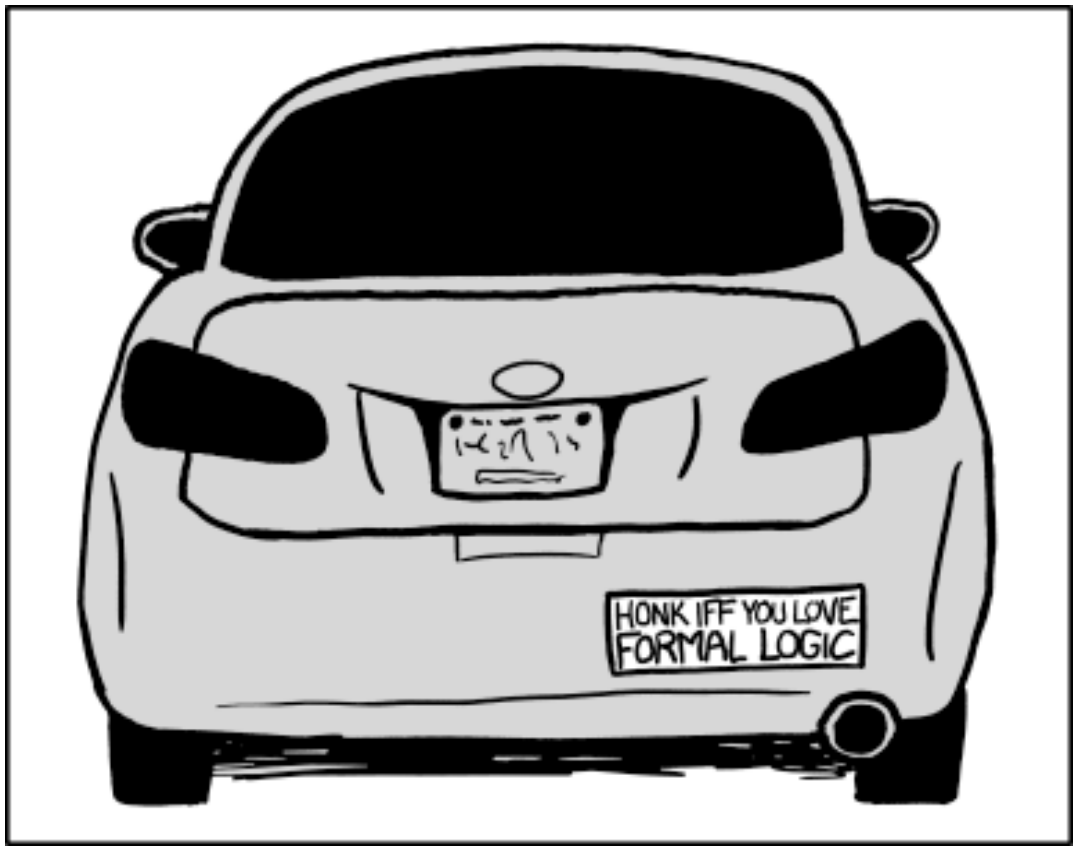
CSE 311: Foundations of Computing I

Topic 1: Formal Logic



What is logic and why do we need it?

Logic is a language, like English or Java, with its own

- words and rules for combining words into sentences (syntax)
- ways to assign meaning to words and sentences (semantics)

Compared to English, Logic is more

- concise (useful)
- precise (critical!)

Importantly, Logic comes with its own toolkit

Why not use English?

- Turn right here…

  Does “right” mean the direction or now?

- We saw her duck

  Does “duck” mean the animal or crouch down?

- Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo

  This means “Bison from Buffalo, that bison from Buffalo bully, themselves bully bison from Buffalo.

Natural languages can be unclear / imprecise

Propositions: building blocks of logic

A ***proposition*** is a statement that

- is "well-formed" (syntactically valid)
- is either true or false

Propositions: building blocks of logic

A ***proposition*** is a statement that

- is "well-formed"
- is either true or false

Garfield is a mammal and Garfield is a cat

true

Odie is a mammal and Odie is a cat

false

Are These Propositions?

$2 + 2 = 5$

This is a proposition. It's okay for propositions to be false.

$x + 2 = 5389$, where x is my PIN number

This is a proposition. We don't need to know what x is.

Akjsdf!

Not a proposition because it's gibberish.

Who are you?

This is a question which means it doesn't have a truth value.

Every positive even integer can be written as the sum of two primes.

This is a proposition. We don't know if it's true or false, but we know it's one of them!

Propositions

We need a way of talking about *arbitrary* ideas…

Propositional Variables: p, q, r, s, …

Constant truth values:

- T for true
- F for false

Familiar from Java

Java `boolean` represents a truth value

- constants `true` and `false`
- variables hold *unknown* values

Operators calculate new values from given ones

- unary: not (`!`)
- binary: and (`&&`), or (`||`)

Logical Connectives

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

con	with	p with q (i.e., both)
dis-	apart from	not necessarily both

Logical Connectives

Negation (not)	$\neg p$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$

$p \vee q$ at least one of p or q

$p \oplus q$ exactly one of p or q

Logic forces us to distinguish $\vee$ from $\oplus$

Logical Connectives

Negation (not)	$\neg p$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$

Syntax of Logical Connectives

		Precedence
Negation (not)	$\neg p$	highest
Conjunction (and)	$p \wedge q$	
Disjunction (or)	$p \vee q$	
Exclusive Or	$p \oplus q$	
Implication	$p \longrightarrow r$	
Biconditional	$p \longleftrightarrow q$	lowest

$p \vee q \wedge r \longrightarrow t$ means $(p \vee (q \wedge r)) \longrightarrow t$

Syntax of Logical Connectives

		Associativity
Conjunction (and)	$p \wedge q$	left-to-right
Disjunction (or)	$p \vee q$	left-to-right
Exclusive Or	$p \oplus q$	
Implication	$p \longrightarrow r$	right-to-left
Biconditional	$p \longleftrightarrow q$	

$p \vee q \vee r \vee t$ means $((p \vee q) \vee r) \vee t$

$p \longrightarrow q \longrightarrow r$ means $p \longrightarrow (q \longrightarrow r)$

Some Truth Tables

p	$\neg p$
T	
F	

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

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

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

Some Truth Tables

p	$\neg p$
T	F
F	T

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

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

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

Another Truth Table

p	*r*	$p \rightarrow r$
T	T	
T	F	
F	T	
F	F	

With implication ($\rightarrow$), *p* is called the "premise" and r is called the "conclusion".

The implication is true when p and r are true.

The implication is true ("vacuously") when p is false.

Another Truth Table

p	*r*	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

With implication ($\rightarrow$), *p* is called the "premise" and r is called the "conclusion".

The implication is true when p and r are true.

The implication is true ("vacuously") when p is false.

Implication

"If it was raining, then I had my umbrella"

p	r	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

*It's useful to think of implications as promises. That is "Was I **wrong**?"*

	It's raining	It's not raining
I have my umbrella		
I do not have my umbrella		

Implication

"If it was raining, then I had my umbrella"

p	r	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

*It's useful to think of implications as promises. That is "Was I **wrong**?"*

	It's raining	It's not raining
I have my umbrella	No	No
I do not have my umbrella	**Yes**	No

I am only wrong when:

(a) It's raining AND

(b) I don't have my umbrella

Implication

"If the Seahawks won, then I was at the game."

p	r	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

In what scenario was I wrong?

	I was at the game	I wasn't at the game
Seahawks won		
Seahawks lost		

Implication

"If the Seahawks won, then I was at the game."

p	r	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

In what scenario was I wrong?

	I was at the game	I wasn't at the game
Seahawks won	Ok	**Doh!**
Seahawks lost	Ok	Ok

Implication

"If it's raining, then I have my umbrella"

p	r	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

Are these true?

$2 + 2 = 4 \rightarrow$ *earth is a planet*

The fact that these are unrelated doesn't make the statement false! "2 + 2 = 4" is true; "earth is a planet" is true. T→T is true. So, the statement is true.

$2 + 2 = 5 \rightarrow$ *26 is prime*

Again, these statements may or may not be related. "2 + 2 = 5" is false; so, the implication is true. (Whether 26 is prime or not is irrelevant).

Implication is not a causal relationship!

$$p \rightarrow r$$

---

(1) "I have collected all 151 Pokémon if I am a Pokémon master"

(2) "I have collected all 151 Pokémon only if I am a Pokémon master"

In English, the "if" can be written at the end of the sentence rather than at the beginning of the sentence (followed by a ",").

$$p \rightarrow r$$

---

(1) "I have collected all 151 Pokémon if I am a Pokémon master"

(2) "I have collected all 151 Pokémon only if I am a Pokémon master"

The implications are:

(1) If I am a Pokémon master, *then* I have collected all 151 Pokémon.

(2) If I have collected all 151 Pokémon, *then* I am a Pokémon master.

$p \rightarrow r$

Implication:

- *p* implies *r*
- whenever *p* is true, *r* must be true
- if *p*, then *r*
- *r* if *p*
- *p* only if *r*
- *p* is sufficient for *r*
- *r is necessary for p*

p	*r*	$p \rightarrow r$
T	T	T
T	F	F
F	T	T
F	F	T

Biconditional: $p \leftrightarrow q$

- ***p* if and only if *q***
- ***p* "iff" *q***
  - ***p and q have the same value truth value***

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

A Compound Proposition (Practical Example)

"Show the notification to the user if its their second login or they've used it for two weeks and haven't tried the feature X unless they did use the feature Y."

Not at all clear what exactly this means!

Can use logic to understand exactly when to show it

A Compound Proposition (Silly Example)

"Garfield has black stripes if he is an orange cat and likes lasagna, and he is an orange cat or does not like lasagna"

We'd like to *understand* what this proposition means.

A Compound Proposition

"Garfield has black stripes if he is an orange cat and likes lasagna, and he is an orange cat or does not like lasagna"

We'd like to *understand* what this proposition means.

First find the simplest (**atomic) propositions**:

- q "Garfield has black stripes"
- r "Garfield is an orange cat"
- s "Garfield likes lasagna"

(q if (r and s)) and (r or (not s))

Logical Connectives

Negation (not)	$\neg p$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$

q	"Garfield has black stripes"
r	"Garfield is an orange cat"
s	"Garfield likes lasagna"

"Garfield has black stripes if he is an orange cat and likes lasagna, and he is an orange cat or does not like lasagna"

(q if (r and s)) and (r or (not s))

Logical Connectives

Negation (not)	$\neg p$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$

q	"Garfield has black stripes"
r	"Garfield is an orange cat"
s	"Garfield likes lasagna"

"Garfield has black stripes if he is an orange cat and likes lasagna, and he is an orange cat or does not like lasagna"

(q if (r and s)) and (r or (not s))

((r ∧ s) ⟶ q) ∧ (r ∨ ¬s)

Analyzing the Garfield Sentence with a Truth Table

q	r	s	$((r \wedge s) \rightarrow q) \wedge (r \vee \neg s)$
F	F	F	
F	F	T	
F	T	F	
F	T	T	
T	F	F	
T	F	T	
T	T	F	
T	T	T	

subexpressions are not (yet) columns in this table

we will always include **all subexpressions** (easiest to verify)

Analyzing the Garfield Sentence with a Truth Table

q	r	s	$r \vee \neg s$	$(r \wedge s) \rightarrow q$	$((r \wedge s) \rightarrow q) \wedge (r \vee \neg s)$
F	F	F			
F	F	T			
F	T	F			
F	T	T			
T	F	F			
T	F	T			
T	T	F			
T	T	T			

Analyzing the Garfield Sentence with a Truth Table

q	r	s	$\neg s$	$r \vee \neg s$	$r \wedge s$	$(r \wedge s) \rightarrow q$	$((r \wedge s) \rightarrow q) \wedge (r \vee \neg s)$
F	F	F					
F	F	T					
F	T	F					
F	T	T					
T	F	F					
T	F	T					
T	T	F					
T	T	T					

Analyzing the Garfield Sentence with a Truth Table

q	r	s	$\neg s$	$r \vee \neg s$	$r \wedge s$	$(r \wedge s) \rightarrow q$	$((r \wedge s) \rightarrow q) \wedge (r \vee \neg s)$
F	F	F	T	T	F	T	T
F	F	T	F	F	F	T	F
F	T	F	T	T	F	T	T
F	T	T	F	T	T	F	F
T	F	F	T	T	F	T	T
T	F	T	F	F	F	T	F
T	T	F	T	T	F	T	T
T	T	T	F	T	T	T	T

Understanding Garfield Claim

"Garfield has black stripes if he is an orange cat and likes lasagna, and he is an orange cat or does not like lasagna"

Black Stripes	Orange	Likes Lasagna	Claim
F	F	F	**T**
F	F	T	F
F	T	F	**T**
F	T	T	F
T	F	F	**T**
T	F	T	F
T	T	F	**T**
T	T	T	**T**

Propositional Logic makes clear exactly what is being claimed.

