Three-Valued Logic

Phil 296 - November 28

Neither True Nor False

So far we've assumed that every sentence is either true or false, and not both. We did make one qualification to that when we started doing modal logic, and said that sentences can be true at some worlds and false at other worlds. But still everything is, at a particular world, either true or false, and not both. There are, however, a series of reasons for rejecting this assumption. I'll start by going over four such reasons.

I do want to stress that these are *reasons*, not necessarily *good reasons*. Indeed, the assumption that all sentences are true or false, and not both, is very widely shared. So most people reject these reasons. But still, they are reasons, and it's worth knowing about them.

Open Future

Going back at least as far as Aristotle, many people have thought that contingent claims about the future are not either true or false. This is especially true for sentences that are about free choices about the future. The standard example here (again as far back as Aristotle) is "There will be a sea battle tomorrow". The admirals (or generals) might or might not decide in the morning whether to give battle. And saying that it is now true (or false) that there will be a sea battle might be thought to undermine their free will.

Paradoxes

Let L be the sentence *L is false*. Assume L is true. Then what L says is the case; i.e., L is false. That contradicts the assumption that L is true. Now assume L is false. Then things are the way that L says things are. That is to say, L is true. This contradicts the assumption that L is false.

So assuming that L is either true or false leads to a contradiction. So L is neither true nor false. So we need a third truth value for sentences like L.

Nonsense and Non-Reference

Let C be the sentence *Green ideas sleep furiously*. This has the syntactic form of a sentence, but maybe it doesn't really make a meaningful claim. And maybe we should say that this really is a sentence, it does have the syntactic form of one, but isn't true or false. It isn't true because green ideas are not the kind of thing that sleep furiously. But also green ideas don't sleep furiously, so it isn't false.

The same kind of thing can happen with definite descriptions that are don't pick out anyone. Let Q be the sentence *The (present) Queen of the US lives in Ann Arbor*. Is that true or false? Well, it isn't true; if it was I suspect I would have met her by now. But is it false? That suggests that the Queen of the US lives somewhere else, and that isn't true.

So these sentences like C and Q seem like they are neither true nor false.

Vagueness

Consider the following four claims.

0. Whether someone is rich is a function of how much money they have (in some sense of money they have).
1. Someone with no money is not rich.
2. Giving someone who is not rich a penny does not make them rich.
3. Jeff Bezos is rich.

These seem inconsistent. Premise 2 suggests that you could take the person with no money, give them a penny over and over again roughly 10 trillion times, and they'd still not be rich, despite having as much money as Bezos. Either they are rich (contradicting 1 and 2), or they are not despite having as much money as Bezos (contradicting 0 and 3).

What could have happened here? One natural thought is that the problem is that there are *borderline* cases of richness. Some people are neither clearly rich nor clearly not rich. And the reason that premise 2 is plausible is that no penny can jump one over that borderline area.

If someone is in that borderline, the sentence *That person is rich* is arguably neither true nor false.

Four Reasons

So we have four reasons to think that sentences can be neither true nor false. There are two philosophical problems, however.

1. Arguably none of these are very good reasons; though in each case we could literally spend a semester (or more) on the literature on them.
2. They are very different reasons, and the kind of non-standard logic they suggest is different in every case.

We'll be mostly interested here in two.

Logics and Valuations

Valuation Functions

Assume that we don't have quantifiers, names, predicates, modals, or anything else beyond the symbols from chapter one of the book. To be more precise, assume we're working in the language with the following characteristics.

- There are some sentence letters, p, q, r, etc.
- There are connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$. We won't even use $\equiv$, and we will use $\rightarrow$ rather than $\supset$ for *if*, because we're sort of trying to capture natural language *if*.
- The sentence letters all stand for sentences.
- If A, B are sentences, so are $\neg A, A \wedge B, A \vee B, A \rightarrow B$.
- There are no sentences other than what are generated through those last two points.

Then one can specify a logic for that language by fixing the values for the following three things.

1. What the **truth-values** are.
2. Which of those truth-values are **designated**.
3. What the values of $\neg A, A \wedge B, A \vee B, A \rightarrow B$ are given the values of A, B.

We then use two definitions which will be constant across all the logics we'll study the rest of the way in this course.

