Based on K. H. Rosen: Discrete Mathematics and its Applications.

Lecture 2: Translation into propositional language. Boolean searches. Sections 1.2

1 Translation into propositional language. Boolean searches.

Day-to-day language is ambiguous. Translating sentences into compound statements removes the ambiguity. Once we have translated sentences from English into logical expressions we can analyze these logical expressions to determine their truth values.

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."
Let a, b 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

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

What about if we change the sentence to:
"You can access the Internet from campus **if** you are a computer science major or you are not a freshman."
Now being a computer science major or not a freshman is a sufficient condition to be able to access the internet, therefore:

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

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."
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. The most common way to translate "unless" is to use instead **"if not"**. With this in mind our sentence translates into
"If you are not older than 16 years, then you cannot ride the roller coaster if you are under 4 feet tall".
Then, the sentence can be translated to

$$\neg s \to (r \to \neg q)$$

or

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

Example 3. 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.
Express the specification "The automated reply cannot be sent when the file system is full" using logical connectives.
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 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.

Example 4. 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."
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. 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.

Example 5. Are the specifications above still consistent if we add "The diagnostic message is not retransmitted" to our list?

Logical connectives are used extensively in searches of large collections of information, such as indexes of Web pages. Because these searches employ techniques from propositional logic, they are called Boolean searches. In Boolean searches, the connective AND is used to match records that contain both of two search terms, the connective OR is used to match one or both of two search terms, and the connective NOT (sometimes written as AND NOT) is used to exclude a particular search term.

Questions:

(1) An explorer is captured by a group of cannibals. There are two types of cannibals—those who always tell the truth and those who always lie. The cannibals will barbecue the explorer unless he can determine whether a particular cannibal always lies or always tells the truth. He is allowed to ask the cannibal exactly one question.
(a) Explain why the question "Are you a liar?" does not work.
(b) Find a question that the explorer can use to determine whether the cannibal always lies or always tells the truth.