Proof Theory & Computation:
or, why study logic?

Thomas Waring

Philosophy & foundations

Historically, logic was studied as a *foundation* — for thought, philosophy, mathematics…

Gottlob Frege in C.XIX: *Concept-notation: A Formal Language for Pure Thought Modeled on that of Arithmetic*

Foundation for mathematics

David Hilbert wanted an axiomatic system of logic to underlie all of mathematics.

"... the problem of the **solvability** in principle of **every mathematical question** ..."

"... the theory of axioms must ... show that within every field of knowledge **contradictions** based on the underlying axiom-system **are absolutely impossible**."

(Taken from *Axiomatic Thought*, 1918. My emphasis.)

First efforts

The usual axiomatic system is based on **set theory**, due to Cantor & others.

Everything is a **set**, sets have things in them (which are also sets), we can combine them, etc…

$$\mathbb{N} = \{0, 1, 2, \dots \}$$

The key operation is **comprehension**:

$$\{x : x \text{ is an even number} \} = \{0, 2, 4, \dots\}$$

Russell’s paradox

The barber shaves everyone who doesn’t shave themself. So

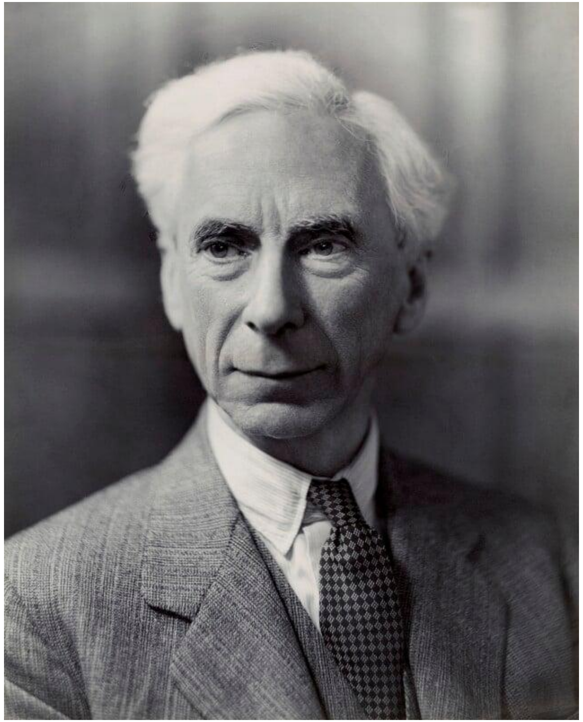
$$B = \{x : x \notin x\}.$$

Who shaves the barber? We have

$$B \in B \Rightarrow B \notin B,$$

but also

$$B \notin B \Rightarrow B \in B.$$

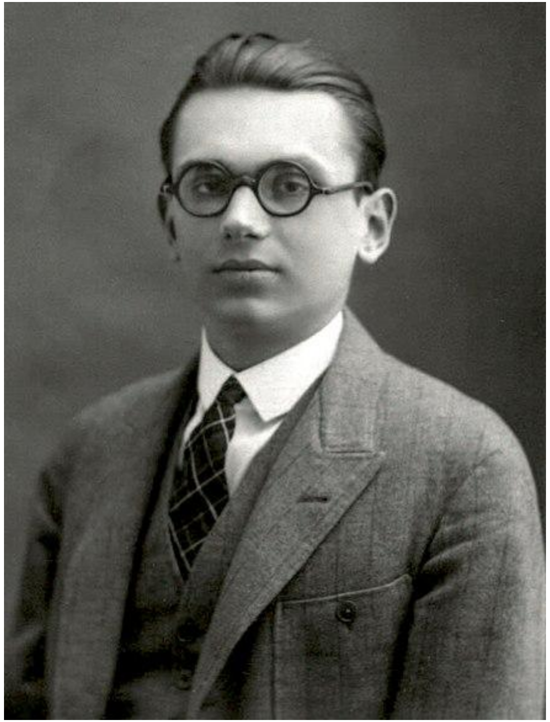


The problem runs deeper

Gödel’s Incompleteness Theorems (1931)

1. In any* consistent system, there is a formula *G* which is **true** but not **provable**.
2. A consistent system **cannot prove** its own **consistency**.

More or less, *G* = “I am not provable.”



