Propositional Logic (cont.)

Assylbek Issakhov,
Ph.D., professor

Applications of Propositional Logic

- Logic has many important applications to mathematics, computer science, and numerous other disciplines. Statements in mathematics and the sciences and in natural language often are imprecise or ambiguous. To make such statements precise, they can be translated into the language of logic. For example, logic is used in the specification of software and hardware, because these specifications need to be precise before development begins.

Applications of Propositional Logic

- Once we have translated sentences from English into logical expressions we can analyze these logical expressions to determine their truth values, we can manipulate them, and we can use rules of inference to reason about them.
- To illustrate the process of translating an English sentence into a logical expression, consider the next examples.

Applications of Propositional Logic: Example

- How can this English sentence be translated into a logical expression?

"You can access the Internet from campus only if you are a computer science major or you are not a freshman."

- *Solution:* There are many ways to translate this sentence into a logical expression. Although it is possible to represent the sentence by a single propositional variable, such as p, this would not be useful when analyzing its meaning or reasoning with it.

Applications of Propositional Logic: Example

- *Solution (cont.):* Instead, we will use propositional variables to represent each sentence part and determine the appropriate logical connectives between them. In particular, we let a, c, and f represent "You can access the Internet from campus," "You are a computer science major," and "You are a freshman," respectively. Noting that "only if" is one way a conditional statement can be expressed, this sentence can be represented as

$$a \rightarrow (c \vee \neg f).$$

Applications of Propositional Logic: Example

- How can this English sentence be translated into a logical expression?

"You cannot ride the roller coaster if you are under 4 feet tall unless you are older than 16 years old."

- *Solution:* Let q, r, and s represent "You can ride the roller coaster," "You are under 4 feet tall," and "You are older than 16 years old," respectively. Then the sentence can be translated to

$$(r \wedge \neg s) \rightarrow \neg q.$$

System Specifications

- Translating sentences in natural language (such as English) into logical expressions is an essential part of specifying both hardware and software systems. System and software engineers take requirements in natural language and produce precise and unambiguous specifications that can be used as the basis for system development.

System Specifications: Example

- Express the specification "The automated reply cannot be sent when the file system is full" using logical connectives.
- *Solution:* One way to translate this is to let p denote "The automated reply can be sent" and q denote "The file system is full." Then $\neg p$ represents "It is not the case that the automated reply can be sent," which can also be expressed as "The automated reply cannot be sent." Consequently, our specification can be represented by the conditional statement $q \rightarrow \neg p$.

System Specifications

- System specifications should be **consistent**, that is, they should not contain conflicting requirements that could be used to derive a contradiction. When specifications are not consistent, there would be no way to develop a system that satisfies all specifications.

System Specifications: Example

- Determine whether these system specifications are consistent:

"The diagnostic message is stored in the buffer or it is retransmitted."

"The diagnostic message is not stored in the buffer."

"If the diagnostic message is stored in the buffer, then it is retransmitted."

System Specifications: Example

- *Solution:* To determine whether these specifications are consistent, we first express them using logical expressions. Let p denote "The diagnostic message is stored in the buffer" and let q denote "The diagnostic message is retransmitted." The specifications can then be written as $p \vee q$, $\neg p$, and $p \rightarrow q$. An assignment of truth values that makes all three specifications true must have p false to make $\neg p$ true.

System Specifications: Example

- *Solution (cont.):* Because we want $p \vee q$ to be true but p must be false, q must be true. Because $p \rightarrow q$ is true when p is false and q is true, we conclude that these specifications are consistent, because they are all true when p is false and q is true. We could come to the same conclusion by use of a truth table to examine the four possible assignments of truth values to p and q.

Logic Puzzles

- Puzzles that can be solved using logical reasoning are known as **logic puzzles**. Solving logic puzzles is an excellent way to practice working with the rules of logic. Also, computer programs designed to carry out logical reasoning often use well-known logic puzzles to illustrate their capabilities. Many people enjoy solving logic puzzles, published in periodicals, books, and on the Web, as a recreational activity.

