PART 1: TYPE THEORY · CHAPTER 2

Hilbert Proofs

Alexander V. Gheorghiu
COMP0009 Logic · University College London

CHAPTER 2

Hilbert Proofs

The logic behind the checker.

OPENING

An argument

You come home after dark and the windows are black, and before the key is out of the lock you have decided that nobody is in. Written out in full:

LINE	STATEMENT	JUSTIFICATION
1.	The lights are off.	seen
2.	If the lights are off, nobody is home.	known
3.	So nobody is home.	from 2 and 1
4.	If nobody is home, the dog has not been fed.	known
5.	So the dog has not been fed.	from 4 and 3

Three starting points. Two lines that follow from earlier lines, by one move, used twice.

OPENING

A proof

- A proof is a list.
- Each line is either a starting point or follows from lines above it by a rule.
- The last line is the thing proved.

Give or take some notation, that is what Hilbert said a proof was.

OPENING

Hilbert

David Hilbert, 1862–1943.

In the 1920s, with Paul Bernays, he proposed a mathematical theory of mathematical arguments: proof theory.

A proof is a finite sequence of formulas, each an axiom, or something one has been told, or got from earlier formulas by one of a short fixed list of rules.

The form used in this chapter, a short list of axioms, one rule, proofs as lists, is the one in the textbook he wrote with Ackermann in 1928.

OPENING

The clerk

Who can check the argument? You do not need to know what a dog is. You need to see that line 3 is the right-hand half of line 2, and that line 1 is its left-hand half.

I shall call the one doing the checking the clerk, after Turing, who in 1936 described not a machine but a person: a clerk with a pencil and squared paper, who looks at one square at a time and does one of a few fixed things.

The clerk **understands nothing** and **checks everything**.

Everything in this chapter is addressed to him.

Not Hughes: she reasons, and wants her reasoning checked. The clerk is whoever, or whatever, does the checking; in Section 2.2 it will be her compiler.

OPENING

Two ends

CHAPTER 1: THE COMPUTING END

A compiler refused to run a program. What had it been looking at? A list of lines, each justified by one of four rules.

CHAPTER 2: THE LOGIC END

Seen from here, the four rules are a **logic**, a very small one. And a logic can be **grown**.

The reason for coming twice is that the view is different.

OPENING

Where we are going

SECTION	THE QUESTION
2.1	Formulas, axioms, and proofs: what does the clerk read, and what may he write?
2.2	Proofs are programs: where did the two shapes come from?
2.3	More connectives: is “if … then” enough?
2.4	Proof theory: what can we prove *about* proofs?
2.5	The Deduction Theorem: the clerk cannot suppose. Does he need to?

And on the way: the rule for parameters that Chapter 1’s checker did not have.

SECTION 2.1

Formulas, axioms, and proofs

A story that runs through the chapter.

2.1 · FORMULAS, AXIOMS, AND PROOFS

Hughes's first Monday

Hughes has just become a maintainer of an open-source library. Her first job is the release bot. A maintainer called Smith comments on a pull request: "please release 2.3". She opens MAINTAINERS.md.

LINE	STATEMENT	JUSTIFICATION
1.	Smith is a maintainer.	told
2.	If Smith is a maintainer, Smith has write access.	told
3.	So Smith has write access.	from 2 and 1
4.	If Smith has write access, Smith may publish a release.	told
5.	So Smith may publish a release.	from 4 and 3

Nothing clever, which is rather the point. Anybody could check it, including someone who had never met Smith.

2.1 · FORMULAS, AXIOMS, AND PROOFS

The same afternoon

A contributor called Jones asks the bot to rotate the signing key, which only a maintainer may do. He has an argument ready:

LINE	STATEMENT	JUSTIFICATION
1.	Maintainers have write access.	told
2.	I have write access.	told
3.	So I am a maintainer.	from 1 and 2?

Hughes is fairly sure something is wrong with that. She would like to be able to say what*, without knowing Jones.*

That is what a logic is for. Write both arguments out, and the same check serves for both: look at the shape of each line. What Hughes wants is a clerk.

Letters

English will not do: its grammar is irregular and opaque. Replace each complete thought by a letter, and keep a small grammar that records nothing but the connections between thoughts.

LETTER	SMITH
p	Smith is a maintainer
q	Smith has write access
r	Smith may publish a release

Told: p, $p \to q$, $q \to r$. Wanted: r.

LETTER	JONES
j	Jones is a maintainer
k	Jones has write access

Told: $j \to k$, k. Wanted: j.

2.1 · FORMULAS, AXIOMS, AND PROOFS

Definition 2.1: formulas

Fix a stock of letters $p, q, r, \ldots$. The formulas are defined inductively:

1. Every letter is a formula.
2. If φ and ψ are formulas, so is $\varphi \to \psi$, read "if φ then ψ".
3. Nothing else is a formula.

IN ONE LINE

$$\varphi, \psi \ ::= \ p \mid \varphi \to \psi$$

The arrow groups to the right: $\varphi \to \psi \to \chi$ is $\varphi \to (\psi \to \chi)$.

Small Greek letters stand for formulas, as capital Latin ones stood for types. Formulas have the same grammar as types; Section 2.2 says why.

Example: reading formulas

With p "Smith is a maintainer", q "Smith has write access", r "Smith may publish a release".

FORMULA	WITH BRACKETS	IN ENGLISH
$p \to q$		If Smith is a maintainer, he has write access.
$p \to q \to r$	$p \to (q \to r)$	If Smith is a maintainer, then, if he has write access, he may publish.
$(p \to q) \to r$	*already*	If being a maintainer gives Smith write access, then he may publish.
$p \to q \to p$	$p \to (q \to p)$	If Smith is a maintainer, then, if he has write access, he is a maintainer.

