Lecture Notes on
What is (Constructive) Logic?

15-317: Constructive Logic
Frank Pfenning

Lecture 1
Tuesday, January 17, 2023

1 Introduction

According to Wikipedia, logic is the study of the principles of valid inferences and demonstration. From the breadth of this definition it is immediately clear that logic constitutes an important area in the disciplines of philosophy and mathematics. Logical tools and methods also play an essential role in the design, specification, and verification of computer hardware and software. It is these applications of logic in computer science which will be the focus of this course. In order to gain a proper understanding of logic and its relevance to computer science, we will need to draw heavily on the much older logical traditions in philosophy and mathematics. We will discuss some of the relevant history of logic and pointers to further reading throughout these notes. In this introduction, we give only a brief introduction. For further general reading, we recommend the introduction to Troelstra and van Dalen [1988] and the entries on Intuitionistic Logic and Intuitionism in the Philosophy of Mathematics in the Stanford Encyclopedia of Philosophy.

2 Russell's Paradox

In the latter part of the 19th and the beginning of the 20th century mathematics was in what one might call a foundational crisis, occasioned by the increasingly more abstract mathematics. This crisis was deepened by the discovery of logical paradoxes. Russell's paradox (discovered in 1901) in particular dealt a blow to Frege's attempt to develop a universal logic in which all of mathematics can be carried out [Frege, 1879]. In modern language, Russell's paradox may be most easily understood via sets. He defined the R as the set of all sets that do not contain themselves.

$$R = \{x \mid x \notin x\}$$

Is R a member of R? If yes, then by definition of R it must be the case that $R \notin R$. If not, the again by the definition of R, it must be the case that $R \in R$. So we find

$$R \in R \leftrightarrow R \notin R$$

so we have a paradoxical proposition that is true if and only it is false. Russell's answer was to introduce *types* so that the set R cannot be formed because a set cannot be a member of itself. Whitehead and Russell's ramified type theory Whitehead and Russell [1910–13], later evolved by Church [1940] into his *simple theory of types* solved this particular problem. Other solutions exist in set theory.

3 Intuitionism

Besides a problem in a particular proposed system, one can ask more generally what we are doing when we are doing mathematics. The Platonic view is that mathematical objects such as integers, rationals, or reals, have an objective existence and that mathematicians are just trying to understand this reality. Brouwer [1907], the founder of intuitionism, instead considered mathematics as carrying out mental constructions. Mathematics then becomes more about construction and knowledge rather than (some elusive) absolute truth. In his view, we can assert $A \vee B$ only if we know whether A is true or B is true. In particular, we cannot necessarily assert the *law of the excluded middle* $A \vee \neg A$ in general, because we may know neither a proof of A nor a refutation of A. An example would be Goldbach's conjecture (let's call it G) that every even number is the sum of two primes. Since Goldbach's conjecture has been open for almost 300 years, we don't know of a proof and we don't know of a counterexample. Consequently we cannot say that $G \vee \neg G$ is true. But we also cannot rule out that we may be able to say this in the future.

We will not go into detail on Brouwer's specific form of intuitionism, but we can extract the following intuitionistic (and purposely somewhat informal) definition of truth [Troelstra and van Dalen, 1988]:

- A proof of $A \wedge B$ is given by presenting a proof of A and a proof of B.
- A proof of $A \vee B$ is given by presenting a proof of A or a proof of B.
- A proof of $A \supset B$ is a construction that permits us to transform a proof of A into a proof of B.
- There is no proof of $\bot$. Also, a proof of $\neg A$ is a construction with transforms a (hypothetical) proof of A into a contradiction.
- A proof of $\exists x.\, A$ is given by providing a witness d in the domain of quantification and a proof of $[d/x]A$.[1]
- A proof of $\forall x.\, A$ is a construction that takes an arbitrary element d in the domain of quantification to a proof of $[d/x]A$.

Where this definition says *construction* we can tilt the definition towards computer science and think of *function*, which must be effectively computable (so the construction can in fact be carried out). Under such an interpretation intuitionistic proofs correspond to functions, and particular systems of nintuitionistic inference give rise to particular programming languages.

