Predicates and Quantifiers

CSE 311 Autumn 23
Lecture 5

Announcements

Late Day Policy

Up to 6 late days to use during the quarter.

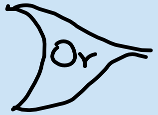
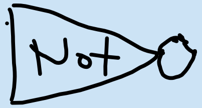
- Lets you submit a homework up to 24 hours later than normal.
- Max of 3 late days per assignment.

Just submit on gradescope, don't need to tell us you're using late days.

If you have extra late days after the last homework, we turn each late day into a "full credit concept check"

- Logisitcally too much to count late days on the small checks; we assume you would have gotten full credit.

Meet Boolean Algebra

Name	Variables	"True/False"	"And"	"Or"	"Not"	Implication
Java Code	`boolean b`	`true,false`	`&&`	`||`	`!`	No special symbol
Propositional Logic	"p, q, r"	`T, F`	$\wedge$	$\vee$	$\neg$	$\rightarrow$
Circuits	Wires	1, 0				No special symbol
Boolean Algebra	a, b, c	1,0	$\cdot$ ("multiplication")	$+$ ("addition")	$'$ (apostrophe after variable)	No special symbol

Propositional logic

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

Boolean Algebra

$$pqr + s + t'$$

Canonical Forms

Back to the old notation.

Canonical Forms

A truth table is a unique representation of a Boolean Function.

If you describe a function, there's only one possible truth table for it.

Given a truth table you can find many circuits and many compound prepositions to represent it.

Think back to when we were developing the law of implication…

It would be nice to have a "standard" proposition (or standard circuit) we could always write as a starting point.

So we have a (possibly) shorter way of telling if we have the same function.

Using Our Rules

WOW that was a lot of rules.

Why do we need them? Simplification!

Let's go back to the "law of implication" example.

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

When is the implication true? Just "or" each of the three "true" lines!

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

Also seems pretty reasonable

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

i.e. are these both alternative representations of $p \rightarrow q$?

Disjunctive Normal Form (DNF)

a.k.a. OR of ANDs

a.k.a Sum-of-Products Form

a.k.a. Minterm Expansion

1. Read the true rows of the truth table
2. AND together all the settings in a given (true) row.
3. OR together the true rows.

Disjunctive Normal Form

p	q	$G(p,q)$
T	T	T
T	F	F
F	T	T
F	F	F

$p \wedge q$

$\neg p \wedge q$

1. Read the true rows of the truth table
2. AND together all the settings in a given (true) row.
3. OR together the true rows.

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

Another Canonical Form

DNF is a great way to represent functions that are usually false.

If there are only a few true rows, the representation is short.

What about functions that are usually true?

Well G is equivalent to $\neg\neg G$, and $\neg G$ is a function that is usually false.

Let's try taking the Sum-of-Products of $\neg G$ and negating it.

Another Canonical Form

p	q	$G(p,q)$	$\neg G(p,q)$	
T	T	T	F	
T	F	F	T	$p \wedge \neg q$
F	T	T	F	
F	F	F	T	$\neg p \wedge \neg q$

1. Read the true rows of the truth table
2. AND together all the settings in a given (true) row.
3. OR together the true rows.

$$\neg G(p,q) \equiv (p \wedge \neg q) \vee (\neg p \wedge \neg q)$$
$$G(p,q) \equiv \neg[(p \wedge \neg q) \vee (\neg p \wedge \neg q)]$$
$$G(p,q) \equiv [\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge \neg q)]$$
$$G(p,q) \equiv [(\neg p \vee q) \wedge (p \vee q)]$$

This is not in Disjunctive Normal Form! It's something else, though…

Conjunctive Normal Form

a.k.a. AND of ORs

a.k.a. Product-of-Sums Form

a.k.a. Maxterm Expansion

1. Read the false rows of the truth table
2. OR together the negations of all the settings in the false rows.
3. AND together the false rows.

Or take the DNF of the negation of the function you care about, and distribute the negation.

Normal Forms

Don’t simplify any further! Don’t factor anything out (even if you can). The point of the canonical form is we know exactly what it looks like, you might simplify differently than someone else.

Why? Easier to understand for people.

Inside the parentheses are only ORs between the parentheses are only ANDs (or vice versa).

You’ll use these more in later courses.

Predicates!

Predicate Logic

So far our propositions have worked great for fixed objects.

What if we want to say "If $x > 10$ then $x^2 > 100$."

$x > 10$ isn't a proposition. Its truth value depends on x.

We need a function that can take in a value for x and output True or False as appropriate.