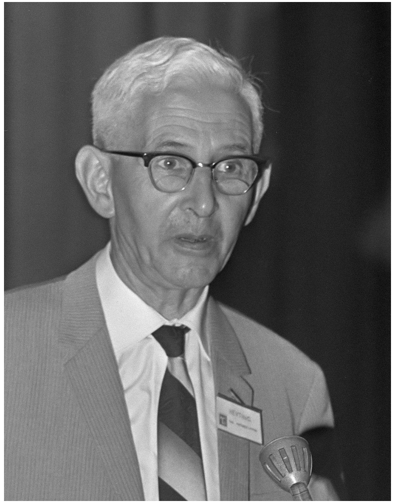
So, why study logic?

- If we want a theory of logic to be a foundation, it had better be consistent.
- To prove the consistency of our theory, we need some “bigger” theory.
- But to justify the new theory, we need a bigger theory again…

It’s turtles all the way down!

“From Why? to How?” — BHK interpretation

Logic can’t (completely) show is **why** things are true — our interest turns to **how**, that is, to *proofs*. What is a proof of P? We build it up from the pieces of the statement P.

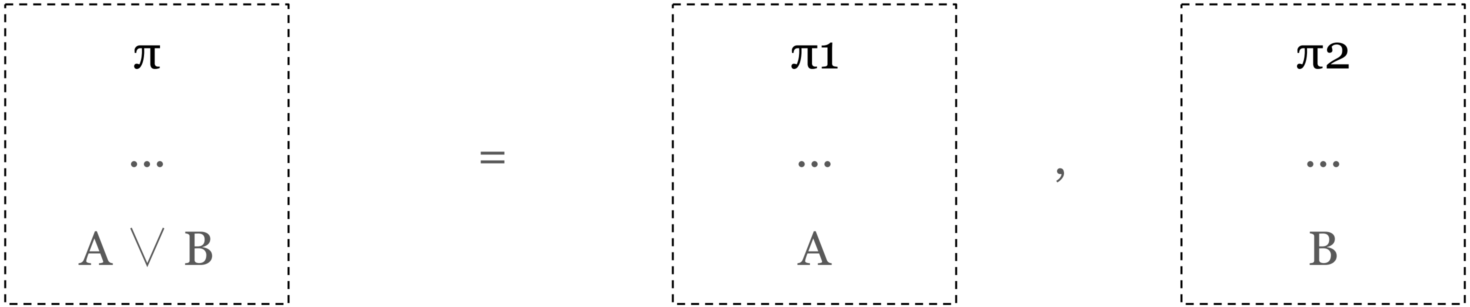


BHK for: Brouwer, Heyting and Kolmogorov (left to right)

“From Why? to How?” — BHK interpretation

Logic can’t (completely) show is **why** things are true — our interest turns to **how**, that is, to *proofs*. What is a proof of *P*? We build it up from the pieces of the statement *P*.

If *P* = “*A* and *B*”, a proof π of *P* is a pair (π_1, π_2), where π_1 proves *A* and π_2 proves *B*.



π

…

A ∨ B

=

π1

…

A

,

π2

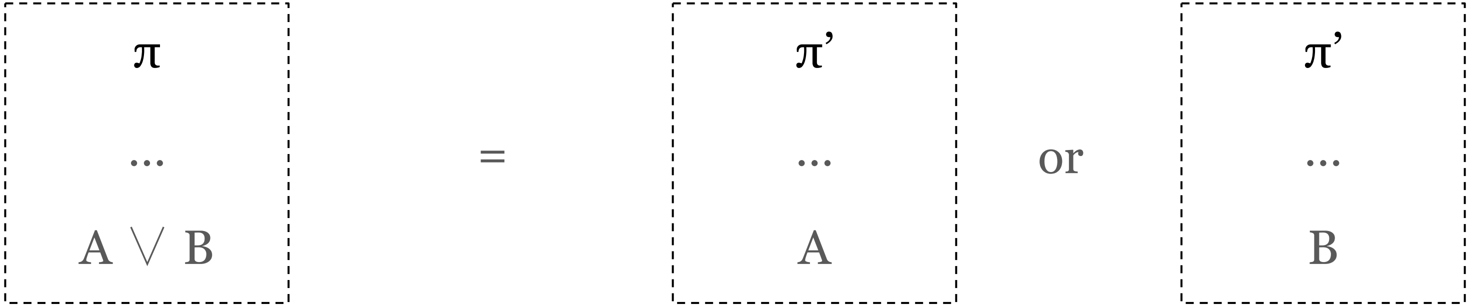
…

B

“From Why? to How?” — BHK interpretation

Logic can’t (completely) show is **why** things are true — our interest turns to **how**, that is, to *proofs*. What is a proof of *P*? We build it up from the pieces of the statement *P*.