Brackets change what is said. The last one sounds idle; it is about to become important.

2.1 · FORMULAS, AXIOMS, AND PROOFS

Your turn: which are formulas?

With p, q, r as before, write each sentence as a formula, or say why you cannot.

1. If Smith is a maintainer, then, if he has write access, he may publish.
2. If being a maintainer gives Smith write access, he may publish.
3. Smith is a maintainer, so he has write access.
4. Smith is a maintainer or he has write access.

ONE MINUTE

Use Definition 2.1 and nothing else.

2.1 · FORMULAS, AXIOMS, AND PROOFS

Your turn: the answer

ITEM	FORMULA	VERDICT
1.	$p \to q \to r$	formula: $p \to (q \to r)$
2.	$(p \to q) \to r$	formula: the brackets matter
3.	*none*	“so” reports an argument: two claims side by side
4.	*none*	“or” is not in the language yet

A formula is one claim built with if. *An argument is a list of them.*

Hold on to item 4. Section 2.3 comes back to it.

2.1 · FORMULAS, AXIOMS, AND PROOFS

Modus ponens

What do we mean by "if …, then …"? At the least this: whenever the *if* part holds, the *then* part may be concluded. That is how Hughes reasoned, twice.

$$\frac{\varphi \to \psi \qquad \varphi}{\psi}$$

The mood that affirms. It is the only move anyone makes in this chapter.

The name is medieval; the rule is the first of the Stoics' five indemonstrables: "if it is day, it is light; it is day; so it is light".

Starting points

Is that all there is to say about implication? No. Some implications hold whatever the policy says:

"If Smith is a maintainer, then Smith is a maintainer."

Nobody should have to write that into MAINTAINERS.md. If such things are to come *out of* the logic rather than be told to it, the rule needs some starting points to work on.

Two will do.

Each is written with φ, ψ, χ standing for any formulas whatever: not one axiom but the shape of infinitely many.

2.1 · FORMULAS, AXIOMS, AND PROOFS

The first shape

$$\varphi \to \psi \to \varphi$$

A truth is implied by anything: if φ holds, then φ holds whatever else, say ψ, does.

If Smith is a maintainer, then, whether or not he has write access, he is a maintainer. A fact once told stays told.

The second shape

$$(\varphi \to \psi \to \chi) \to (\varphi \to \psi) \to \varphi \to \chi$$

Modus ponens carried out under a hypothesis*: if, given φ, you have $\psi \to \chi$, and, given φ, you have ψ, then, given φ, you have χ.*

If, when Smith is a maintainer, write access lets him publish, and, when Smith is a maintainer, he has write access, then, when Smith is a maintainer, he may publish.

Definition 2.4: proof from hypotheses

The axioms are all instances of two schemas, where φ, ψ, χ stand for arbitrary formulas:

$$\text{(I)} \quad \varphi \to \psi \to \varphi$$

$$\text{(II)} \quad (\varphi \to \psi \to \chi) \to (\varphi \to \psi) \to \varphi \to \chi$$

Let Γ be a set of formulas, the hypotheses. A proof of φ from Γ is a finite sequence of formulas $\varphi_1, \dots, \varphi_n$ whose last line is φ, in which each line φ_k is justified in one of three ways:

1. (*axiom*) φ_k is an instance of (I) or (II);
2. (*hypothesis*) φ_k is a member of Γ;
3. (*modus ponens*) there are earlier lines φ_i and φ_j, with $i, j < k$, such that φ_i is the formula $\varphi_j \to \varphi_k$.

No other kind of line is allowed.

Definition 2.4: theorems

If such a proof exists, we write

$$\Gamma \vdash \varphi$$

read "φ is provable from Γ". A proof from Γ does not know whether its hypotheses hold; it says what follows if they do.

When Γ is empty the second clause never applies. We write $\vdash \varphi$ and call φ a theorem: it holds by virtue of its logical structure alone, with nothing told.

A proof is written as a box in the Fitch style of the notes: the hypotheses above the line, the numbered lines below it, each with its justification on the right. "MP i, j": line i is the implication, line j its antecedent.

Example: instances of (I)

A schema stands for infinitely many axioms, all of one shape. To get an instance, substitute a formula for each of φ and ψ.

SUBSTITUTION	INSTANCE OF (I)	IN ENGLISH
$\varphi := p,\ \psi := q$	$p \to q \to p$	If Smith is a maintainer, then, if he has write access, he is a maintainer.
$\varphi := p,\ \psi := p$	$p \to p \to p$	If Smith is a maintainer, then, if he is a maintainer, he is a maintainer.
$\varphi := p \to q,\ \psi := r$	$(p \to q) \to r \to (p \to q)$	If being a maintainer gives Smith write access, then, if he may publish, being a maintainer gives him write access.

Same formula for φ at both ends. ψ is free.

Example: instances of (II)

Substitute for φ, ψ, χ in $(\varphi \to \psi \to \chi) \to (\varphi \to \psi) \to \varphi \to \chi$.

SUBSTITUTION	INSTANCE OF (II)
$\varphi := p,\ \psi := q,\ \chi := r$	$(p \to q \to r) \to (p \to q) \to p \to r$
$\varphi := p,\ \psi := p \to p,\ \chi := p$	$(p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$
$\varphi := q,\ \psi := p,\ \chi := q$	$(q \to p \to q) \to (q \to p) \to q \to q$

The second line is the one we will need in a few minutes. Check it against the schema.

