جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

Discrete Mathematics

Lecture 2

Applications of Propositional Logic, Propositional Equivalences

By

Asst. Lect. Ali Saleem Haleem

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. Furthermore, propositional logic and its rules can be used to design computer circuits, to construct computer programs, to verify the correctness of programs, and to build expert systems. Logic can be used to analyze and solve many familiar puzzles.

1. Translating English Sentences

There are many reasons to translate English sentences into expressions involving propositional variables and logical connectives. In particular, English (and every other human language) is often ambiguous. Translating sentences into compound statements (and other types of logical expressions) removes the ambiguity. Note that this may involve making a set of reasonable assumptions based on the intended meaning of the sentence. Moreover, 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.

EXAMPLE 1 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. 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 $a \rightarrow (c \vee \neg f)$.

EXAMPLE 2 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 $(r \wedge \neg s) \rightarrow \neg q$. Of course, there are other ways to represent the original sentence as a logical expression, but the one we have used should meet our needs.

2. Logic Circuits

Propositional logic can be applied to the design of computer hardware. This was first observed in 1938 by Claude Shannon in his MIT master's thesis. A logic circuit (or digital circuit) receives input signals ***p1, p2, ..., pn***, each a bit [either 0 (off) or 1 (on)], and produces output signals ***1, s2..., sn***, each a bit. In this section we will restrict our attention to logic circuits with a single output signal; in general, digital circuits may have multiple outputs. Complicated digital circuits can be constructed from three basic circuits, called ***gates***, shown in Figure 1. The inverter, or NOT gate, takes an input bit ***p,*** and produces as output $\neg p$. The OR gate takes two input signals ***p*** and ***q***, each a bit, and produces as output the signal $p \vee q$. Finally, the AND gate takes two input signals ***p*** and ***q***, each a bit, and produces as output the signal $p \wedge q$. We use combinations of these three basic gates to build more complicated circuits, such as that shown in Figure 2.

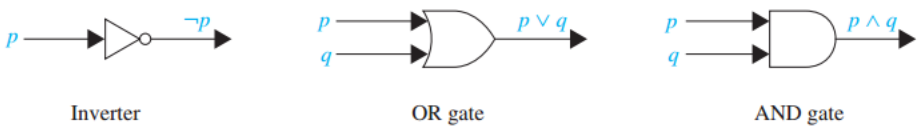

FIGURE 1 **Basic logic gates.**

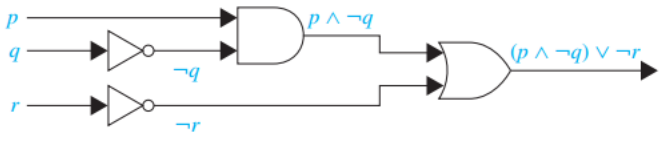

FIGURE 2 **A combinatorial circuit.**

EXAMPLE 3 Determine the output for the combinatorial circuit in Figure 2.

Solution: In Figure 2 we display the output of each logic gate in the circuit. We see that the AND gate takes input of p and ¬q, the output of the inverter with input q, and produces p∧¬q. Next, we note that the OR gate takes input p∧¬q and ¬r, the output of the inverter with input r, and produces the final output(p∧¬q) ∨¬r.

EXAMPLE 4 Build a digital circuit that produces the output (p∨¬r) ∧(¬p∨(q∨¬r)) when given input bits p, q, and r.

Solution: To construct the desired circuit, we build separate circuits for p∨¬r and for ¬p ∨ (q∨¬r) and combine them using an AND gate. To construct a circuit for p∨¬r, we use an inverter to produce ¬r from the input r. Then, we use an OR gate to combine p and ¬r. To build a circuit for ¬p∨(q∨¬r), we first use an inverter to obtain ¬r. Then we use an OR gate with inputs q and ¬r to obtain q∨¬r. Finally, we use another inverter and an OR gate to get ¬p∨(q∨¬r) from the inputs p and q∨¬r.

To complete the construction, we employ a final AND gate, with inputs p∨¬r and ¬p∨ (q∨¬r). The resulting circuit is displayed in Figure 3.

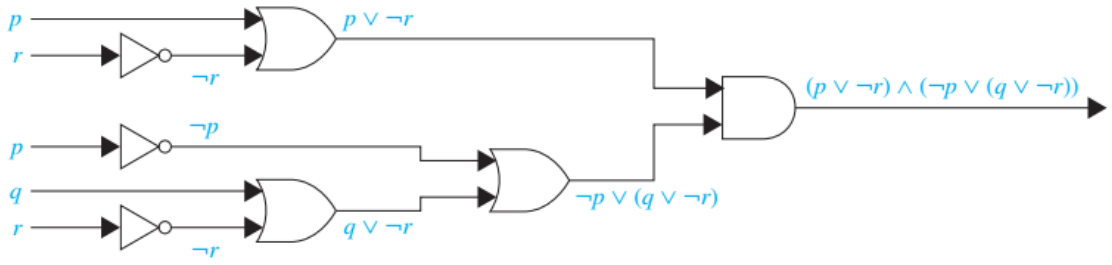

FIGURE 3 **The circuit for $(p \vee \neg r) \wedge (\neg p \vee (q \vee \neg r))$.**

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 ∧ q. We begin our discussion with a classification of compound propositions according to their possible truth values.

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*.

Tautologies and contradictions are often important in mathematical reasoning. Example 1 illustrates these types of compound propositions.

EXAMPLE 5 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.

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.

Not: 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.