Predicate Logic: Predicates and Quantifiers

Section 1.4

Propositional Logic Not Enough

- If we have:

  "All men are mortal."

  "Socrates is a man."

  ∴"Socrates is mortal"

Compare to:
"If it is snowing, then I will study discrete math."
"It is snowing."

∴"I will study discrete math."

- This *not* a valid argument in propositional logic.

➔Need a language that talks about objects, their properties, and their relations.

Introducing Predicate Logic

- Predicate logic uses the following new features:
  - Variables: x, y, z
  - Predicates: P, M
  - Quantifiers: $\forall, \exists$
- *Propositional functions* are a generalization of propositions.
  - They contain variables and a predicate, e.g., $P(x)$
  - Variables can be replaced by elements from their *domain*, e.g. the domain of integers.

Propositional Functions

- Propositional functions become propositions (and have truth values) when their variables are each replaced by a value from the *domain* (or *bound* by a quantifier, as we will see later).
- The statement *P(x)* is said to be the value of the propositional function *P(x)* at *x*.
- For example, let *P(x)* denote "$x > 0$" and the domain be the integers. Then:

  P(-3) is false.

  P(0) is false.

  P(3) is true.
- Often the domain is denoted by *U*. So in this example *U* is the integers.

Examples of Propositional Functions

- Let "$x + y = z$" be denoted by $R(x, y, z)$ and U (for all three variables) be the integers. Find these truth values:

  R(2,-1,5)

  Solution: F

  R(3,4,7)

  Solution: T

  R(x, 3, z)

  Solution: Not a Proposition

- Now let "$x - y = z$" be denoted by $Q(x, y, z)$, with U as the integers. Find these truth values:

  Q(2,-1,3)

  Solution: T

  Q(3,4,7)

  Solution: F

  Q(x, 3, z)

  Solution: Not a Proposition

Compound Expressions

- Connectives from propositional logic carry over to predicate logic.
- If *P(x)* denotes "$x > 0$," find these truth values:

  P(3) ∨ P(-1) **Solution**: T

  P(3) ∧ P(-1) **Solution**: F

  P(3) → P(-1) **Solution**: F

  P(3) → P(-1) **Solution**: T
- Expressions with variables are not propositions and therefore do not have truth values. For example,

  P(3) ∧ P(*y*)

  P(*x*) → P(*y*)
- When used with quantifiers (to be introduced next), these expressions (propositional functions) become propositions.

Quantifiers

- We need *quantifiers* to express the meaning of English words including *all* and *some*:
  - "All men are Mortal."
  - "Some cats do not have fur."
- The two most important quantifiers are:
  - *Universal Quantifier*, "For All," symbol: $\forall$
  - *Existential Quantifier*, "There Exists," symbol: $\exists$
- We write as in $\forall x\, P(x)$ and $\exists x\, P(x)$.
- $\forall x\, P(x)$ asserts $P(x)$ is true for every x in the *domain*.
- $\exists x\, P(x)$ asserts $P(x)$ is true for some x in the *domain*.
- The quantifiers are said to bind the variable x in these expressions.

Universal Quantifier

- $\forall x\, P(x)$ is read as "For All x, P(x)"

  Examples:

  1) If $P(x)$ denotes "$x > 0$" and U is the integers, then $\forall x\, P(x)$ is false.
  2) If $P(x)$ denotes "$x > 0$" and U is the positive integers, then $\forall x\, P(x)$ is true.
  3) If $P(x)$ denotes "x is even" and U is the integers, then $\forall x\, P(x)$ is false.

Existential Quantifier

- $\exists x\, P(x)$ is read as "There Exists an x such that P(x)"

Examples:

1. If $P(x)$ denotes "$x > 0$" and U is the integers, then $\exists x\, P(x)$ is true. It is also true if U is the positive integers.
2. If $P(x)$ denotes "$x < 0$" and U is the positive integers, then $\exists x\, P(x)$ is false.
3. If $P(x)$ denotes "x is even" and U is the integers, then $\exists x\, P(x)$ is true.