Logic Puzzles

- We consider a puzzle originally posed by Raymond Smullyan, a master of logic puzzles, who has published more than a dozen books containing challenging puzzles that involve logical reasoning.

Logic Puzzles: Example

- Smullyan posed many puzzles about an island that has two kinds of inhabitants, knights, who always tell the truth, and their opposites, knaves, who always lie. You encounter two people A and B. What are A and B if A says "B is a knight" and B says "The two of us are opposite types?"

Logic Puzzles: Example

- *Solution:* Let p and q be the statements that "A is a knight" and "B is a knight", respectively, so that $\neg p$ and $\neg q$ are the statements that A is a knave and *B* is a knave, respectively.
- We first consider the possibility that A is a knight; this is the statement that p is true. If A is a knight, then he is telling the truth when he says that B is a knight, so that q is true, and A and B are the same type.

Logic Puzzles: Example

- *Solution (cont.):* However, if B is a knight, then B's statement that A and B are of opposite types, the statement

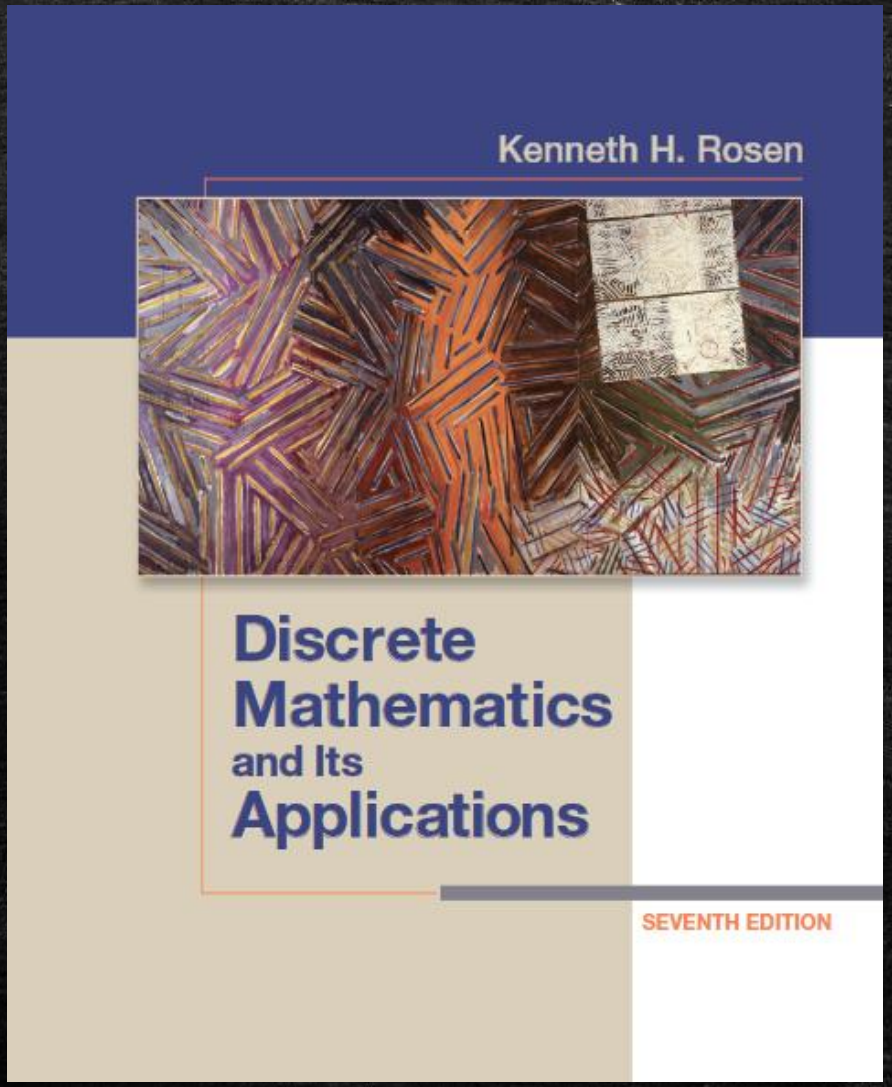
$$(p \wedge \neg q) \vee (\neg p \wedge q),$$

- would have to be true, which it is not, because A and B are both knights. Consequently, we can conclude that A is not a knight, that is, that p is false.