Predicates

Predicate

A function that outputs true or false.

`Cat(x)` := "`x` is a cat"

`Prime(x)` := "`x` is prime"

`LessThan(x,y)` := "`x<y`"

`Sum(x,y,z)` := "`x+y=z`"

`HasNChars(s,n)` := "string `s` has length `n`"

Numbers and types of inputs can change. Only requirement is output is Boolean.

Analogy

Propositions were like Boolean variables.

What are predicates? Functions that return Booleans

```
public boolean predicate(…)
```

Translation

Translation works a lot like when we just had propositions.

Let's try it…

x is prime or x^2 is odd or $x = 2$.

$\texttt{Prime}(x) \vee \texttt{Odd}(x^2) \vee \texttt{Equals}(x, 2)$

Domain of Discourse

x is prime or x^2 is odd or $x = 2$.

$\texttt{Prime}(x) \vee \texttt{Odd}(x^2) \vee \texttt{Equals}(x, 2)$

Can x be 4.5? What about "abc" ?

I never intended you to plug 4.5 or "abc" into x.

When you read the sentence you probably didn't imagine plugging those values in….

Domain of Discourse

x is prime or x^2 is odd or $x = 2$.

$\texttt{Prime}(x) \vee \texttt{Odd}(x^2) \vee \texttt{Equals}(x, 2)$

To make sure we **can't** plug in 4.5 for x, predicate logic requires deciding on the types we'll allow

Domain of Discourse

The set of all inputs allowed as inputs to our predicates.

Often we give the type(s) of allowed inputs, like "all integers" or "all real numbers.

Try it…

What’s a possible domain of discourse for these lists of predicates?

1. “x is a cat”, “x barks”, “x likes to take walks”
2. “x is prime”, “x=5” “$x < 20$“ “x is a power of two”
3. “x is enrolled in course y”, “y is a pre-req for z"

Try it…

What’s a possible domain of discourse for these lists of predicates?

1. “x is a cat”, “x barks”, “x likes to take walks”

   “Mammals”, “pets”, “dogs and cats”, …

2. “x is prime”, “x=5” “$x < 20$“ “x is a power of two”

   “positive integers”, “integers”, “numbers”, …

3. “x is enrolled in course y”, “y is a pre-req for z"

   “objects in the university course enrollment system”, “university entities”, “students and courses”, …

More than one domain of discourse might be reasonable…if it might affect the meaning of the statement, we specify it.

Quantifiers

Now that we have variables, let's really use them…

We tend to use variables for two reasons:

1. The statement is true for every x, we just want to put a name on it.
2. There's some x out there that works, (but I might not know which it is, so I'm using a variable).

Quantifiers

We have two extra symbols to indicate which way we're using the variable.

1. The statement is true for every x, we just want to put a name on it.

$\forall x\ (\mathrm{p}(\mathrm{x}) \wedge q(x))$ means "for every x in our domain, $p(x)$ and $q(x)$ both evaluate to true."

2. There's some x out there that works, (but I might not know which it is, so I'm using a variable).

$\exists x(p(x) \wedge q(x))$ means "there is an x in our domain, such that $p(x)$ and $q(x)$ are both true.

Quantifiers

We have two extra symbols to indicate which way we're using the variable.

1. The statement is true for every x, we just want to put a name on it.

$\forall x\ (\mathrm{p}(\mathrm{x}) \land q(x))$ means "for every x in our domain, $p(x)$ and $q(x)$ both evaluate to true."

Universal Quantifier

"$\forall x$"

"for each x", "for every x", "for all x" are common translations

Remember: upside-down-A for All.

Quantifiers

Existential Quantifier

"$\exists x$"

"there is an x", "there exists an x", "for some x" are common translations

Remember: backwards-E for Exists.

2. There's some x out there that works, (but I might not know which it is, so I'm using a variable).

$\exists x(p(x) \wedge q(x))$ means "there is an x in our domain, for which $p(x)$ and $q(x)$ are both true.

Translations

"For every x, if x is even, then $x = 2$."

"There are x, y such that $\mathrm{x} < y$."

$\exists x\ (\texttt{Odd}(x) \wedge \texttt{LessThan}(x, 5))$

$\forall y\ (\text{Even}(y) \wedge \text{Odd}(y))$

pollev.com/robbie

Help me adjust my explanation!

