uncertain times, make sure you invest in family, friends and giving.
- This is the last quarter before the summer, let's make it count!
- And find a way to have fun.



What is Mathematics, really?

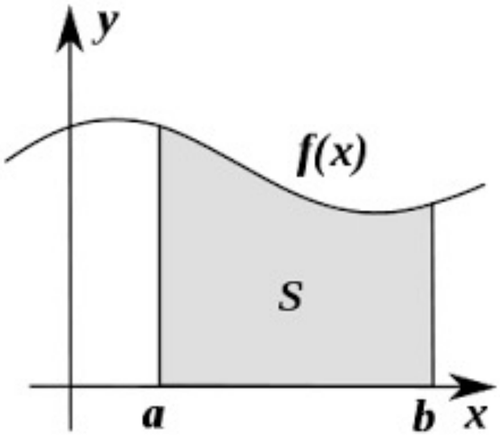
- It's *not* just about numbers!
- Mathematics is *much* more than that:

> Mathematics is, most generally, the study of any and all *absolutely certain* truths about any and all *perfectly well-defined* concepts.

- But, these concepts can be *about* numbers, symbols, objects, images, sounds, *anything*!

Calculus is for continuous systems

- Calculus *f(x)* versus *x,* considers a continuous variable *x.*
  - Continuous numbers: *x* is a number which can have any number of decimal places,
    - E.g. 3.1415
    - Transcendental numbers: pi=3.1415926…..



Discrete systems and structures

- "***Discrete***" ($\neq$ "discreet"!) - Composed of distinct, separable parts. (Opposite of ***continuous***.)
  discrete:***continuous*** :: *digital*:***analog***
- "***Structures***" - Objects built up from simpler objects according to some definite pattern.
- "***Discrete Mathematics***" - The study of discrete, mathematical objects and structures.

Discrete Structures We'll Study

- Propositions
- Predicates
- Proofs
- Sets
- Functions
- Orders of Growth
- Algorithms
- Integers
- Summations
- Sequences
- Strings
- Permutations
- Combinations
- Relations
- Graphs
- Trees
- Logic Circuits
- Automata

Uses for Discrete Math in Computer Science

- Data Science and machine learning
- Advanced algorithms & data structures
- Programming language compilers & interpreters.
- Computer networks
- Operating systems
- Computer architecture

- Database management systems
- Cryptography
- Error correction codes
- Graphics & animation algorithms, game engines, *etc*.…
- *I.e.*, the whole field!

Course Objectives

- Upon completion of this course, the student should be able to: **Think!** (Critical reasoning)
    - Check validity of simple logical arguments (proofs).
    - Check the correctness of simple algorithms.
    - Creatively construct simple instances of valid logical arguments and correct algorithms.
    - Describe the definitions and properties of a variety of specific types of discrete structures.
    - Correctly read, represent and analyze various types of discrete structures using standard notations.

Part #1: Foundations of Logic

- *Mathematical Logic* is a tool for working with elaborate *compound* statements. It includes:

- A formal language for expressing them.
- A concise notation for writing them.
- A methodology for objectively reasoning about their truth or falsehood.
- It is the foundation for expressing formal proofs in all branches of mathematics.

Examples: English is ambiguous

- Are you not going to the party tonight?
- You must be at least 5 feet tall or over the age of 14 to ride the rollercoaster.
- The lawn is wet, so either it rained last night or the sprinklers went on this morning, but not both happened.

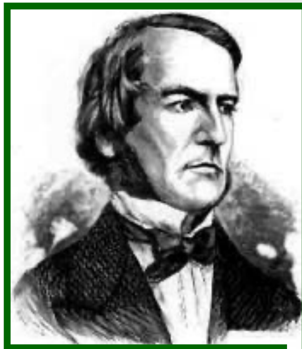
Foundations of Logic: Overview

- Propositional logic:
  - Basic definitions.
  - Equivalence rules & derivations.

- Predicate logic
  - Predicates.
  - Quantified predicate expressions.
  - Equivalences & derivations.

Propositional Logic

- *Propositional Logic* is the logic of compound statements built from simpler statements using so-called *Boolean connectives.*



George Boole
(1815-1864)

Some applications in computer science:

- Design of digital electronic circuits.
- Expressing conditions in programs.
- Queries to databases & search engines.