Understanding Garfield Claim

Black Stripes	Orange	Likes Lasagna	Claim
F	F	F	T
…	…	…	…
T	T	T	T

Consistent with

but also

Converse, Contrapositive

Implication:

$p \rightarrow r$

Contrapositive:

$\neg r \rightarrow \neg p$

Converse:

$r \rightarrow p$

Inverse:

$\neg p \rightarrow \neg r$

Consider

p: 6 is divisible by 2

r: 6 is divisible by 4

$p \rightarrow r$	
$r \rightarrow p$	
$\neg r \rightarrow \neg p$	
$\neg p \rightarrow \neg r$	

Converse, Contrapositive

Implication:

$p \rightarrow r$

Converse:

$r \rightarrow p$

Contrapositive:

$\neg r \rightarrow \neg p$

Inverse:

$\neg p \rightarrow \neg r$

Consider

p: 6 is divisible by 2

r: 6 is divisible by 4

$p \rightarrow r$	F
$r \rightarrow p$	T
$\neg r \rightarrow \neg p$	F
$\neg p \rightarrow \neg r$	T

Converse, Contrapositive

Implication:

$p \rightarrow r$

Contrapositive:

$\neg r \rightarrow \neg p$

Converse:

$r \rightarrow p$

Inverse:

$\neg p \rightarrow \neg r$

How do these relate to each other?

p	r	$p \rightarrow r$	$r \rightarrow p$	$\neg p$	$\neg r$	$\neg p \rightarrow \neg r$	$\neg r \rightarrow \neg p$
T	T						
T	F						
F	T						
F	F						

Converse, Contrapositive

Implication:

$p \rightarrow r$

Contrapositive:

$\neg r \rightarrow \neg p$

Converse:

$r \rightarrow p$

Inverse:

$\neg p \rightarrow \neg r$

An implication and its contrapositive have the same truth value!

p	r	$p \rightarrow r$	$r \rightarrow p$	$\neg p$	$\neg r$	$\neg p \rightarrow \neg r$	$\neg r \rightarrow \neg p$
T	T	T	T	F	F	T	T
T	F	F	T	F	T	T	F
F	T	T	F	T	F	F	T
F	F	T	T	T	T	T	T

Converse, Contrapositive

Implication:

$p \rightarrow r$

Contrapositive:

$\neg r \rightarrow \neg p$

Converse:

$r \rightarrow p$

Inverse:

$\neg p \rightarrow \neg r$

An implication and its inverse
do not have the same truth value!

p	r	$p \rightarrow r$	$r \rightarrow p$	$\neg p$	$\neg r$	$\neg p \rightarrow \neg r$	$\neg r \rightarrow \neg p$
T	T	T	T	F	F	T	T
T	F	F	T	F	T	T	F
F	T	T	F	T	F	F	T
F	F	T	T	T	T	T	T

Equivalence

- **Propositional Logic expressions with the same truth table are called "equivalent"**

- **Examples:**
  - **implication and its contrapositive are equivalent**

    e.g., $(p \vee q) \rightarrow (q \wedge r)$ is equivalent to $\neg(q \wedge r) \rightarrow \neg(p \vee q)$
  - **implication and its inverse are not equivalent**

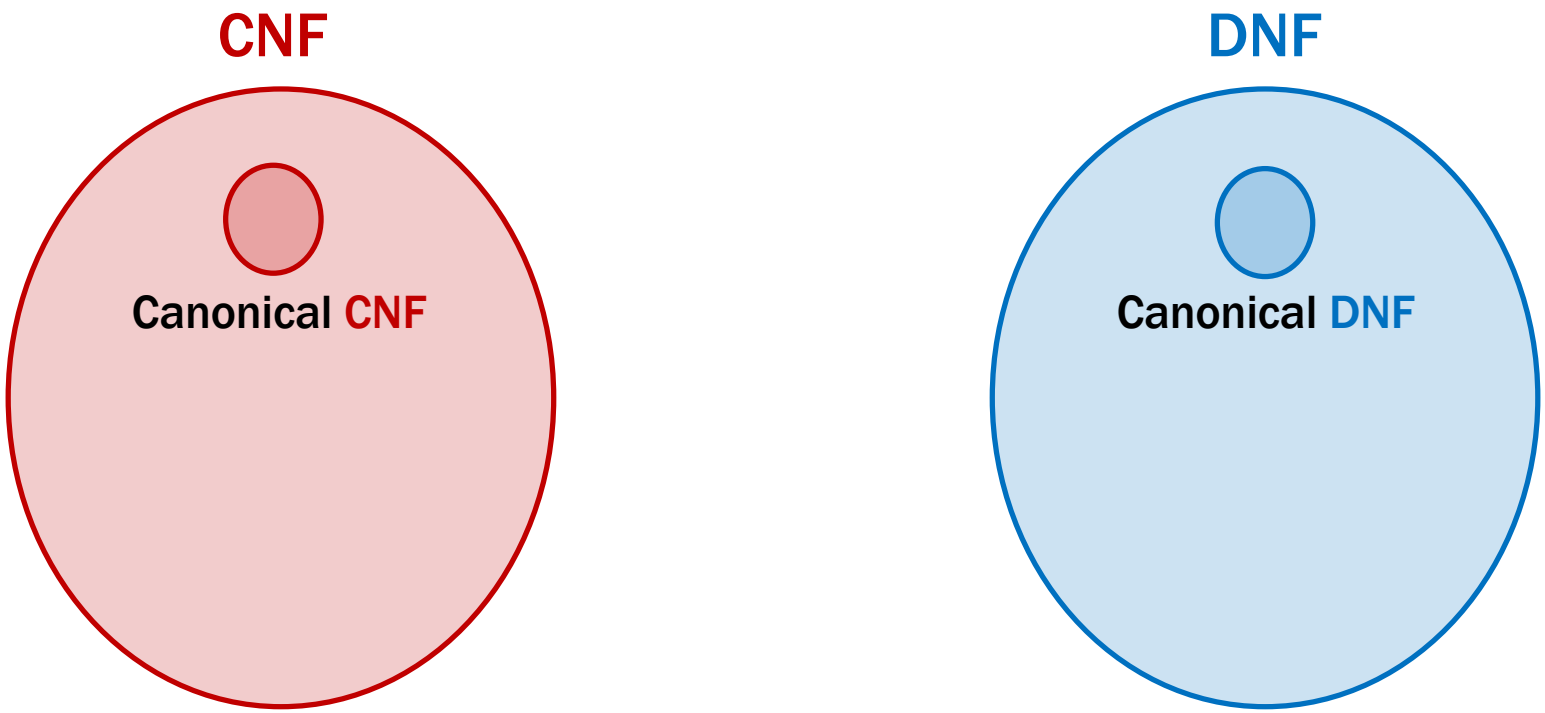
    e.g., $(p \vee q) \rightarrow (q \wedge r)$ is **not** equivalent to $\neg(p \vee q) \rightarrow \neg(q \wedge r)$

    assuming they are the same is the "fallacy of the inverse"

- **Greatly expand on equivalence next time**
  - **prove equivalence without a truth table**

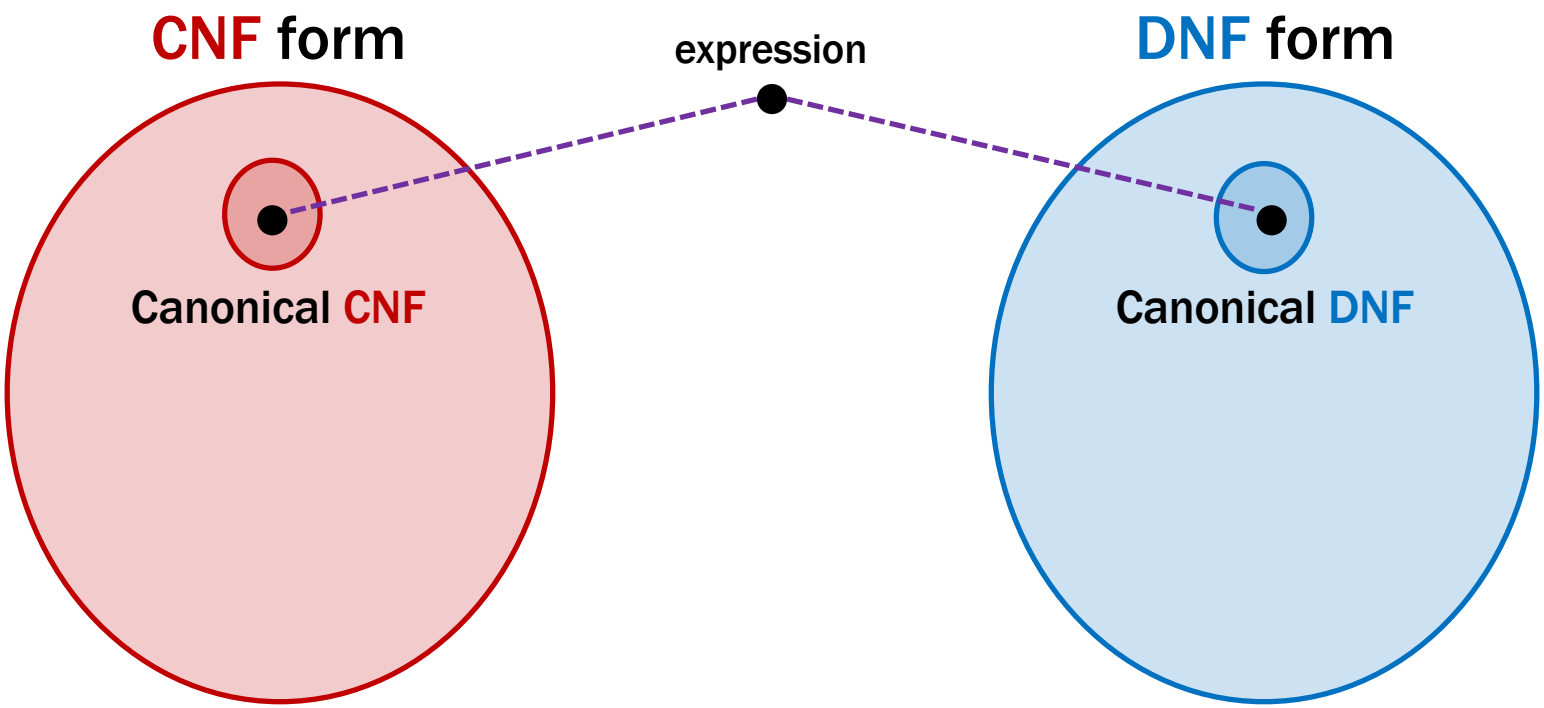
Canonical Forms

CNF & DNF



All Logic Expressions

CNF & DNF



equivalent to exactly <u>one</u> in canonical CNF (up to reordering)

if our expressions are in canonical CNF,
then they are equivalent iff they are the same

Canonical Forms

- Canonical is from Latin "canon" (ruler)
  - compare against to see if equivalent
- We saw one way to do this already: truth table
- Canonical forms are a second way...

