From English to Math in 3 Steps:

A Quick Guide to Writing Logical Expressions on Binary Variables as Linear Inequalities

by Tallys Yunes (updated March 2008)
Department of Management Science, University of Miami, Coral Gables, FL 33124-8237

1 Notation and Terminology (STEP 1)

In this guide, I assume you are familiar with Integer Linear Programming and the use of binary variables to express yes/no or true/false decisions.

Consider the following statement: "I will either go to the movies or watch TV." There are two parts of the statement that can be true (represented by the value 1) or false (represented by the value 0). Let the binary variable x be equal to 1 if I go to the movies and equal to 0 otherwise. Similarly, let the binary variable y indicate whether or not I watch TV. So I can rewrite that sentence as "x or y". In mathematics, we use the symbol "$\vee$" to represent "or" and the symbol "$\wedge$" to represent "and". Hence, we can write "$x \vee y$". If the sentence were "I will either not go to the movies or watch TV", we could have rewritten it in math as "$\neg x \vee y$", where the symbol "$\neg$" represents negation.

Now consider a more elaborate sentence: "If it rains, the streets get wet." This is of the form "if A then B", which is called an *implication*. The mathematical symbol for an implication is "$\Rightarrow$". So we can rewrite the last sentence in math as "$r \Rightarrow w$", where the binary variables r and w represent the rain and wetness conditions, respectively. Note that "A only if B" translates into math as "$A \Rightarrow B$", not "$B \Rightarrow A$".

Not all implications work in both directions. In the previous example, it is not correct to say "$w \Rightarrow r$" because the streets might be wet for another reason other than the rain. When an implication works both ways, it is an "if-and-only-if", whose mathematical symbol is "$\Leftrightarrow$".

Expressions that contain binary variables and the logical operators "$\vee$", "$\wedge$", "$\neg$", "$\Rightarrow$", and "$\Leftrightarrow$" are called logical expressions[1]. A *disjunction* is a logical expression of the form $E_1 \vee E_2 \vee \cdots \vee E_n$, where $n \geq 1$ and each E_i is either a single variable or a larger logical expression in parenthesis. Similarly, a *conjunction* is a logical expression of the form $E_1 \wedge E_2 \wedge \cdots \wedge E_n$.

<u>STEP 1</u>: Write the condition you are trying to model as a logical expression.

Example (a): "If it rains or snows tomorrow, then I will either go to the theater or rent a movie." Use variables r, s, t, m to represent rain, snow, going to the theater, and renting a movie. The corresponding logical expression is $r \vee s \Rightarrow t \vee m$.

2 Properties of Logical Operators

The logical operators mentioned above have several properties that are important for the type of manipulations we need to be able to perform (see Section 3). The most relevant ones are listed below. Note that A, B and C can be single variables (possibly negated) or larger expressions inside parentheses.

[1]This is an abuse of notation/terminology because, in general, logical expressions can be a lot more complicated than this. For the purposes of this explanation, however, let's pretend we don't know that.

1. $\neg(\neg A) = A$
2. $\neg(A \vee B) = \neg A \wedge \neg B$
3. $\neg(A \wedge B) = \neg A \vee \neg B$
4. $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$
5. $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$
6. $A \Rightarrow B = \neg A \vee B$
7. $A \Leftrightarrow B = (A \Rightarrow B) \wedge (B \Rightarrow A)$

The second and third properties above are known as De Morgan's rules. The fourth and fifth properties are the distributive rules.

The above properties allow us to rewrite a logical expression into another logical expression that is *equivalent* to it. This will be useful in the next section. By the way, "$\wedge$" has priority over "$\vee$". This means that the expression $x \vee y \wedge z$ is equivalent to $x \vee (y \wedge z)$, not $(x \vee y) \wedge z$.

3 Conjunctive Normal Form (STEP 2)

Clearly, logical expressions can assume many different forms. We are interested in one specific form called Conjunctive Normal Form (CNF). The reason for that will be made clear in Section 4.

We say that a logical expression is in CNF if it is a conjunction of disjunctions. For example, the expression x is in CNF because it is a conjunction of one disjunction that contains only one variable. Other examples of expressions in CNF are: $x \wedge y$, $(x \vee y) \wedge \neg z$, and $(x \vee y) \wedge (y \vee z) \wedge (z \vee \neg w)$. Examples of expressions that are *not* in CNF are: $x \vee y \wedge z$ and $(x \wedge y) \vee (y \wedge z)$.

To convert any logical expression into CNF, we use the properties described in Section 2.

STEP 2: Once you have written your English statement (i.e. the condition you are trying to model) as a logical expression, you must convert it into CNF.

