Day 3: 6.1—6.8, Truth Tables, Validity, and Translation

Let A = Ann is agile.
B = Bob is boring.

Consider:

(A • B)
∴ A

This is a valid argument. All valid arguments are like this. The conclusion is implicitly contained in the premises. In this case it is very straightforward. In other cases, it is much harder to see. But it is always just like this case. When you assert or assume that the premises are true, you have already asserted or assumed that the conclusion is true.

An argument is a piece of reasoning that attempts to give reason to believe something, on the basis of a common ground. If I reason, argue, with you, I want to start with premises that we both already believe, and that show you that some conclusion follows from those agreed upon beliefs. In mathematical reasoning, we are starting from some statement or set of statements that are already given (or assumed) as true. We are attempting to show that something follows from these given statements.

But, as the example shows, a deductive argument can never derive anything really "new." To say that an argument is valid is to say that it is impossible for the conclusion to be false if the premises are all true. If the premises are all true, there is already "enough" (without any other evidence or reasoning) to show that the conclusion is true.

Consider:

(A ∨ B)
∴ A

This is an invalid argument. We can see that there is a way for the premise to be true and the conclusion false. The premise is true if even one of the two disjuncts is true. So it is true, say, when B is true but A is false. Right? But what we have just done by talking this through is essentially what we do with a truth a table. We come up with an assignment of truth-values to the atomic components of the statements in the argument that makes the premises true and the conclusion false. If there is any such truth-value assignment, then the argument is invalid.

A	B	||	(A ∨ B)	∴A
0	0	||	0	0
0	1	||	1	0
1	0	||	1	1
1	1	||	1	1

We see here that on the second line of the truth table, the premise is true and the conclusion is false. We have thus demonstrated or proven the invalidity of the argument by coming up with a truth-value assignment to the atomic components of the statements in the argument that make all the premises true and the conclusion false.

Can everyone see that what we have done with the truth table is exactly the same as what we did in the previous paragraph?

Lets create a truth table for our first argument:

A	B	||	(A • B)	∴A
0	0	||	0	0
0	1	||	0	0
1	0	||	0	1
1	1	||	1	1

Here we see that there is no line where all the premises are true and the conclusion is false—i.e., there is no truth-value assignment to the atomic components of the statements in the argument that makes the premises all true and the conclusion false. And this tells us that the argument is valid.

Comparing these two cases tells us something about the nature of the strategy we are using to prove validity or invalidity: we set out trying to find a line that makes the premises true and the conclusion false. If we find one, it is invalid, it not, it is valid. So, we essentially try to prove the argument to be invalid. If we fail, then it is valid. This is an important point to keep in mind. The argument is valid only when we don't find what we looking for. This is important to keep in mind in using the validity test introduced in 6.7 of the text.

Review how the exercises build up to this.
Truth Evaluations 6.3: Determining the truth-value of a compound statement given an assignment of truth-values to its atomic components

Examples: (A • B)
~(A • B)
(~A ⊃ (B v ~(C ≡ A)))

Let A=1, B=0, C=1

```
      1=True on this assignment
  0         0
  0      0    0
  0      0         1
    1    0      1    1
(~A ⊃ (B v ~(C ≡ A)))
```

<u>Complex Truth Tables 6.5</u>: Determine the truth table for a complex formula (a truth functionally compound statement) (see below)

<u>Truth Table Test for Validity 6.6</u>: Add multiple columns to the truth table, one for each of the premises and then the conclusion. And then look for a line (a truth-value assignment to the atomic components of the statements of the argument) where all of the premises come out true and the conclusion false. If you find even one such line, you have shown that the argument is invalid. If you find none, you have shown that the argument is valid.

A	B	C	(~A ⊃ (B ∨ ~(C ≡ A)))	(A ∨ B)	∴ (A • C)
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	1	0
1	1	1	1	1	1

<u>Constructing Truth Tables with 3 (or more!) atomic components</u>
Above.

6.7 The Truth Assignment Test for Determining Validity and Invalidity

(Explain while illustrating with an example from the exercises in the text.)

The technique is to try to see if we can make the premises true and the conclusion false. If so, then we have shown that the argument is invalid. If we cannot do this, we have shown that the argument is valid.