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

Definition of a *Proposition*

- **Definition:** A *proposition* (denoted *p*, *q*, *r*, …) is simply:
  - A *declarative statement with some definite meaning*, (not vague or ambiguous)
  - having a *truth value* that is either *true* (**T**) or *false* (**F**)
  - 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.

Note, we will also use the notation:
T=1, F=0

Examples of Propositions

- "Beijing is the capital of China."
- "1 + 2 = 5"
- "It is raining." (T/F assessed given situation.)

- But, the following are **NOT** propositions:
- "Who's there?" (interrogative, question)
- "La la la la la." (meaningless interjection)
- "Just do it!" (imperative, command)
- "1 + 2" (expression with a non-true/false value; does not assert anything)

Absurd statements can be propositions

- “Pigs can fly”
- “The moon is made of green cheese”
- “1+1 = 10”

Example –
Are these statements propositions?

- P = “This statement is true”
- P = “This statement is false”

Example –
Are these statements propositions?

- P = "This statement is true"
  - ***Yes***, *and the truth value is* ***T***

- P = "This statement is false"
  - ***No***, *not a proposition; cannot assign* ***T*** *or* ***F***
  - *It is not logically consistent:*
    - *If P is T then the statement is F (i.e., P is F)*
    - *If P is F then the statement is T (i.e., P is T)*
  - *A proposition must be T or F but not both.*

Boolean Operators / Connectives

- An *operator* or *connective* combines one or more *operand* expressions into a larger expression. (*E.g.*, "+" in numeric expressions.)
  - *Unary* operators take 1 operand (*e.g., negation,* −3);
  - *binary* operators take 2 operands (*e.g., multiplication,* 3 × 4).

- *Propositional* or *Boolean* operators operate on propositions (or their truth values) instead of on numbers.

Some Popular Boolean Operators

Formal Name	Nickname	Arity	Symbol
Negation operator	NOT	/jūn/ Unary only one	¬
Conjunction operator	AND	Binary	∧
Disjunction operator	OR	Binary	∨
Exclusive-OR operator	XOR (Exclusive or) either A or B, not both	Binary	⊕
Implication operator	IMPLIES if A ✓, B must ✓	Binary	→
Biconditional operator	IFF { B ✓ ; A ✓ A ✓ ; B ✓ If and only if	Binary	↔

Truth tables

- To evaluate the T or F status of a compound proposition
- Enumerate over all T and F assignments of all the propositions

The Negation Operator

- The unary *negation operator* "¬" (*NOT*) transforms a prop. into its logical *negation*.
- *E.g.* If p = "I have brown hair."

  then $\neg p$ = "I do **not** have brown hair."
- The *truth table* for NOT:

p	$\neg p$
T	F
F	T

Operand column

Result column

T :≡ True; F :≡ False

":≡" means "is defined as"

The Conjunction Operator

$\wedge \Rightarrow$ And

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

$\wedge$ND

- *E.g.* If
  p= "I will have salad for lunch." and
  q= "I will have steak for dinner.",

- then $p \wedge q$= "I will have salad for lunch **and** I will have steak for dinner."

Remember: "$\wedge$" points up like an "A", and it means "$\wedge$ND"

Conjunction Truth Table

- 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

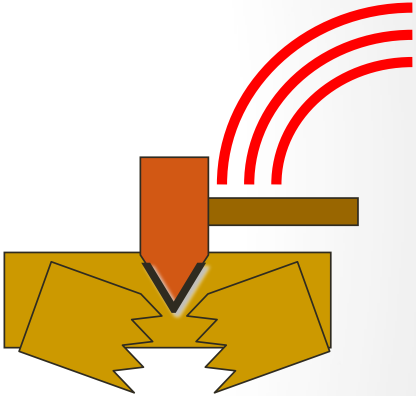
p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

- Remark. ¬ and ∧ operations together are sufficient to express *any* Boolean truth table!

The Disjunction Operator

- The binary *disjunction operator* "∨" (*OR*) combines two propositions to form their logical *disjunction*.

- *p*= "My car has a bad engine."
- *q*= "My car has a bad carburetor."
- $p \lor q$= "Either my car has a bad engine, **or** my car has a bad carburetor."