Logic Puzzles: Example

- If A is a knave, then because everything a knave says is false, A's statement that B is a knight, that is, that q is true, is a lie. This means that q is false and B is also a knave. Furthermore, if B is a knave, then B's statement that A and B are opposite types is a lie, which is consistent with both A and B being knaves. We can conclude that both A and B are knaves.

HOMEWORK: Exercises 2, 4, 8, 10, 12, 20, 22 on pp. 22-23



Propositional Equivalences

- An important type of step used in a mathematical argument is the replacement of a statement with another statement with the same truth value. Because of this, methods that produce propositions with the same truth value as a given compound proposition are used extensively in the construction of mathematical arguments. Note that we will use the term "compound proposition" to refer to an expression formed from propositional variables using logical operators, such as $p \wedge q$.

Propositional Equivalences

- **DEFINITION 1.** A compound proposition that is always true, no matter what the truth values of the propositional variables that occur in it, is called a *tautology*. A compound proposition that is always false is called a *contradiction*. A compound proposition that is neither a tautology nor a contradiction is called a *contingency*.

Propositional Equivalences

- Tautologies and contradictions are often important in mathematical reasoning. Example below illustrates these types of compound propositions.
- **EXAMPLE.** We can construct examples of tautologies and contradictions using just one propositional variable. Consider the truth tables of $p \vee \neg p$ and $p \wedge \neg p$, shown in Table 1. Because $p \vee \neg p$ is always true, it is a tautology. Because $p \wedge \neg p$ is always false, it is a contradiction.

Propositional Equivalences

TABLE 1 Examples of a Tautology and a Contradiction.

p	$\neg p$	$p \vee \neg p$	$p \wedge \neg p$
T	F	T	F
F	T	T	F

Logical Equivalences

- Compound propositions that have the same truth values in all possible cases are called **logically equivalent**. We can also define this notion as follows.
- **DEFINITION 2.** The compound propositions p and q are called *logically equivalent* if $p \leftrightarrow q$ is a tautology. The notation $p \equiv q$ denotes that p and q are logically equivalent.

Logical Equivalences

- ***Remark:*** The symbol $\equiv$ is not a logical connective, and $p \equiv q$ is not a compound proposition but rather is the statement that $p \leftrightarrow q$ is a tautology. The symbol $\Leftrightarrow$ is sometimes used instead of $\equiv$ to denote logical equivalence.
- One way to determine whether two compound propositions are equivalent is to use a truth table. In particular, the compound propositions p and q are equivalent if and only if the columns giving their truth values agree.

Logical Equivalences

- This logical equivalence is one of the two **De Morgan laws**, shown in Table 2, named after the English mathematician Augustus De Morgan, of the mid-nineteenth century.

TABLE 2 De Morgan's Laws.

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

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

Logical Equivalences: Example

- Show that $\neg(p \vee q)$ and $\neg p \wedge \neg q$ are logically equivalent.
- *Solution:* The truth tables for these compound propositions are displayed in Table 3. Because the truth values of the compound propositions $\neg(p \vee q)$ and $\neg p \wedge \neg q$ agree for all possible combinations of the truth values of p and q, it follows that $\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$ is a tautology and that these compound propositions are logically equivalent.

Logical Equivalences: Example

TABLE 3 Truth Tables for $\neg(p \vee q)$ and $\neg p \wedge \neg q$.

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

Logical Equivalences: Example

- Show that $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ are logically equivalent. This is the *distributive law* of disjunction over conjunction.