Logical Truth A sentence is a logical truth if it gets a *designated* value no matter what value the sentence letters that make it up take. These logical truths are sometimes called *tautologies*.

Validity An argument is valid if whenever its premises take a designated value, so does its conclusion.

Note that how we label the values doesn't matter for the logic. It's the same logic whether you call the values True and False, or T and F, or 1 and 0. I'll mostly use numbers, because I think it makes it easier to see some patterns in what we're doing.

Classical Logic

To get **classical** logic, the logic we studied back in chapter 1, you can set these values as follows. The truth values are $\{1, 0\}$, and the only designated value is 1. And the values of compound sentences are given by these rules.[1]

$$\begin{aligned} v(\neg A) &= 1 - v(A) \\ v(A \wedge B) &= min(v(A), v(B)) \\ v(A \vee B) &= max(v(A), v(B)) \\ v(A \to B) &= v(\neg A \vee B) = min(1, 1 - v(A) + v(B)) \end{aligned}$$

I've done two definitions for $\to$ which are equivalent in this setting because they very soon will become not equivalent.

It's not much work to check given these definitions that the logical truths, and the validities, are just the ones we covered in chapter 1. Indeed, the definitions we've used are pretty similar to the definitions in chapter 1. But let's note one special case. Consider the sentence $p \vee \neg p$. There is only one sentence letter in this, p, and there are only two values in the logic, $1, 0$, so there are only two cases to check. If $v(p) = 1$, then the maximum of $v(p), v(\neg p)$ will be 1, so $v(p \vee \neg p)$ will be 1. If If $v(p) = 0$, then $v(\neg p) = 1$, so again the maximum of $v(p), v(\neg p)$ will be 1, so $v(p \vee \neg p)$ will be 1. Either way, $v(p \vee \neg p) = 1$, so $p \vee \neg p$ is a logical truth *in this logic*.

Strong Kleene Logic

The first **non-classical** logic we'll look at is called the Strong Kleene logic. It's named after Stephen Kleene (1909-1994, pronounced Klay-nee), a mathematician who spent most of his career at Wisconsin-Madison, and developed a number of key concepts in theoretical computer science. (He also apparently invented regular expressions, so even if you haven't heard his name, you've probably used his work.)

- The logic has three truth values: $1, \frac{1}{2}, 0$.
- The only designated value is 1.
- The rules for the connectives are mostly just like classical logic.

$$\begin{aligned} v(\neg A) &= 1 - v(A) \\ v(A \wedge B) &= min(v(A), v(B)) \\ v(A \vee B) &= max(v(A), v(B)) \\ v(A \to B) &= v(\neg A \vee B) \end{aligned}$$

[1] I'm going to list equations for these, rather than the tables as in the book for two reasons. One is that the book has the tables (in fact both books do), and seeing the equations is a useful comparison. The other is that the software I'm using doesn't make tables like those particularly easy to draw.

The only change I had to make was to only use one of the two definitions for $\rightarrow$, since the two definitions are not equivalent when v(A) = v(B) = ½.

If all the sentence letters in a longer sentence get value ½, so does the whole sentence. You can check this by going through the four rules. In each case, if all the inputs are ½, so is the output. And ½ is not a designated value. So there are *no logical truths* in this logic.

Despite this, there are validities in the logic. In any logic defined using designated values, $A \vdash A$ will be a logical truth. Whenever A is designated, A is designated. But there are more substantive validities in this logic, such as $A, A \rightarrow B \vdash B$. Note that there are nine cases to check to see that this works, but in all nine cases, either one of the premises is not designated, or the conclusion is designated.

The Strong Kleene logic is plausibly an account of the paradoxes. Let's not start with the paradoxical sentence L, the so-called *liar sentence*, but instead with T, the *truth-teller sentence*. T is the sentence *T is true*. It is consistent to say that it is true; then things are just the way it says they are. But it is also consistent to say that it is false. If T is false, then things are not the way T says they are, so T is false. That's consistent as well. Nothing seems to settle whether T is true or false.

Interpret 1 as meaning that the world settles that the sentence is true, 0 as that the world settles that the sentence is false, and ½ as that the world does not settle the sentence. If you understand *settling* in terms of a recursive process, the last one will be a situation where the process does not terminate. On that understanding, it makes sense to think that the value of T is ½; if you try to work out the value for T, you'll have to first work out the value for T, and you'll end up in a loop.