Meaning is like "and/or" in English.

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!
- So, this operation is also called *inclusive or,* because **it includes** the possibility that both p and q are true.

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

or

Note difference from AND

- Remark. "¬" and "∨" together are also universal. (We can write any Boolean logic function in terms of those operators.)

Nested Propositional Expressions

- Use parentheses to *group sub-expressions*:
  "I just saw my old friend, and either he's grown or I've shrunk."

- First break it down into propositions:
  - f = "I just saw my old friend"
  - g = "he's grown"
  - s = "I've shrunk"

- $= f \overset{\text{and}}{\wedge} (g \overset{\text{or}}{\vee} s)$
  - $(f \overset{\text{and}}{\wedge} g) \overset{\text{or}}{\vee} s$ would mean something different
  - $f \overset{\text{and}}{\wedge} g \overset{\text{or}}{\vee} s$ would be ambiguous

- By convention, "¬" takes *precedence* over both "$\wedge$" and "$\vee$".
  - $\neg s \wedge f$ means $(\neg s) \wedge f$, it does **not** mean $\neg (s \wedge f)$

A Simple Exercise

- Let
  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:
- $\neg p$ = not rained
- $r \overset{\text{and}}{\wedge} \neg p$ =
- $\underline{\neg r} \overset{\text{or}}{\vee} p \overset{\text{or}}{\vee} q$ =

The *Exclusive Or* Operator

- The binary *exclusive-or operator* "$\oplus$" (*XOR*) combines two propositions to form their logical "exclusive or" (exclusive disjunction)

- 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!)"

- A more common phrase: "Your entrée comes with either soup or salad"

Exclusive-Or Truth Table

- Note that $p \oplus q$ means that p is true, or q is true, but **not both**!

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

Note difference from OR.

- This operation is called *exclusive or,* because it **excludes** the possibility that both p and q are true.
- Remark. "$\neg$" and "$\oplus$" together are **not** universal.

Natural Language is Ambiguous

- Note that English "or" can be ambiguous regarding the "both" case!
- "Pat is a singer or Pat is a writer." - $\vee$
- "Pat is alive or Pat is deceased." - $\oplus$

p	q	p "or" q
F	F	F
F	T	T
T	F	T
T	T	?

- Need context to disambiguate the meaning!
- For this class, assume "or" means inclusive.

$\wedge$

The *Implication* Operator

- The *implication* $p \rightarrow q$ states that $\overbrace{p}^{\text{hypothesis/antecedent}}$ implies $\overbrace{q}^{\text{conclusion/consequent}}$.

- *i.e.*, If p is true, then q is true; but if p is not true, then q could be either true or false.

- *E.g.*, let p = "You master ECS20."
  q = "You will get a good job."

- $p \rightarrow q$ = "If you master ECS20, then you will get a good job."

(But note, some good jobs do not require discrete math so having a good job does not necessarily mean that you mastered ECS20).

Let's build the truth table for $p \rightarrow q$

Implication Truth Table

- $p \rightarrow q$ is **false** only when p is true but q is **not** true.
- $p \rightarrow q$ does **not** say that p causes q!
- $p \rightarrow q$ does **not** require that p or q **are ever true**!

Start from Falsehood ⇑ anything can be true

- *E.g.* "$(1{=}0) \rightarrow$ pigs can fly" is TRUE!

starting with F, everything is true.

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	**F**
T	T	T

The only False case!

Examples of Implications

- "If this lecture ever ends, then the sun will rise tomorrow." *True* or *False*?

- "If Tuesday is a day of the week, then I am a penguin." *True* or *False*?

  ② (False) ① (True) ⇒ False

  Q if P

- "1+1=6, if Biden is president."
  True or *False*?

  P implies Q

- "If the moon is made of green cheese, then I am richer than Elon Musk." *True* or *False*?

Examples of Implications

- "If this lecture ever ends, then the sun will rise tomorrow." *True* or *False*? (p=T and q=T)

- "If Tuesday is a day of the week, then I am a penguin." *True* or *False*? (p=T and q=F)

Q if P

- "1+1=6, if Biden is president." *True* or *False*? (p=T and q=F)

- "If the moon is made of green cheese, then I am richer than Elon Musk." *True* or *False*? (p=F and q=F)

Logic cares about T/F values and not about if implications are sensical

- Consider a sentence like,
  - "If I wear a red shirt tomorrow, then global peace will prevail"
- In logic, the sentence is **True** so long as either I don't wear a red shirt, or global peace is achieved.
- But, in normal English conversation, if I were to make this claim, you would think that I was crazy.
- Why this discrepancy between logic & language?
  - Logic is about self-consistency.