TABLE 5 A Demonstration That $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ Are Logically Equivalent.

p	q	r	$q \wedge r$	$p \vee (q \wedge r)$	$p \vee q$	$p \vee r$	$(p \vee q) \wedge (p \vee r)$
T	T	T	T	T	T	T	T
T	T	F	F	T	T	T	T
T	F	T	F	T	T	T	T
T	F	F	F	T	T	T	T
F	T	T	T	T	T	T	T
F	T	F	F	F	T	F	F
F	F	T	F	F	F	T	F
F	F	F	F	F	F	F	F

Logical Equivalences: Example

- The next Tables 6, 7, 8 contains some important equivalences. In these equivalences, $\boldsymbol{T}$ denotes the compound proposition that is always true and $\boldsymbol{F}$ denotes the compound proposition that is always false.

Logical Equivalences: Example

TABLE 6 Logical Equivalences.

Equivalence	*Name*
$p \wedge \mathbf{T} \equiv p$ $p \vee \mathbf{F} \equiv p$	Identity laws
$p \vee \mathbf{T} \equiv \mathbf{T}$ $p \wedge \mathbf{F} \equiv \mathbf{F}$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws

Logical Equivalences: Example

$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's laws
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv \mathbf{T}$ $p \wedge \neg p \equiv \mathbf{F}$	Negation laws

Logical Equivalences: Example

TABLE 7 Logical Equivalences Involving Conditional Statements.

$$p \to q \equiv \neg p \vee q$$

$$p \to q \equiv \neg q \to \neg p$$

$$p \vee q \equiv \neg p \to q$$

$$p \wedge q \equiv \neg(p \to \neg q)$$

$$\neg(p \to q) \equiv p \wedge \neg q$$

$$(p \to q) \wedge (p \to r) \equiv p \to (q \wedge r)$$

$$(p \to r) \wedge (q \to r) \equiv (p \vee q) \to r$$

$$(p \to q) \vee (p \to r) \equiv p \to (q \vee r)$$

$$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$$

TABLE 8 Logical Equivalences Involving Biconditional Statements.

$$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$$

$$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$$

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

$$\neg(p \leftrightarrow q) \equiv p \leftrightarrow \neg q$$

Using De Morgan's Laws

- The two logical equivalences known as De Morgan's laws are particularly important. They tell us how to negate conjunctions and how to negate disjunctions. In particular, the equivalence $\neg(p \vee q) \equiv \neg p \wedge \neg q$ tells us that the negation of a disjunction is formed by taking the conjunction of the negations of the component propositions. Similarly, the equivalence $\neg(p \wedge q) \equiv \neg p \vee \neg q$ tells us that the negation of a conjunction is formed by taking the disjunction of the negations of the component propositions.

Using De Morgan's Laws: Example

- Use De Morgan's laws to express the negations of "Asan has a cellphone and he has a laptop computer" and "Aizhan will go to the concert or Timur will go to the concert."
- *Solution:* Let p be "Asan has a cellphone" and q be "Asan has a laptop computer." Then "Asan has a cellphone and he has a laptop computer" can be represented by $p \wedge q$. By the first of De Morgan's laws, $\neg(p \wedge q)$ is equivalent to $\neg p \vee \neg q$.

Using De Morgan's Laws: Example

- Consequently, we can express the negation of our original statement as "Asan does not have a cellphone or he does not have a laptop computer." Let r be "Aizhan will go to the concert" and s be "Timur will go to the concert." Then "Aizhan will go to the concert or Timur will go to the concert" can be represented by $r \vee s$. By the second of De Morgan's laws, $\neg(r \vee s)$ is equivalent to $\neg r \wedge \neg s$. Consequently, we can express the negation of our original statement as "Aizhan will not go to the concert and Timur will not go to the concert."

Constructing New Logical Equivalences

- **EXAMPLE.** Show that $\neg\big(p \vee (\neg p \wedge q)\big)$ and $\neg p \wedge \neg q$ are logically equivalent by developing a series of logical equivalences.
- *Solution:* We will use one of the equivalences in Table 6 at a time, starting with $\neg(p \vee (\neg p \wedge q))$ and ending with $\neg p \wedge \neg q$. (*Note:* we could also easily establish this equivalence using a truth table.) We have the following equivalences.

