2.16. Construction Meets Semantics: Building Sentences, Building Truth Tables

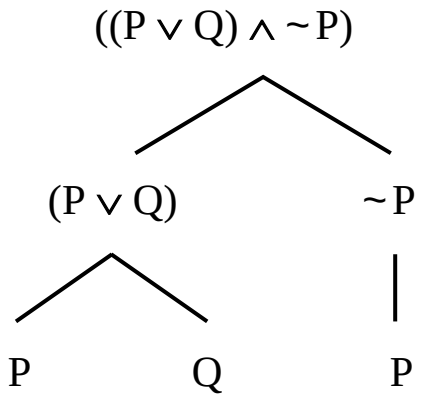
> "…logic explores the truth conditions of sentences in the light of how the sentences are grammatically constructed. Logic chases truth up the tree of grammar."
>
> –W. V. Quine, **Philosophy of Logic** p. 35

1. Truth and Construction. So far we've built truth tables only for fairly simple sentences. But since our goal in formal logic is to develop techniques which work equally in complex cases, where our intuitions are overwhelmed, we need a method for building truth tables even for complex formal sentences. Here **sentence construction** is a reliable guide.

Consider this formal sentence.

$$((P \vee Q) \wedge \sim P)$$

Its construction tree explains how this sentence was built up from atoms.



The truth table for this sentence will begin the same way: with sentence letters. 'Skimming off' the sentence letters from the bottom of the tree, we find only "P" and "Q" as atoms ("P" appearing twice).

So the truth table for this sentence likewise begins with “P” and “Q”.

P	**Q**

Since there are two sentence letters, we need **four** valuations.

2 x 2 = 4

P	**Q**

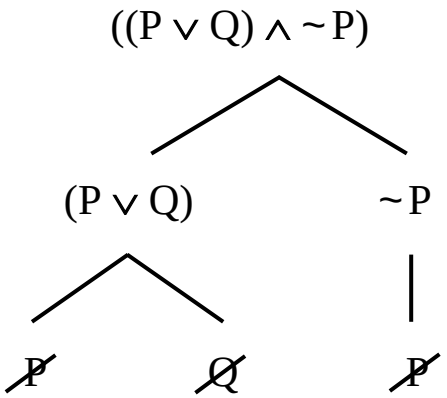
As always, we start with the **right-most** sentence letter, alternating “1” and “0” for the required number of times.

P	**Q**
	1
	0
	1
	0

Since there’s another sentence letter to the left of “Q,” we **double up** the numbers of 1’s and 0’s: with “P” we use **two** of each.

P	Q
1	1
1	0
0	1
0	0

Next in the construction of this sentence, "P" and "Q" are joined into the disjunction "(P ∨ Q").



So the truth table does the same.

P	**Q**	**(P ∨ Q)**
1	1	
1	0	
0	1	
0	0	

Here we appeal to the semantic rule for disjunctions: disjunctions are **only false when both their parts are false**.

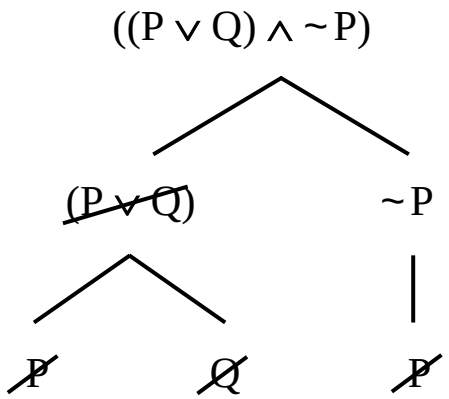
Disjunction Rule:

●	▲	(● ∨ ▲)
1	1	1
1	0	1
0	1	1
0	**0**	**0**

“(P ∨ Q)” has both parts false only in the last valuation. So it’s false there, and true in the other three valuations.

P	**Q**	**(P ∨ Q)**
1	1	**1**
1	0	**1**
0	1	**1**
0	0	**0**

Next in construction, “~P” was built from the atom “P”.



Once again the truth table matches sentence construction, step for step.

P	**Q**	**(P ∨ Q)**	**~P**
1	1	1	
1	0	1	
0	1	1	
0	0	0	

“~P” follows the semantic rule for negations.

Negation Rule

▲	~▲
1	0
0	1

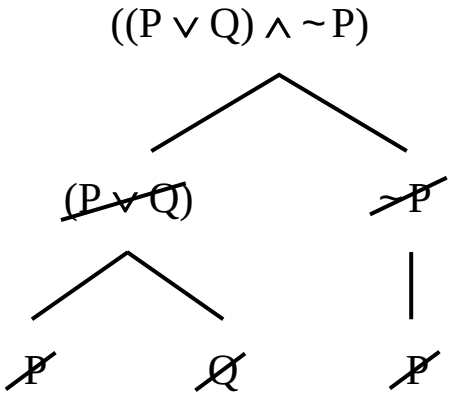
Where “P” is true (the first and second valuations), “~P” is false.