Canonical DNF

Suppose F is an expression using the variables a, b, c

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**

1. Find the T rows in the truth table

Canonical DNF

1. Find the T rows in the truth table

2. For each T row, write an expression that is T in that row but *no others* ("min term")

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**

$\neg a \wedge \neg b \wedge c$

This is only T if a = F, b = F, and c = T
(AND requires *all* arguments to be T)

Canonical DNF

1. Find the T rows in the truth table

2. For each T row, write an expression that is T in that row but *no others* ("min term")

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**

$\neg a \wedge \neg b \wedge c$

$\neg a \wedge b \wedge c$

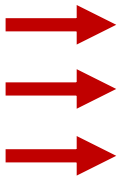
This is only T if a = F, b = T, and c = T

Canonical DNF

1. Find the T rows in the truth table

2. For each T row, write an expression that is T in that row but *no others* ("min term")

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**



$\neg a \wedge \neg b \wedge c$

$\neg a \wedge b \wedge c$

$a \wedge \neg b \wedge c$

$a \wedge b \wedge \neg c$

$a \wedge b \wedge c$

A min term includes every variable exactly once, either negated or unnegated, AND-ed together

Canonical DNF

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**

1. **Find the T rows in the truth table**

2. **For each T row, write an expression that is T in that row but *no others* ("min term")**

$\neg a \wedge \neg b \wedge c$

$\neg a \wedge b \wedge c$

$a \wedge \neg b \wedge c$

$a \wedge b \wedge \neg c$

$a \wedge b \wedge c$

3. **Form the disjunction (OR) of the min terms**

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

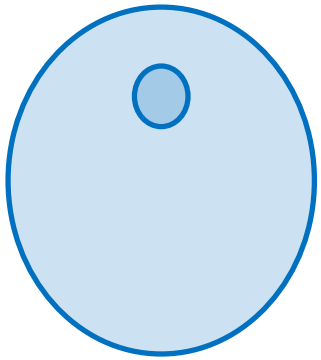
DNF: Canonical and Non

- Stands for "Disjunctive Normal Form"
  - **outermost** operation is disjunction (OR)
  - operands are conjunctions (ANDs) of variables or their negations

$(a \wedge c) \vee (\neg a) \vee (\neg a \wedge \neg b)$ non-canonical DNF

$(a \wedge b \wedge \neg c) \vee (\neg a \wedge b \wedge c) \vee (a \wedge \neg b \wedge \neg c)$ canonical DNF

(every disjunct is a min term)



Canonical CNF

1. Find the F rows in the truth table

a	b	c	F
F	F	F	F
F	F	**T**	**T**
F	**T**	F	F
F	**T**	**T**	**T**
T	F	F	F
T	F	**T**	**T**
T	**T**	F	**T**
T	**T**	**T**	**T**

Canonical CNF

1. Find the F rows in the truth table

2. For each F row, write an expression that is T in *every row but that one* ("max term")

a	b	c	F
F	F	F	F
F	F	T	T
F	T	F	F
F	T	T	T
T	F	F	F
T	F	T	T
T	T	F	T
T	T	T	T

a ∨ b ∨ c

This is only F if a = F, b = F, and c = F
(OR is T if *any* arguments is a T)

Canonical CNF

1. Find the F rows in the truth table

2. For each F row, write an expression that is T in *every row but that one* ("max term")

a	b	c	F
F	F	F	F
F	F	T	T
F	T	F	F
F	T	T	T
T	F	F	F
T	F	T	T
T	T	F	T
T	T	T	T

$a \vee b \vee c$

$a \vee \neg b \vee c$

This is only F if a = F, b = T, and c = F

Canonical CNF

1. Find the F rows in the truth table

2. For each F row, write an expression that is T in *every row but that one* ("max term")

a	b	c	F
F	F	F	F
F	F	T	T
F	T	F	F
F	T	T	T
T	F	F	F
T	F	T	T
T	T	F	T
T	T	T	T

$a \vee b \vee c$

$a \vee \neg b \vee c$

$\neg a \vee b \vee c$

This is only F if a = T, b = F, and c = F

Canonical CNF

a	b	c	F
F	F	F	F
F	F	T	T
F	T	F	F
F	T	T	T
T	F	F	F
T	F	T	T
T	T	F	T
T	T	T	T

1. Find the F rows in the truth table

2. For each F row, write an expression that is T in *every row but that one* ("max term")

$a \vee b \vee c$

$a \vee \neg b \vee c$

$\neg a \vee b \vee c$

3. Form the conjunction (AND) of the max terms

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

CNF: Canonical and Non

- Stands for "Conjunctive Normal Form"
  - **outermost** operation is conjunction (AND)
  - operands (conjuncts) are disjunctions (ORs) of variables or their negations

$(a \vee b \vee \neg c) \wedge (\neg a \vee b \vee c) \wedge (a \vee \neg b \vee \neg c)$ canonical CNF

$(a \vee c) \wedge (\neg a) \wedge (\neg a \vee \neg b)$ non-canonical CNF

Comparing DNF and CNF

	DNF	CNF
operation	disjunction (OR)	conjunction (AND)
operands	conjunctions (ANDs)	disjunctions (ORs)
	(of only variables or their negations)	
canonical iff	all conjunctions are **min** terms	all disjunctions are **max** terms

Comparing Min and Max Terms

- **Min/Max term if *every* variable appears exactly once**

	Min Term	Max Term
operation	conjunction (AND)	disjunction (OR)
	(of every variables or its negation)	
result	only one T row	only one F row

Important Corollary of DNF Construction

$\neg$, $\wedge$, $\vee$ can implement any Boolean function!

no need for anything else

Why? Because this construction only uses $\neg$, $\wedge$, $\vee$

DNF conversion works for any boolean function

Digital Circuits

Application: Digital Circuits

Computing With Logic

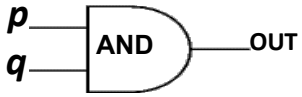
- T corresponds to 1 or "high" voltage
- F corresponds to 0 or "low" voltage

Gates

- Take inputs and produce outputs (functions)
- Several kinds of gates
- Correspond to propositional connectives