Constructing New Logical Equivalences

$$
\begin{aligned}
\neg(p \vee (\neg p \wedge q)) &\equiv \neg p \wedge \neg(\neg p \wedge q) && \text{by the second De Morgan law} \\
&\equiv \neg p \wedge [\neg(\neg p) \vee \neg q] && \text{by the first De Morgan law} \\
&\equiv \neg p \wedge (p \vee \neg q) && \text{by the double negation law} \\
&\equiv (\neg p \wedge p) \vee (\neg p \wedge \neg q) && \text{by the second distributive law} \\
&\equiv \mathbf{F} \vee (\neg p \wedge \neg q) && \text{because } \neg p \wedge p \equiv \mathbf{F} \\
&\equiv (\neg p \wedge \neg q) \vee \mathbf{F} && \text{by the commutative law for disjunction} \\
&\equiv \neg p \wedge \neg q && \text{by the identity law for } \mathbf{F}
\end{aligned}
$$

- Consequently $\neg\big(p \vee (\neg p \wedge q)\big)$ and $\neg p \wedge \neg q$ are logically equivalent.

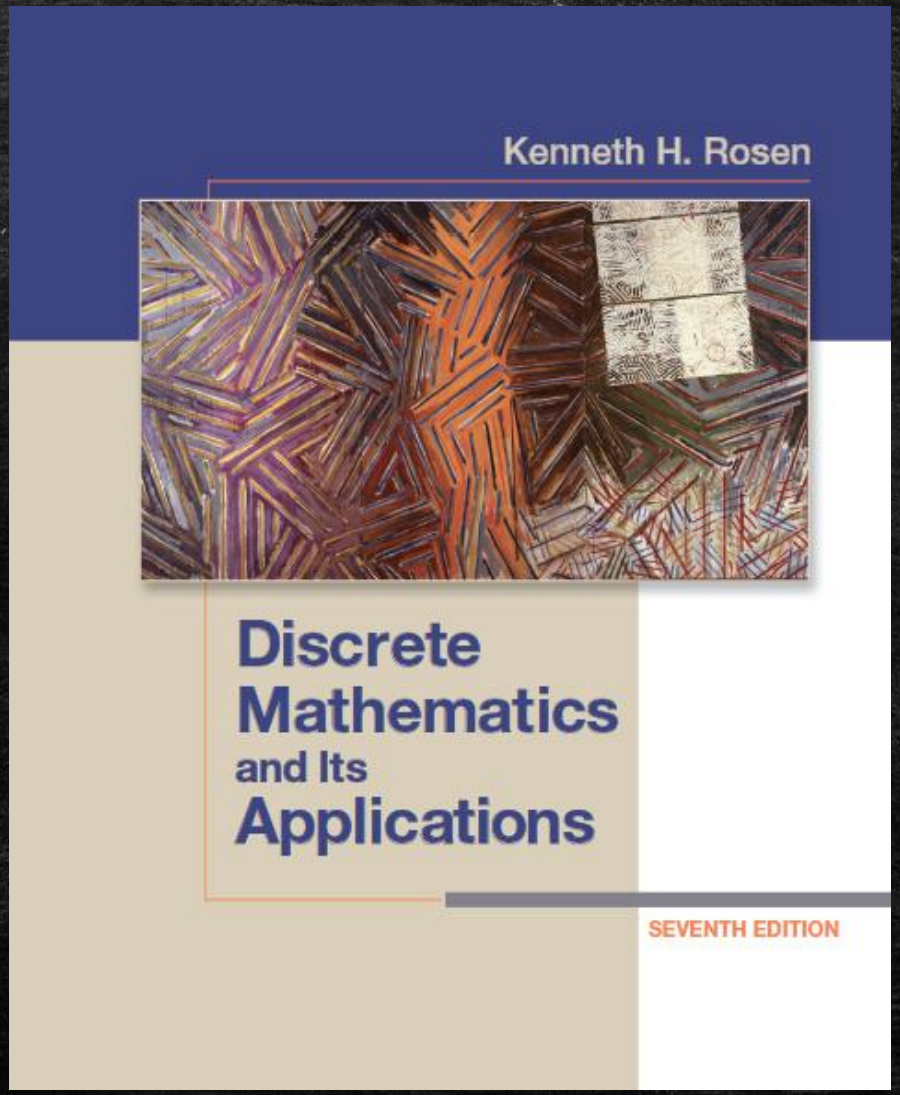
Propositional Satisfiability

- A compound proposition is **satisfiable** if there is an assignment of truth values to its variables that makes it true. When no such assignments exists, that is, when the compound proposition is false for all assignments of truth values to its variables, the compound proposition is **unsatisfiable**.
- Note that a compound proposition is unsatisfiable if and only if its negation is true for all assignments of truth values to the variables, that is, if and only if its negation is a tautology.

Propositional Satisfiability

- When we find a particular assignment of truth values that makes a compound proposition true, we have shown that it is satisfiable; such an assignment is called a **solution** of this particular satisfiability problem. However, to show that a compound proposition is unsatisfiable, we need to show that *every* assignment of truth values to its variables makes it false. Although we can always use a truth table to determine whether a compound proposition is satisfiable, it is often more efficient.

HOMEWORK: 4, 8, 10, 16, 18, 20, 24, 32, 62 Exercises on pp. 34-36



Predicates and Quantifiers

- Propositional logic cannot adequately express the meaning of all statements in mathematics and in natural language. For example, suppose that we know that

  "Every computer connected to the university network is functioning properly."