The technique is less "mechanical" than using a truth table (i.e., it takes more thinking), but it is considerably shorter.

There is also a question of what to do with "unknown" truth-values.

So, begin by assuming that the premises are true and the conclusion false. These examples all begin with a complex first premise, with the remaining premises and conclusion either atomic or negated atomic. These atomic or negated atomic statements give us truth-values on the assumption that the premises are true and the conclusion false. Plug these truth-values into the complex initial premise. If they make the premise true, then you have a truth-value assignment that makes the premises true and the conclusion false. ***So, if the first premise comes out true, the argument is invalid.*** If the first premise comes out false, the argument is valid. The truth-value assignment you plugged into the first premise came from assuming the remaining premises true and the conclusion false. It was the only such assignment that made the remaining premises true and the conclusion false. So if it doesn't make the first premise true, no truth assignment will make all the premises true and the conclusion false, and so the argument is valid

Sometimes, not all the atomic components in the first premise will get a truth-value assignment from assuming the remaining premises are true and the conclusion false. You then have to evaluate the first premise with these componenents having an unknown truth-value.

Remember, if there is any way to make all the premises true and the conclusion false, then the argument is invalid. So when you have an unknown truth-value, evaluate the statement twice, once using each truth-value. If it comes out true on either of them, the argument is invalid. So, if you have an unknown truth-value in that intial complex premise, the argument is valid only if the first premise comes out false no matter if that unknown component turns out to be true or turns out to be false.

(For more discussion about the "Truth Assignment Test" (from 6.7) for determining validity, see http://homepages.wmich.edu/~baldner/s67.pdf)

Translations:

"Ann is agile." "Bob is boring." We have two (truth functionally atomic) statements.

"Ann is agile and Bob is boring." This is a single (truth functionally compound statement. We translate it: (A •B). There are multiple ways that we could say this same statement in English. These other ways of saying it would be true and false in the very same conditions as "Ann is agile and Bob is boring." So they would also be translated as (A • B). When we "translate" from English into our formal language, what we are trying to do is make visible the conditions that would make the statement true or false. All statements that are true or false under the same conditions are truth functionally equivalent.

So we could also say, "*It is true both that Ann is agile and that Bob is boring.*"
This way of saying it is wordier, but it makes the truth functional structure of the statement completely explicit.

We could also say, "*Bob is boring and Ann is agile*," changing the order of the two conjuncts. A conjunction is true only if both of its conjuncts are true, so the order in which they are stated does not affect the conditions under which the conjunction is true or false. This is also the case with disjunctions and biconditionals. We will generally translate the parts of conjunctions, disjunctions, and biconditionals in the order in which they actually occur, but it is not mistaken to change their order precisely because doing so does not affect the truth-value of the compound statement. Translating into our formal language is all about making explicit truth conditions. So if the truth conditions are the same, it is logically equivalent.

Note that the order of conditional statements *does* matter. That is why we have different terms for the two parts. The "if" clause is known as the "antecedent," and the "then" clause is known as the consequent. "If A, then B" is *not* true or false in the same conditions as "If B, then A."

But let us go back to other ways of saying, "Ann is agile and Bob is boring." Consider, "*Ann is agile but Bob is boring.*"

The word "but" has connotations that are not captured by the word "and" (or by "•"). But remember, our purpose in translating into this formal language is to make explicit the truth conditions, and the truth conditions of "A and B" are the same as "A but B." In each case we are saying that both are true. "But" suggest some kind of contrast. Perhaps we were not expecting that B would also be true. But we are still saying that both are true. So, when we translate from English into our formal language, we can simply treat "but" as "•."

"Ann is agile or Bob is boring."
We translated this as "(A ∨ B)," and we noted in constructing our truth table for "∨" that this statement is true as long as one or the other is true, and even when both are true. This is known as the "*inclusive*" use of the word "or." But sometimes we use the word "exclusively," where we mean to exclude the possibility that both are true. I said there would be a way to capture that with the 5 connectives we introduced. Let's look at how.

When we use "or" *exclusively*, we want to exclude the possibility that both disjuncts are true. We could capture this explicitly in English by saying:

Either Ann is agile or Bob is boring, but not both.

