Mathematics deals with absolute truth, right? Errr, well, absolute truth is an incredibly loaded term. Properly unpacking it requires a healthy amount of philosophy and mathematics. Wait. Don't go away. It's fun. You'll be grappling with paradoxical statements like 'this sentence is false' in a productive and well-defined way. You'll find statements that are neither true nor false, yet are critical to our modern understanding of mathematics.

1 Philosophical Underpinnings

1.1 won't get fooled again

> I'll tip my hat to the new constitution
> Take a bow for the new revolution
> Smile and grin at the change all around
> Pick up my guitar and play
> Just like yesterday
> Then I'll get on my knees and pray
> We don't get fooled again
> Don't get fooled again
> No, no!
>
> The Who

Rhetoric is an ancient tradition of disguising opinions as sound logic. Through millions of years of evolution, we carbon sack computers have been tuned to take shortcuts. Taking shortcuts in thought is necessary: no one would teach an infant to count by formally proving $2+2=4$. But with these shortcuts come logical fallacies, passing off emotions as fact, perception as reality.

We could start with assertions and then work toward a conclusion by requiring each new claim we make must logically follow from what we have already established, but English is too ambiguous and irregular. Consider the sentence

> Every mouse fears some cat.

Does this mean there is one terribly frightening cat that every single mouse fears? Or does it mean for each mouse there is a corresponding scary cat? [Wik15b]

We need to move toward logical deductions governed by rules expressed in an appropriate notation (not English). These rules should be straight-forward enough that they can be checked by a computer, but expressive enough that we can work out all mathematics in terms of these logical deductions.

We look to answer a nebulous philosophical question, 'is mathematics absolutely true?' by building a mathematical model that can handle mathematics, and then asking our model what it can tell us about truth in mathematics. Instead of saying, yes it is absolutely true, or no it is not, we will look for algorithms that will check our proofs when possible. Such an approach discards much nuance of the philosophical question; we're punting. This punting is necessary, because the question 'is mathematics absolutely true' is rife with uncertainty. What is mathematics? What is absolute truth? Does absolute truth exist? How can a human judge truth if 'to err is human'? One can object *ad nauseum.*