Your turn: is it an axiom?

Is each an instance of (I), of (II), or neither?

1. $p \to q \to q$
2. $q \to (p \to p) \to q$
3. $(p \to q) \to (q \to r) \to p \to r$
4. $(p \to p \to p) \to (p \to p) \to p \to p$

ONE MINUTE

Find the substitution, or say why none exists.

Your turn: the answer

ITEM	FORMULA	VERDICT
1.	$p \to q \to q$	neither: (I) needs the same formula at both ends
2.	$q \to (p \to p) \to q$	(I) with $\varphi := q,\ \psi := p \to p$
3.	$(p \to q) \to (q \to r) \to p \to r$	neither: it is what chaining two implications *concludes*, not an instance of a schema
4.	$(p \to p \to p) \to (p \to p) \to p \to p$	(II) with $\varphi := p,\ \psi := p,\ \chi := p$

Checking a shape is a matter of comparing the pieces. No understanding required.

2.1 · FORMULAS, AXIOMS, AND PROOFS

The clerk's job

AN AXIOM

Does the line match (I) or (II)? A simple parser decides, symbol by symbol.

MODUS PONENS

Find the two cited lines. Confirm, symbol for symbol, that one of them reads ⟨the other⟩ → ⟨the new line⟩.

That is the whole job. Neither step asks what the letters mean.

A line may cite only lines above it, so the check is a single pass down the page.

Example: Monday's box

Monday's list, with "hypothesis" for "told" and "MP" for "from". $\Gamma = \{p,\ p \to q,\ q \to r\}$; show $\Gamma \vdash r$.

LINE	FORMULA	JUSTIFICATION
	$p,\ p \to q,\ q \to r$	
1.	p	hypothesis
2.	$p \to q$	hypothesis
3.	q	MP 2, 1
4.	$q \to r$	hypothesis
5.	r	MP 4, 3

Nothing else has changed. A proved line is used exactly like a hypothesis: line 3 is cited by line 5.

Example: Jones's box

Jones's argument, written the same way. $\Gamma = \{j \to k,\ k\}$.

LINE	FORMULA	JUSTIFICATION
	$j \to k,\ k$	
1.	$j \to k$	hypothesis
2.	k	hypothesis
3.	j	MP 1, 2 refused

For "MP 1, 2" to justify j, line 1 would have to read $\langle \text{line 2} \rangle \to j$, that is, $k \to j$. It reads $j \to k$. No other clause puts j on a line.

The clerk did not need to know who Jones is. Jones's mistake is a mistake of ***shape****.*

Your turn: spot the bad line

Someone says: we were told only that if Smith is a maintainer, he has write access; so he has write access. They write:

LINE	FORMULA	JUSTIFICATION
	$p \to q$	
1.	$p \to q$	hypothesis
2.	p	hypothesis
3.	q	MP 1, 2

They claim $\{p \to q\} \vdash q$. Check it against Definition 2.4.

ONE MINUTE

Which line, if any, breaks the rules?

Your turn: the answer

- Line 2 is justified as a hypothesis, but p is *not* a member of $\Gamma = \{p \to q\}$. We were never told that Smith is a maintainer. The list is not a proof.
- The same list *is* a proof of q from $\{p \to q,\ p\}$: the first three lines of Monday's box.

A hypothesis is what you were told. You cannot tell yourself more.

Supposing

Hughes would like to file the rule that a maintainer may publish, $p \to r$, derived from the policy alone. Here is how a person proves it.

LINE	STATEMENT	JUSTIFICATION
1.	Suppose Smith is a maintainer.	supposition
2.	Then, by the policy, he has write access.	
3.	Then, by the policy again, he may publish.	
4.	So, if he is a maintainer, he may publish.	supposition paid off

Every if *anyone has ever proved was proved that way: suppose the* if*, reach the* then*, and discharge.*

The clerk cannot suppose

Look at the three kinds of line. “Suppose” is not one of them.

A hypothesis is for keeps. It is a member of Γ, and whatever is proved with its help is proved *from* Γ, the hypothesis still owed at the end.

What the clerk can do is build the implication as a formula, out of instances of (I) and (II) that carry the supposition inside them. The most trivial *if* of all shows what that costs:

$$\vdash p \to p$$

A person supposes p and is done. The clerk needs five lines, and the first of them is the one that is hard to find.

Lemma 2.7: $\vdash p \to p$

No hypotheses. The square brackets give the formulas substituted for φ, ψ, χ, in that order.

LINE	FORMULA	JUSTIFICATION
	no hypotheses	
1.	$(p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$	(II) $[p,\ p \to p,\ p]$
2.	$p \to (p \to p) \to p$	(I) $[p,\ p \to p]$
3.	$(p \to (p \to p)) \to p \to p$	MP 1, 2
4.	$p \to (p \to p)$	(I) $[p,\ p]$
5.	$p \to p$	MP 3, 4

Each line is trivial to check. *Line 1 is not trivial to* find.

Lemma 2.7: in English

Let S be the claim "if Smith is a maintainer, he is a maintainer". We use it as a stepping stone, before having proved it.

- Line 4 (first shape): *given that Smith is a maintainer, S holds*: any claim then implies that he is a maintainer, and S is one such claim.
- Line 2 (first shape): *given that Smith is a maintainer, "if S then he is a maintainer" holds*, as that is already in hand.
- Line 1 (second shape): from those two, *given that Smith is a maintainer, he is a maintainer*.

Lines 3 and 5 are modus ponens: bookkeeping.