AND, OR, NOT Gates

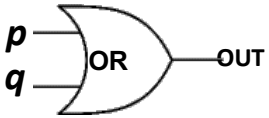
AND Gate



p	*q*	OUT
1	1	1
1	0	0
0	1	0
0	0	0

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

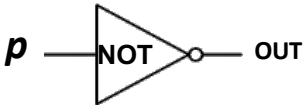
OR Gate



p	*q*	OUT
1	1	1
1	0	1
0	1	1
0	0	0

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

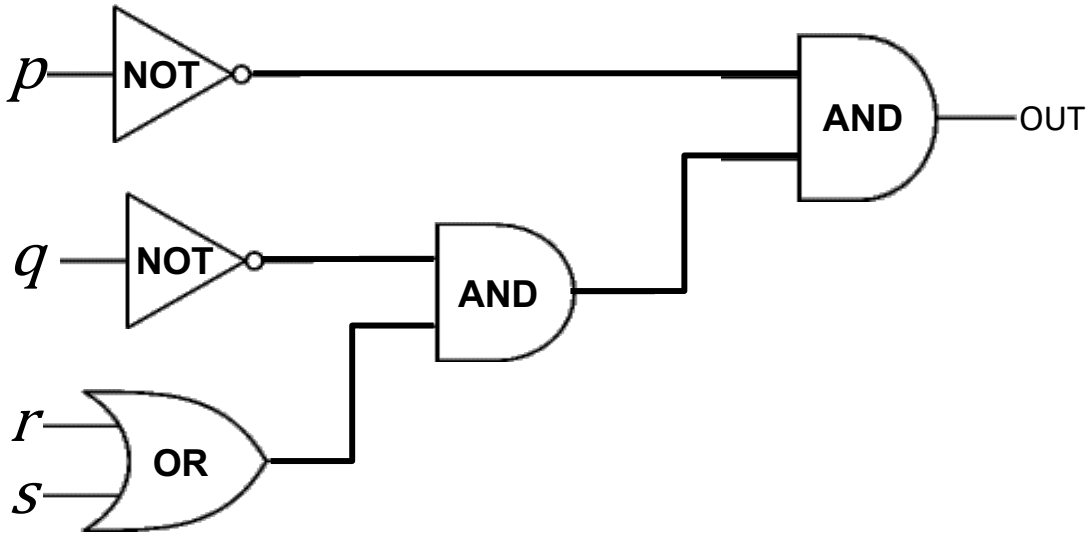
NOT Gate



p	OUT
1	0
0	1

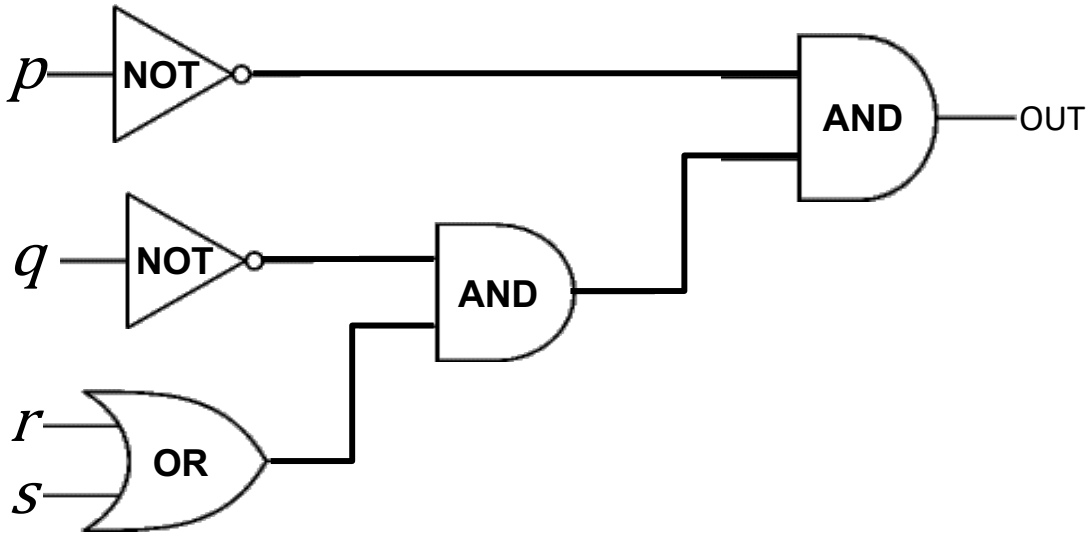
p	$\neg p$
T	F
F	T

Combinational Logic Circuits



Values get sent along wires connecting gates

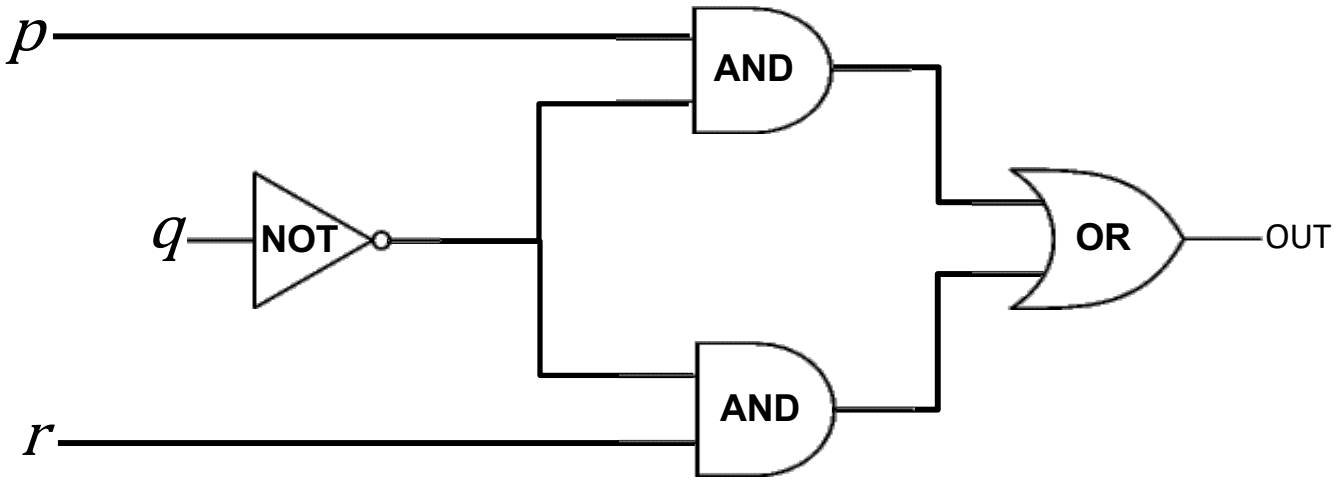
Combinational Logic Circuits



Values get sent along wires connecting gates

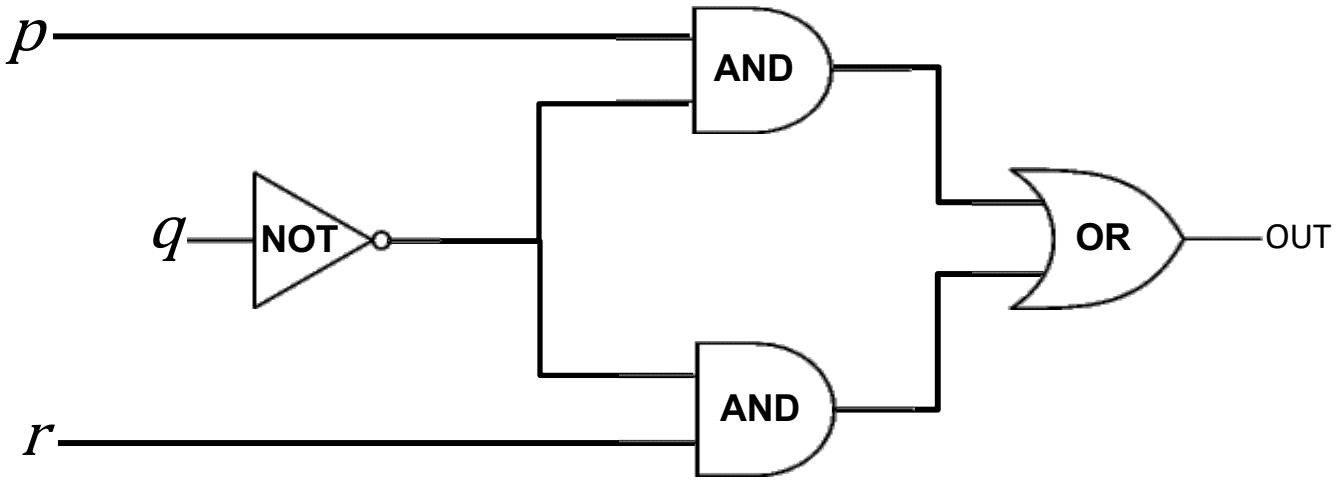
$$\neg p \wedge (\neg q \wedge (r \vee s))$$

Combinational Logic Circuits



Wires can send one value to multiple gates!

Combinational Logic Circuits



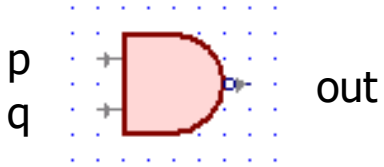
Wires can send one value to multiple gates!

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

Other Useful Gates

NAND

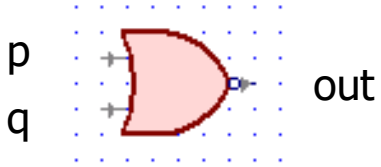
$\neg(p \wedge q)$



p	q	out
0	0	1
0	1	1
1	0	1
1	1	0

NOR

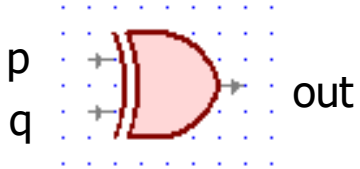
$\neg(p \vee q)$



p	q	out
0	0	1
0	1	0
1	0	0
1	1	0

XOR

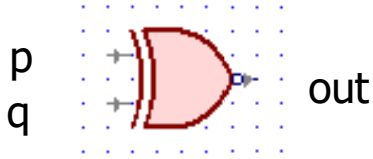
$p \oplus q$



p	q	out
0	0	0
0	1	1
1	0	1
1	1	0

XNOR

$p \leftrightarrow q$

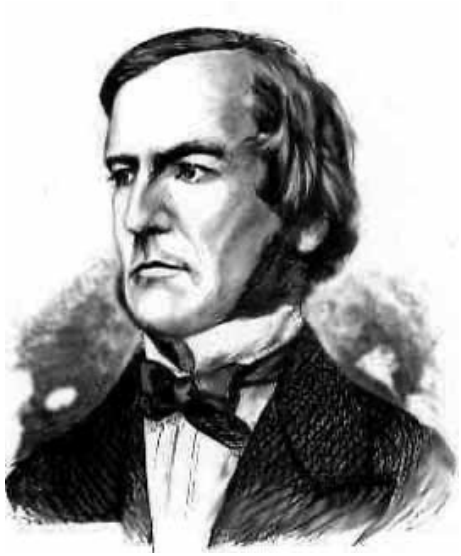


p	q	out
0	0	1
0	1	0
1	0	0
1	1	1

Boolean Algebra

- **Usual notation used in circuit design**

- **Boolean algebra**
  - a set of elements B containing {0, 1}
  - binary operations { + , • }
  - and a unary operation { a' } or { $\bar{a}$ }



Write these in Boolean Algebra:

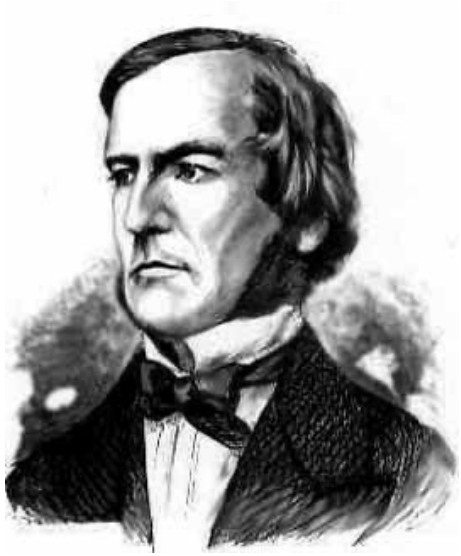
$\neg p \wedge (\neg q \wedge (r \vee s))$

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

Boolean Algebra

- **Usual notation used in circuit design**

- **Boolean algebra**
  - a set of elements B containing {0, 1}
  - binary operations { + , • }
  - and a unary operation { a' } or { $\bar{a}$ }



Write these in Boolean Algebra:

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

$p'q'(r+s)$

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

$pq' + q'r$

A Combinational Logic Example

Sessions of Class:

We would like to compute the number of lectures or quiz sections remaining *at the start* of a given day of the week.

- **Inputs:** Day of the Week, Lecture/Section flag
- **Output:** Number of sessions left

Examples: Input: (Wednesday, Lecture) Output: **2**

Input: (Monday, Section) Output: **1**

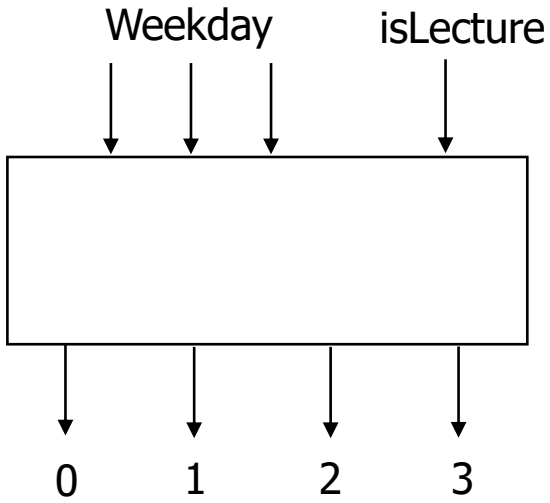
Implementation in Software

```
public int classesLeftInMorning(int weekday, boolean isLecture) {
    switch (weekday) {
        case SUNDAY:
        case MONDAY:
            return isLecture ? 3 : 1;
        case TUESDAY:
        case WEDNESDAY:
            return isLecture ? 2 : 1;
        case THURSDAY:
            return isLecture ? 1 : 1;
        case FRIDAY:
            return isLecture ? 1 : 0;
        case SATURDAY:
            return isLecture ? 0 : 0;
   }
}
```

Implementation with Hardware

Encoding:

- How many bits for each input/output?
- Binary number for weekday
- One bit for each possible output



Defining Our Inputs!

Weekday Input:

- Binary number for weekday
- Sunday = 0, Monday = 1, …
- We care about these in binary:

Weekday	Number	Binary
Sunday	0	000
Monday	1	001
Tuesday	2	010
Wednesday	3	011
Thursday	4	100
Friday	5	101
Saturday	6	110

Converting to a Truth Table!

```
case SUNDAY or MONDAY:
    return isLecture ? 3 : 1;
case TUESDAY or WEDNESDAY:
    return isLecture ? 2 : 1;
case THURSDAY:
    return isLecture ? 1 : 1;
case FRIDAY:
    return isLecture ? 1 : 0;
case SATURDAY:
    return isLecture ? 0 : 0;
```

Weekday		isLecture	c_0	c_1	c_2	c_3
SUN	000	0				
SUN	000	1				
MON	001	0				
MON	001	1				
TUE	010	0				
TUE	010	1				
WED	011	0				
WED	011	1				
THU	100	-				
FRI	101	0				
FRI	101	1				
SAT	110	-				

Converting to a Truth Table!

```
case SUNDAY or MONDAY:
    return isLecture ? 3 : 1;
case TUESDAY or WEDNESDAY:
    return isLecture ? 2 : 1;
case THURSDAY:
    return isLecture ? 1 : 1;
case FRIDAY:
    return isLecture ? 1 : 0;
case SATURDAY:
    return isLecture ? 0 : 0;
```

Weekday		isLecture	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

Let's begin by finding an expression for c_3. To do this, we look at the rows where $c_3 = 1$ (true).

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

DAY == SUN && L == 1

DAY == MON && L == 1

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$d_2d_1d_0$ == 000 && L == 1

$d_2d_1d_0$ == 001 && L == 1

Substituting DAY for the binary representation.

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$d_2 == 0 \;\&\&\; d_1 == 0 \;\&\&\; d_0 == 0 \;\&\&\; L == 1$

$d_2 == 0 \;\&\&\; d_1 == 0 \;\&\&\; d_0 == 1 \;\&\&\; L == 1$

Splitting up the bits of the day;
so, we can write a formula.

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$d_2' \cdot d_1' \cdot d_0' \cdot L$

$d_2' \cdot d_1' \cdot d_0 \cdot L$

Replacing with Boolean Algebra…

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$d_2' \cdot d_1' \cdot d_0' \cdot L$

$d_2' \cdot d_1' \cdot d_0 \cdot L$

How do we combine them?

Truth Table to Logic

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$d_2' \cdot d_1' \cdot d_0' \cdot L$

$d_2' \cdot d_1' \cdot d_0 \cdot L$

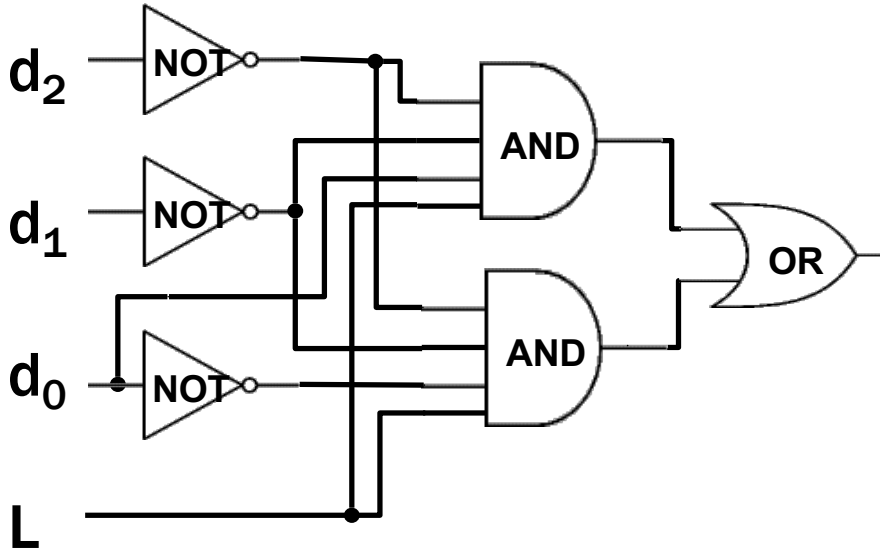
Either situation causes c_3 to be true. So, we "or" them.

$$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$$

Truth Table to Logic

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

Here's c_3 as a circuit:



Truth Table to Logic (Part 2)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

Now, we do c_2.