[1]We write $[d/x]A$ for the result of substituting d for x in the proposition A.

4 A First Example

We call a logic that accepts that law of excluded middle *classical*. As example of the difference between classical and intuitionistic proof, we consider the following theorem and proof.

Theorem 1 *There are two irrational numbers a and b such that a^b is rational.*

Proof: Consider $\sqrt{2}^{\sqrt{2}}$. There are two cases:

Case: $\sqrt{2}^{\sqrt{2}}$ is rational. Then $a = b = \sqrt{2}$ satisfies the claim.

Case: $\sqrt{2}^{\sqrt{2}}$ is irrational. Then $a = \sqrt{2}^{\sqrt{2}}$ and $b = \sqrt{2}$ satisfy the claim, since $a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^2 = 2$.

□

At this point, the classical mathematician is profoundly happy, since this is an extremely short and elegant proof of a prima facie nontrivial theorem. The intuitionist is profoundly unhappy, since it does not actually exhibit irrational witnesses a and b such that a^b is rational. They might $a = b = \sqrt{2}$, or they might be $a = \sqrt{2}^{\sqrt{2}}$ and $b = \sqrt{2}$. Therefore an intuitionist rejects this proof.

The step which turns out to be incorrect here is to assume that there are two cases (either $\sqrt{2}^{\sqrt{2}}$ is rational or not) without knowing which of the cases hold, which is an instance of the (rejected) law of the excluded middle.

However, all is not lost! As an intuitionist, I look at the above proof and say:

> *Oh, I understand your proof, but it is for a different theorem! What you have proven is:*
>
> > ***Theorem.*** *If $\sqrt{2}^{\sqrt{2}}$ is rational or not, then there are two irrational numbers a and b such that a^b is rational.*

Surprisingly, as long as we stick to pure logic, or perhaps the theory of natural numbers, any classical proof can be reinterpreted as an intuitionistic proof of a different theorem! This suggests that, once we accept that intuitionism can in fact also be formalized, intuitionistic and classical are no longer in conflict. Instead, intuitionistic logic is a *generalization* of classical logic in the sense that has a constructive existential quantifier and a constructive disjunction, which are absent from classical logic. At the same time, all classical theorems and proofs can be uniformly imported into intuitionistic logic under some translation.

5 Another Paradox

Possibly beyond intuitionism, we call a logic *constructive* if its proofs describe effective constructions. The emphasis here is on *effective* which is to say that the construction conveyed by a proof can actually be carried out mechanically. In other words, constructive proofs describe algorithms. At first one might think that all proofs describe constructions of this form, and this was historically true for a long time. At some point in the 19th century this direct link between mathematics and computation seemed to get lost, which gave rise to intuitionism and other stricter forms of mathematical reasoning. Even today, though, most mathematicians accept the law of the excluded middle and it was computer science that revived interest in constructive logic and mathematics.

In order to understand this distinction better, we use consider the so-called *Banach-Tarski Paradox*.[2]

Theorem 2 *Given a solid ball in 3-dimensional space, there exists a decomposition of the ball into a finite number of disjoint subsets, which can then be put back together to yield two identical copies of the original ball. Indeed, the reassembly process involves only moving the pieces around and rotating them, without changing their shape. The reconstruction can work with as few as five pieces (but no fewer).*

This is considered paradoxical, since we obviously cannot carry out such a decomposition. The intermediate pieces are in fact non-measurable infinite scatterings of points. The decomposition relies critically on the axiom of choice in set theory, which is highly non-constructive.

6 On the Connection Between Proofs and Programs

We now give an concrete example of the connection between intuitionistic proofs in arithmetic and programs. We define

$$\begin{array}{lcl} \mathsf{even}(x) & \triangleq & \exists y.\, x = 2y \\ \mathsf{odd}(x) & \triangleq & \exists y.\, x = 2y+1 \end{array}$$

Theorem 3 $\forall x.\, \mathsf{even}(x) \lor \mathsf{odd}(x)$

Proof: By mathematical induction on x.

Case: $x = 0$. Then $\mathsf{even}(x)$ since for $y = 0$ we have $x = 2y$.