How would we translate this? The comma tells us that what follows it is one half of the compound. What follows it is the word "but," which we treat as "and," and so we know that the entire compound is a conjunction. The first conjunct begins with "either," so we know that it is disjunction. The second conjunct is "not both," so it is the negation of the conjunction of the two components in the disjunction that comprises the first conjunct of the statement as a whole. We translate the above as:

$((A \lor B) \bullet {\sim}(A \bullet B))$

Either or A or B is true, and it is not the case that both A and B are true. This is the "*exclusive*" use of the word "or." If we mean to say that one or the other disjunct is true, but not both, this is how we translate that statement.

Things get more complicated with the ***conditional***.
"If P then Q" is translated

$(P \supset Q)$

But we can say the same thing as
"**Q, if P**." So this is also:

$(P \supset Q)$

So, what follows the word "if" is the antecedent. Well, but what about,
"**P, only if Q.**"
What does this say? It says that P is true only if Q is true. So, if Q isn't true, then neither is P. Right? So, one way to translate "P only if Q" is

$(\sim Q \supset \sim P)$.

This says that if Q isn't true, then neither is P. And that means that if P is true, so must Q be true. Which gives us:

$(P \supset Q)$

So, two things come out of this. One is that, in general

<u>$(P \supset Q)$ is logically equivalent to $(\sim Q \supset \sim P)$.</u>

So, take any conditional, reverse the order of the antecedent and consequent and change the signs (from "~"), and what you end up with is logically equivalent.

But what we began with was "only if." We saw that "P, only if Q" was equivalent to $(\sim Q \supset \sim P)$ which was equivalent to $(P \supset Q)$. So we translate "P, only if Q" as $(P \supset Q)$. And that means that what follows "only if" is the consequent of a conditional.

So, what follows "if" (by itself) is the antecedent of a conditional, while what follows "only if" is the consequent of a conditional. This is a good rule of thumb to remember.

So, **P iff Q** is equivalent to

P if Q, and P only if Q

i.e.,

$((Q \supset P) \bullet (P \supset Q))$

and that is logically equivalent to

$(P \equiv Q)$

which is how we translate "if and only if."

P, provided that Q
"Provided that" just means "if"
So, "P provided that Q" means "P, if Q," which is translated
$(Q \supset P)$

P just if Q.
The author treats "just if" as "if and only if." My "ear" tells me that "just if" is the same as "only if." Ultimately, this is a matter of how one understands the English language. I just disagree with the author here. When you're doing LogiCola, just treat "just if" as saying "if and only if," and then translate that as a biconditional. I won't put any examples of "just if" on the test.

P is sufficient for Q.

Think about what this says. P, by itself, is "*enough*" for P. So, if P is true (or if is happens or is the case), then Q is too. So, it is translated as
$(P \supset Q)$.

P is necessary for Q.
This one is harder to see. Perhaps you can just "see" that it is somehow the compliment of "sufficient." To say that P is necessary for Q is to say that Q can't be true unless P is. So, if P is not true, then neither is Q. This gives us
$(\sim P \supset \sim Q)$,
and that is logically equivalent to
$(Q \supset P)$

It is important for you to understand what we have just said. But then at some point, you just need to memorize it.

P is necessary and sufficient for Q.
You should see where this is going. If P is sufficient for Q is $(P \supset Q)$ and P is necessary for Q is $(Q \supset P)$, then P is necessary *and* sufficient for Q will be the conjunction of these two conditionals, which is exactly what is captured by
$(P \equiv Q)$.

(So, we see that not all of these connectives are strictly necessary, in that some can be defined in terms of others.)

P unless Q.
My ear hears "unless" as "if not." P unless Q means that P is true if Q isn't, i.e., if not Q, then P.
$(\sim Q \supset P)$.
This is equivalent to P or Q, i.e.,
$(P \vee Q)$
And also to If not P, then Q, i.e.,

(~P ⊃ Q)

All of these have the same truth table. Show it.:

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

But the easy rule of thumb is to translate "unless" as "or."

(For a list of the translation forms you need to memorize for the test, see http://homepages.wmich.edu/~baldner/transguide.pdf)