Predicates and Quantifiers

- Now we will introduce a more powerful type of logic called **predicate logic**. We will see how predicate logic can be used to express the meaning of a wide range of statements in mathematics and computer science in ways that permit us to reason and explore relationships between objects. To understand predicate logic, we first need to introduce the concept of a predicate.

Predicates

- Statements involving variables, such as

$$\text{“}x > 3\text{,”}\ \text{“}x = y + 3\text{,”}\ \text{“}x + y = z\text{,”}$$

and

“computer x is under attack by an intruder,”

and

“computer x is functioning properly,”

are often found in mathematical assertions, in computer programs, and in system specifications.

Predicates

- These statements are neither true nor false when the values of the variables are not specified. Here, we will discuss the ways that propositions can be produced from such statements.
- The statement "x is greater than 3" has two parts. The first part, the variable x, is the subject of the statement. The second part – the **predicate**, "is greater than 3" – refers to a property that the subject of the statement can have.

Predicates

- We can denote the statement "x is greater than 3" by $P(x)$, where P denotes the predicate "is greater than 3" and x is the variable. The statement $P(x)$ is also said to be the value of the **propositional function** P at x. Once a value has been assigned to the variable x, the statement $P(x)$ becomes a proposition and has a truth value.

Predicates: Example

- Let $P(x)$ denote the statement "$x > 3$." What are the truth values of $P(4)$ and $P(2)$?
- *Solution:* We obtain the statement $P(4)$ by setting $x = 4$ in the statement "$x > 3$." Hence, $P(4)$, which is the statement "$4 > 3$," is true. However, $P(2)$, which is the statement "$2 > 3$," is false.

Predicates: Example

- Let $Q(x, y)$ denote the statement "$x = y + 3$." What are the truth values of the propositions $Q(1, 2)$ and $Q(3, 0)$?
- *Solution:* To obtain $Q(1, 2)$, set $x = 1$ and $y = 2$ in the statement $Q(x, y)$. Hence, $Q(1, 2)$ is the statement "1 = 2 + 3," which is false. The statement $Q(3, 0)$ is the proposition "$3 = 0 + 3$," which is true.