And, crucially, the same thing will go for L. What we want is that L takes the opposite value to what it itself takes. On the one hand, that looks impossible. But more importantly, there isn't a process for working that out; we don't define the value for L in terms of something simpler. So it makes sense that our theory gives it the *unsettled* value ½.

Kleene did a lot of work on recursion theory, and developed this logic in part for this purpose, so this isn't entirely a made-up, just-so, story about the logic. And Saul Kripke (who we met in the context of modal logic) developed a theory of the semantic paradoxes that used the Strong Kleene logic in just this way.

Even though there are no logical truths in the Strong Kleene logic, we can recover something like classical logic in it. Let A be any sentence that is a logical truth in classical logic, and let $p_1, \ldots, p_n$ be the sentence letters in it. Then the argument $p_1 \vee \neg p_1, \ldots, p_n \vee \neg p_n \vdash A$ will be a logical truth. The point of the premises here is just to specify that none of the sentence letters take the value ½. And when that happens, we're back in classical logic. In computing terms, when we know that every process will halt, we can use classical logic, but when we don't know that, we have to be more careful.

Łukasiewicz Logic

Jan Łukasiewicz (1878-1956, pronounced very roughly Wook-a-say-vich) was a Polish logician who is most famous for his development of Polish notation.[2] He was one of many refugees from central Europe in the 1930s who became central to the development of English-language philosophy after the war. Most of these refugees ended up in America, and several in Britain, but Łukasiewicz landed in Dublin.[3]

[2]We're not going into this. It was a way of writing arbitrarily complex logical/mathematical expressions without using brackets. On the one hand, it massively increased the range of things one could write in text-only systems, and produced much more compact expressions than standard notation. On the other hand, unless you worked in it for years, it was impossible to use intuitively. For both these reasons, it became very important in computing, and was the basis of some important early programming languages.

[3]According to Wikipedia, this was at the invitation of the Irish President, though the sourcing on that seems a little thin to me.

Łukasiewicz enters our story because of his interest in the open future. He thought that we needed a third truth value to account for sentences like *There will be a sea battle tomorrow.* So he developed the following logic.

- The logic has three truth values: $1, \frac{1}{2}, 0$.
- The only designated value is 1.
- The rules for the connectives are mostly just like classical logic.

$$\begin{aligned} v(\neg A) &= 1 - v(A) \\ v(A \wedge B) &= min(v(A), v(B)) \\ v(A \vee B) &= max(v(A), v(B)) \\ v(A \rightarrow B) &= min(1, 1 - v(A) + v(B)) \end{aligned}$$

The only difference between this and the Strong Kleene logic is in the last clause. We've used the other of the two definitions of $\rightarrow$. The only difference is in what happens when both A and B are ½. In Strong Kleene, then $A \rightarrow B$ is ½. In Łukasiewicz, in that case $A \rightarrow B$ is 1.

This in turn makes a huge difference. It means that in Łukasiewicz, there *are* logical truths. They all involve $\rightarrow$. In any sentence that does not have $\rightarrow$, if all the sentence letters get value ½, so does the whole. But the sentence $p \rightarrow p$ gets value 1 no matter what the value of p, so it is a logical truth. This makes a certain amount of sense given Łukasiewicz's interests. We might think that it is uncertain whether there will be a sea battle tomorrow, and hence uncertain whether any ships will be sunk tomorrow, but certain that if there is a sea battle tomorrow, some ships will be sunk tomorrow.

In classical logic, the following two things are equivalent.

- $A \rightarrow B$ is a logical truth.
- $A \vdash B$ is a valid argument.

In Strong Kleene these can't be equivalent, since there are no logical truths, but there are valid arguments. You might hope that this change would mean they would be back to being equivalent in Łukasiewicz. But they are not. Here is a counterexample:

- $(p \wedge \neg p) \rightarrow q$ is **not** a logical truth. (Why not?!)
- $p \wedge \neg p \vdash q$ **is** a valid argument. (Why?!)

Is this a bad thing for Łukasiewicz logic? It does look a bit weird to me, but maybe when we are working with the open future, weird things will happen.