Lemma 2.7: finding line 1

The thinking runs backwards from the goal. That is how all proofs are found.

1. The goal $p \to p$ is not an axiom, so it comes by modus ponens: from some χ and $\chi \to (p \to p)$.
2. Could $\chi \to (p \to p)$ be an instance of (I)? Then χ would be p, and we would need a proof of p on its own. There cannot be one: nothing in the system proves a bare letter.
3. So try (II). Its last part, $\varphi \to \chi$, must be $p \to p$. That fixes $\varphi = \chi = p$ and leaves ψ free.
4. The two premises are then $p \to (\psi \to p)$ and $p \to \psi$. Choose $\psi := p \to p$, and both are instances of (I).

That is line 1. The search took a paragraph; the check takes none.

SECTION 2.2

Proofs are programs

Where the two shapes came from.

2.2 · PROOFS ARE PROGRAMS

Frege

Where did the two shapes come from? Not from Hilbert. He found the form of a proof in *Principia*, and Russell had found it in Frege, whose *Begriffsschrift* of 1879 is the book with which modern logic began.

Frege found ordinary language too loose: an argument in words can lean on something nobody stated. So he built a notation in which every step follows from earlier ones by a rule stated in advance.

> "To prevent anything intuitive from penetrating here unnoticed, I had to bend every effort to keep the chain of inferences free of gaps."

For if, *Frege gave three starting points. (I) and (II) are the first two, letter for letter.*

The third follows from the other two, as Łukasiewicz noticed fifty years later (Exercise 2.3).

2.2 · PROOFS ARE PROGRAMS

Curry, 1934

Curry worked out what types K and S must have, the types that (K) and (S) give them in Section 1.6, and found that he had seen them before.

SOURCE	FORMULA
(K), the type of K	$A \to B \to A$
(I), Frege's first axiom	$\varphi \to \psi \to \varphi$
(S), the type of S	$(A \to B \to C) \to (A \to B) \to A \to C$
(II), Frege's second axiom	$(\varphi \to \psi \to \chi) \to (\varphi \to \psi) \to \varphi \to \chi$

Write Greek for Latin and each pair is one formula written twice.

2.2 · PROOFS ARE PROGRAMS

Types and formulas

To you, $A \to B$ is the type of a function from A to B. To Frege, it was the statement "if A then B". Is that a pun, or is something going on? Put a proof and a typing side by side.

IN A TYPING	IN A PROOF
$\mathsf{K} : A \to B \to A$	schema (I), with the K left off
$\mathsf{S} : (A \to B \to C) \to \cdots$	schema (II), with the S left off
$x : A$ above the line	a hypothesis, with the x left off
(app): from $M : A \to B$ and $N : A$, $M\,N : B$	modus ponens, with the M and N left off

Let us see it in action.

2.2 · PROOFS ARE PROGRAMS

The policy, as types

Monday's reasoning was done by hand. Hughes would rather the release bot did it. A bot is a program, so she writes the policy in the one language the compiler reads: types.

```
smith            :: Maintainer                    -- MAINTAINERS.md
maintainersWrite :: Maintainer -> WriteAccess     -- policy, sentence 1
writeMayPublish  :: WriteAccess -> MayPublish     -- policy, sentence 2
publish          :: MayPublish -> Release -> IO ()
```

Each of the three propositions is a type, and each of the two policy sentences a function.

A value of type `Maintainer` is evidence that somebody is a maintainer; `smith` is the evidence the MAINTAINERS file supplies.

2.2 · PROOFS ARE PROGRAMS

The ticket

`publish` does not take a yes or a no, which a careless workflow could skip asking for.

It takes a value of type `MayPublish`: a ticket. Nothing in the program can make one except the policy functions, applied in the right order.

Read the types as sentences about Smith: `Maintainer` *is* p, `WriteAccess` *is* q, `MayPublish` *is* r, *and the two functions have types* $p \to q$ *and* $q \to r$.

In Haskell she gets the guarantee by exporting the three types from a module without their constructors.

2.2 · PROOFS ARE PROGRAMS

Demonstration: the typing

To release 2.3 Hughes writes `publish (writeMayPublish (maintainersWrite smith)) rel`. The compiler types the ticket, with `smith` and the two policy functions declared above the line.

LINE	JUDGEMENT	JUSTIFICATION
	`smith : Maintainer`	
	`maintainersWrite : Maintainer → WriteAccess`	
	`writeMayPublish : WriteAccess → MayPublish`	
1.	`smith : Maintainer`	(var)
2.	`maintainersWrite : Maintainer → WriteAccess`	(var)
3.	`maintainersWrite smith : WriteAccess`	(app) 2, 1
4.	`writeMayPublish : WriteAccess → MayPublish`	(var)
5.	`writeMayPublish (maintainersWrite smith) : MayPublish`	(app) 4, 3

Line 3 applies line 2 to line 1; line 5 applies line 4 to line 3. By the bottom, the program has been put together from its parts.

2.2 · PROOFS ARE PROGRAMS

Demonstration: rub out the programs

Cover the left-hand column with your thumb, the programs and the colons, and read the types as sentences about Smith.

LINE	TYPE	RULE	AS A FORMULA	AS A PROOF
1.	`Maintainer`	(var)	p	hypothesis
2.	`Maintainer -> WriteAccess`	(var)	$p \to q$	hypothesis
3.	`WriteAccess`	(app) 2, 1	q	MP 2, 1
4.	`WriteAccess -> MayPublish`	(var)	$q \to r$	hypothesis
5.	`MayPublish`	(app) 4, 3	r	MP 4, 3

Monday's box, line for line. Each (var) is a told*; each (app) is a* from*.*

The expression Hughes wrote to get Smith his ticket *is* the proof that Smith may have one. She set out to write a program; the compiler, in accepting it, checked a proof. The compiler is the clerk.