Case: $x = x' + 1$. Then $\mathsf{even}(x') \lor \mathsf{odd}(x')$ by induction hypothesis on x'. We now proceed by cases.

Subcase: $\mathsf{even}(x')$. Then $x' = 2y'$ for some y'.
Then $x = x' + 1 = 2y' + 1$ and $\mathsf{odd}(x)$.

Subcase: $\mathsf{odd}(x')$. Then $x' = 2y' + 1$ for some y'.
Then $x = x' + 1 = (2y' + 1) + 1 = 2(y' + 1)$ and $\mathsf{even}(x)$.

□

[2] See https://en.wikipedia.org/wiki/Banach-Tarski_paradox

Let's now extract a function from this proof. According to the intuitionistic interpretation of truth, it should be a function from natural numbers to an indication if the number is even or odd. We could use the booleans for that, but let's be more explicit and define a specific type for that. We write our function in ML, but of course it can be transliterated into any functional language. Since ML has only built-in integers, we define natural numbers (type nat) as Peano did from zero and a successor function.

```
datatype nat = zero | succ of nat
datatype eo = ev | od

(* even_or_odd : nat -> eo *)
fun even_or_odd x = case x of
    zero => ev
  | succ(x') => case even_or_odd x'
                 of ev => od
                  | od => ev
```

We see that an appeal to the induction hypothesis corresponds to a recursive call. The case distinctions turn into **case** constructs.

Should an intuitionist accept induction as a valid reasoning principle? Or, in a related question, should an intuitionist accept the given even_or_odd function as a valid construction? The answer is yes because it is clear that this function is well-defined for every natural number because (a) it supplies a case for 0, and (b) it reduces the argument from $x' + 1$ to x' in the recursive call. This is an example of the general schema of *primitive recursion* over natural numbers. We say that a function f is defined by primitive recursion of its definition has the form

$$\begin{array}{lcl} f(0) & = & c \\ f(x'+1) & = & g(x', f(x')) \end{array}$$

This means that any recursive call of f must be on x'. In general, the function extracted from an intuitionistic proof by mathematical induction will be in the form of primitive recursion.

Looking back at the original proof, we see we lost some information along the way. If we expand the definitions of even and odd we see an existential quantifier. And, indeed, the proof gives information about the witness for the existential. In case x' is even (with witness y'), the witness for the fact that $x' + 1$ is odd is the same y'. In case x' is odd (with witness y'), the witness for the fact that $x' + 1$ is even is $y' + 1$. In order to incorporate this into the function, we just change the eo datatype to carry the witness.

```
datatype nat = zero | succ of nat
datatype eo = ev of nat | od of nat
```

This can be carried into the function as prescribed by the proof.

```
(* even_or_odd : nat -> eo *)
fun even_or_odd x = case x of
    zero => ev(zero)
  | succ(x') => case even_or_odd x'
```

```
            of ev(y') => od(y')
             | od(y') => ev(succ(y'))
```

Now the function actually not only computes an indication of whether the argument x was even or odd and also returns $y = \lfloor x/2 \rfloor$.

We want to highlight that we did not write this program and then prove it correct (which we can of course also do), but that we provided a constructive proof and extracted a program. It is also possible to go in the other direction: given a program we may be able to read off the kind of proof it represents. This connection between proofs and programs will occupy quite a bit of time in the first section of the course.

7 Hiding Computational Contents

It is common (although not necessary, as we will explore in a future assignment) to define negation via implication and falsehood:

$$\neg A \triangleq A \supset \bot$$

Now, intuitionistically a proof of $A \supset \bot$ is a function that, when given a proof of A returns a proof of $\bot$. However, since we have a refutation of A, there can be no proof of A, so such a function can never be applied. The informal conclusion is that a proof of $\neg A$ has no computational contents. This means we can "hide" computational contents by replacing A with $\neg\neg A$. For example, if we did not want a witness and just wanted to know if a number x is even or odd, we might specify:

$$\forall x.\, \neg\neg\mathsf{even}(x) \vee \neg\neg\mathsf{odd}(x)$$

But we have to be very careful, because this might fundamentally change the structure of the proof. In particular, if we now try to prove this statement directly by induction, suddenly induction hypothesis is much weaker.