Weak Kleene Logic

Given we had a Strong Kleene logic, you might be expecting a Weak Kleene logic, and you'd be right. The tables for it are in both of the books, so I won't repeat them here, but I will give a short description of them.

- If either A or B get value ½, any combination of them does as well.
- If neither A nor B get value ½, then the classical truth tables apply.

On this picture, the value ½ is 'infectious'; if it turns up at any part of the sentence, it spreads to the whole. This fits with a perhaps more realistic picture of recursion. Remember I motivated the Strong Kleene logic by sayingt it was a picture of how recursive assessment of truth might work: 1 means that the sentence resolves to true, 0 means that it resolves to false, and ½ means that it does not resolve. Well, I also had to add that it was a clever recursion process, that halted when it saw, for instance, that a disjunction had a true disjunct. Imagine that our recursion process wasn't quite so

clever, and it does not halt until it has worked out the truth values of all parts of the sentence. Then you'll get the Weak Kleene tables; any sentence with a part that does not resolve will not resolve, and will have value ½.

This is apparently a useful logic to use for various mathematical purposes, but it's really not much use to us. It doesn't even let you make simple inferences like $A \vdash A \vee B$, since this fails when A is 1 and B is ½. So I'm not going to talk about it much more, but I wanted to make sure we knew it existed.

I'm going to spend a bit of time now working out what we can prove and not prove in these three logics. We'll eventually want to come back to logics that allow for even more values, but these three-valued logics will be enough to get us started

Working Out What's Valid

Brute Force

In the Strong Kleene and Łukasiewicz logics, is this valid:

- $A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)$

Let's solve this by brute force, with a table.[4]

A	B	C	B v C	A & (B v C)	A & B	A & C	(A & B) v (A & C)
1	1	1	1	1	1	1	1
1	1	0.5	1	1	1	0.5	1
1	1	0	1	1	1	0	1
1	0.5	1	1	1	0.5	1	1
1	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.5	0	0.5	0.5	0.5	0	0.5
1	0	1	1	1	0	1	1
1	0	0.5	0.5	0.5	0	0.5	0.5
1	0	0	0	0	0	0	0
0.5	1	1	1	0.5	0.5	0.5	0.5
0.5	1	0.5	1	0.5	0.5	0.5	0.5
0.5	1	0	1	0.5	0.5	0	0.5
0.5	0.5	1	1	0.5	0.5	0.5	0.5
0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.5	0.5	0	0.5	0.5	0.5	0	0.5
0.5	0	1	1	0.5	0	0.5	0.5
0.5	0	0.5	0.5	0.5	0	0.5	0.5
0.5	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0.5	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0.5	1	1	0	0	0	0
0	0.5	0.5	0.5	0	0	0	0
0	0.5	0	0.5	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0.5	0.5	0	0	0	0
0	0	0	0	0	0	0	0

[4]For some reason $\vee$ and $\wedge$ aren't showing up in headers in the software I'm using, so I'm using & for *and*.

And it looks like it is. We just need the one table for the two logics, because the symbol $\rightarrow$ doesn't appear, and it's the only difference between the logics. But it's a big table. In three valued logic, a table for an argument with k sentence letters has 3^k rows. So even adding a fourth sentence letter would take us to 81 rows. I built this table with a little bit of code, but that's still some work.

But brute force isn't too bad for some things. Let's look at a simpler argument. Is this valid in the logics we're looking at?

- $A \vee B \vdash \neg B \rightarrow A$

First we'll do it in Strong Kleene:

A	B	A v B	$\neg$B	$\neg$B $\rightarrow$ A
1	1	1	0	1
1	0.5	1	0.5	1
1	0	1	1	1
0.5	1	1	0	1
0.5	0.5	0.5	0.5	0.5
0.5	0	0.5	1	0.5
0	1	1	0	1
0	0.5	0.5	0.5	0.5
0	0	0	1	0

That looks good. In fact, it loks like the two are identical. We'll come back to this point. What about in Łukasiewicz?

A	B	A v B	$\neg$B	$\neg$B $\rightarrow$ A
1	1	1	0	1
1	0.5	1	0.5	1
1	0	1	1	1
0.5	1	1	0	1
0.5	0.5	0.5	0.5	1
0.5	0	0.5	1	0.5
0	1	1	0	1
0	0.5	0.5	0.5	0.5
0	0	0	1	0

Again, it is fine. Every row where the premise is 1, the conclusion is 1. Now think about the reverse implication.