2.2 · PROOFS ARE PROGRAMS

Demonstration: the mistakes

SKIPPING THE POLICY

```
publish smith rel

Couldn't match expected type 'MayPublish'
with actual type 'Maintainer'
```

Chapter 1's red line. Programs rubbed out: r from p, with no implication to fire.

JONES

```
rotateKey :: Maintainer -> IO ()
```

Hughes has a `WriteAccess` for Jones. She would need a function `WriteAccess -> Maintainer`, and there is none: the policy says no such thing.

Jones's bad argument is a type error. The compiler refuses him for the reason the clerk did: the line he needs is not there.

2.2 · PROOFS ARE PROGRAMS

Demonstration: S K K

Hughes's program is not special. Example 1.27, the typing of S K K at $A \to A$, with p for A.

LINE	JUDGEMENT	JUSTIFICATION
	empty context	
1.	$\mathsf{S} : (p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$	(S) $[p,\ p \to p,\ p]$
2.	$\mathsf{K} : p \to (p \to p) \to p$	(K) $[p,\ p \to p]$
3.	$\mathsf{S}\,\mathsf{K} : (p \to (p \to p)) \to p \to p$	(app) 1, 2
4.	$\mathsf{K} : p \to (p \to p)$	(K) $[p,\ p]$
5.	$\mathsf{S}\,\mathsf{K}\,\mathsf{K} : p \to p$	(app) 3, 4

Now rub out the programs.

2.2 · PROOFS ARE PROGRAMS

Demonstration: rubbed out

Line 1 loses its S; line 3 loses its "(app) 1, 2" and gains "MP 1, 2"; the empty context becomes no hypotheses.

	LINE FORMULA	JUSTIFICATION
	no hypotheses	
1.	$(p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$	(II) $[p,\ p \to p,\ p]$
2.	$p \to (p \to p) \to p$	(I) $[p,\ p \to p]$
3.	$(p \to (p \to p)) \to p \to p$	MP 1, 2
4.	$p \to (p \to p)$	(I) $[p,\ p]$
5.	$p \to p$	MP 3, 4

Lemma 2.7, line for line, down to the first line that nobody would write unprompted.

The identity function *is* the proof that p implies p.

2.2 · PROOFS ARE PROGRAMS

Demonstration: the other way round

Go down the five lines of Lemma 2.7 writing K beside each use of (I), S beside each use of (II), and an application beside each modus ponens.

LINE	FORMULA	BY	WRITE BESIDE IT
1.	$(p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$	(II)	S
2.	$p \to (p \to p) \to p$	(I)	K
3.	$(p \to (p \to p)) \to p \to p$	MP 1, 2	SK
4.	$p \to (p \to p)$	(I)	K
5.	$p \to p$	MP 3, 4	SKK

When you reach the bottom you have written SKK. *Keep track of how a proof was put together and what you have built is a program.*

2.2 · PROOFS ARE PROGRAMS

The dictionary

For a context $\Gamma = x_1 : A_1, \dots, x_n : A_n$ write $|\Gamma| = \{A_1, \dots, A_n\}$: the types declared in it, read as formulas.

LINE OF A TYPING	LINE OF A PROOF
$\mathsf{K} : B \to C \to B$, by (K)	$B \to C \to B$, an instance of (I)
$\mathsf{S} : (B \to C \to D) \to \cdots$, by (S)	$(B \to C \to D) \to \cdots$, an instance of (II)
$x : B$, by (var), $x : B$ being in Γ	B, a hypothesis, B being in $\lvert\Gamma\rvert$
$M\,N : C$, by (app) i, j	C, by MP i, j

Read left to right, the table rubs programs out. Read right to left, it writes them in.

2.2 · PROOFS ARE PROGRAMS

Theorem 2.13: Curry's correspondence

THEOREM 2.13 · CURRY'S CORRESPONDENCE

(i) (*Erasure.*) Replace every line of a typing of $\Gamma \vdash M : A$ by its partner in the right-hand column. The result is a proof of A from $|\Gamma|$.
(ii) (*Decoration.*) Take a proof of φ from some hypotheses, give each hypothesis a variable, and replace every line by its partner in the left-hand column, working downwards. The result is a typing of $\Gamma \vdash M : \varphi$ for some term M.

In one line: $\Gamma \vdash M : A$ *for some term* M *if and only if* $|\Gamma| \vdash A$.

Theorem 2.13: why

All that needs checking is that each line's justification survives the change. Every line of a typing has the same context Γ (Definition 1.22), so take the four kinds of line in turn.

1. A line by (K) has type $B \to C \to B$: erased, an instance of (I).
2. A line by (S) has type $(B \to C \to D) \to (B \to C) \to B \to D$: erased, an instance of (II).
3. A line by (var) has a type listed in Γ: erased, a member of $|\Gamma|$, a hypothesis.
4. A line by "(app) i, j" has type C, where line i has type $B \to C$ and line j has type B. Erased, "MP i, j" is a correct justification for C.

That is (i). For (ii), run the same four cases upwards.

2.2 · PROOFS ARE PROGRAMS

What the theorem says

It does not say that proofs *resemble* programs, or that there is a useful analogy.