If *P* = “*A* or *B*”, a proof π of *P* is a pair (*X*, π’), where *X* = *A* or *B* and π’ proves *X*.



π

…

A ∨ B

=

π’

…

A

or

π’

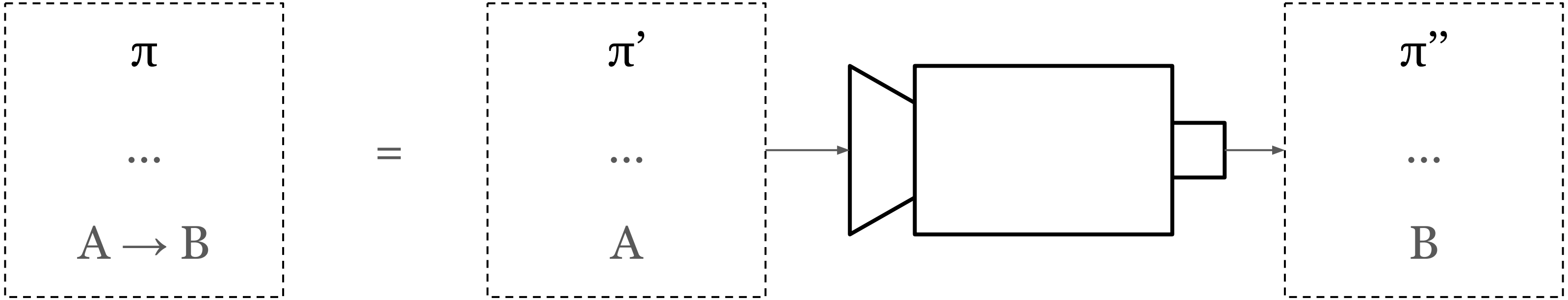
…

B

“From Why? to How?” — BHK interpretation

Logic can’t (completely) show is **why** things are true — our interest turns to **how**, that is, to *proofs*. What is a proof of *P*? We build it up from the pieces of the statement *P*.

If *P* = “*A* implies *B*”, a proof π of *P* is a machine (or function, algorithm) which turns proofs of *A* into proofs of *B*.

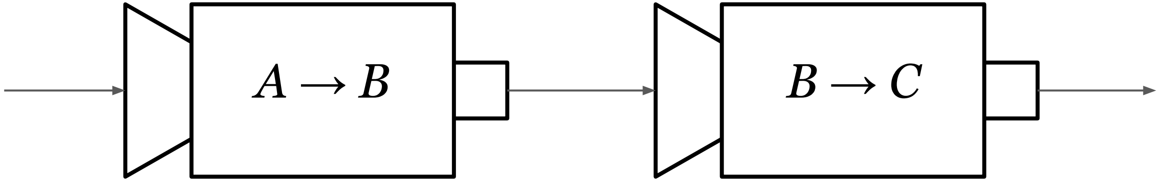


What is a function?

What is a function? According to Church, we can do two things:

Apply a function to something: from “A” and “$A \rightarrow B$” get “B” — *modus ponens*.

Compose functions: from “$B \rightarrow C$” and “$A \rightarrow B$” get “$A \rightarrow C$”.



Composing proofs — the cut rule

What is composition for proofs? As formalised by Gentzen, we write “$A \vdash B$” for “A proves B”. Therefore, composing proofs looks like:

$$\dfrac{\begin{matrix}\vdots \\ A \vdash B\end{matrix} \qquad \begin{matrix}\vdots \\ B \vdash C\end{matrix}}{A \vdash C}\ (\text{cut})$$

Cut elimination

Some cuts are unnecessary, viz:

$$\dfrac{\dfrac{}{A \vdash A}\,(\text{axiom}) \qquad \begin{matrix}\vdots\\ A \vdash B\end{matrix}}{A \vdash B}\,(\text{cut}) \qquad \rightsquigarrow \qquad \begin{matrix}\vdots\\ A \vdash B\end{matrix}$$