- $\neg B \rightarrow A \vdash A \vee B$

We have all the tables we need for that; they are just back-to-front. But we can still read them. What do they say? Well, in Strong Kleene every line where the premises are 1, so is the conclusion. But in Łukasiewicz, if both A and B are ½, the premise gets value 1, and the conclusion gets value ½. So the argument is *invalid*.

So just to recap, that's how we solve things by brute force.

- Build a table with 3^k lines, where k is the number of sentence letters. This might be easiest in Excel/Google Sheets/some coding language, because each of the sentences can be computed via a formula. (I was using Excel, but there are surely more efficient ways to do it.)

- Is there a row where all the premises get designated values and the conclusion gets an undesignated value?
- If so, the argument is invalid.
- If not, it is valid.

In the case of checking for a logical truth, the process is the same, but instead we just ask does the sentence take a designated value in every row. If so, it is a logical truth. If not, it is not.

Search Methods

Brute force has its limits. Think about whether this sentence is a logical truth in either of these logics.

- $((p \to q) \vee (q \to r)) \vee (r \to s)$

A table for that would have 81 rows, and it wouldn't even print with the format I'm using. But we can work out whether it is a logical truth in the two logics somewhat more easily.

It is not a logical truth in Strong Kleene for the same reason that there are literally no logical truths in Strong Kleene. Consider the row where all four sentences get value ½. In that row, every disjunct will get value ½, so the whole sentence will get value ½. And ½ is not designated.

What about in Łukasiewicz? Well, what would it take for the sentence to not have value 1. In Łukasiewicz (and in Strong Kleene), $\vee$ is a maximum operator. If either side of $A \vee B$ is 1, the whole is 1. So a row where the sentence gets a value other than 1 has to be a row where all three of $p \to q$, $q \to r$ and $r \to s$ get values less than 1. And when does a conditional $A \to B$ get value less than 1 in Łukasiewicz? Answer: when and only $v(A) > v(B)$. So for there to be a row where this sentence fails to take value 1, the following would all have to be true.

1. $v(p) > v(q)$
2. $v(q) > v(r)$
3. $v(r) > v(s)$

Is that possible? No it isn't; there aren't enough values! The first implies that $v(q)$ is at most ½. That plus 2 implies that $v(r)$ is 0. And that's inconsistent with 3. So in Łukasiewicz, this weird looking thing is in fact a logical truth.

Let's end with one more. Is this valid in either of the two logics?

$p \to (q \wedge \neg q) \vdash \neg p$

In Strong Kleene, we can reason as follows. The consequent of the premise, $q \wedge \neg q$, has value either 0 or ½. A conditional with a consequent whose value is not 1 is only value 1 when the antecedent has value 0. So p must have value 0 if this conditional has value 1. And if p has value 0, then $\neg p$ has value 1. So whenever the premise is designated, so is the conclusion, so the argument is **valid**.

In Łukasiewicz, that reasoning helps us, but not in the same way. The premise could have value 1 if both sides of the conditional have value ½. That happens when both p and q have value ½. And in that case, $\neg p$ has value ½, which is not designated. So there is a line where the premise is designated and the conclusion is not, so the argument is **invalid**.

There are only two variables here, so we can double check this with brute force. First the table for Strong Kleene, which shows that every line where the premise is designated, so is the conclusion. (But not vice versa, which is interesting.)

p	q	$p \rightarrow (q \,\&\, \neg q)$	$\neg p$
1	1	0	0
1	0.5	0.5	0
1	0	0	0
0.5	1	0.5	0.5
0.5	0.5	0.5	0.5
0.5	0	0.5	0.5
0	1	1	1
0	0.5	1	1
0	0	1	1

What about in Łukasiewicz?

p	q	$p \rightarrow (q \,\&\, \neg q)$	$\neg p$
1	1	0	0
1	0.5	0.5	0
1	0	0	0
0.5	1	0.5	0.5
0.5	0.5	1	0.5
0.5	0	0.5	0.5
0	1	1	1
0	0.5	1	1
0	0	1	1

And we see the line where the premise is designated and the conclusion is not designated.