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

For c_1, let's look at the 0s:

$d_2 + d_1 + d_0 + L'$

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3	
SUN	000	0	0	1	0	0	
SUN	000	1	0	0	0	1	$d_2 + d_1 + d_0 + L'$
MON	001	0	0	1	0	0	
MON	001	1	0	0	0	1	$d_2 + d_1 + d_0' + L'$
TUE	010	0	0	1	0	0	
TUE	010	1	0	0	1	0	$d_2 + d_1' + d_0 + L'$
WED	011	0	0	1	0	0	
WED	011	1	0	0	1	0	$d_2 + d_1' + d_0' + L'$
THU	100	-	0	1	0	0	
FRI	101	0	1	0	0	0	$d_2' + d_1 + d_0' + L$
FRI	101	1	0	1	0	0	
SAT	110	-	1	0	0	0	???

For c_1, let's look at the 0s:

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

For c_1, let's look at the 0s:

- $d_2 + d_1 + d_0 + L'$
- $d_2 + d_1 + d_0' + L'$
- $d_2 + d_1' + d_0 + L'$
- $d_2 + d_1' + d_0' + L'$
- $d_2' + d_1 + d_0' + L$
- $d_2' + d_1' + d_0$

No matter what L is, we always say it's 1.
So, we don't need L in the expression.

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

For c_1, let's look at the 0s:

$d_2 + d_1 + d_0 + L'$

$d_2 + d_1 + d_0' + L'$

$d_2 + d_1' + d_0 + L'$

$d_2 + d_1' + d_0' + L'$

$d_2' + d_1 + d_0' + L$

$d_2' + d_1' + d_0$

How do we combine them?

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

For c_1, let's look at the 0s:

$d_2 + d_1 + d_0 + L'$

$d_2 + d_1 + d_0' + L'$

$d_2 + d_1' + d_0 + L'$

$d_2 + d_1' + d_0' + L'$

$d_2' + d_1 + d_0' + L$

$d_2' + d_1' + d_0$

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')(d_2 + d_1' + d_0 + L')$$
$$(d_2 + d_1' + d_0' + L')(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

Truth Table to Logic (Part 3)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

For c_1, let's look at the 0s:

Is c_1 still in CNF form?

Yes, but not canonical CNF

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')(d_2 + d_1' + d_0 + L')$$
$$(d_2 + d_1' + d_0' + L')(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

Truth Table to Logic (Part 4)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')$$
$$(d_2 + d_1' + d_0 + L')(d_2 + d_1' + d_0' + L')$$
$$(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

$$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$$

$$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$$

Truth Table to Logic (Part 4)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')$$
$$(d_2 + d_1' + d_0 + L')(d_2 + d_1' + d_0' + L')$$
$$(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

$$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$$

$$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$$

Finally, we do c_0:

Truth Table to Logic (Part 4)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')$$
$$(d_2 + d_1' + d_0 + L')(d_2 + d_1' + d_0' + L')$$
$$(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

$$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$$

$$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$$

Finally, we do c_0:

$d_2 \cdot d_1' \cdot d_0 \cdot L'$

$d_2 \cdot d_1 \cdot d_0'$

Truth Table to Logic (Part 4)

	$d_2d_1d_0$	L	c_0	c_1	c_2	c_3
SUN	000	0	0	1	0	0
SUN	000	1	0	0	0	1
MON	001	0	0	1	0	0
MON	001	1	0	0	0	1
TUE	010	0	0	1	0	0
TUE	010	1	0	0	1	0
WED	011	0	0	1	0	0
WED	011	1	0	0	1	0
THU	100	-	0	1	0	0
FRI	101	0	1	0	0	0
FRI	101	1	0	1	0	0
SAT	110	-	1	0	0	0

$$c_1 = (d_2 + d_1 + d_0 + L')(d_2 + d_1 + d_0' + L')$$
$$(d_2 + d_1' + d_0 + L')(d_2 + d_1' + d_0' + L')$$
$$(d_2' + d_1 + d_0' + L)(d_2' + d_1' + d_0)$$

$$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$$

$$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$$

$$c_0 = d_2 \cdot d_1' \cdot d_0 \cdot L' + d_2 \cdot d_1 \cdot d_0'$$

Mapping Truth Tables to Logic Gates

Given a truth table:

1. Write the output in a table
2. Write the Boolean expression
3. Draw as gates

Equivalence

One Application of Equivalence

Given a truth table:

1. Write the output in a table
2. Write the Boolean expression
3. Draw as gates

This will give us *some* circuit.
But is it the best circuit?

Recall: Truth Table to Logic

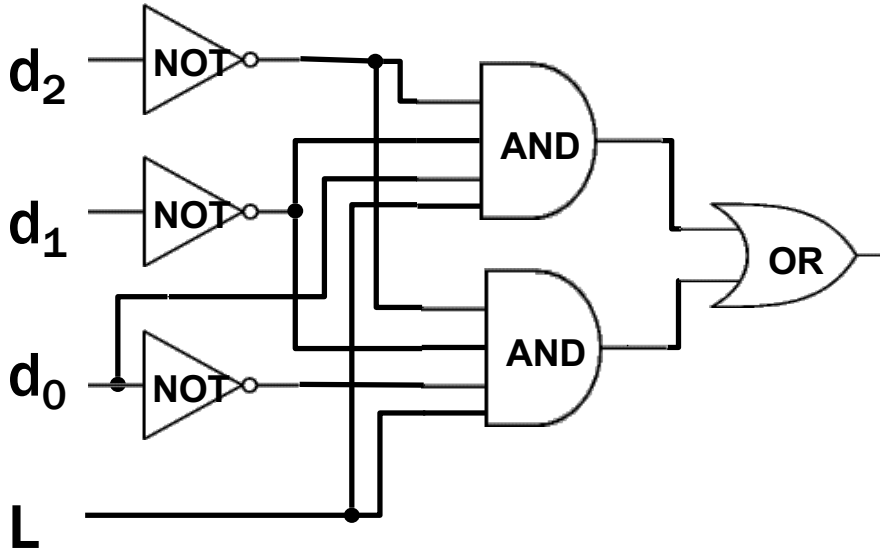
$c_0 = d_2 \cdot d_1' \cdot d_0 \cdot L' + d_2 \cdot d_1 \cdot d_0' + d_2 \cdot d_1 \cdot d_0$

$c_1 = d_2' \cdot d_1' \cdot d_0' \cdot L' + d_2' \cdot d_1' \cdot d_0 \cdot L' + d_2' \cdot d_1 \cdot d_0' \cdot L' + d_2' \cdot d_1 \cdot d_0 \cdot L' + d_2 \cdot d_1' \cdot d_0' + d_2 \cdot d_1' \cdot d_0 \cdot L$

$c_2 = d_2' \cdot d_1 \cdot d_0' \cdot L + d_2' \cdot d_1 \cdot d_0 \cdot L$

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

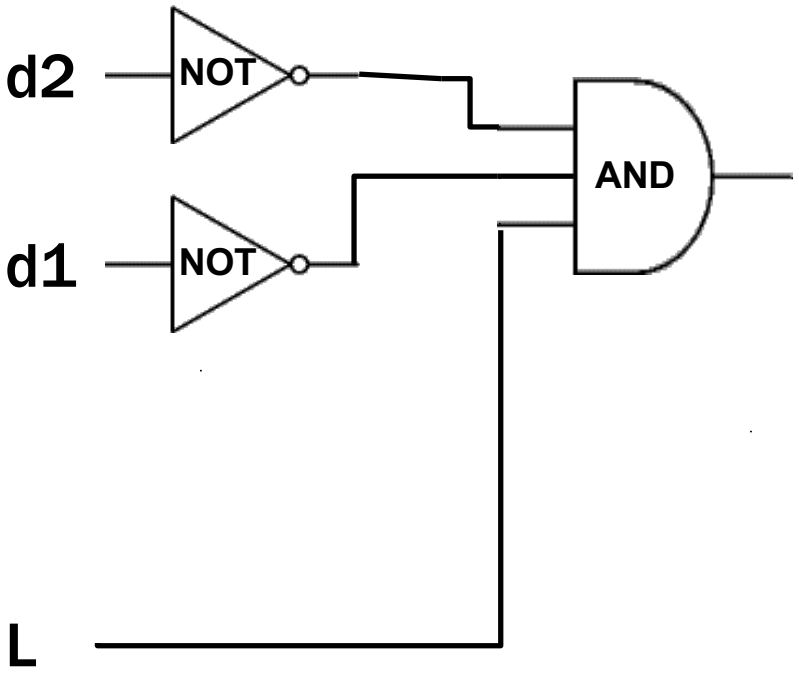
Here's c_3 as a circuit:



Simplifying using Boolean Algebra

$c_3 = d_2' \cdot d_1' \cdot d_0' \cdot L + d_2' \cdot d_1' \cdot d_0 \cdot L$

$c_3 = d_2' \cdot d_1' \cdot L$



Uses of Equivalence

- Hardware applications
  - hardware optimization
  - hardware verification
- Software applications
  - program optimization
  - program verification
  - query optimization and caching
  - artificial intelligence

Tautologies!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

a	*b*	…	*f(a,b)*	*g(a,b)*
T	T	…	**T**	**F**
F	T	…	**T**	**F**
T	F	…	**T**	**F**
F	F	…	**T**	**F**
…	…	…	…	…

Tautologies!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

$p \vee \neg p$

$p \oplus p$

$(p \rightarrow r) \wedge p$

Tautologies!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

$p \vee \neg p$

This is a tautology. It's called the "law of the excluded middle".
If p is true, then $p \vee \neg p$ is true. If p is false, then $p \vee \neg p$ is true.

$p \oplus p$

This is a contradiction. It's always false no matter what truth value p takes on.

$(p \rightarrow r) \wedge p$

This is a contingency. When p=T, r=T, $(T \rightarrow T) \wedge T$ is true.
When p=T, r=F, $(T \rightarrow F) \wedge T$ is false.

Logical Equivalence

A = B means **A** and **B** are the same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

Logical Equivalence

A = B means A and B are the same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

- $p \wedge q \wedge r = (p \wedge q) \wedge r$

  These are equal. The parentheses are *implicit* in the first case.
  (Both describe the same operations applied in the same order)

Logical Equivalence

A = B means A and B are the same thing written twice:

- $p \wedge r = p \wedge r$

  These are equal, because they are character-for-character identical.

- $p \wedge q \wedge r = (p \wedge q) \wedge r$

  These are equal. The parentheses are *implicit* in the first case. (Both describe the same operations applied in the same order)

- $p \wedge r \neq r \wedge p$

  These are NOT equal, because they are different sequences of characters. They "mean" the same thing though.

Logical Equivalence

A = B means A and B are the same thing written twice:

A $\equiv$ B means A and B have identical truth values:

- $p \wedge r \equiv p \wedge r$

  Two formulas that are **equal** also are equivalent.

Logical Equivalence

A = B means **A** and **B** are the same thing written twice:

A ≡ B means **A** and **B** have identical truth values:

- $p \wedge r \equiv p \wedge r$

  Two formulas that are **equal** also are equivalent.

- $p \wedge r \equiv r \wedge p$

  These two formulas have the same truth table!

Logical Equivalence

A = B means A and B are the same thing written twice:

A $\equiv$ B means A and B have identical truth values:

- $p \wedge r \equiv p \wedge r$

  Two formulas that are **equal** also are equivalent.

- $p \wedge r \equiv r \wedge p$

  These two formulas have the same truth table!

- $p \wedge r \not\equiv r \vee p$

  When *p*=T and *r*=F, $p \wedge r$ is false, but $p \vee r$ is true!

$A \leftrightarrow B$ vs. $A \equiv B$

A ↔ B is a **proposition** that may be true or false depending on the truth values of **A** and **B**.

Example: $(p \wedge r) \leftrightarrow (r \vee p)$ has 4 rows in its truth table

A ≡ B is an **assertion** over all possible truth values that **A** and **B** always have the same truth values.

