Equivalences:

Often when proving statements we need to replace claims with others. The purpose can be varied but a common one is to make claims easier to prove.

In order to replace claims they must have the same truth values.

Def: A compound proposition that is always true no matter the truth values of the varibles is a tautology.

Ex: Can you think of a tautology?

anything of the form $p \vee \neg p$.

Def The compound propositions p & q are logically equivalent if $p \leftrightarrow q$ is a tautology.

The notation $p \equiv q$ is sometimes used.

Ex: De Morgan's Laws allow us to work with negations.

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

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

we'll prove the first, the second is Hmwk!

We must show they have the same truth values!

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

Thus $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ is a tautology

Ex: Show $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ are logically equivalent

Table:

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

This is one of the distributive laws.

In a Logic course we would spend much more time learning all the laws & proving them, but here the important part is knowing that there are laws & how to prove logical equivalences.

Using De Morgan's laws:

We can use these laws on English sentences as well:

Negate the sentence: I will have my cake and eat it too.

This is of the form $p \wedge q$ De Morgan's law says

$$\neg(p \wedge q) \overset{?}{=} (\neg p \vee \neg q)$$

So the negation is "I will not have cake or I won't eat it"

SATisfiability: Now a brief digression onto hard problems.

You may have heard of the classic problem P vs NP. We're gonna explain that real quick. This is dealing with algorithm complexity (we'll touch on this a bit later). P = a class of problem which have solutions that can be executed in polynomial time, based on inputs.

Ex: Adding numbers. if a, b are n-digit numbers it will take ~ n operations to add them,

polynomial in n.

NP = class of problems whose solution can be verified in polynomial time

NP = non-deterministic polynomial

This has nothing to do with finding a solution just verifying one.

Ex: Imagine a very long and complicated compound proposition

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

Finding a solution could be difficult, but if I provide values for each variable

Checking that the equation is Satisfied is quick.

that is NP.

Clearly $P \subseteq NP$ basically P is inside NP why? Ignore given solution (ignore this)

Solve it using poly alg, check if equal.

but $NP \subseteq P$? unknown. We'd need to show every polynomially verifiable problem can also be solved in polynomial time - a LOT of work.

Enter NP-Complete. - a class of problems that are in NP and any other NP problem can be transformed into it in polynomial time.

Exs are outside scope of this class (talk during break or OH)

But if we can show any single NP-complete problem is in P

Then all of NP is! $NP \xrightarrow{p\ time} NP\ \text{Complete} \xrightarrow{p\ time} \text{Solution}$

Remember the last example of satisfiability? Thats an NP-Complete problem. Way too complicated though.

Theres a "simpler" problem that is still NP-Complete

3 SAT: logical satisfiability with any number of variables in clauses only containing 3 variables. Variables are Or'd clauses are AND'd

e.g. $(x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee \neg x_3 \vee x_4) \ldots$

Any logical statement can be re-written in this form! usually this complicates the statement.

Lots of work goes into solving these!