Thinking about Quantifiers

- When the domain of discourse is finite, we can think of quantification as looping through the elements of the domain.
- To evaluate $\forall x\, P(x)$ loop through all x in the domain.
  - If at every step P(x) is true, then $\forall x\, P(x)$ is true.
  - If at a step P(x) is false, then $\forall x\, P(x)$ is false and the loop terminates.
- To evaluate $\exists x\, P(x)$ loop through all x in the domain.
  - If at some step, P(x) is true, then $\exists x\, P(x)$ is true and the loop terminates.
  - If the loop ends without finding an x for which P(x) is true, then $\exists x\, P(x)$ is false.
- Even if the domains are infinite, we can still think of the quantifiers this fashion, but it would not be practical to implement it this way…

Properties of Quantifiers

- The truth value of $\exists x\, P(x)$ and $\forall x\, P(x)$ depend on both the propositional function $P(x)$ and on the domain U.
- **Examples**:
  1. If U is the positive integers and $P(x)$ is the statement "$x < 2$", then $\exists x\, P(x)$ is true, but $\forall x\, P(x)$ is false.
  2. If U is the negative integers and $P(x)$ is the statement "$x < 2$", then both $\exists x\, P(x)$ and $\forall x\, P(x)$ are true.
  3. If U consists of 3, 4, and 5, and $P(x)$ is the statement "$x > 2$", then both $\exists x\, P(x)$ and $\forall x\, P(x)$ are true. But if $P(x)$ is the statement "$x < 2$", then both $\exists x\, P(x)$ and $\forall x\, P(x)$ are false.

Precedence of Quantifiers

- The quantifiers ∀ and ∃ have higher precedence than all the logical operators.
- For example, $\forall x\, P(x) \vee Q(x)$ means $(\forall x\, P(x)) \vee Q(x)$
- $\forall x\, (P(x) \vee Q(x))$ means something different.
- Unfortunately, often people write $\forall x\, P(x) \vee Q(x)$ when they mean $\forall x\, (P(x) \vee Q(x))$.
- To avoid any confusion just put brackets right after every quantifier you use, i.e.

  $\forall x\, [P(x) \vee Q(x)]$
- Proposition then becomes very easy to read

Translating from English to Logic

Example 1: Translate the following sentence into predicate logic: “Every student in this class has taken a course in Java.”

Solution:

First decide on the domain *U*.

Solution 1: If *U* is all students in this class, define a propositional function J(x) denoting “x has taken a course in Java” and translate as $\forall x\, J(x)$.

Solution 2: But if *U* is all people, also define a propositional function S(x) denoting “x is a student in this class” and translate as $\forall x\, [S(x) \rightarrow J(x)]$.

Translating from English to Logic

Example 2: Translate the following sentence into predicate logic: "Some student in this class has taken a course in Java."

Solution:

First decide on the domain *U*.

Solution 1: If *U* is all students in this class, translate as

$$\exists x\, J(x)$$

Solution 1: But if *U* is all people, then translate as

$$\exists x\, [S(x) \land J(x)]$$

Returning to the Socrates Example

- Introduce the propositional functions *man(x)* denoting "*x* is a man" and *mortal(x)* denoting "*x* is mortal." Specify the domain as all people.
- The two premises are:

$$\forall x[man(x) \rightarrow mortal(x)]$$
$$man(Socrates)$$

- The conclusion is:

$$\therefore mortal(Socrates)$$

- Later we will show how to prove that the conclusion follows from the premises.

Equivalences in Predicate Logic

- Statements involving predicates and quantifiers are *logically equivalent* if and only if they have the same truth value
  - for every predicate substituted into these statements and
  - for every domain of discourse used for the variables in the expressions.
- The notation $S \equiv T$ indicates that S and T are logically equivalent.
- **Example**: $\forall x \neg\neg S(x) \equiv \forall x\, S(x)$

Equivalences

- To show that two quantified expressions are equivalent, we need to show that both sides will be true under all predicates and all domains.
- Here is a way to prove it.

$\forall x[\neg\neg P(x)] \equiv \forall x[P(x)]$

Assume that the right side holds, also

assume that $a \in U$ is *any* element in U,

where U is *any* domain, then

$\forall x[P(x)]$ implies $P(a)$ implies $\neg\neg P(a)$ implies $\forall x[\neg\neg P(x)]$

Now, assume that the left side holds, then

$\forall x[\neg\neg P(x)]$ implies $\neg\neg P(a)$implies $P(a)$ implies $\forall x[P(x)]$

$\therefore \forall x[\neg\neg P(x)] \equiv \forall x[P(x)]$

Negating Quantified Expressions

- Consider $\forall x\, J(x)$

  "Every student in your class has taken a course in Java."