It says that a typing of Chapter 1 and a proof of this chapter are the **same list**, the one with an extra column; and that the four rules the compiler was checking your code against are, column removed, the rules Frege wrote down in 1879.

Types are formulas, and a typing is a proof. Not "resembles": is.

Curry saw it in 1934; he and Feys printed it in 1958. Howard said it for the richer logics of Chapters 3 and 4 in notes of 1969, printed in 1980: the Curry–Howard correspondence.

2.2 · PROOFS ARE PROGRAMS

Access control

Hughes's ticket is not a toy. It is how access control is done when it is done properly.

- Abadi, Burrows, Lampson and Plotkin (1993): a policy is a set of formulas, and a request is granted exactly when the resource is provable from them.
- Appel and Felten (1999): the *request* carries its proof, so the server, like the clerk, does nothing but check.
- Necula (1997): a program arriving over the network carries a proof that it is safe to run.

Whoever wants the access does the expensive part, finding the proof. The trusted part is a few dozen lines of pattern matching.

What stands between this and a system you could deploy is expressiveness: who delegated what to whom, and every member of a group at once. The next two chapters.

SECTION 2.3

More connectives

And, or, and not.

2.3 · MORE CONNECTIVES

Three things Hughes must write

Everything so far has used one connective. Can each of these be written with $\to$ alone?

	THE SENTENCE	WITH $\to$ ALONE
1.	People with write access may publish a release *and* edit the release notes.	✗ no: two formulas, but no single claim "both"
2.	The release was triggered by the scheduled job *or* by a maintainer pressing the button.	✗ no: nothing says "one or the other"
3.	A request that comes from a fork may *not* publish a release.	✗ no: an arrow only ever permits

So the language grows by two connectives and one constant, and the clerk's list of shapes grows with it.

2.3 · MORE CONNECTIVES

Definition 2.15: formulas, extended

Fix a stock of letters $p, q, r, \ldots$, as before. The formulas are defined inductively:

1. every letter is a formula;
2. $\bot$ is a formula, read "absurd" or "that cannot happen";
3. if φ, ψ are formulas, so is $\varphi \to \psi$;
4. if φ, ψ are formulas, so is $\varphi \wedge \psi$, "φ and ψ";
5. if φ, ψ are formulas, so is $\varphi \vee \psi$, "φ or ψ";
6. nothing else is a formula.

IN ONE LINE

$\varphi, \psi \ ::= \ p \mid \bot \mid \varphi \to \psi \mid \varphi \wedge \psi \mid \varphi \vee \psi$

NOT

$\neg\varphi$ abbreviates $\varphi \to \bot$: an arrow into $\bot$.

$\neg$ takes precedence over $\wedge$ and $\vee$, and they over $\to$. Clauses 1, 3 and 6 are Definition 2.1.

2.3 · MORE CONNECTIVES

Definition 2.17: nine schemas

The **axioms** are all instances of nine schemas, where φ, ψ, χ stand for arbitrary formulas.

SCHEMA	SHAPE
(I)	$\varphi \to \psi \to \varphi$
(II)	$(\varphi \to \psi \to \chi) \to (\varphi \to \psi) \to \varphi \to \chi$
(pair)	$\varphi \to \psi \to \varphi \wedge \psi$
(fst), (snd)	$\varphi \wedge \psi \to \varphi, \quad \varphi \wedge \psi \to \psi$
(left), (right)	$\varphi \to \varphi \vee \psi, \quad \psi \to \varphi \vee \psi$
(either)	$(\varphi \to \chi) \to (\psi \to \chi) \to \varphi \vee \psi \to \chi$
(absurd)	$\bot \to \varphi$

The names are those of the Haskell functions whose types the schemas are: `(,)`, `fst`, `snd`, `Left`, `Right`, `either`, `absurd`. (I) and (II) could as well have been called (K) and (S).

2.3 · MORE CONNECTIVES

Definition 2.17: the same three kinds of line

A proof of φ from Γ is a box, as before, in which each line is

1. (*axiom*) an instance of one of the nine schemas;
2. (*hypothesis*) a member of Γ;
3. (*modus ponens*) the φ_k of earlier lines $\varphi_j \to \varphi_k$ and φ_j.

Put this beside Definition 2.4. Same box, same three kinds of line, same rule. Only the list of shapes has grown, from two to nine.

Every proof of the earlier sections is still a proof. The clerk is the same clerk with a longer list.

2.3 · MORE CONNECTIVES

And

With r "Smith may publish", a "Smith may edit the release notes".

SCHEMA	SHAPE	HASKELL	FOR HUGHES
(pair)	$\varphi \to \psi \to \varphi \wedge \psi$	`(,)`	if Smith may publish, then, if he may edit the notes, he may publish and edit the notes
(fst)	$\varphi \wedge \psi \to \varphi$	`fst`	someone who may publish and edit the notes may publish
(snd)	$\varphi \wedge \psi \to \psi$	`snd`	and may edit the notes

To make a conjunction you need both halves. To use one you take a half.

2.3 · MORE CONNECTIVES

Or

With b "the scheduled job triggered the release", h "a maintainer pressed the button", r "the release may go ahead".

SCHEMA	SHAPE	HASKELL	FOR HUGHES
(left)	$\varphi \to \varphi \vee \psi$	`Left`	if the job triggered it, the job or a maintainer did
(right)	$\psi \to \varphi \vee \psi$	`Right`	likewise for the maintainer
(either)	$(\varphi \to \chi) \to (\psi \to \chi) \to \varphi \vee \psi \to \chi$	`either`	if the job may publish, and a maintainer may, then whichever did it, it may go ahead

To make a disjunction you need one side. To use one you must handle both: (either) is proof by cases.

