This is from Chapter 2 of textbook “Logic in Action” (www.logicinaction.org)

2.5 Semantic Situations, Truth Tables, Binary Arithmetic

Differences in formal syntax often correspond to differences in meaning: the above two trees are an example. To explain this in more detail, we now need a semantics that, for a start, relates syntactic objects like formulas to truth and falsity in semantic situations. Thus, formulas acquire meaning in specific settings, and differences in meaning between formulas are often signalled by differences in truth in some situation.

Truth values and valuations for atoms As we said already, each set of proposition letters $p, q, r, \ldots$ generates a set of different *situations*, different ways the actual world might be, or different states that it could be in (all these interpretations make sense in applications). Three proposition letters generate $2^3 = 8$ situations:

$$\{pqr, pq\overline{r}, p\overline{q}r, p\overline{q}\overline{r}, \overline{p}qr, \overline{p}q\overline{r}, \overline{p}\overline{q}r, \overline{p}\overline{q}\overline{r}\} \tag{2.17}$$

Here proposition letters stand for 'atomic propositions', while logical operations form 'molecules'. Of course this is just a manner of speaking, since what counts as 'atomic' in a given application is usually just our decision 'not to look any further inside' the proposition. A convenient mathematical view of situations is as functions from atomic propositions to truth values 1 ('true') and 0 ('false'). For instance, the above situation $p\overline{q}r$ corresponds to the function sending p to 1, q to 0, and r to 1. An alternative notation for truth values is t and f, but we use numbers for their suggestive analogy with binary arithmetic (the heart of computers). We call these functions V *valuations*; $V(\varphi) = 1$ says that the formula φ is true in the situation (represented by) V, and $V(\varphi) = 0$ says that the formula φ is false in the situation V. For $V(\varphi) = 1$ we also write $V \models \varphi$ and for $V(\varphi) = 0$ we also write $V \not\models \varphi$. One can read $V \models \varphi$ as "V makes true φ", or as "V *satisfies* φ" or "V is a *model* of φ". The notation using $\models$ will reappear in later chapters.

Boolean operations on truth values Any complex sentence constructed from the relevant atomic proposition letters is either true or false in each situation. To see how this works, we first need an account for the meaning of the logical operations. This is achieved by assigning them Boolean operations on the numbers 0, 1, in a way that respects (as far as reasonable) their intuitive usage. For instance, if $V(\varphi) = 0$, then $V(\neg\varphi) = 1$, and vice versa; and if $V(\varphi) = 1$, then $V(\neg\varphi) = 0$, and vice versa. Such relations are easier formatted in a table.

Definition 2.10 (Semantics of propositional logic) A valuation V is a function from proposition letters to truth values 0 and 1. The value or meaning of complex sentences is computed from the value of basic propositions according to the following truth tables.

φ	$\neg\varphi$
0	**1**
1	**0**

φ	ψ	$\varphi \wedge \psi$	$\varphi \vee \psi$	$\varphi \rightarrow \psi$	$\varphi \leftrightarrow \psi$
0	0	**0**	**0**	**1**	**1**
0	1	**0**	**1**	**1**	**0**
1	0	**0**	**1**	**0**	**0**
1	1	**1**	**1**	**1**	**1**

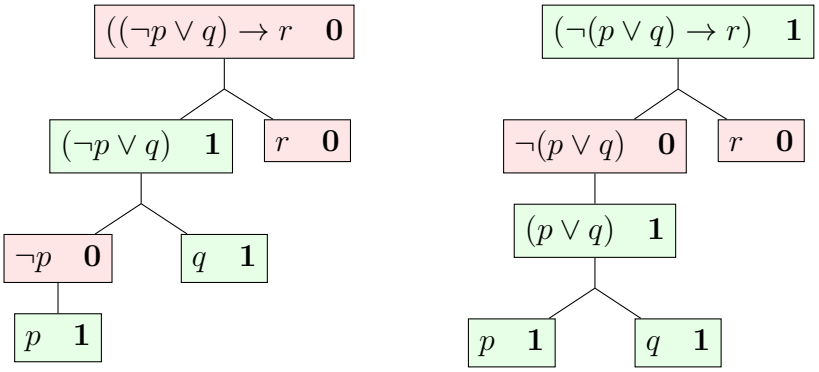
(2.18)

Bold-face numbers give the truth values for all relevant combinations of argument values: four in the case of connectives with two arguments, two in the case of the connective with one argument, the negation.

Explanation The tables for negation, conjunction, disjunction, and equivalence are quite intuitive, but the same does not hold for the table for implication. The table for implication has generated perennial debate, since it does not match the word 'implies' in natural language very well. E.g., does having a false *antecedent* (condition) φ and a true *consequent* ψ really make the implication if-φ-then-ψ true? But we are just doing the best we can in our simple two-valued setting. Here is a thought that has helped many students. You will certainly accept the following assertion as true: 'All numbers greater than 13 are greater than 12'. Put differently, 'if a number n is greater than 13 (p), then n is greater than 12 (q)'. But now, just fill in different numbers n, and you get all combinations in the truth table. For instance, $n = 14$ motivates the truth-value 1 for $p \to q$ at pq, $n = 13$ motivates 1 for $p \to q$ at $\overline{p}q$, and $n = 12$ motivates 1 for $p \to q$ at $\overline{pq}$.