English Phrases Meaning $p \rightarrow q$

- "p implies q"
- "if p, then q"
- "if p, q"
- "when p, q"
- "whenever p, q"
- "q if p"
- "q when p"
- "q whenever p"
- "p only if q"
- "p is sufficient for q"
- "q is necessary for p"
- "q follows from p"
- "q is implied by p"

- We will see some equivalent logic expressions later.

In this class we will use the phrases in red above

Translating between written English and propositional logic

- Isolate the constituent propositions of a compound proposition. (You can name them with letters that remind you what the proposition is about.)

- For conditional statements, note if it is written in terms of "if p then q" or in terms of "q if p".

- Identify the Boolean operators being used in the compound proposition.

- Write the sentence in propositional logic.

Example 1

- “You are at least 16 years old if you have a US Driver’s License and live in CA”.

- d = “you have a US Driver’s License”
- c = “you live in CA”
- s = “you are at least 16 years old”

If _ and _, then _

- Symbolic logic translation: $(d \wedge c) \rightarrow s$

Example 2

- "You can access the Internet from campus only if you are a computer science major or you are not a Freshman."

- i = "You can access the Internet from campus"
- c = "you are a computer science major"
- f = "you are a Freshman"

p only if q

- Symbolic logic translation: $(c \vee \neg f) \rightarrow i$

Example 3

- “You cannot ride the roller coaster if you are under 4 feet tall unless you are older than 16 years old.”

- r = ”ride the roller coaster”
- f = ”you are under 4 feet tall”
- s = “you are older then 16”

- Symbolic logic translation: $(\neg f \overset{\text{or}}{\vee} s) \rightarrow r$

Converse, Inverse, Contrapositive

- Some terminology, for an implication $p \rightarrow q$:
- Its *converse* is: $q \rightarrow p$.
- Its *inverse* is: $\neg p \rightarrow \neg q$.
- Its *contrapositive*: $\neg q \rightarrow \neg p$.
- One of these three has the *same meaning* (same truth table) as $p \rightarrow q$. Can you figure out which?

Contrapositive

- Proving the equivalence of $p \rightarrow q$ and its contrapositive using truth tables:

T → F ⇒F

p	q	$\neg q$	$\neg p$	$p \rightarrow q$	$\neg q \rightarrow \neg p$
F	F	T	T	T	T
F	T	F	T	T	T
T	F	T	F	F	F
T	T	F	F	T	T

Foundations of logic come from the implication operator and its contrapositive

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

  called *modus ponens* (the mode that affirms)

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

  called *modus tollens* (the mode that denies)

The *biconditional* operator

- The *biconditional* $p \leftrightarrow q$ states that $p \rightarrow q$ and $q \rightarrow p$
- In other words, p is true *if and only if (IFF)* q is true.

- p = "Italy wins the 2022 FIFA World Cup."
- q = "Italy will be World Cup Champion for all of 2023."

- $p \leftrightarrow q$ = "If, and only if, Italy wins the 2022 World Cup, Italy will be World Cup Champion for all of 2023."

Biconditional Truth Table

- $p \leftrightarrow q$ means that p and q have the **same** truth value.
- Remark. This truth table is the exact **opposite** of $\oplus$'s!
  - Thus, $p \leftrightarrow q$ means $\neg(p \oplus q)$

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

- $p \leftrightarrow q$ does **not** imply that p and q are true, or that either of them causes the other, or that they have a common cause.