2.3 · MORE CONNECTIVES

Not

A new formula $\bot$, read "absurd": *that cannot happen.* Nothing makes $\bot$.

$$\text{(absurd)} \quad \bot \to \varphi$$

From what cannot happen, anything follows. Haskell: `absurd :: Void -> a`.

And $\neg\varphi$ abbreviates $\varphi \to \bot$, read "not φ". "A fork may not publish" is $u \to \neg r$, that is, $u \to r \to \bot$. u: "the request comes from a fork".

A prohibition is an arrow into absurdity.

2.3 · MORE CONNECTIVES

Your turn: is it an instance?

Which schema, with which substitution? Or none?

1. $q \to p \vee q$
2. $p \wedge q \to q \wedge p$
3. $(p \to r) \to (q \to r) \to p \vee q \to r$
4. $\bot \to p \wedge q$

ONE MINUTE

Same check as in Section 2.1, with nine shapes instead of two.

2.3 · MORE CONNECTIVES

Your turn: the answer

ITEM	FORMULA	VERDICT
1.	$q \to p \vee q$	(right) with $\varphi := p,\ \psi := q$
2.	$p \wedge q \to q \wedge p$	none: to be proved (Example 2.21), not an axiom
3.	$(p \to r) \to (q \to r) \to p \vee q \to r$	(either) with $\varphi, \psi, \chi := p,\ q,\ r$
4.	$\bot \to p \wedge q$	(absurd) with $\varphi := p \wedge q$

Still pattern matching. The clerk is the same clerk, with seven more shapes on his list.

2.3 · MORE CONNECTIVES

Example 2.18: proof by cases

The bot may publish when the job triggers it ($b \to r$), and when a maintainer presses the button ($h \to r$). The log says one of the two happened ($b \vee h$), not which. May the release go ahead?

LINE	FORMULA	JUSTIFICATION
	$b \to r,\ h \to r,\ b \vee h$	
1.	$b \to r$	hypothesis
2.	$h \to r$	hypothesis
3.	$(b \to r) \to (h \to r) \to b \vee h \to r$	(either) $[b,\ h,\ r]$
4.	$(h \to r) \to b \vee h \to r$	MP 3, 1
5.	$b \vee h \to r$	MP 4, 2
6.	$b \vee h$	hypothesis
7.	r	MP 5, 6

Two rules with the same conclusion are as good as one rule whose condition is "the one or the other". That is what (either) says. Hughes never learns which of the two it was.

2.3 · MORE CONNECTIVES

Example 2.19: commutativity of $\vee$

A colleague writes the log's sentence the other way round. Does it matter? Claim: $\vdash b \vee h \to h \vee b$. Write ε for $h \vee b$, the thing wanted.

LINE	FORMULA	JUSTIFICATION
	no hypotheses	
1.	$(b \to \varepsilon) \to (h \to \varepsilon) \to b \vee h \to \varepsilon$	(either) $[b,\ h,\ \varepsilon]$
2.	$b \to \varepsilon$	(right) $[h,\ b]$
3.	$(h \to \varepsilon) \to b \vee h \to \varepsilon$	MP 1, 2
4.	$h \to \varepsilon$	(left) $[h,\ b]$
5.	$b \vee h \to \varepsilon$	MP 3, 4

No hypotheses and no discharge: the two cases were both axioms, and (either) joined them.

If it was the job, then certainly a maintainer or the job did it; if a maintainer, likewise. The other direction is the same five lines with b and h exchanged.

2.3 · MORE CONNECTIVES

Example 2.20: from a contradiction, anything

A careless edit leaves the policy saying both that Smith has write access and that he does not. What follows? Everything: any r, "Jones may rotate the signing key", say.

LINE	FORMULA	JUSTIFICATION
	$q,\ \neg q$	
1.	q	hypothesis
2.	$q \to \bot$	hypothesis (that is, $\neg q$)
3.	$\bot$	MP 2, 1
4.	$\bot \to r$	(absurd) $[r]$
5.	r	MP 4, 3

"Not q" means that q leads to what cannot happen; but q holds; so what cannot happen has happened; and from that, anything. A conflict between an allow rule and a deny rule is an emergency.

2.3 · MORE CONNECTIVES

Example 2.21: commutativity of $\wedge$

"May publish and edit the notes", written by the colleague the other way round. r "may publish", a "may edit the notes". For now: told both, conclude both the other way round.

LINE	FORMULA	JUSTIFICATION
	$r \wedge a$	
1.	$r \wedge a$	hypothesis
2.	$r \wedge a \to a$	(snd) $[r,\ a]$
3.	a	MP 2, 1
4.	$r \wedge a \to r$	(fst) $[r,\ a]$
5.	r	MP 4, 1
6.	$a \to r \to a \wedge r$	(pair) $[a,\ r]$
7.	$r \to a \wedge r$	MP 6, 3
8.	$a \wedge r$	MP 7, 5

What Hughes would really like to file is the rule itself, $r \wedge a \to a \wedge r$, with nothing above the line. The clerk, who cannot suppose, cannot get it from this box.

2.3 · MORE CONNECTIVES

Example 2.22: currying

The two-factor rule: q "has write access", s "has two-factor on", r "may publish". A rule with two conditions can be written $q \wedge s \to r$ or $q \to s \to r$. Does it matter which?

