University of Colorado **Boulder**

Department of Computer Science

CSCI 2824: Discrete Structures

Chris Ketelsen

# Predicate Logic, Quantifiers, and Rules of Inference (Lecture 2):

# Nested Quantifiers

# Quantifier Recap

**Last Time:**

- Introduced predicates and propositional functions
- Started on universal and existential quantifiers

**Universal Quantifier:**

- $\forall x\ P(x)$: "For all $x$ in my domain $P(x)$ is true "

**Existential Quantifier:**

- $\exists x\ P(x)$: "There exists an $x$ in my domain s.t. $P(x)$ is true"

# Quantifier Recap

Last time we showed the following equivalences

**DeMorgan's Laws for Quantifiers:**

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

**Distribution Laws for Quantifiers:**

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

**Note**: Distribution of $\forall$ over $\vee$ and $\exists$ over $\wedge$ didn't work

# Quantifiers as Loops

### A Computer Sciency Way of Viewing Quantifiers

Think of quantified statements as loops that do logic checks

**Example:** $\forall x \, P(x)$

In [ ]:
```
for x in domain:
    if P(x) == False:
        return False
return True
```

- If we find an $x$ in domain where $P(x)$ is False, return False
- If we make it through loop then return True

# Quantifiers as Loops

## A Computer Sciency Way of Viewing Quantifiers

Think of quantified statements as loops that do logic checks

**Example:** $\exists x\ P(x)$

In [ ]:
```
for x in domain:
    if P(x) == True:
        return True
return False
```

- If we find an $x$ in domain where $P(x)$ is True, return True
- If we make it through loop without finding one, return False

# Nested Quantifiers

Interesting things happen when we include multiple quantifiers

**Example**: What does this say: $\forall x \; \exists y \; (x + y = 0)$ ?

# Nested Quantifiers

Interesting things happen when we include multiple quantifiers

**Example**: What does this say: $\forall x \; \exists y \; (x + y = 0)$ ?

It really helps to read these outloud: ``For all $x$, there exists a $y$, such that the sum of $x$ and $y$ is zero"

What do you think? Is this true or false?

# Nested Quantifiers

Interesting things happen when we include multiple quantifiers

**Example**: What does this say: $\forall x \, \exists y \, (x + y = 0)$ ?

It really helps to read these outloud: ``For all $x$, there exists a $y$, such that the sum of $x$ and $y$ is zero"

What do you think? Is this true or false?

This is totally **true.** It's the expression of the fact that all numbers have an **additive inverse**.

# Nested Quantifiers as Loops

**Example:** $\forall x \, \exists y \, P(x, y)$ ?

In [ ]:
```
for x in domain:
    exists_y = False
    for y in domain:
        if P(x,y) == True:
            exists_y = True
    if exists_y == False:
        return False
return True
```

- If we make it through $y$-loop without finding a True, return False
- If we make it through entire $x$-loop then return True

# Nested Quantifiers as Loops

## Example: $\forall x\ \exists y\ (x + y = 0)$ ?

In [7]:
```python
def check_additive_inverse(domain):

    for x in domain:
        exists_y = False
        for y in domain:
            if x + y == 0:
                exists_y = True
        if exists_y == False:
            return False
    return True

domain = [-3, -2, -1, 0, 1, 2, 3]
check_additive_inverse(domain)
```

Out[7]:
```
True
```

# Nested Quantifiers as Loops

## **Example:** $\forall x \; \exists y \; (x + y = 0)$ ?

In [8]:
```
def check_additive_inverse(domain):

    for x in domain:
        exists_y = False
        for y in domain:
            if x + y == 0:
                exists_y = True
        if exists_y == False:
            return False
    return True

domain = [-2, -1, 0, 1, 2, 3]
check_additive_inverse(domain)
```

Out[8]: False

# Nested Quantifiers

**Example:** Express the law of **commutativity of addition**:

$$x + y = y + x$$

using quantifiers

# Nested Quantifiers as Loops

**Example**: $\forall x \; \forall y \; P(x, y)$ ?

In [ ]:
```
for x in domain:
    for y in domain:
        if P(x,y) == False:
            return False
return True
```

- If we ever find an $(x, y)$-pair that makes $P(x, y)$ False, return False
- If we make it through both loops, return True

# Nested Quantifiers

**Example:** Express the law of **commutativity of addition**:

$$x + y = y + x$$

using quantifiers

**Solution:** $\forall x \, \forall y \, (x + y = y + x)$

**Question:** What happens if we change the order of $\forall x$ and $\forall y$?

# Nested Quantifiers

Let's go back to the **additive inverse** example:

**Example:** $\forall x \ \exists y \ (x + y = 0)$

**Question:** What happens if we change the order here?

# Nested Quantifiers

**Rules for Switching Quantifiers:**

- OK to switch $\forall x$ and $\forall y$
- OK to switch $\exists x$ and $\exists y$ (**EFY**: Check that this is true!)
- **NOT** OK to switch $\forall x$ and $\exists y$

OK, let's do a few more examples

# Nested Quantifiers

Now we'll let the domain be all real numbers

**Example**: How can you express the fact that all numbers have a **multiplicative inverse**

That is, a number that you can multiply by to get 1?

First of all, is it really true that **all** numbers of have a multiplicative inverse?

# Nested Quantifiers

Now we'll let the domain be all real numbers

**Example:** How can you express the fact that all numbers have a **multiplicative inverse**

That is, a number that you can multiply by to get 1?

First of all, is it really true that **all** numbers of have a multiplicative inverse?

**Nope**! But all **nonzero** numbers do

So how could we say this with quantifiers?

# Nested Quantifiers

Let's say it in logic-y English

``For all $x$'s that aren't zero, there exists a $y$ such that $xy = 1$"

# Nested Quantifiers

Let's say it in logic-y English

``For all $x$'s that aren't zero, there exists a $y$ such that $xy = 1$"

Note that $x$ not being zero is a **condition** that has to happen before we consider looking for an inverse. Let's rephrase:

``For all $x$, if $x \neq 0$ then there exists a $y$ such that $xy = 1$"

# Nested Quantifiers

**Example:** How could you express that if you multiply two negative numbers together you get a positive number?

# Nested Quantifiers

OK, lets practice some non-mathy translations

**Example:** Translate the statement ``You can fool some of the people all of the time"

We need to define a propositional function that says a person can be fooled at a particular time

Let $F(p, t)$ represent ``you can fool person $p$ at time $t$"

# Nested Quantifiers

**Example:** Translate the statement ``You can't fool all of the people all of the time"

# Nested Quantifiers

## End of Representational Logic

- We now know how to represent standard propositions
- We know how to represent propositions with quantifiers
- We know how to prove and derive logical equivalences

## Next Time We Start Learning to Argue

- Rules of inference
- Valid and sound arguments