Example: $p \wedge r \equiv r \wedge p$ is true

A ≡ B and **(A ↔ B) ≡ T** have the same meaning as does "**A ↔ B** is a tautology"

Logical Equivalence $A \equiv B$

A ≡ B is an assertion that ***two propositions*** **A** and **B** always have the same truth values.

A ≡ B and **(A ↔ B) ≡ T** have the same meaning.

$p \wedge r \equiv r \wedge p$

p	r	$p \wedge r$	$r \wedge p$	$(p \wedge r) \leftrightarrow (r \wedge p)$
T	T			
T	F			
F	T			
F	F			

Logical Equivalence $A \equiv B$

A ≡ B is an assertion that ***two propositions*** **A** and **B** always have the same truth values.

A ≡ B and **(A ↔ B) ≡ T** have the same meaning.

$p \wedge r \equiv r \wedge p$

p	r	$p \wedge r$	$r \wedge p$	$(p \wedge r) \leftrightarrow (r \wedge p)$
T	T	T	T	T
T	F	F	F	T
F	T	F	F	T
F	F	F	F	T

Familiar Equivalence

Double Negation

$$p \equiv \neg \neg p$$

p	$\neg p$	$\neg \neg p$
T	F	T
F	T	F

De Morgan's Laws

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

Negate the statement:

"My code compiles or there is a bug."

To negate the statement,

ask "when is the original statement false".

It's false when not(my code compiles) AND not(there is a bug).

Translating back into English, we get:

My code doesn't compile and there is not a bug.

De Morgan's Laws

Example: $\neg(p \wedge r) \equiv \neg p \vee \neg r$

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

De Morgan's Laws

Example: $\neg(p \wedge r) \equiv \neg p \vee \neg r$

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

De Morgan's Laws

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

```
if (!(front != null && value > front.data)) {
    front = new ListNode(value, front);
} else {
    ListNode current = front;
    while (current.next != null && current.next.data < value))
        current = current.next;
    current.next = new ListNode(value, current.next);
}
```

De Morgan's Laws

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

```
!(front != null && value > front.data)
```

$\equiv$

```
front == null || value <= front.data
```

Law of Implication

$$p \rightarrow r \equiv \neg p \vee r$$

p	r	$p \rightarrow r$	$\neg p$	$\neg p \vee r$
T	T			
T	F			
F	T			
F	F			

Law of Implication

$$p \rightarrow r \equiv \neg p \vee r$$

p	r	$p \rightarrow r$	$\neg p$	$\neg p \vee r$
T	T	T	F	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

Biconditional: $p \leftrightarrow r$

- *p* if and only if *r* *(p* iff *r)*
- *p* implies *r* and *r* implies *p*
- *p* is necessary and sufficient for *r*

p	r	$p \leftrightarrow r$	$p \rightarrow r$	$r \rightarrow p$	$(p \rightarrow r) \wedge (r \rightarrow p)$
T	T	T	T	T	
T	F	F	F	T	
F	T	F	T	F	
F	F	T	T	T	

Biconditional: $p \leftrightarrow r$

- *p* if and only if *r* *(p* iff *r)*
- *p* implies *r* and *r* implies *p*
- *p* is necessary and sufficient for *r*

p	r	$p \leftrightarrow r$	$p \rightarrow r$	$r \rightarrow p$	$(p \rightarrow r) \wedge (r \rightarrow p)$
T	T	T	T	T	T
T	F	F	F	T	F
F	T	F	T	F	F
F	F	T	T	T	T

Some Familiar Properties of Arithmetic

- $x + y = y + x$ **(Commutativity)**
- $x \cdot (y + z) = x \cdot y + x \cdot z$ **(Distributivity)**
- $(x + y) + z = x + (y + z)$ **(Associativity)**

Important Equivalences

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- Associative
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- Negation
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

Some Familiar Properties of Arithmetic

- $x \cdot 1 = x$ **(Identity)**
- $x + 0 = x$

- $x \cdot 0 = 0$ **(Domination)**

Important Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

Some Familiar Properties of Arithmetic

- Usual properties hold under relabeling:
  - 0, 1 becomes F, T
  - "+" becomes "$\vee$"
  - " $\cdot$ " becomes "$\wedge$"
- But there are some new facts:
  - Distributivity works for both "$\wedge$" and "$\vee$"
  - Domination works with T
- There are some other facts specific to logic…

Important Equivalences

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- Associative
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- Negation
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

Important Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

Using Equivalences

- Note that p, q, and r can be **any** propositions (not just atomic propositions)

- Ex: $(r \rightarrow s) \wedge (\neg t) \equiv (\neg t) \wedge (r \rightarrow s)$

  – apply commutativity: $p \wedge q \equiv q \wedge p$
  with $p := r \rightarrow s$
  and $q := \neg t$

Computing Equivalence

Given two propositions, can we write an algorithm to determine if they are equivalent?

What is the runtime of our algorithm?

Computing Equivalence

Given two propositions, can we write an algorithm to determine if they are equivalent?

Yes! Generate the truth tables for both propositions and check if they are the same for every entry.

What is the runtime of our algorithm?

Every atomic proposition has two possibilities (T, F). If there are n atomic propositions, there are 2^n rows in the truth table.

Another approach: Equivalence Chains

To show A is equivalent to B

– Apply a series of logical equivalences to sub-expressions to convert A to B

To show A is a tautology

– Apply a series of logical equivalences to sub-expressions to convert A to T

Another approach: Equivalence Chains

To show A is equivalent to B

– Apply a series of logical equivalences to sub-expressions to convert A to B

Example:

Let A be "$p \vee (p \wedge p)$", and B be "p".

Our general equivalence proof looks like:

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

Another approach: Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Example:

Let A be "$p \vee (p \wedge p)$", and B be "p".

Our general equivalence proof looks like:

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

Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Example:

Let A be "$p \vee (p \wedge p)$", and B be "p".

Our general equivalence proof looks like:

$$p \vee (p \wedge p) \equiv p \vee p \qquad \text{Idempotent}$$
$$\equiv p \qquad \text{Idempotent}$$

Logical Equivalences

To show A is a tautology

– Apply a series of logical equivalences to sub-expressions to convert A to T

Example:

Let A be "$\neg p \vee (p \vee p)$".

Our general equivalence proof looks like:

$$\begin{aligned} \neg p \vee (p \vee p) &\equiv \\ &\equiv \\ &\equiv \mathbf{T} \end{aligned}$$

Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Example:

Let A be "$\neg p \vee (p \vee p)$".

Our general equivalence proof looks like:

$$\neg p \vee (p \vee p) \equiv$$
$$\equiv$$
$$\equiv \mathrm{T}$$

Logical Equivalences

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Example:

Let A be "$\neg p \vee (p \vee p)$".

Our general equivalence proof looks like:

$$\neg p \vee (p \vee p) \equiv \neg p \vee p \quad \text{Idempotent}$$
$$\equiv p \vee \neg p \quad \text{Commutative}$$
$$\equiv \mathbf{T} \quad \text{Negation}$$

Prove these propositions are equivalent: Option 1

Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

Make a Truth Table and show:

$$(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r) \equiv \mathbf{T}$$

p	r	$p \rightarrow r$	$(p \wedge (p \rightarrow r))$	$p \wedge r$	$(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r)$
T	T				
T	F				
F	T				
F	F				

Prove these propositions are equivalent: Option 1

Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

Make a Truth Table and show:

$$(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r) \equiv \mathbf{T}$$

p	r	$p \rightarrow r$	$(p \wedge (p \rightarrow r))$	$p \wedge r$	$(p \wedge (p \rightarrow r)) \leftrightarrow (p \wedge r)$
T	T	T	T	T	T
T	F	F	F	F	T
F	T	T	F	F	T
F	F	T	F	F	T

Prove these propositions are equivalent: Option 2

Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

$$
\begin{aligned}
p \wedge (p \rightarrow r) &\equiv \\
&\equiv \\
&\equiv \\
&\equiv \\
&\equiv p \wedge r
\end{aligned}
$$

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Prove these propositions are equivalent: Option 2

Prove: $p \wedge (p \rightarrow r) \equiv p \wedge r$

$$
\begin{aligned}
p \wedge (p \rightarrow r) &\equiv p \wedge (\neg p \vee r) && \text{Law of Implication} \\
&\equiv (p \wedge \neg p) \vee (p \wedge r) && \text{Distributive} \\
&\equiv \mathbf{F} \vee (p \wedge r) && \text{Negation} \\
&\equiv (p \wedge r) \vee \mathbf{F} && \text{Commutative} \\
&\equiv p \wedge r && \text{Identity}
\end{aligned}
$$

- **Identity**
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- **Domination**
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- **Idempotent**
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- **Commutative**
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$
- **Associative**
  - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
  - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- **Distributive**
  - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
  - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- **Absorption**
  - $p \vee (p \wedge q) \equiv p$
  - $p \wedge (p \vee q) \equiv p$
- **Negation**
  - $p \vee \neg p \equiv \mathrm{T}$
  - $p \wedge \neg p \equiv \mathrm{F}$

De Morgan's Laws

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

Law of Implication

$$p \rightarrow q \equiv \neg p \vee q$$

Contrapositive

$$p \rightarrow q \equiv \neg q \rightarrow \neg p$$

Biconditional

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

Double Negation

$$p \equiv \neg \neg p$$

Prove this is a Tautology: Option 1

$$(p \wedge r) \rightarrow (r \vee p)$$

Make a Truth Table and show:

$$(p \wedge r) \rightarrow (r \vee p) \equiv \mathbf{T}$$

p	r	$p \wedge r$	$r \vee p$	$(p \wedge r) \rightarrow (r \vee p)$
T	T			
T	F			
F	T			
F	F			

Prove this is a Tautology: Option 1

$$(p \wedge r) \rightarrow (r \vee p)$$

Make a Truth Table and show:

$$(p \wedge r) \rightarrow (r \vee p) \equiv \mathbf{T}$$

p	r	$p \wedge r$	$r \vee p$	$(p \wedge r) \rightarrow (r \vee p)$
T	T	T	T	T
T	F	F	T	T
F	T	F	T	T
F	F	F	F	T

Prove this is a Tautology: Option 2

$$(p \land r) \rightarrow (r \lor p)$$

Use a series of equivalences like so:

$$(p \land r) \rightarrow (r \lor p) \equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv$$
$$\equiv \mathrm{T}$$

Associative
- $(p \lor q) \lor r \equiv p \lor (q \lor r)$
- $(p \land q) \land r \equiv p \land (q \land r)$

Distributive
- $p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
- $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$

Absorption
- $p \lor (p \land q) \equiv p$
- $p \land (p \lor q) \equiv p$

Negation
- $p \lor \neg p \equiv \mathrm{T}$
- $p \land \neg p \equiv \mathrm{F}$

Identity
- $p \land \mathrm{T} \equiv p$
- $p \lor \mathrm{F} \equiv p$

Domination
- $p \lor \mathrm{T} \equiv \mathrm{T}$
- $p \land \mathrm{F} \equiv \mathrm{F}$

Idempotent
- $p \lor p \equiv p$
- $p \land p \equiv p$

Commutative
- $p \lor q \equiv q \lor p$
- $p \land q \equiv q \land p$

Prove this is a Tautology: Option 2

$$(p \wedge r) \rightarrow (r \vee p)$$

Use a series of equivalences like so:

$(p \wedge r) \rightarrow (r \vee p)$	$\equiv \neg(p \wedge r) \vee (r \vee p)$	Law of Implication
	$\equiv (\neg p \vee \neg r) \vee (r \vee p)$	De Morgan
	$\equiv \neg p \vee (\neg r \vee (r \vee p))$	Associative
	$\equiv \neg p \vee ((\neg r \vee r) \vee p)$	Associative
	$\equiv \neg p \vee (p \vee (\neg r \vee r))$	Commutative
	$\equiv (\neg p \vee p) \vee (\neg r \vee r)$	Associative
	$\equiv (p \vee \neg p) \vee (r \vee \neg r)$	Commutative (twice)
	$\equiv \mathbf{T} \vee \mathbf{T}$	Negation (twice)
	$\equiv \mathbf{T}$	Domination/Identity

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

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

Absorption
- $p \vee (p \wedge q) \equiv p$
- $p \wedge (p \vee q) \equiv p$

Negation
- $p \vee \neg p \equiv \mathrm{T}$
- $p \wedge \neg p \equiv \mathrm{F}$

Identity
- $p \wedge \mathrm{T} \equiv p$
- $p \vee \mathrm{F} \equiv p$

Domination
- $p \vee \mathrm{T} \equiv \mathrm{T}$
- $p \wedge \mathrm{F} \equiv \mathrm{F}$

Idempotent
- $p \vee p \equiv p$
- $p \wedge p \equiv p$

Commutative
- $p \vee q \equiv q \vee p$
- $p \wedge q \equiv q \wedge p$

Chains of Equivalence/Tautology

- Not smaller than truth tables when there are only a few propositional variables…
- …but usually ***much shorter*** than truth table proofs when there are many propositional variables
- A big advantage will be that we can extend them to a more in-depth understanding of logic for which truth tables don't apply.