Here are two similar (informal) examples that we will come back to later in the course.

$$\forall u, v.\, (\exists p.\, \mathsf{path}(p, u, v)) \vee \neg\exists p.\, \mathsf{path}(p, u, v)$$

Assuming the right domains of quantification and a predicate $\mathsf{path}(p, u, v)$ that holds if there is a path in a graph connecting u and v, this proposition can serve as a specification of graph reachability. We mean that a *constructive proof* of this proposition must contain an algorithm for determining whether a path between given vertices u and v exists and, if so, also return the path. We can prove this in a number of different ways, leading to a number of different algorithms (by induction on the length of the path, by induction on the set of unvisited notes, by induction on the set of edges on the path, etc.). If we do not want the path, just an indication of whether there is one, we could precede the first existential quantifier by a double negation.

The other point is that, *classically*, this is an entirely trivial statement since it has the form $\forall u, v.\, A(u, v) \vee \neg A(u, v)$ and is thus an instance of the law of excluded middle.

Finally, in the domain of programming languages we could specify a type checker as

$$\forall e, \tau.\, \mathsf{type}(e, \tau) \vee \neg\mathsf{type}(e, \tau)$$

where the predicate $\mathsf{type}(e, \tau)$ holds if expression e has type τ. A variation on this theme would be *type inference*:

$$\forall e.\, (\exists \tau.\, \mathsf{type}(e, \tau)) \vee \neg \exists \tau.\, \mathsf{type}(e, \tau)$$

From an implementation perspective, both of these are a bit unsatisfactory because we obtain no information when the program is not well-typed. In order solve this problem we could define another predicate $\mathsf{not\_type}(e, \tau)$ such that its proof would carry the required information. This is somehow the reverse from our even/odd example where might have said $\forall x.\, \mathsf{even}(x) \vee \neg \mathsf{even}(x)$, but we chose instead to use an explicit definition for odd.

8 The Status of Some Propositions

In lecture, there was some discussion about whether certain propositions may have a proof in intuitionistic logic or not. After the next lecture we will have some tools to prove some of them, but it will be a few lectures until we can show some of them are not intuitionistically provable. Here are some that I recall:

We have $A \supset \neg\neg A$: if A is true it is definitely not false. On the other hand, $\neg\neg A$ does not imply A in general since we may not have enough information in the proof of $\neg\neg A$. But we have $\neg A \leftrightarrow \neg\neg\neg A$ and similarly for further iterations of negation. Intuitively, that's because a proof of $\neg A$ does not have computational contents.

As mentioned before, an intuitionist does not accept the law of excluded middle $A \vee \neg A$. But we can prove that it is not false: $\neg\neg(A \vee \neg A)$ is intuitionistically true.

The law of the excluded middle is somewhat related to the rule of indirect proof, which intuitionism also rejects. The rule of indirect proofs allows a proof of A by assuming $\neg A$ and deriving a contradiction. But we have to be careful! The proof principle that to prove $\neg A$ we assume A and derive a contraction is fundamentally intuitionistic. Classically, these two forms of proof are in some sense indistinguishable, but intuitionistically they are very different (one is valid, the other one isn't!).

This is a common occurrence: when reasoning constructively we have to be careful about the precise formulation of a theorem since many classically equivalent propositions are not intuitionistically equivalent.

References

L. E. J. Brouwer. *Over de Grondslagen der Wiskunde*. PhD thesis, Universiteit van Amsterdam, 1907. English translation "On the Foundations of Mathematics" in *Collected Works. I: Philosophy and Foundations of Mathematics*, Arend Heyting (ed.), North-Holland, 1975.

Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5: 56–68, 1940.

Gottlob Frege. *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Verlag von Louis Nebert, 1879. English translation *Begriffsschrift, a formula language, modeled upon that of arithmatic, for pure thought* in J. van Heijenoort, editor, *From Frege to Gödel; A Source Book in Mathematical Logic, 1879–1931*, pp. 1–82, Harvard University Press, 1967.

A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction*, volume 1. North-Holland, 1988.

Alfred North Whitehead and Bertrand Russell. *Principia Mathematica*. Cambridge University Press, 1910–13. 3 volumes.