Example (a) continued: The expression $r \vee s \Rightarrow t \vee m$ is not in CNF. We'll first get rid of the "$\Rightarrow$" by using property (6) of Section 2. This gives $\neg(r \vee s) \vee (t \vee m)$. Then we use property (2) to get $(\neg r \wedge \neg s) \vee (t \vee m)$, which is equivalent to $(t \vee m) \vee (\neg r \wedge \neg s)$. To use property (5), we let $A = (t \vee m)$, $B = \neg r$ and $C = \neg s$. After distributing terms, we get $(t \vee m \vee \neg r) \wedge (t \vee m \vee \neg s)$, which happens to be in CNF.

4 From CNF to Linear Inequalities (STEP 3)

Once you have converted your logical expression into CNF, you are basically done. You just need to follow the steps below.

1. Every disjunction in the CNF expression will give rise to one inequality of the form LHS ≥ 1, where LHS is the left-hand side of the inequality;

2. For each disjunction, the left-hand side of the corresponding linear inequality is constructed as follows:

(a) Replace every "$\vee$" by a "+";

(b) Every variable that is not negated appears on the left-hand side as itself;

(c) Every variable that is negated appears on the left-hand side as 1 minus itself;

3. Optionally, when you are finished, rearrange terms so that all constants are moved to the right-hand sides of each inequality.

STEP 3: Use the above explanation to convert your expression from CNF to a system of linear inequalities and you are done!

Example (a) continued: Because the expression $(t \vee m \vee \neg r) \wedge (t \vee m \vee \neg s)$ is a conjunction of two disjunctions, it will give rise to two linear constraints: $t + m + 1 - r \geq 1$ and $t + m + 1 - s \geq 1$, which are equivalent to $t + m - r \geq 0$ and $t + m - s \geq 0$.

4.1 Additional Examples

Consider the expression $\neg x \vee y \vee \neg z$. Since this is a conjunction of only one disjunction, it is equivalent to a single inequality. Because x and z are negated, they appear on the left as $1 - x$ and $1 - z$, whereas y will appear as itself. This gives $(1 - x) + y + (1 - z) \geq 1$. After moving all 1's to the right, we have $-x + y - z \geq -1$, or equivalently, $x - y + z \leq 1$.

Consider the logical expression $(\neg x \vee y) \wedge (y \vee z) \wedge (x \vee w \vee \neg z)$. Because it is a conjunction of 3 disjunctions, this logical expression is equivalent to a system of 3 inequalities, as follows

$$\begin{aligned} (1 - x) + y &\geq 1 \\ y + z &\geq 1 \\ x + w + (1 - z) &\geq 1 \end{aligned}$$

Although following these 3 steps may seem too complicated at first sight, once you get some practice you will realize that the whole process is quite simple.

5 Practice Problems

Convert the sentences below into systems of linear inequalities on binary variables.

(b) I will play basketball and go to the gym only if I get an A+ in Math and an A in History.

(c) If a factory is built in cities A and B, or no factory is built in city C, then a factory must be built in city D and no factory can be built in city E.

6 Solution to Practice Problems

The next page contains solutions to the above problems. Try them by yourself first before you turn the page!

These are possible ways to solve the practice problems. There are other sequences of transformations that take you to the same end result.

(b) $b \wedge g \Rightarrow m \wedge h$
$(\neg b \vee \neg g) \vee (m \wedge h)$
$(\neg b \vee \neg g \vee m) \wedge (\neg b \vee \neg g \vee h)$

This generates two constraints:
$1 - b + 1 - g + m \geq 1$
$1 - b + 1 - g + h \geq 1$

Rearranging terms we get:
$1 + m \geq b + g$
$1 + h \geq b + g$

Now check the original condition and convince yourself that the above constraints make sense.

(c) $(a \wedge b) \vee (\neg c) \Rightarrow d \wedge \neg e$
$((\neg a \vee \neg b) \wedge c) \vee (d \wedge \neg e)$
$(((\neg a \vee \neg b) \wedge c) \vee d) \wedge (((\neg a \vee \neg b) \wedge c) \vee \neg e)$
$(((\neg a \vee \neg b) \vee d) \wedge (c \vee d)) \wedge (((\neg a \vee \neg b) \vee \neg e) \wedge (c \vee \neg e))$
$(\neg a \vee \neg b \vee d) \wedge (c \vee d) \wedge (\neg a \vee \neg b \vee \neg e) \wedge (c \vee \neg e)$

This generates four constraints:
$1 - a + 1 - b + d \geq 1$
$c + d \geq 1$
$1 - a + 1 - b + 1 - e \geq 1$
$c + 1 - e \geq 1$

Rearranging terms we get:
$1 + d \geq a + b$
$c + d \geq 1$
$a + b + e \leq 2$
$c \geq e$

Now check the original condition and convince yourself that the above constraints make sense.