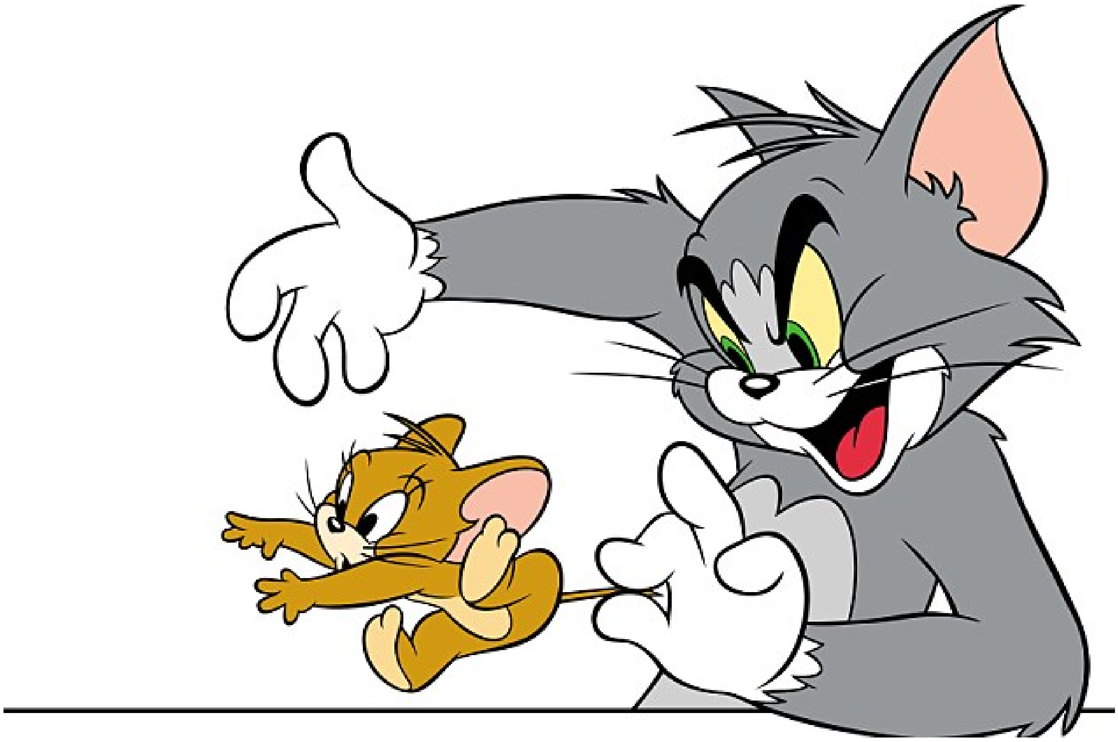

Figure 1.1: A scary cat

A similar problem: we cannot get something from nothing. We'll need to think logically in order to set up our mathematical model of logic. This circular reasoning can't be helped. It does pose a philosophical problem—we may very well be in the matrix, in which case everything we know is a lie, so our mathematical model could be dead wrong. I will assume without proof that we are not in the matrix. I will assume without proof that logic-checking computers do not lie. I will assume without proof that this bit of circular reasoning we're stuck with is innocent. The proof, of course, is left as an exercise.

2 boolean logic

The simplest interesting model of logic we will discuss. There are two possible values, true and false. When it is not ambiguous, we will abbreviate these as 0 and 1. Letters are variables. There are three operators we start with: and ($\wedge$), or ($\vee$), and not ($\neg$).

`Definition 2.0.1 (and ∧)` $x \wedge y$ is true exactly when x and y are both true. This is also called conjunction.

$\wedge$	0	1
0	0	0
1	0	1

`Definition 2.0.2 (or ∨)` $x \vee y$ is true exactly when either x or y or both are true. This is also called disjunction.

$\vee$	0	1
0	0	1
1	1	1

And

`Definition 2.0.3 (not ¬)` Negation flips 0 and 1:

x	$\neg x$
0	1
1	0

`Theorem 2.0.4 (De Morgan's law)`

$$\neg(x \wedge y) = (\neg x) \vee (\neg y) \tag{2.1}$$
$$\neg(x \vee y) = (\neg x) \wedge (\neg y) \tag{2.2}$$

(see also [Wik15a])

`Proof.` First show eq. (2.1) by checking truth tables:

x	y	$x \wedge y$	$\neg(x \wedge y)$	$\neg x$	$\neg y$	$(\neg x) \vee (\neg y)$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Then negate both sides of eq. (2.1), giving

$$x \wedge y = \neg(\neg x \vee \neg y)$$

Make the substitution $\bar{x} = \neg x$ and $\bar{y} = \neg y$:

$$(\neg \bar{x}) \wedge (\neg \bar{y}) = \neg(\bar{x} \vee \neg \bar{y})$$

or, equivalently, eq. (2.2). $\square$

2.1 the conditional

We would like to model causality. Causality is vital in logical reasoning. But defining causality is a black hole ready to swallow up even the cleverest of thinkers. So hark the olde adage 'the best mathematician is a lazy one', and punt. We will define the operator implies, a.k.a. the conditional, denoted $\rightarrow$ to act as our stripped-down notion of causality.

`Definition 2.1.1 (implies →)` $x \rightarrow y$ is true for all inputs except $1 \rightarrow 0$.

$\rightarrow$	0	1
0	1	1
1	0	1

We like this definition because $x \rightarrow y$ captures the idea y must follow from x being true, while allowing for the possibility that y may occur even when x is false. For example, if x means 'that's a cat', and y means 'that's a mammal', $x \rightarrow y$ means 'that's a cat, hence it's a mammal', with the unspoken caveat that dogs are not cats ($x = 0$), but they are mammals ($y = 1$).

That example is awkward because we have no good way of saying 'all cats are mammals, but not all mammals are cats' in boolean logic. We can say this, but not in a particularly useful way. In the next chapter, we will construct a more sophisticated logic that allows such statements.

We defined the conditional through a truth table; however, our definition of boolean logic does not mention truth tables. We ought to give a definition of $\rightarrow$ using $\wedge$, $\vee$, and $\neg$:

`Theorem 2.1.2` $x \rightarrow y$ is equivalent to $y \vee (\neg x)$

`Proof.` Both expressions are false only when $x = 1$ and $y = 0$. $\square$

`Definition 2.1.3 (logical equivalence ↔)` $x \leftrightarrow y$ exactly when $x = y$.

Likewise, we can define this in terms of $\wedge$, $\vee$, and $\neg$:

`Theorem 2.1.4` $x \leftrightarrow y$ is equivalent to $(x \wedge y) \vee (\neg x \wedge \neg y)$.