P	**Q**	**(P ∨ Q)**	**~P**
1	1	1	**0**
1	0	1	**0**
0	1	1	
0	0	0	

Where “P” is false (the third and fourth valuations), “~P” is true.

P	**Q**	**(P ∨ Q)**	**~P**
1	1	1	0
1	0	1	0
0	1	1	**1**
0	0	0	**1**

The finished sentence, “((P ∨ Q) ∧ ~P),” is the last construction step.



This sentence is added to the truth table.

P	Q	(P ∨ Q)	~P	((P ∨ Q) ∧ ~P)
1	1	1	0	
1	0	1	0	
0	1	1	1	
0	0	0	1	

"((P ∨ Q) ∧ ~P)"is a conjunction, following the semantic conjunction rule. A conjunction is **true only when both its left and right parts are true**.

Conjunction Rule

●	▲	(● ∧ ▲)
1	1	1
1	0	0
0	1	0
0	0	0

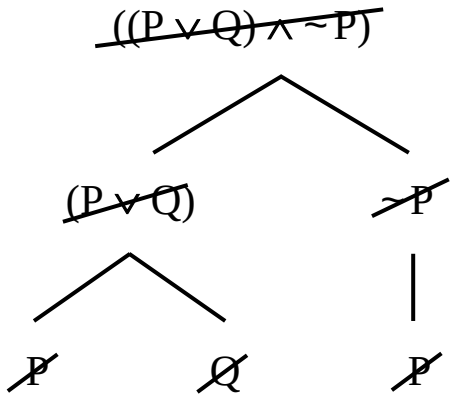
The left part of this sentence is "(P ∨ Q)"; its right part is "~P". Both these parts are true (together) only in the **third** valuation; so the whole conjunction is true just in that valuation.

		Left Part	**Right Part**	
P	Q	**(P ∨ Q)**	**~P**	((P ∨ Q) ∧ ~P)
1	1	1	0	
1	0	1	0	
0	1	**1**	**1**	**1**
0	0	0	1	

It's false in the other valuations.

		Left Part	**Right Part**	
P	Q	**(P ∨ Q)**	**~P**	((P ∨ Q) ∧ ~P)
1	1	1	**0**	**0**
1	0	1	**0**	**0**
0	1	1	1	1
0	0	**0**	1	**0**

The truth table for this sentence is now complete.
Its construction tree explains how this sentence was built up from atoms.



P	Q	(P ∨ Q)	~P	((P ∨ Q) ∧ ~P)
1	1	1	0	0
1	0	1	0	0
0	1	1	1	1
0	0	0	1	0

2. Semantics and Compositionality. We see now that our earlier focus on construction was no fastidious waste of time, but rather an essential ingredient for a truly general approach to semantics: so long as we shadow each use of a **construction rule** with its matching **semantic rule**, we can automatically produce a truth table for any formal sentence, big or small. That means: there couldn't possibly be a formal sentence which lacked a matching truth table. Since every formal sentence is built via one or more of the construction rules, and every construction rule has a parallel semantic rule, **every sentence in the formal language has a truth table**.

In general: **truth and falsehood** of a sentence **mirrors construction** of that sentence. So **truth tables follow construction trees**.

In the jargon of logic and linguistics, we say that our formal semantics is **compositional**: for each molecular sentence, its truth table depends only on (i) the truth tables of that sentence's immediate parts, and (ii) how those parts are put together to form the whole.

So the truth table for “((P ∨ Q) ∧ ~P)” depends only on (i) the truth tables for its left and right parts, “(P ∨ Q)” and “~P”, and (ii) the fact that these were put together into a conjunction – occasioning use of the semantic Conjunction Rule.

Had we instead disjoined those two sentences together – and so used the semantic Disjunction Rule – a different whole sentence would have resulted, with a different truth table. Likewise if we conjoined different left and right parts – say, the sentences “((P ∧ Q)” and “~Q” – again the final truth table would have been different.

Compositionality is nothing very new; for we find this same semantic feature in any natural language. Note that Sentences (1) and (2) mean different things because the parts, “kicked” and “kissed,” mean different things.

(1) Neko kissed Jack.
(2) Neko kicked Jack.

And while Sentences (2) and (3) have all the same parts, meaning the same things, those parts are put together differently – resulting again in whole sentences with different meanings.

(2) Neko kicked Jack.
(3) Jack kicked Neko.

In both English and the formal language, the semantic profile of the whole depends on (i) the semantic profile of its parts, and (ii) how those parts are put together.