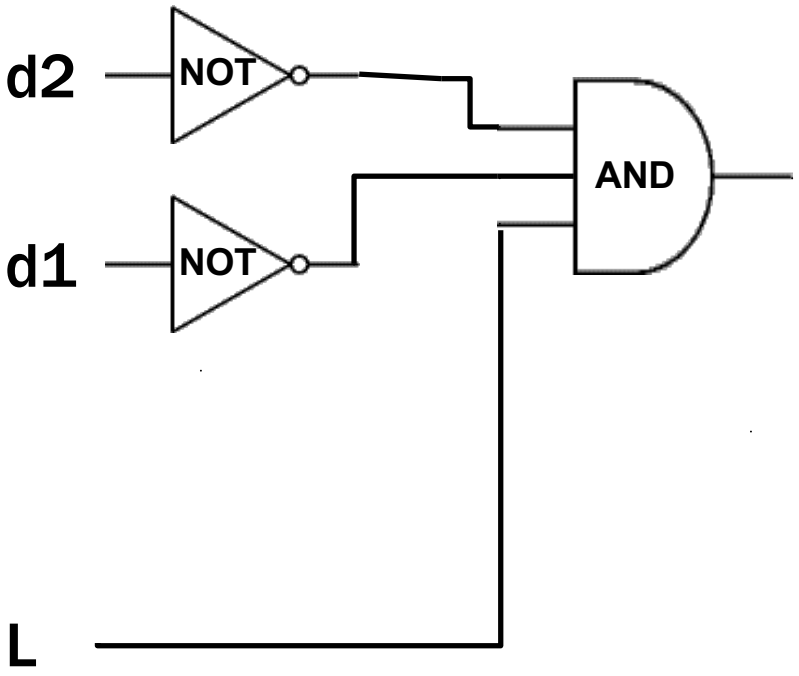
Simplifying using Boolean Algebra

$c3 = d2' \cdot d1' \cdot d0' \cdot L + d2' \cdot d1' \cdot d0 \cdot L$

$= d2' \cdot d1' \cdot (d0' + d0) \cdot L$ Distributivity

$= d2' \cdot d1' \cdot 1 \cdot L$ Negation

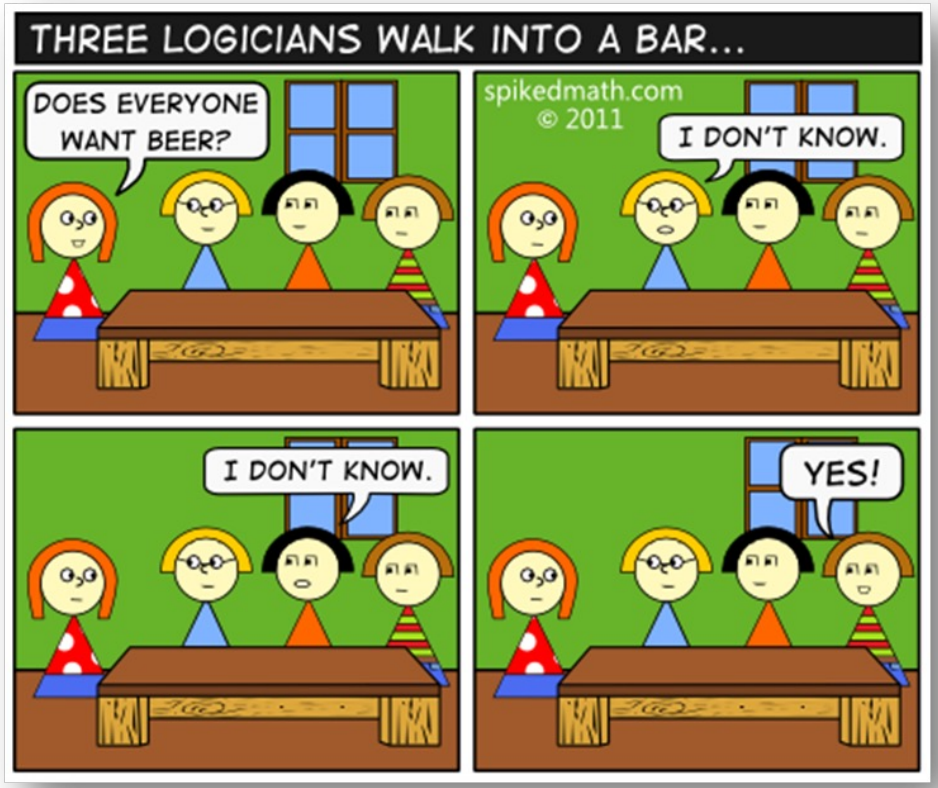
$= d2' \cdot d1' \cdot L$ Identity



In Boolean Algebra, we skip Associativity, Commutativity, and Identity steps

In Boolean Algebra, write "=" instead of "≡"

Predicate Logic



Predicate Logic

- **Propositional Logic**
  - Allows us to analyze complex propositions in terms of their simpler constituent parts (a.k.a. atomic propositions) joined by connectives

- **Predicate Logic**
  - Lets us express how truth values depend on the objects they are talking about

"All positive integers x, y, and z satisfy $x^3 + y^3 \neq z^3$."

Predicate Logic

Adds two key notions to propositional logic

- Predicates
- Quantifiers

Predicates

Predicate

– **A function that returns a truth value, e.g.,**

Cat(x) := “x is a cat”

Prime(x) := “x is prime”

GreaterThan5(x) := “x > 5”

HasTaken(x, y) := “student x has taken course y”

LessThan(x, y) := “x < y”

HasNChars(s, n) := “string s has length n”

Sum(x, y, z) := “x + y = z”

Predicates can have varying numbers of arguments and input types.

Domain of Discourse

For ease of use, we define one "type"/"domain" that we work over. This non-empty set of objects is called the "**domain of discourse**".

For each of the following, what might the domain be?

(1) "x is a cat", "x barks", "x ruined my couch"

"mammals" or "sentient beings" or "cats and dogs" or …

(2) "x is prime", "x = 0", "x > 0", "x is a power of two"

"numbers" or "integers" or "non-negative integers" or …

(3) "student x has taken course y" "x is a pre-req for z"

"students and courses" or "university entities" or …

Quantifiers

We use ***quantifiers*** to talk about collections of objects.

$\forall x\ P(x)$

$P(x)$ is true for every x in the domain

read as "for all x, P of x"

$\exists x\ P(x)$

There is an x in the domain for which $P(x)$ is true

read as "there exists x, P of x"

Truth Depends on the Domain of Discourse

- Example: "Everybody was Kung Fu fighting"
  - **true** if the domain is people in the arena
  - **false** if the domain is the whole world

Quantifiers

We use ***quantifiers*** to talk about collections of objects.

Universal Quantifier ("for all"): $\forall x\ P(x)$

P(x) is true for **every** x in the domain

read as "**for all x, P of x**"

Examples: Are these true?

- $\forall x$ Odd(x)
- $\forall x$ LessThan4(x)

Quantifiers

We use ***quantifiers*** to talk about collections of objects.

Universal Quantifier ("for all"): $\forall x\ P(x)$

P(x) is true for **every** x in the domain

read as "**for all x, P of x**"

Examples:

Are these true? It depends on the domain. For example:

- $\forall x$ Odd(x)
- $\forall x$ LessThan4(x)

{1, 3, -1, -27}	Integers	Odd Integers
True	False	True
True	False	False

Quantifiers

We use ***quantifiers*** to talk about collections of objects.

Existential Quantifier ("exists"): $\exists x\ P(x)$

There is an x in the domain for which $P(x)$ is true

read as "there exists x, P of x"

Examples:

Are these true? It depends on the domain. For example:

- $\exists x$ Odd(x)
- $\exists x$ LessThan4(x)

{1, 3, -1, -27}	Integers	Positive Multiples of 5
True	True	True
True	True	False

Statements with Quantifiers

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Determine the truth values of each of these statements:

$\exists x$ Even(x)	**T**	**e.g. 2, 4, 6, …**
$\forall x$ Odd(x)	**F**	**e.g. 2, 4, 6, …**
$\forall x$ (Even(x) $\vee$ Odd(x))	**T**	**every integer is either even or odd**
$\exists x$ (Even(x) $\wedge$ Odd(x))	**F**	**no integer is both even and odd**
$\forall x$ Greater(x+1, x)	**T**	**adding 1 makes a bigger number**
$\exists x$ (Even(x) $\wedge$ Prime(x))	**T**	**Even(2) is true and Prime(2) is true**

Syntax of Quantifiers

		Precedence
Negation (not)	$\neg p$	highest
For all	$\forall x\ P(x)$	
Exists	$\exists x\ P(x)$	
Conjunction (and)	$p \wedge q$	
Disjunction (or)	$p \vee q$	
Exclusive Or	$p \oplus q$	
Implication	$p \longrightarrow r$	
Biconditional	$p \longleftrightarrow q$	lowest

$\forall x\ \neg P(x) \wedge Q(y)$ **means** $(\forall x\ \neg P(x)) \wedge Q(y)$

Syntax of Quantifiers

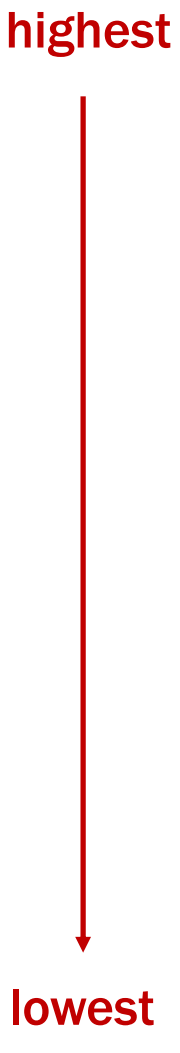
Negation (not)	$\neg p$
For all	$\forall x\, P(x)$
Exists	$\exists x\, P(x)$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$

Not everyone uses this convention!

We will try to accommodate both approaches...

Syntax of Quantifiers (Two Conventions)

Negation (not)	$\neg p$
For all	$\forall x\ P(x)$
Exists	$\exists x\ P(x)$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$
For all	$\forall x, P(x)$
Exists	$\exists x, P(x)$



Syntax of Quantifiers (Two Conventions)

Negation (not)	$\neg p$
For all	$\forall x\ P(x)$
Exists	$\exists x\ P(x)$
Conjunction (and)	$p \wedge q$
Disjunction (or)	$p \vee q$
Exclusive Or	$p \oplus q$
Implication	$p \longrightarrow r$
Biconditional	$p \longleftrightarrow q$
For all	$\forall x, P(x)$
Exists	$\exists x, P(x)$

$\forall x, \neg P(x) \wedge Q(y)$

means

$\forall x\ (\neg P(x) \wedge Q(y))$

Statements with Quantifiers (Literal Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := "x is even"	Greater(x, y) := "x > y"
Odd(x) := "x is odd"	Equal(x, y) := "x = y"
Prime(x) := "x is prime"	Sum(x, y, z) := "x + y = z"

Translate the following statements to English

$\forall x \, \exists y$ Greater(y, x)

For every positive integer x, there is a positive integer y, such that y > x.

$\exists y \, \forall x$ Greater(y, x)

There is a positive integer y such that, for every pos. int. x, we have y > x.

$\forall x \, \exists y$ (Prime(y) $\wedge$ Greater(y, x))

For every positive integer x, there is a pos. int. y such that y is prime and y > x.