A mismatch with natural language can actually be very useful. Conditionals are a 'hot spot' in logic, and it is a challenge to create systems that get closer to their behaviour. Propositional logic is the simplest treatment that exists, but other logical systems today deal with further aspects of conditionals in natural language and ordinary reasoning. You will see a few examples later in this course.

Computing truth tables for complex formulas How exactly can we compute truth values for complex formulas? This is done using our tables by following the construction stages of syntax trees. Here is how this works. Take the valuation V with $V(p) = V(q) = 1, V(r) = 0$ and consider two earlier formulas:



Incidentally, this difference in truth value explains our earlier point that these two variant formulas are different readings of the earlier natural language sentence.

Computing in this manner for all valuations, we can systematically tabulate the truth value

behaviour of complex propositional formulas in all relevant situations:

p	q	r	$((\neg p \vee q) \rightarrow r)$	$(\neg(p \vee q) \rightarrow r)$
0	0	0	**0**	**0**
0	0	1	**1**	**1**
0	1	0	**0**	**1**
0	1	1	**1**	**1**
1	0	0	**1**	**1**
1	0	1	**1**	**1**
1	1	0	**0**	**1**
1	1	1	**1**	**1**

(2.19)

Paying attention to the proper placement of brackets in formulas, you can compute truth-tables step by step for all situations. As an example we take the second formula from (2.19). First, start with summing up the situations and copy the truth-values under the proposition letters as has been done in the following table.

p	q	r	$(\neg$	$(p$	$\vee$	$q)$	$\rightarrow$	$r)$
0	0	0	$\cdot$	0	$\cdot$	0	$\cdot$	0
0	0	1	$\cdot$	0	$\cdot$	0	$\cdot$	1
0	1	0	$\cdot$	0	$\cdot$	1	$\cdot$	0
0	1	1	$\cdot$	0	$\cdot$	1	$\cdot$	1
1	0	0	$\cdot$	1	$\cdot$	0	$\cdot$	0
1	0	1	$\cdot$	1	$\cdot$	1	$\cdot$	1
1	1	0	$\cdot$	1	$\cdot$	0	$\cdot$	0
1	1	1	$\cdot$	1	$\cdot$	1	$\cdot$	1

(2.20)

Then start filling in the truth-values for the first possible operator. Here it is the disjunction: it can be computed because the values of its arguments are given (you can also see this from the construction tree). $(p \vee q)$ gets value 0 if and only if both p and q have the value 0. The intermediate result is given in the first table in (2.21). The next steps are the

negation and then the conjunction. This gives the following results:

$(\neg$	$(p$	$\vee$	$q)$	$\rightarrow$	$r)$
$\cdot$	0	0	0	$\cdot$	0
$\cdot$	0	0	0	$\cdot$	1
$\cdot$	0	1	1	$\cdot$	0
$\cdot$	0	1	1	$\cdot$	1
$\cdot$	1	1	0	$\cdot$	0
$\cdot$	1	1	0	$\cdot$	1
$\cdot$	1	1	1	$\cdot$	0
$\cdot$	1	1	1	$\cdot$	1

$(\neg$	$(p$	$\vee$	$q)$	$\rightarrow$	$r)$
1	0	0	0	$\cdot$	0
1	0	0	0	$\cdot$	1
0	0	1	1	$\cdot$	0
0	0	1	1	$\cdot$	1
0	1	1	0	$\cdot$	0
0	1	1	0	$\cdot$	1
0	1	1	1	$\cdot$	0
0	1	1	1	$\cdot$	1

$(\neg$	$(p$	$\vee$	$q)$	$\rightarrow$	$r)$
1	0	0	0	**0**	0
1	0	0	0	**1**	1
0	0	1	1	**1**	0
0	0	1	1	**1**	1
0	1	1	0	**1**	0
0	1	1	0	**1**	1
0	1	1	1	**1**	0
0	1	1	1	**1**	1

(2.21)

One does not have to draw three separate tables. All the work can be done in a single table. We just meant to indicate the right order of filling in truth-values.

Exercise 2.11 Construct truth tables for the following formulas:

- $(p \rightarrow q) \vee (q \rightarrow p)$,
- $((p \vee \neg q) \wedge r) \leftrightarrow (\neg(p \wedge r) \vee q)$.

Exercise 2.12 Using truth tables, investigate all formulas that can be readings of

$$\neg p \rightarrow q \vee r$$

(by inserting brackets in appropriate places), and show that they are not equivalent.

If, Only If, If and Only If Here is a useful list of different ways to express implications:

If p then q	$p \rightarrow q$
p if q	$q \rightarrow p$
p only if q	$p \rightarrow q$

The third item on this list may come as a surprise. To see that the third item is correct, reflect on how one can check whether "We will help you only if you help us" is false. This can can happen only if "We help you" is true, but "You help us" is false.

These uses of 'if' and 'only if' explain the use of the common abbreviation 'if and only if' for an equivalence. "We will help you if and only if you help us" states that "you help us" implies "we help you", and vice versa. A common abbreviation for 'if and only if' that we will use occasionally is *iff*.