Table 2.1: some truthtable

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

2.2 tables to expressions

We have just translated two truth-tables into expressions consisting only of $\wedge$, $\vee$, and $\neg$. You might wonder if we can always do this. We can.

`Theorem 2.2.1 (completeness)` Every truth table has a corresponding boolean expression.

`Proof.` It's best to start with an example. Consider the truth table: We want an expression that is true exactly when (x_1, x_2, x_3) is $(0,0,1)$ or $(1,0,0)$ or $(1,1,0)$. A straightforward way to do this is to or together a statement that is true only on the input $(0,0,1)$, a statement true only on $(1,0,0)$, and a statement that is true only on $(1,1,0)$.

Consider $(x_1, x_2, x_3) = (0,0,1)$. To find an expression that is true for this input and false for all others, we can simply assert $x_1 = 0$ and $x_2 = 0$ and $x_3 = 1$, *i.e.*

$$\neg x_1 \wedge \neg x_2 \wedge x_3$$

By the same argument, we can find the remaining expressions:

$$\begin{array}{cc} (x_1, x_2, x_3) = (0,0,1) & \neg x_1 \wedge \neg x_2 \wedge x_3 \\ (x_1, x_2, x_3) = (1,0,0) & x_1 \wedge \neg x_2 \wedge \neg x_3 \\ (x_1, x_2, x_3) = (1,1,0) & x_1 \wedge x_2 \wedge \neg x_3 \end{array}$$

Hence table 2.1 can be written as

$$(\neg x_1 \wedge \neg x_2 \wedge x_3) \vee (\wedge \neg x_2 \wedge \neg x_3) \vee (x_1 \wedge x_2 \wedge \neg x_3)$$

To prove the theorem, we just need to generalize the work we've done. Suppose we are given a truth table for the function $f(x_1, \ldots, x_n)$. Mark off the rows in the truth table where $f(x_1, \ldots, x_n) = 1$. For each such row, we construct the expression that asserts the input matches. If the current row is for the input

$$(x_1, \ldots, x_n) = (\alpha_1, \ldots, \alpha_n)$$

then the expression is

$$\begin{pmatrix} x_1 & \text{if } \alpha_1 = 1 \\ \neg x_1 & \text{if } \alpha_1 = 0 \end{pmatrix} \wedge \cdots \wedge \begin{pmatrix} x_n & \text{if } \alpha_n = 1 \\ \neg x_n & \text{if } \alpha_n = 0 \end{pmatrix}$$

Abbreviate this as $(x_1 = \alpha_1) \wedge \cdots \wedge (x_n = \alpha_n)$. This is just a notational convenience; the final answer will not actually contain $=$ or refer to the α_i.

Then we or over each of the above expressions:

$$f(x_1, \ldots, x_n) = \bigvee_{\alpha_1,\ldots,\alpha_n : f(\alpha_1,\ldots,\alpha_n)=1} (x_1 = \alpha_1) \wedge \cdots \wedge (x_n = \alpha_n)$$

giving a boolean expression with the desired truth table. □

This tells us that boolean logic is big enough to express any truth table. An interesting implication is that general purpose computers can be constructed!

2.3 absolute truth?

We are supposed to shy away from notions of absolute truth, but boolean logic is simple enough that we can talk about it without making a mess. A boolean expression's truth value can be computed directly from the inputs. Since each input can be either true or false, there are only finitely many possible inputs. Hence, everything you need to know is contained in the expression's truth table. These computations can be carried out unambiguously by a machine in finite time, so things are looking pretty good.

We can also show there are no contradictions. In boolean logic, a contradiction would be of the form

$$p \wedge \neg p$$

for some expression p. No matter what the expression p is, it must either evaluate to 1 or 0. Suppose $p = 1$. Then $\neg p = 0$. So $p \wedge \neg p = 1 \wedge 0 = 0$. Likewise, if $p = 0$, we get $0 \wedge 1 = 0$. Thus all contradictions are false.

Bibliography

[Wik15a] Wikipedia. *De Morgan's laws.* 2015. URL: https://en.wikipedia.org/wiki/De_Morgan%27s_laws.

[Wik15b] Wikipedia. *Problem of multiple generality.* 2015. URL: https://en.wikipedia.org/wiki/Problem_of_multiple_generality.