Quantifiers

- When the variables in a propositional function are assigned values, the resulting statement becomes a proposition with a certain truth value. However, there is another important way, called **quantification**, to create a proposition from a propositional function. Quantification expresses the extent to which a predicate is true over a range of elements. In English, the words *all*, *some*, *many*, *none*, and *few* are used in quantifications. The area of logic that deals with predicates and quantifiers is called the **predicate calculus**.

Quantifiers

- **DEFINITION 1.** The *universal quantification* of $P(x)$ is the statement

  "$P(x)$ for all values of x in the domain."

- The notation $\forall x P(x)$ denotes the universal quantification of $P(x)$. Here $\forall$ is called the **universal quantifier.** We read $\forall x P(x)$ as "for all x $P(x)$" or "for every x $P(x)$." An element for which $P(x)$ is false is called a **counterexample** of $\forall x\ P(x)$.

Quantifiers: Example

- Let $P(x)$ be the statement "$x + 1 > x$." What is the truth value of the quantification $\forall x\, P(x)$, where the domain consists of all real numbers?
- *Solution:* Because $P(x)$ is true for all real numbers x, the quantification $\forall x\, P(x)$ is true.

Quantifiers

- ***Remark:*** Generally, an implicit assumption is made that all domains of discourse for quantifiers are nonempty. Note that if the domain is empty, then $\forall x\, P(x)$ is true for any propositional function $P(x)$ because there are no elements x in the domain for which $P(x)$ is false.
- Besides "for all" and "for every," universal quantification can be expressed in many other ways, including "all of," "for each," "given any," "for arbitrary," "for each," and "for any."

Quantifiers: Example

- Let $Q(x)$ be the statement "$x < 2$." What is the truth value of the quantification $\forall x\, Q(x)$, where the domain consists of all real numbers?
- *Solution:* $Q(x)$ is not true for every real number x, because, for instance, $Q(3)$ is false. That is, $x = 3$ is a counterexample for the statement $\forall x\, Q(x)$. Thus $\forall x\, Q(x)$ is false.

Quantifiers

- **DEFINITION 2.** The *existential quantification* of $P(x)$ is the proposition

  "There exists an element x in the domain such that $P(x)$."

- We use the notation $\exists x\ P(x)$ for the existential quantification of $P(x)$. Here $\exists$ is called the *existential quantifier*.

Quantifiers

- A domain must always be specified when a statement $\exists x\ P(x)$ is used. Furthermore, the meaning of $\exists x\ P(x)$ changes when the domain changes. Without specifying the domain, the statement $\exists x\ P(x)$ has no meaning.
- Besides the phrase "there exists," we can also express existential quantification in many other ways, such as by using the words "for some," "for at least one," or "there is."

Quantifiers

- The existential quantification $\exists x\ P(x)$ is read as

"There is an x such that $P(x)$,"

"There is at least one x such that $P(x)$,"

or

"For some x $P(x)$."

Quantifiers: Example

- Let $P(x)$ denote the statement "$x > 3$." What is the truth value of the quantification $\exists x\, P(x)$, where the domain consists of all real numbers?
- *Solution:* Because "$x > 3$" is sometimes true—for instance, when $x = 4$ – the existential quantification of $P(x)$, which is $\exists x\, P(x)$, is true.

Quantifiers

- Observe that the statement $\exists x\ P(x)$ is false if and only if there is no element x in the domain for which $P(x)$ is true. That is, $\exists x\ P(x)$ is false if and only if $P(x)$ is false for every element of the domain.
- ***Remark:*** Generally, an implicit assumption is made that all domains of discourse for quantifiers are nonempty. If the domain is empty, then $\exists x\ Q(x)$ is false whenever $Q(x)$ is a propositional function because when the domain is empty, there can be no element x in the domain for which $Q(x)$ is true.

Quantifiers

- When all elements in the domain can be listed – say, $x_1, x_2, \ldots, x_n$ – the existential quantification $\exists x\, P(x)$ is the same as the disjunction

$$P(x_1) \vee P(x_2) \vee \ \ldots \ \vee P(x_n),$$

because this disjunction is true if and only if at least one of $P(x_1), P(x_2), \ldots, P(x_n)$ is true.