Boolean Operations Summary

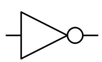
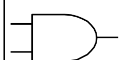
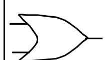
p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
F	F	T	F	F	F	T	T
F	T	T	F	T	T	T	F
T	F	F	F	T	T	F	F
T	T	F	T	T	F	T	T

Order of operation: $\neg$, $\wedge$, $\vee$, $\oplus$, $\rightarrow$, $\leftrightarrow$

i.e., $p \vee \neg q \rightarrow p \wedge q$ means $(p \vee (\neg q)) \rightarrow (p \wedge q)$

(Note, precedence of $\vee$, $\oplus$ is ambiguous and often depends on the programming language)

Some Alternative Notations

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

Bits and Bit Operations

John Tukey
(1915-2000)

- A *bit* is a <u>bi</u>nary (base 2) dig<u>it</u>: 0 or 1.
- Bits may be used to represent truth values.
- By convention:
  0 represents "false";
  1 represents "true".

- *Boolean algebra* is like ordinary algebra except that variables stand for bits,
  \+ means "or", and
  multiplication means "and".
  - See module 23 (chapter 10) for more details.

Propositional Consistency

Propositional Consistency

- Life is complex: we often have to satisfy multiple logical compound propositions
- Eg. A=, B=
- Two different compound propositions may be True at the same time. We call them *consistent*.

Learn:

- How to *prove* propositional consistency using truth *tables*.

Logical Consistency

- **Definition:** Compound proposition *p* is *logically consistent with* compound proposition *q*, **IFF** *p and q can be true simultaneously*.
- Compound propositions *p* and *q* are logically consistent to each other **IFF** *p* and *q* contain T simultaneously in at least one row of their truth tables.

E.g., Where Is the Treasure?

- Among four people, P1, P2, P3, P4, at least one of is truthful, and at least one is lying
- One of the truthful ones has a treasure in their pocket.
- They each know who has the treasure and each of them makes a statement:

S1 (by P1): I don’t have the treasure.

S2 (by P2): My pockets are empty.

S3 (by P3): P1 is lying.

S4 (by P4): P1 is ~~lying~~ truthful.

Where is the treasure?

Where Is the Treasure?, contd.

- How to solve?
- Name the statements, and create a truth table where the inputs are the truthfulness of the people (Truthful, Lies)
- Find a row for which P1-P4 assignments are consistent with all S1-S4 ~~are True~~, R1, R2

R1 : #T > 0 , # L > 0.

R2 : Treasure in a T.

P1	P2	P3	P4	S1-S4 consistent	Why?
T	T	T	T	No	All truthful R1 violated
T	T	T	L	No	S3 violated
T	T	L	L	NO	P1 & P4, S4 contrad.
T	T	L	T	Yes!	

S1 (by P1): I don't have the treasure.

S2 (by P2): My pockets are empty.

S3 (by P3): P1 is lying.

S4 (by P4): P1 is ~~lying~~ Truthful.

Propositional Equivalence

Propositional Equivalence

- Two *syntactically* (*i.e.,* textually) different compound propositions may be the *semantically* identical (*i.e.,* have the same meaning). We call them *equivalent*. Learn:

- Various *equivalence rules* or *laws*.
- How to *prove* equivalences using *symbolic derivations*.

Tautologies and Contradictions

- A *tautology* is a compound proposition that is always **true** *no matter what* the truth values of its atomic propositions are!
- *Ex.* $p \overset{\text{or}}{\vee} \neg p = T$ always

- A *contradiction* is a compound proposition that is **false** no matter what!
- *Ex.* $p \overset{\text{and}}{\wedge} \neg p = F$ always

* Contingency (other)

- Otherwise the compou

(i.e. most propositions are cor

TABLE 1 Examples of a Tautology and a Contradiction.

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

Logical Equivalence

- **Definition:** Compound proposition p is *logically equivalent* to compound proposition q, written $p \Leftrightarrow q$, **IFF** the compound proposition $p \leftrightarrow q$ is a tautology.
- Note, $\Leftrightarrow$ is often denoted by $\equiv$

  (We will use both notations in this class)
- Compound propositions p and q are logically equivalent to each other **IFF** p and q contain the same truth values as each other in <u>all</u> rows of their truth tables.

Proving Equivalence via Truth Tables

or and

- Prove that $p \vee q \Leftrightarrow \neg(\neg p \wedge \neg q)$.