Translations

"For every x, if x is even, then $x = 2$."

$$\forall x(\texttt{Even}(x) \rightarrow \texttt{Equal}(x, 2))$$

"There are x, y such that $\mathrm{x} < y$."

$$\exists x \exists y(\texttt{LessThan}(x, y))$$

$$\exists x\ (\texttt{Odd}(x) \wedge \texttt{LessThan}(x, 5))$$

There is an odd number that is less than 5.

$$\forall y\ (\mathrm{Even}(y) \wedge \mathrm{Odd}(y))$$

All numbers are both even and odd.

Translations

More practice in section and on homework.

Also a reading on the webpage –

- An explanation of why “for any” is not a great way to translate $\forall$ (even though it looks like a good option on the surface)
- More information on what happens with multiple quantifiers (we’ll discuss more on Wednesday).

Evaluating Predicate Logic

"For every x, if x is even, then $x = 2$." / $\forall x(\texttt{Even}(x) \rightarrow \texttt{Equal}(x, 2))$

Is this true?

Evaluating Predicate Logic

"For every x, if x is even, then $x = 2$." / $\forall x(\texttt{Even}(x) \rightarrow \texttt{Equal}(x, 2))$

Is this true?

TRICK QUESTION! It depends on the domain.

Prime Numbers	Positive Integers	Odd integers
True	False	True (vacuously)

One Technical Matter

How do we parse sentences with quantifiers?

What's the "order of operations?"

We will usually put parentheses right after the quantifier and variable to make it clear what's included. If we don't, it's the rest of the expression.

Be careful with repeated variables…they don't always mean what you think they mean.

$\forall x\big(P(x)\big) \wedge \forall x(Q(x))$ are different x's.

Bound Variables

What happens if we repeat a variable?

Whenever you introduce a new quantifier with an already existing variable, it "takes over" that name until its expression ends.

$$\forall x(P(x) \wedge \forall x[Q(x)] \wedge R(x))$$

It's common (albeit somewhat confusing) practice to reuse a variables when it "wouldn't matter".

Never do something like the above: where a single name switches from gold to purple back to gold. Switching from gold to purple only is usually fine…but names are cheap.

More Practice

Let your domain of discourse be fruits. Translate these

There is a fruit that is tasty and ripe.

For every fruit, if it is not ripe then it is not tasty.

There is a fruit that is sliced and diced.

More Practice

Let your domain of discourse be fruits. Translate these

There is a fruit that is tasty and ripe.

$$\exists x(\texttt{Tasty}(x) \land \texttt{Ripe}(x))$$

For every fruit, if it is not ripe then it is not tasty.

$$\forall x(\neg\texttt{Ripe}(x) \rightarrow \neg\texttt{Tasty}(x))$$

There is a fruit that is sliced and diced.

$$\exists x(\texttt{Sliced}(x) \land \texttt{Diced}(x))$$

Domain Restriction

Quantifiers

∀ (for **A**ll) and ∃ (there **E**xists)

Write these statements in predicate logic with quantifiers. Let your domain of discourse be "cats"

This sentence implicitly makes a statement about all cats!

If a cat is fat, then it is happy.

$$\forall x[\texttt{Fat}(x) \rightarrow \texttt{Happy}(x)]$$

Quantifiers

Writing implications can be tricky when we change the domain of discourse.

For every cat: if the cat is fat, then it is happy.

Domain of Discourse: cats $\qquad \forall x[\texttt{Fat}(x) \to \texttt{Happy}(x)]$

What if we change our domain of discourse to be all mammals?

We need to limit x to be a cat. How do we do that?

$\forall x[(\texttt{Cat}(x) \wedge \texttt{Fat}(x)) \to \texttt{Happy}(x)] \qquad \forall x[\texttt{Cat}(x) \wedge (\texttt{Fat}(x) \to \texttt{Happy}(x))]$

Quantifiers

Which of these translates "For every cat: if a cat is fat then it is happy."
when our domain of discourse is "mammals"?

$\forall x[(\texttt{Cat}(x) \wedge \texttt{Fat}(x)) \rightarrow \texttt{Happy}(x)]$

For all mammals, if x is a cat and fat then it is happy
[if x is not a cat, the claim is vacuously true, you can't use the promise for anything]

$\forall x[\texttt{Cat}(x) \wedge (\texttt{Fat}(x) \rightarrow \texttt{Happy}(x))]$

For all mammals, that mammal is a cat and if it is fat then it is happy.
[what if x is a dog? Dogs are in the domain, but…uh-oh. This isn't what we meant.]

To "limit" variables to a portion of your domain of discourse under a universal quantifier add a hypothesis to an implication.

