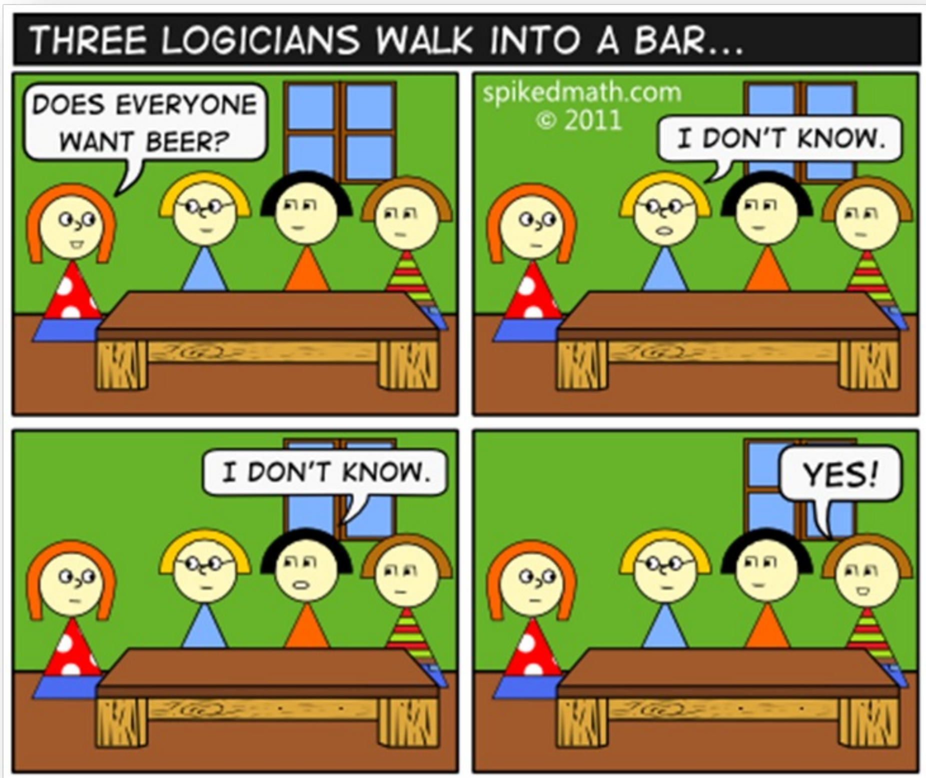
CSE 311: Foundations of Computing

Topic 3: 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 analyze them at a deeper level by expressing how those propositions 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"

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

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

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

GreaterThan5(x) ::= "x > 5"

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

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"

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:

	{1, 3, -1, -27}	Integers	Odd Integers
• $\forall x$ **Odd(x)**	True	False	True
• $\forall x$ **LessThan4(x)**	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?

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

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:

	{1, 3, -1, -27}	Integers	Positive Multiples of 5
• $\exists x$ Odd(x)	True	True	True
• $\exists x$ LessThan4(x)	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**

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.

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

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

$\forall x$ (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.

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

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

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 other positive integer.

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

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

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

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

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

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

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

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

Every prime number is either 2 or odd.

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

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

There exist prime numbers that are 2 apart.

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 often (not always) 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 $\forall$

Three people that I know are all friends with Mark Zuckerberg $\exists$

Formal logic removes this ambiguity

- quantifiers can always be specified
- unquantified variables that are not known constants (e.g, π) are **implicitly** $\forall$–quantified

Negations of Quantifiers

Predicate Definitions
PurpleFruit(x) ::= “x is a purple fruit”

(*) $\forall$x PurpleFruit(x) (“All fruits are purple”)

What is the negation of (*)?

(a) “there exists a purple fruit”

(b) “there exists a non-purple fruit”

(c) “all fruits are not purple”

Try your intuition! Which one seems right?

Negations of Quantifiers

Predicate Definitions
PurpleFruit(x) ::= "x is a purple fruit"

(*) $\forall$x PurpleFruit(x) ("All fruits are purple")

What is the negation of (*)?

(a) "there exists a purple fruit"

(b) "there exists a non-purple fruit"

(c) "all fruits are not purple"

Domain of Discourse
{plum, apple}

(*) PurpleFruit(plum) $\wedge$ PurpleFruit(apple)

(a) PurpleFruit(plum) $\vee$ PurpleFruit(apple)

(b) $\neg$ PurpleFruit(plum) $\vee$ $\neg$ PurpleFruit(apple)

(c) $\neg$ PurpleFruit(plum) $\wedge$ $\neg$ PurpleFruit(apple)

De Morgan's Laws for Quantifiers

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

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

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 integer larger than every other integer"

$$\neg \exists x \forall y\ (x \geq y)$$
$$\equiv \forall x \neg \forall y\ (x \geq y)$$
$$\equiv \forall x\ \exists y \neg (x \geq y)$$
$$\equiv \forall x\ \exists y\ (y > x)$$

"For every integer, there is a larger integer"

De Morgan's Laws for Quantifiers

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

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

They have different English translations, e.g.:

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

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

De Morgan's Laws for Quantifiers

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

"No even prime is greater than 2"

$$\neg\ \exists x\ (\text{Even}(x) \wedge \text{Prime}(x) \wedge \text{Greater}(x, 2))$$
$$\equiv\ \forall x\ \neg(\text{Even}(x) \wedge \text{Prime}(x) \wedge \text{Greater}(x, 2))$$
$$\equiv\ \forall x\ (\neg(\text{Even}(x) \wedge \text{Prime}(x)) \vee \neg\text{Greater}(x, 2))$$
$$\equiv\ \forall x\ (\neg(\text{Even}(x) \wedge \text{Prime}(x)) \vee \text{LessEq}(x, 2))$$
$$\equiv\ \forall x\ ((\text{Even}(x) \wedge \text{Prime}(x)) \rightarrow \text{LessEq}(x, 2))$$

"Every even prime is less than or equal to 2."

De Morgan's Laws for Quantifiers

We just saw that

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

Can similarly show 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.)

Quantifiers in Java

- **For finite domains of discourse, we could implement quantifiers in Java:**

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

$\forall x\ P(x)$

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

$\exists x\ P(x)$

Scope of Quantifiers

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

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.

Scope of Quantifiers

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

Example: NotLargest(x) ::= $\exists$ y Greater (y, x)

$\equiv$ $\exists$ z Greater (z, x)

truth value:

doesn't depend on y **or** z **"bound variables"**

does depend on x **"free variable"**

Scope of Quantifiers

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

Example: NotLargest(x) ::= $\exists$ y Greater (y, x)

$\equiv$ $\exists$ z Greater (z, x)

truth value:

doesn't depend on y **or** z **"bound variables"**

does depend on x **"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"

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

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…

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"

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

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

x \ y	1	2	3	4
1	T	F	F	F
2	T	T	F	F
3	T	T	T	F
4	T	T	T	T

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.”

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

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

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

x \ y	1	2	3	4
1	T	F	F	F
2	T	T	F	F
3	T	T	T	F
4	T	T	T	T

Quantifier Order Can Matter

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

Predicate Definitions
GreaterEq(x, y) ::= "x ≥ y"

x \ y	1	2	3	4
1	T	F	F	F
2	T	T	F	F
3	T	T	T	F
4	T	T	T	T

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

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

"Every number has a number greater than or equal to it."

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

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.