Proving Equivalence via Truth Tables

- *Ex.* Prove that $p \vee q \Leftrightarrow \neg(\neg p \wedge \neg q)$.

p	q	or $p \vee q$	$\neg p$	$\neg q$	and $\neg p \wedge \neg q$	not $\neg(\neg p \wedge \neg q)$
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

- These are similar to the arithmetic identities you may have learned in algebra, but for propositional equivalences instead.
- They 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.
- Summarized in Table 6, Sec 1.3 of Rosen (and posted on Canvas)

Equivalence Laws - Examples

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

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

More Equivalence Laws

- *Distributive*: $p \overset{\text{or}}{\vee} (q \overset{\text{and}}{\wedge} r) \Leftrightarrow (p \overset{\text{or}}{\vee} q) \overset{\text{and}}{\wedge} (p \overset{\text{or}}{\vee} r)$
  $p \overset{\text{and}}{\wedge} (q \overset{\text{or}}{\vee} r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$

- *De Morgan's*: $\equiv$
  $\neg(p \overset{\text{and}}{\wedge} q) \Leftrightarrow \neg p \overset{\text{or}}{\vee} \neg q$
  $\neg(p \overset{\text{or}}{\vee} q) \Leftrightarrow \neg p \overset{\text{and}}{\wedge} \neg q$

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

Augustus De Morgan (1806-1871)

- *Trivial tautology/contradiction*:
  $p \overset{\text{or}}{\vee} \neg p \Leftrightarrow \mathbf{T}$ $\quad$ $p \overset{\text{and}}{\wedge} \neg p \Leftrightarrow \mathbf{F}$

De Morgan's law

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

Not (p or q): $\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$

Not (p and q): $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

Summary of basic equivalences

TABLE 6 **Logical Equivalences.**

Equivalence	*Name*
$p \wedge \mathbf{T} \equiv p$ $p \vee \mathbf{F} \equiv p$	Identity laws
$p \vee \mathbf{T} \equiv \mathbf{T}$ $p \wedge \mathbf{F} \equiv \mathbf{F}$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws
$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's laws
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv \mathbf{T}$ $p \wedge \neg p \equiv \mathbf{F}$	Negation laws

Defining Operators via Equivalences

- Using equivalences, we can *define* operators in terms of other operators.
- Exclusive or: $p \oplus q \Leftrightarrow (p \vee q) \wedge \neg(p \wedge q)$

  $p \oplus q \Leftrightarrow (p \wedge \neg q) \vee (q \wedge \neg p)$
- Implication: $p \rightarrow q \Leftrightarrow \neg p \vee q$
- Biconditional: $p \leftrightarrow q \Leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$

  $p \leftrightarrow q \Leftrightarrow \neg(p \oplus q)$

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	**F**
T	T	T

Logical equivalences for conditional statements

TABLE 7 **Logical Equivalences Involving Conditional Statements.**

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

TABLE 8 **Logical Equivalences Involving Biconditional Statements.**

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

Tables 7-8, Sec1.3 Rosen

Example 1 for logical equiv.

- Using logical equivalences, show that

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

- Soln:

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

$\Leftrightarrow (\neg p \vee q) \wedge (\neg p \vee r)$ [Expand definition of $\to$]

$\Leftrightarrow \neg p \vee (q \wedge r)$ [distributive law]

$\Leftrightarrow p \to (q \wedge r)$ [logical equivalence for $\to$]

Example 2 for logical equiv.

- Using logical equivalences, show that

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

- Let's do this one together on pen and paper.

Example 3 – an involved calculation

- Check using a symbolic derivation whether $(p \wedge \neg q) \rightarrow (p \oplus r) \Leftrightarrow \neg p \vee q \vee \neg r$.

- $(p \wedge \neg q) \rightarrow (p \oplus r)$
- $\Leftrightarrow \neg(p \wedge \neg q) \vee (p \oplus r)$ [Expand definition of $\rightarrow$]
- $\Leftrightarrow \neg(p \wedge \neg q) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [Expand defn. of $\oplus$]
- $\Leftrightarrow (\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [DeMorgan's Law]
- *cont.*

Example Continued...