Quantifiers

Existential quantifiers need a different rule:

To "limit" variables to a portion of your domain of discourse under an existential quantifier AND the limitation together with the rest of the statement.

There is a dog who is not happy.

Domain of discourse: dogs

$\exists x(\neg \texttt{Happy}(x))$

Quantifiers

Which of these translates "There is a dog who is not happy."
when our domain of discourse is "mammals"?

$\exists x[\texttt{Dog}(x) \rightarrow \neg\texttt{Happy}(x))]$

There is a mammal, such that if x is a dog then it is not happy.
[this can't be right – plug in a cat for x and the implication is true]

$\exists x[(\texttt{Dog}(x) \wedge \neg\texttt{Happy}(x)]$

There is a mammal that is both a dog and not happy.
[this one is correct!]

To "limit" variables to a portion of your domain of discourse under an existential quantifier AND the limitation together with the rest of the statement.

Why are the rules what they are?

A universal quantifier is a "Big AND"

For a domain of discourse of $\{e_1, e_2, \ldots, e_k\}$

$\forall x(P(x))$ means $P(e_1) \wedge P(e_2) \wedge \cdots \wedge P(e_k)$

Now let's say our domain is $\{e_1, e_2, \ldots, e_k, f_1, f_2, \ldots, f_j\}$ where f_i are the irrelevant parts of the bigger domain (non-cat-mammals). We want the expression to be

$P(e_1) \wedge P(e_2) \wedge \cdots \wedge P(e_k) \wedge T \wedge T \ldots \wedge T$

$\forall x(RightSubDomain(x) \rightarrow P(x))$ does that!

Why are the rules what they are?

An existential quantifier is a "Big OR"

For a domain of discourse of $\{e_1, e_2, \dots, e_k\}$

$\exists x(P(x))$ means $P(e_1) \vee P(e_2) \vee \cdots \vee P(e_k)$

Now let's say our domain is $\{e_1, e_2, \dots, e_k, f_1, f_2, \dots, f_j\}$ where f_i are the irrelevant parts of the bigger domain (non-cat-mammals). We want the expression to be

$P(e_1) \vee P(e_2) \vee \cdots \vee P(e_k) \vee F \vee F \dots \vee F$

$\exists x(RightSubDomain(x) \wedge P(x))$ does that!

Negating Quantifiers

What happens when we negate an expression with quantifiers?

What does your intuition say?

Original	Negation
Every positive integer is prime	There is a positive integer that is not prime.
$\forall x\ \texttt{Prime}(x)$ Domain of discourse: positive integers	$\exists x(\neg\ \texttt{Prime}(x))$ Domain of discourse: positive integers

Negating Quantifiers

Let's try on an existential quantifier…

Original	Negation
There is a positive integer which is prime and even.	Every positive integer is composite or odd.
$\exists x(\texttt{Prime}(x) \wedge \texttt{Even}(x))$ Domain of discourse: positive integers	$\forall x(\neg\texttt{Prime}(x) \vee \neg\texttt{Even}(x))$ Domain of discourse: positive integers

To negate an expression with a quantifier

1. Switch the quantifier ($\forall$ becomes $\exists$, $\exists$ becomes $\forall$)
2. Negate the expression inside

Negation

Translate these sentences to predicate logic, then negate them.

All cats have nine lives.

$$\forall x\left(Cat(x) \rightarrow NumLives(x, 9)\right)$$

$\exists x(Cat(x) \wedge \neg\left(NumLives(x, 9)\right))$ “There is a cat without 9 lives.

All dogs love every person.

$$\forall x \forall y\left(Dog(x) \wedge Human(y) \rightarrow Love(x, y)\right)$$

$\exists x \exists y(Dog(x) \wedge Human(y) \wedge \neg Love(x, y))$ “There is a dog who does not love someone.” “There is a dog and a person such that the dog doesn’t love that person.”

There is a cat that loves someone.

$$\exists x \exists y(Cat(x) \wedge Human(y) \wedge Love(x, y)$$

$$\forall x \forall y(Cat(x) \wedge Human(y) \rightarrow \neg Love(x, y))$$

“For every cat and every human, the cat does not love that human.”

“Every cat does not love any human” (“no cat loves any human”)

Negation with Domain Restriction

$\exists x \exists y (Cat(x) \land Human(y) \land Love(x, y))$

$\forall x \forall y ([Cat(x) \land Human(y)] \rightarrow \neg Love(x, y))$

There are lots of equivalent expressions to the second. This one is by far the best because it reflects the domain restriction happening. How did we get there?

There's a problem in this week's section handout showing similar algebra.