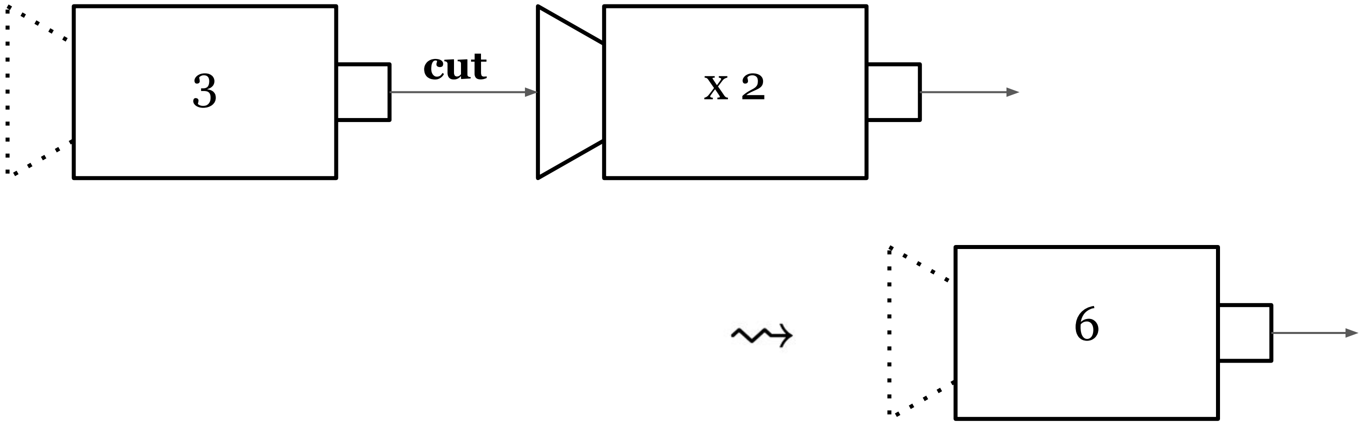
Others we can move around, such as:

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ A \vdash C\end{matrix} \qquad \begin{matrix}\vdots\\ B \vdash C\end{matrix}}{A \lor B \vdash C}\,(L\lor) \qquad \begin{matrix}\vdots\\ C \vdash D\end{matrix}}{A \lor B \vdash D}\,(\text{cut}) \quad \rightsquigarrow \quad \dfrac{\dfrac{\begin{matrix}\vdots\\ A \vdash C\end{matrix} \qquad \begin{matrix}\vdots\\ C \vdash D\end{matrix}}{A \vdash D}\,(\text{cut}) \qquad \dfrac{\begin{matrix}\vdots\\ B \vdash C\end{matrix} \qquad \begin{matrix}\vdots\\ C \vdash D\end{matrix}}{B \vdash D}\,(\text{cut})}{A \lor B \vdash D}\,(L\lor)$$

Cut elimination — proofs & computation

Gentzen’s “Hauptsatz” provides an algorithm to eliminate cuts from a proof — anything proven with cut can be proved without it.

In this way, proofs can simulate *computation*.



Making this precise: Church-Turing thesis

The first step to making this precise is pinning down what we mean by “computation”.

Turing defines a *Turing Machine*, an idealised computer.

The *Church-Turing thesis* is that “**computable**” means “**computable by a Turing Machine**”. Proven to be equivalent to many other abstractions of a computer.

Making this precise: Curry-Howard correspondence

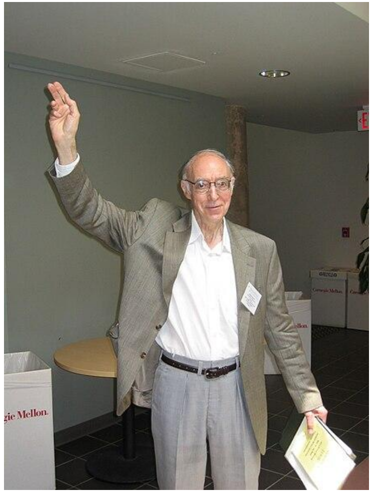
One “Turing complete” abstraction is Church’s λ-calculus.

The *Curry-Howard correspondence* says that **proofs** and **λ-calculus** are two perspectives on **the same thing**.

Philosophically:

Proofs are programs,

cut elimination is execution.



Where to from here?

Proofs

Recall Russell’s paradox, which looked like this:

$$\begin{tikzcd}
B \in B \arrow[dr, bend left] & \\
& B \notin B \arrow[ul, bend left]
\end{tikzcd}$$

Programs

Consider the *Halting Problem*: given a program, give an algorithm to decide whether it runs forever.

Church (1935), Turing (1937): the halting problem is unsolvable.

The method of proof is the same!

If this is interesting to you…

Ask me about it! Now or later, I’m always happy to discuss.

Most of this talk follows the work of J.Y. Girard.

Further reading from Girard:

- Detailed history: *La Théorie de la Démonstration: du programme de Hilbert à la logique linéaire* (1997).
- The proof-program connection: *Proofs and Types* (1989).
- Philosophising (if you’re feeling brave!): *The Blind Spot: lectures on logic* (2010) — adapted from a course in Rome!