  Here $J(x)$ is "x has taken a course in Java" and

  the domain is students in your class.

- Negating the original statement gives "It is not the case that every student in your class has taken Java." This implies that "There is a student in your class who has not studied Java."

  Symbolically $\neg\forall x\, J(x)$ and $\exists x\, \neg J(x)$ are equivalent

Negating Quantified Expressions (*continued*)

- Now Consider $\exists x\, J(x)$

  "There is a student in this class who has taken a course in Java."

  Where $J(x)$ is "x has taken a course in Java."

- Negating the original statement gives "It is not the case that there is a student in this class who has taken Java." This implies that "Every student in this class has not taken Java"

  Symbolically $\neg\exists x\, J(x)$ and $\forall x\, \neg J(x)$ are equivalent

De Morgan's Laws for Quantifiers

- It can be shown that the following holds:

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

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

Translation from English to Logic

Examples:

1. "Some student in this class has visited Mexico."

   Solution: Let $M(x)$ denote "x has visited Mexico" and $S(x)$ denote "x is a student in this class," and U be all people.

   $$\exists x \, [S(x) \wedge M(x)]$$

2. "Every student in this class has visited Canada or Mexico."

   Solution: Add $C(x)$ denoting "x has visited Canada."

   $$\forall x \, [S(x) \rightarrow (M(x) \vee C(x))]$$

Nested Quantifiers

Section 1.5

Nested Quantifiers

- Nested quantifiers are often necessary to express the meaning of sentences in English as well as important concepts in computer science and mathematics.

- **Example**: "Every real number has an inverse" is

  $\forall x \, \exists y[x + y = 0]$

  where the domains of x and y are the real numbers.

Thinking of Nested Quantification

- Nested Loops
  - To see if ∀x∀y[P(x,y)] is true, loop through the values of *x* :
    - At each step, loop through the values for *y*.
    - If for some pair of *x* and *y, P(x,y)* is false, then ∀x ∀y[P(x,y)] is false and both the outer and inner loop terminate.

    ∀x ∀y[P(x,y)] is true if the outer loop ends after stepping through each *x*.
  - To see if ∀x ∃y[P(x,y*)*] is true, loop through the values of *x*:
    - At each step, loop through the values for *y.*
    - The inner loop ends when a pair *x* and *y* is found such that *P*(*x*, y) is true.
    - If no *y* is found such that *P*(*x*, y) is true the outer loop terminates as ∀x ∃y[P(x,y)] has been shown to be false.

    ∀x ∃y[P(x,y)] is true if the outer loop ends after stepping through each *x*.
- If the domains of the variables are infinite, then this process can not actually be carried out.

Order of Quantifiers

The order of quantification matters!

Examples:

1. Let $P(x,y)$ be the statement "$x + y = y + x$." Assume that U is the real numbers. Then $\forall x\, \forall y P(x,y)$ and $\forall y\, \forall x P(x,y)$ have the same truth value.
2. However, let $Q(x,y)$ be the statement "$x + y = 0$." Assume that U is the real numbers. Then $\forall x\, \exists y P(x,y)$ is true, but $\exists y\, \forall x P(x,y)$ is false.

Translating Nested Quantifiers into English

Example : Translate the statement

$$\forall x\,[C(x) \vee \exists y[C(y) \wedge F(x, y)]]$$

where C(x) is "*x* has a computer," and *F*(*x*,*y*) is "*x* and *y* are friends," and the domain for both *x* and *y* consists of all students in your school.

Solution: First we can rewrite the expression:

$$\forall x\,[C(x) \vee \exists y[C(y) \wedge F(x, y)]] \equiv \forall x\,[C(x)] \vee \forall x \exists y[F(x, y) \wedge C(y)]$$

Every student in your school has a computer or has a friend who has a computer.

Translating Mathematical Statements into Predicate Logic

Example : Translate "The sum of two positive integers is always positive" into a logical expression.

Solution:

1. Rewrite the statement to make the implied quantifiers and domains explicit:

   "For every two integers, if these integers are both positive, then the sum of these integers is positive."

2. Introduce the variables *x* and *y*, and specify the domain, to obtain:

   "For all positive integers *x* and *y*, *x* + *y* is positive."

3. The result is:

$$\forall x \, \forall y \, ((x > 0) \wedge (y > 0) \rightarrow (x + y > 0))$$

where the domain of both variables consists of all integers