C241 Pairwork II: Equivalence Simplification, Inference Proofs, Predicates & Quantifiers

Done in class on Monday 9/21/09, Wednesday, 9/23/09

Important Pointers

- Predicates are functions that describe very simple properties or relationships between their variables, properties and relationships that might or might not be true depending on what values we plug in for the variables.
  So, for example this is a predicate: P(x) = "x is chocolate chip", domain of x = cookies. Whether it's true or not depends what value of x we plug in, specifically whether x is a chocolate chip cookie. On the other hand, this is not a predicate: "There is a chocolate chip cookie". This is just a proposition, a flat statement that's either true or false. There's no variable that affects whether its true or false.
  Now, if I take the predicate $P(x)$, and I bind it's only variable with $\exists$, then I get $\exists x : P(x) =$ "There's a cookie which is a chocolate cookie". So *after* you use quantifiers to bind all the variables of a predicate, *then* the fully bound predicate is a proposition: it's a flat statement, a function with no free variables, a constant that's either always true or always false. But the original predicate itself, $P(x)$, must be dependent on x, it's truth must be a function of its variables.

- We want to keep predicates as simple as possible. So if I want to express the english sentence "x is a chocolate chip cookie and it just came out of the oven", it's much better to use two predicates and put them together with a logical operator: $P(x) =$"x is chocolate chip" and $W(x) =$"x is fresh baked", and write $P(x) \wedge W(x)$. For one thing, this gives me a lot more flexibility. Now that I have these two predicates, I can write: $P(x) \wedge \neg W(x) =$ "x is a chocolate chip cookie which is not fresh baked. Or, if I've been baking all day, and the last batch of cookies I did was chocolate chip, I could write: $W(x) \rightarrow P(x) =$ "if x is a freshly baked cookie, then it's chocolate chip". If the most recent batch of cookies I did was sugar cookies, and I haven't baked chocolate chip since this morning, I could write $W(x) \rightarrow \neg P(x) =$ "If x is fresh baked, then it's not a chocolate chip cookie".

- Also make sure you label your steps, on the line that shows the result of using them (as opposed to the line before you use them). So your first label should show up on the second line of the problem, as in the examples below.

- There's a variety of ways to solve (1) and (2) which are all perfectly valid. The solutions below are just examples of correct approaches.

1) Use the Algebraic Laws of Logic (posted on the 'resources' page on the class website) to simplify these logical statements.

a)

$(p \vee (r \wedge \neg r)) \vee (p \wedge q)$	
$(p \vee F) \vee (p \wedge q)$	Complement
$p \vee (p \wedge q)$	Identity
p	Subsumption

b)

$((p \vee q) \wedge \neg p) \rightarrow r$	
$((p \wedge \neg p) \vee (q \wedge \neg p)) \rightarrow r$	Distributive
$(F \vee (q \wedge \neg p)) \rightarrow r$	Complement
$(q \wedge \neg p) \rightarrow r$	Identity (we could stop here, or we can expand $\rightarrow$)
$\neg(q \wedge \neg p) \vee r$	Definition of $\rightarrow$
$(\neg q \vee \neg\neg p) \vee r$	DeMorgan's
$(\neg q \vee p) \vee r$	Double Negation
$\neg q \vee p \vee r$	Associative

2) For each problem below, use the Inference Rules (posted on the 'resources' page on the class website) to prove that if the premises of the problem are all true (the statements above the line), then the conclusion must be true (the statement below the line).

a)

$$\begin{array}{l} p \rightarrow q \\ q \rightarrow r \\ p \wedge s \\ \hline \therefore r \vee u \end{array}$$

1)	$p \to q$	premise
2)	$q \to r$	premise
3)	$p \to r$	lines (1)(2) and Law of Syllogism
4)	$p \wedge s$	premise
5)	p	line (4) and Conjunctive Simplification
6)	r	lines (3)(5) and Modus Ponens
7)	$r \vee u$	line (6) and Disjunctive Amplification
	$\therefore r \vee u$	

b)

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

$\therefore q$

1)	$\neg(\neg p \wedge \neg q)$	premise
2)	$\neg\neg p \vee \neg\neg q$	line (1) and DeMorgan's
3)	$p \vee q$	line (2) and Double Negation
4)	$\neg p \vee r$	premise
5)	$q \vee r$	lines (3)(4) and Proof By Resolution
6)	$\neg r$	premise
7)	q	lines (5)(6) and Disjunctive Syllogism
	$\therefore q$	

3) Write each of the english sentences below as formal logic statements in the predicate calculus, using predicates (very simple functions that evaluate to T or F depending on the values of their free variables) and quantifiers ($\exists$ = "There exists", "There's some", or "There's at least one", and $\forall$ = "for all" or "for every").

a) There is a pirate who is more awesome than every ninja

b) There s a pirate who is less awesome than every ninja

One method: Define $a(p,n)$ ="p is more awesome than n", where the domain of p is pirates, and the domain of n is ninjas. And, define $l(p,n)$ ="p is less awesome than n", where the domain of p is pirates and the domain of n in ninjas.

a)$\exists p \forall n : a(p,n)$

b)$\exists p \forall n : l(p, n)$

Another method: Define $a(x, y) =$"x is more awesome than y", where the domain of x is pirates and ninjas, and the domain of y is pirates and ninjas.

a)$\exists$ pirate p $\forall$ ninja $n : a(p, n)$

b)$\exists$ pirate p $\forall$ ninja $n : a(n, p)$ (there's a pirate such that every ninja is more awesome than him)

Almost, but not quite, another method: Define $a(p, n) =$"p is more awesome than n", where the domain of p is pirates, and the domain of n is ninjas.

a)$\exists p \forall n : a(p, n)$

b)$\exists p \forall n : \neg a(n, p)$ (there's a pirate who is *not* more awesome than any ninja. This statement doesn't mean *quite* the same thing as our english sentence. What happens in the case where the worst pirate and the worst ninja are equally awesome? Then our english sentence will be false (since the worst pirate isn't less awesome than *every* ninja). But this logical statement will evaluate to true (because there is a pirate who's not *better* than any ninja)) Tricky edge cases are tricky.