Precedence of Quantifiers

- The quantifiers ∀ and ∃ have higher precedence than all logical operators from propositional calculus. For example, $\forall x P(x) \vee Q(x)$ is the disjunction of $\forall x\ P(x)$ and $Q(x)$. In other words, it means $(\forall x\ P(x)) \vee Q(x)$ rather than $\forall x(P(x) \vee Q(x))$.

Binding Variables

- When a quantifier is used on the variable x, we say that this occurrence of the variable is **bound**. An occurrence of a variable that is not bound by a quantifier or set equal to a particular value is said to be **free**. All the variables that occur in a propositional function must be bound or set equal to a particular value to turn it into a proposition. This can be done using a combination of universal quantifiers, existential quantifiers, and value assignments.

Binding Variables: Example

- In the statement

$$\exists x(x + y = 1),$$

- the variable x is bound by the existential quantification $\exists x$, but the variable y is free because it is not bound by a quantifier and no value is assigned to this variable. This illustrates that in the statement $\exists x(x + y = 1)$, x is bound, but y is free.

Negating Quantified Expressions

- We will often want to consider the negation of a quantified expression. For instance, consider the negation of the statement

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

This statement is a universal quantification, namely, $\forall x P(x)$, where $P(x)$ is the statement "x has taken a course in calculus" and the domain consists of the students in your class.

Negating Quantified Expressions

- The negation of this statement is “It is not the case that every student in your class has taken a course in calculus.” This is equivalent to “There is a student in your class who has not taken a course in calculus.” And this is simply the existential quantification of the negation of the original propositional function, namely,

$$\exists x \neg P(x).$$

Negating Quantified Expressions

- This example illustrates the following logical equivalence:

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

- To show that $\neg\forall x\, P(x)$ and $\exists x \neg P(x)$ are logically equivalent no matter what the propositional function $P(x)$ is and what the domain is, first note that $\neg\forall x\, P(x)$ is true if and only if $\forall x\, P(x)$ is false. Next, note that $\forall x\, P(x)$ is false if and only if there is an element x in the domain for which $P(x)$ is false.

Negating Quantified Expressions

- This holds if and only if there is an element x in the domain for which $\neg P(x)$ is true. Finally, note that there is an element x in the domain for which $\neg P(x)$ is true if and only if $\exists x \neg P(x)$ is true. Putting these steps together, we can conclude that $\neg \forall x \, P(x)$ is true if and only if $\exists x \neg P(x)$ is true. It follows that $\neg \forall x \, P(x)$ and $\exists x \neg P(x)$ are logically equivalent.

Negating Quantified Expressions

TABLE 2 De Morgan's Laws for Quantifiers.

Negation	*Equivalent Statement*	*When Is Negation True?*	*When False?*
$\neg\exists x\,P(x)$	$\forall x \neg P(x)$	For every x, $P(x)$ is false.	There is an x for which $P(x)$ is true.
$\neg\forall x\,P(x)$	$\exists x \neg P(x)$	There is an x for which $P(x)$ is false.	$P(x)$ is true for every x.

Negating Quantified Expressions: Example

- What are the negations of the statements $\forall x(x^2 > x)$ and $\exists x(x^2 = 2)$?
- *Solution:* The negation of $\forall x(x^2 > x)$ is the statement $\neg\forall x(x^2 > x)$, which is equivalent to $\exists x\neg(x^2 > x)$. This can be rewritten as $\exists x(x^2 \leq x)$. The negation of $\exists x(x^2 = 2)$ is the statement $\neg\exists x(x^2 = 2)$, which is equivalent to $\forall x\neg(x^2 = 2)$. This can be rewritten as $\forall x(x^2 \neq 2)$. The truth values of these statements depend on the domain.

HOMEWORK: Exercises 2, 8, 10, 12, 14, 16, 20, 26, 32, 34, 52 on pp. 53-55