$\forall x$ (Prime(x) $\rightarrow$ (Equal(x, 2) $\vee$ Odd(x)))

For each positive integer x, if x is prime, then x = 2 or x is odd.

$\exists x \, \exists y$ (Prime(x) $\wedge$ Prime(y) $\wedge$ Sum(x, 2, y))

There exist positive integers x and y such that x and y are prime and x + 2 = y.

Statements with Quantifiers (Literal Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Translate the following statements to English

$\forall x \exists y$ Greater(y, x)

For every positive integer x, there is a positive integer y, such that y > x.

$\exists y \forall x$ Greater(y, x)

There is a positive integer y such that, for every pos. int. x, we have y > x.

$\forall x \exists y$ (Prime(y) $\wedge$ Greater(y, x))

For every positive integer x, there is a pos. int. y such that y is prime and y > x.

Sound more natural without introducing variable names…

Statements with Quantifiers (Natural Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Translate the following statements to English

$\forall x \, \exists y$ Greater(y, x)

For every positive integer, there is some larger positive integer.

$\exists y \, \forall x$ Greater(y, x)

There is a positive integer that is larger than every positive integer.

$\forall x \, \exists y$ (Prime(y) $\wedge$ Greater(y, x))

For every positive integer, there is a prime that is larger.

Sound more natural without introducing variable names

Statements with Quantifiers (Literal Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Translate the following statements to English

$\exists x \, \exists y \, (\text{Prime}(x) \wedge \text{Prime}(y) \wedge \text{Sum}(x, 2, y))$

There exist positive integers x and y such that x and y are prime and x + 2 = y.

Spot the domain restriction patterns

Statements with Quantifiers (Natural Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Translate the following statements to English

$\exists x\, \exists y\, (\text{Prime}(x) \wedge \text{Prime}(y) \wedge \text{Sum}(x, 2, y))$

There exist primes x and y such that x + 2 = y.

$\forall x\, (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \vee \text{Odd}(x)))$

For each positive integer x, if x is prime, then x = 2 or x is odd.

Spot the domain restriction patterns

Statements with Quantifiers (Natural Translations)

Domain of Discourse
Positive Integers

Predicate Definitions	
Even(x) := “x is even”	Greater(x, y) := “x > y”
Odd(x) := “x is odd”	Equal(x, y) := “x = y”
Prime(x) := “x is prime”	Sum(x, y, z) := “x + y = z”

Translate the following statements to English

$\exists x \, \exists y \, (\text{Prime}(x) \wedge \text{Prime}(y) \wedge \text{Sum}(x, 2, y))$

There exist primes x and y such that x + 2 = y.

$\forall x \, (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \vee \text{Odd}(x)))$

Every prime number is either 2 or odd.

Spot the domain restriction patterns

English to Predicate Logic

Domain of Discourse
Mammals

Predicate Definitions
Cat(x) := “x is a cat”
Red(x) := “x is red”
LikesTofu(x) := “x likes tofu”

“All red cats like tofu”

$$\forall x ((\text{Red}(x) \wedge \text{Cat}(x)) \rightarrow \text{LikesTofu}(x))$$

“Some red cats don’t like tofu”

$$\exists y ((\text{Red}(y) \wedge \text{Cat}(y)) \wedge \neg \text{LikesTofu}(y))$$

English to Predicate Logic

Domain of Discourse
Mammals

Predicate Definitions
Cat(x) := “x is a cat”
Red(x) := “x is red”
LikesTofu(x) := “x likes tofu”

When putting two predicates together like this, we use an “and”.

“All Red cats like tofu”

When restricting to a smaller domain in a “for all” we use **implication.**

“Some red cats don’t like tofu”

When restricting to a smaller domain in an “exists” we use **and.**

“Some” means “there exists”.

English to Predicate Logic

Domain of Discourse
Mammals

Predicate Definitions
Cat(x) := “x is a cat”
Red(x) := “x is red”
LikesTofu(x) := “x likes tofu”

“All Red cats like tofu”

“Red cats like tofu”

When there’s no leading quantification, it *usually* means “for all”.

“Some red cats don’t like tofu”

“A red cat doesn’t like tofu”

“A” means “there exists”.

Statements with Quantifiers (Natural Translations)

Translations usually sound more natural if we

1. Notice "domain restriction" patterns

$\forall x\ (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \vee \text{Odd}(x)))$

Every prime number is either 2 or odd.

2. Avoid introducing *unnecessary* variable names

$\forall x\ \exists y\ \text{Greater}(y, x)$

For every positive integer, there is some larger positive integer.

3. Can sometimes drop "all" or "there is"

$\neg\ \exists x\ (\text{Even}(x) \wedge \text{Prime}(x) \wedge \text{Greater}(x, 2))$

No even prime is greater than 2.

More English Ambiguity

Implicit quantifiers in English are often **ambiguous**

Three people that are all friends can form a raiding party ∀

Three people that I know were all friends with Paul Allen ∃

Formal logic removes this ambiguity

- quantifiers can always be specified
- unquantified variables that are not known constants (e.g, π) are **implicitly** ∀–quantified (mostly… one special case coming later)

Quantifiers in Java

- **Implementing quantifiers in Java…**

```
boolean forAll(Map<Integer, Boolean> P) {
  for (Integer x : P.keySet()) {
    if (!P.get(x)) return false;
  }
  return true;
}
```

$\forall x\ P(x)$

(Bound) variable names don't matter: $\forall x\ P(x) \equiv \forall a\ P(a)$

```
boolean exists(Map<Integer, Boolean> P) {
  for (Integer x : P.keySet()) {
    if (P.get(x)) return true;
  }
  return false;
}
```

$\exists x\ P(x)$

Scope of Quantifiers

Domain of Discourse
{1, 2, 3, …, 100}

Example: $\exists$ x Greater (x, y) $\equiv$ $\exists$ z Greater (z, y)

truth value:

doesn't depend on x or z "bound variables"

does depend on y "free variable"

Scope of Quantifiers

Domain of Discourse
{1, 2, 3, …, 100}

Example: $\exists$ x Greater (x, y) $\equiv$ $\exists$ z Greater (z, y)

truth value:

doesn't depend on x or z "bound variables"

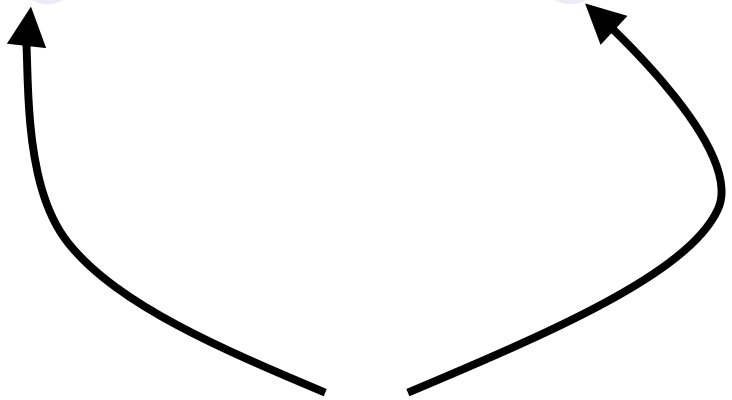
does depend on y "free variable"

quantifiers only act on free variables of the formula

$$\forall x \ \exists y \ (P(x,y) \rightarrow \forall x \ Q(y, x)))$$

Quantifier “Style”

$$\forall x\,(\exists y\,(P(x,y) \wedge \forall x\,Q(y, x)))$$



This isn’t “wrong”, it’s just horrible style.
Don’t confuse your reader by using the same variable multiple times…there are a lot of letters…

Scope of Quantifiers

$\exists x \ (P(x) \wedge Q(x))$ **vs.** $(\exists x \ P(x)) \wedge (\exists x \ Q(x))$

This one asserts P and Q of the *same* x.

This one asserts P and Q of potentially different x's.

Variables with the same name do not necessarily refer to the same object.

Nested Quantifiers

- **Bound variable names don't matter**

$$\forall x\, \exists y\, P(x, y) \equiv \forall a\, \exists b\, P(a, b)$$

- **Positions of quantifiers can <u>sometimes</u> change**

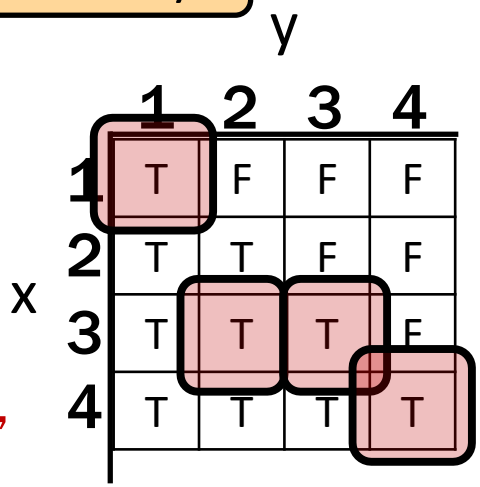
$$\forall x\, (Q(x) \wedge \exists y\, P(x, y)) \equiv \forall x\, \exists y\, (Q(x) \wedge P(x, y))$$

- **But: order is important...**

Quantifier Order Can Matter

Domain of Discourse
{1, 2, 3, 4}

Predicate Definitions
GreaterEq(x, y) ::= “x ≥ y”



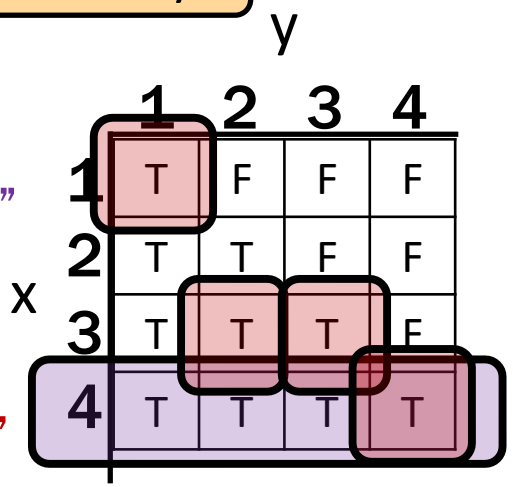
“Every number has a number greater than or equal to it.”

$\forall y \, \exists x$ GreaterEq(x, y)

Quantifier Order Can Matter

Domain of Discourse
{1, 2, 3, 4}

Predicate Definitions
GreaterEq(x, y) ::= “x ≥ y”



“There is a number greater than or equal to all numbers.”

$$\exists x \, \forall y \; GreaterEq(x, y)$$

“Every number has a number greater than or equal to it.”

$$\forall y \, \exists x \; GreaterEq(x, y)$$

The purple statement requires **an entire row** to be true.
The red statement requires one entry in **each column** to be true.

Important: both include the case x = y

Different names does not imply different objects!

Quantification with Two Variables

expression	when true	when false
$\forall x \forall y P(x, y)$	Every pair is true.	At least one pair is false.
$\exists x \exists y P(x, y)$	At least one pair is true.	All pairs are false.
$\forall x \exists y P(x, y)$	We can find a specific y for each x. $(x_1, y_1), (x_2, y_2), (x_3, y_3)$	Some x doesn't have a corresponding y.
$\exists y \forall x P(x, y)$	We can find ONE y that works no matter what x is. $(x_1, y), (x_2, y), (x_3, y)$	For any candidate y, there is an x that it doesn't work for.

De Morgan's Laws for Quantifiers

$$\neg \forall x\ P(x) \equiv \exists x \neg P(x)$$
$$\neg \exists x\ P(x) \equiv \forall x \neg P(x)$$

There is no unicorn $\neg \exists x\ \text{Unicorn}(x)$

Every animal is not a unicorn $\forall x \neg \text{Unicorn}(x)$

These are **equivalent** but not **equal**

De Morgan's Laws for Quantifiers

Eash to check that

$$\neg\, \exists x\, (P(x) \wedge R(x)) \equiv \forall x\, (P(x) \rightarrow \neg\, R(x))$$

and similarly that

$$\neg \forall x\, (P(x) \rightarrow R(x)) \equiv\ \exists x\, (P(x) \wedge \neg\, R(x))$$

De Morgan's Laws respect domain restrictions!
(It leaves them in place and only negates the other parts.)