101100001000 101100001000 101100001000 101100001000 101100001000

CSCI 2824: Discrete Structures

Lecture 7: Nested Quantifiers

Rachel Cox

Department of Computer Science

101100001000 101100001000 101100001000 101100001000 101100001000

Nested Quantifiers

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"

Nested Quantifiers

Warm-Up Problems: Let the domain for x be the set of all Natural Numbers, $\mathbb{N} = \{0, 1, 2, \ldots\}$

Example: Determine the truth value of $\forall n \ (3n \leq 4n)$

Example: Determine the truth value of $\exists x \ (x^2 = x)$

Nested Quantifiers

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

Nested Quantifiers

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

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

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

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

Nested Quantifiers as Loops

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

In [7]:
```
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

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

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: How could we express the law of **commutation of addition** (that is, that $x + y = y + x$)?

Nested Quantifiers

Let's go back to the previous example:

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

Question: What happens if we change the order here?

Answer: A lot! The new expression $\exists y \; \forall x \; (x + y = 0)$ says

- ``There exists some number y such that for every x out there, $x + y = 0$"

Can you think of such a number?

Nested Quantifiers

Rules for Switching Quantifiers:

- OK to switch $\forall x$ and $\forall y$
- OK to switch $\exists x$ and $\exists y$
- **NOT** OK to switch $\forall x$ and $\exists y$

Nested Quantifiers

Example: Now we'll switch the domain to all real numbers

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

Nested Quantifiers

Example: How could you express that there are an infinite number of natural numbers?

If domains for x and y are the set of natural numbers, we could say

$$\forall x \, \exists y \, (y > x)$$

This just says that every natural number has a number that is larger

Nested Quantifiers

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

Nested Quantifiers

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

Nested Quantifiers

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

Nested Quantifiers

Quantifications with more than two quantifiers are also common

Example: Let $Q(x, y, z)$ mean "$x + y = z$". What are the truth values of

- $\forall x \, \forall y \, \exists z \, Q(x, y, z)$
- $\exists z \, \forall x \, \exists y \, Q(x, y, z)$

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
- Proof types and strategies