- $\Leftrightarrow (\neg p \vee q) \vee ((p \vee r) \wedge \neg(p \wedge r))$
- $\Leftrightarrow (q \vee \neg p) \vee ((p \vee r) \wedge \neg(p \wedge r))$ [$\vee$ commutes]
- $\Leftrightarrow q \vee (\neg p \vee ((p \vee r) \wedge \neg(p \wedge r)))$ [$\vee$ associative]
- $\Leftrightarrow q \vee (((\neg p \vee (p \vee r)) \wedge (\neg p \vee \neg(p \wedge r)))$ [distrib. $\vee$ over $\wedge$]

$\Leftrightarrow q \vee (((\neg p \vee p) \vee r) \wedge (\neg p \vee \neg(p \wedge r)))$ [assoc.]

$\Leftrightarrow q \vee ((\mathbf{T} \vee r) \wedge (\neg p \vee \neg(p \wedge r)))$ [trivail taut.]

$\Leftrightarrow q \vee (\mathbf{T} \wedge (\neg p \vee \neg(p \wedge r)))$ [domination]

$\Leftrightarrow q \vee (\neg p \vee \neg(p \wedge r))$ [identity]

cont.

End of Long Example

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

$\Leftrightarrow q \vee (\neg p \vee (\neg p \vee \neg r))$ [DeMorgan's]

$\Leftrightarrow q \vee ((\neg p \vee \neg p) \vee \neg r)$ [Assoc.]

- $\Leftrightarrow q \vee (\neg p \vee \neg r)$ [Idempotent]
- $\Leftrightarrow (q \vee \neg p) \vee \neg r$ [Assoc.]

$\Leftrightarrow \neg p \vee q \vee \neg r$ [Commut.]

Q.E.D.

Remark. *Q.E.D. (quod erat demonstrandum)*

(Which was to be shown.)

Review: Propositional Logic

- Atomic propositions: $p, q, r, \ldots$
- Boolean operators: $\neg \wedge \vee \oplus \rightarrow \leftrightarrow$
- 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. $p \Leftrightarrow q \Leftrightarrow r \ldots$

Foundations of logic

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

  called *modus ponens* (the mode that affirms)

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

  called *modus tollens* (the mode that denies)

Logical equivalence : $(p \Rightarrow q) \equiv (\neg p \lor q)$

p	q	$\neg p$	or $\neg p \lor q$	$p \Rightarrow q$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

emonstrates the fact that $p \Rightarrow q$ is logically equivalent to $\neg p \lor q$.

n table for most commonly used logical operators [edit]

is a truth table that gives definitions of the 7 most commonly used out of the 16 possible truth functions of two Boolean variables P and Q:

P	Q	$P \land Q$	$P \lor Q$	$P \underline{\lor} Q$	$P \underline{\land} Q$	$P \Rightarrow Q$	$P \Leftarrow Q$	$P \Leftrightarrow Q$
T	T	T	T	F	T	T	T	T
T	F	F	T	T	F	F	T	F
F	T	F	T	T	F	T	F	F
F	F	F	F	F	T	T	T	T
P	Q	$P \land Q$	$P \lor Q$	$P \underline{\lor} Q$	$P \underline{\land} Q$	$P \Rightarrow Q$	$P \Leftarrow Q$	$P \Leftrightarrow Q$
		AND (conjunction)	OR (disjunction)	XOR (exclusive or)	XNOR (exclusive nor)	conditional "if-then"	conditional "then-if"	biconditional "if-and-only-if"
where T *means* **true** *and* F *means* **false**								

Condensed truth tables for binary operators [edit]

For binary operators, a condensed form of truth table is also used, where the row headings and the column headings specify the operands and the table cells specify the

Boolean logic uses this condensed truth table notation:

egation of a conjunction: ¬(*p* ∧ *q*), and the disjunction of negations: (¬*p*) ∨ (¬*q*) can be tabulated as follows:

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

ical NOR [edit]

logical NOR is an operation on two logical values, typically the values of two propositions, that produces a value of *true* if both of its operands are false. In other words, it produces a va

lse if at least one of its operands is true. ↓ is also known as the Peirce arrow after its inventor, Charles Sanders Peirce, and is a Sole sufficient operator.

truth table for **p NOR q** (also written as **p ↓ q**, or **Xpq**) is as follows:

Logical NOR

p	*q*	*p* ↓ *q*
T	T	F
T	F	F
F	T	F
F	F	T

he negation of a disjunction ¬(*p* ∨ *q*), and the conjunction of negations (¬*p*) ∧ (¬*q*) can be tabulated as follows:

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