$q \wedge s \to r,\ q,\ s$

LINE	FORMULA	JUSTIFICATION
1.	$q \wedge s \to r$	hypothesis
2.	q	hypothesis
3.	s	hypothesis
4.	$q \to s \to q \wedge s$	(pair) $[q,\ s]$
5.	$s \to q \wedge s$	MP 4, 2
6.	$q \wedge s$	MP 5, 3
7.	r	MP 1, 6

$q \to s \to r,\ q \wedge s$

LINE	FORMULA	JUSTIFICATION
1.	$q \to s \to r$	hypothesis
2.	$q \wedge s$	hypothesis
3.	$q \wedge s \to q$	(fst) $[q,\ s]$
4.	q	MP 3, 2
5.	$s \to r$	MP 1, 4
6.	$q \wedge s \to s$	(snd) $[q,\ s]$
7.	s	MP 6, 2
8.	r	MP 5, 7

That is not what she asked. The boxes say something weaker: from either rule and its conditions*, publish follows.*

Example 2.23: contraposition

The checker refuses Smith a release: he has no write access. That tells Hughes he is not a maintainer. She wants $\vdash (p \to q) \to \neg q \to \neg p$. The clerk can follow her as far as "which cannot happen":

LINE	FORMULA	JUSTIFICATION
	$p \to q,\ \neg q,\ p$	
1.	$p \to q$	hypothesis
2.	$q \to \bot$	hypothesis (that is, $\neg q$)
3.	p	hypothesis
4.	q	MP 1, 3
5.	$\bot$	MP 2, 4

$\{p \to q,\ \neg q,\ p\} \vdash \bot$. *That is what* $\neg p$ means*, but the clerk has* p *above the line, where it is of no use to anyone.*

Nothing about $\neg$ needed a rule of its own: it is an arrow into $\bot$, and the rules for $\to$ did the work.

2.3 · MORE CONNECTIVES

Three of six, a line short

EXAMPLE	THE CLERK REACHED	HUGHES WANTED
2.10	$\{r \wedge a\} \vdash a \wedge r$	$\vdash r \wedge a \to a \wedge r$
2.11	$\{q \wedge s \to r,\ q,\ s\} \vdash r$, and back	$\vdash (q \wedge s \to r) \to q \to s \to r$, and back
2.12	$\{p \to q,\ \neg q,\ p\} \vdash \bot$	$\vdash (p \to q) \to \neg q \to \neg p$

Always for the same reason: a supposition stuck above the line. The Deduction Theorem comes back for them.

SECTION 2.4

Proof theory

Facts about proofs, proved once and used for ever.

2.4 · PROOF THEORY

Hilbert's idea

A Hilbert proof represents reasoning in such minute detail that doing anything serious with one is impractical. Five lines to show that p implies p. And when the situation changes a little, you write out what is essentially the same proof again.

What spares you is a handful of facts about proofs: true of every box whatever, provable once, from the definition, and then usable for ever.

Once a proof is a finite list of formulas it is a mathematical object like any other, a thing one can prove theorems about.

That was Hilbert's idea, and it is what makes proof theory a subject.

2.4 · PROOF THEORY

Proposition 2.9: structural facts

PROPOSITION 2.9 · STRUCTURAL FACTS

For all sets of formulas Γ, Δ and all formulas φ, ψ:

1. (*reflexivity*) if $\varphi \in \Gamma$ then $\Gamma \vdash \varphi$;
2. (*monotonicity*) if $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$ then $\Delta \vdash \varphi$;
3. (*transitivity*) if $\Gamma \vdash \varphi$ and $\Delta \cup \{\varphi\} \vdash \psi$ then $\Gamma \cup \Delta \vdash \psi$.

In each case the job is to exhibit a proof of the required kind, built from the proofs we are given. Nothing is said about what the formulas mean, only about lists.

2.4 · PROOF THEORY

In Hughes's terms

FACT ABOUT ⊢	IN HUGHES'S TERMS	IN A PROGRAM
(1) reflexivity	What you were told, you may use.	A variable in scope may be used.
(2) monotonicity	Being told more never takes access away.	Extra names in scope do no harm.
(3) transitivity	A rule proved once is as good as a rule you were given.	A helper function may be inlined at its call sites.

The third is the one to remember: proofs ***compose****.*

From here on a line may be justified by the name of a fact already proved, and the box is still a proof: the clerk can splice the proof in if anyone asks.

2.4 · PROOF THEORY

Why: reflexivity and monotonicity

(1) The one-line sequence φ is a proof of φ from Γ: its only line is justified as a hypothesis, because φ is a member of Γ.

(2) A proof of φ from Γ is already a proof of φ from Δ: every line justified as a member of Γ is a member of Δ too, and the other two kinds of line do not mention Γ.

In neither case is a proof rewritten.

2.4 · PROOF THEORY

Why: transitivity is splicing

- Write out the proof of φ from Γ.
- After it, write the proof of ψ from $\Delta \cup \{\varphi\}$, with every line justified as "hypothesis φ" deleted.
- Each modus ponens that cited a deleted line now cites the last line of the first proof, which is φ.
- Every remaining hypothesis is in Γ or in Δ: a proof of ψ from $\Gamma \cup \Delta$.

Only this item rewrites a proof.

2.4 · PROOF THEORY

Example: a fact once told stays told

Show $\{p\} \vdash q \to p$: from "Smith is a maintainer", "if he has write access, he is a maintainer".

LINE	FORMULA	JUSTIFICATION
	p	
1.	$p \to q \to p$	(I) $[p,\ q]$
2.	p	hypothesis
3.	$q \to p$	MP 1, 2

Three lines: an instance of (I), the hypothesis, and one modus ponens.

2.4 · PROOF THEORY

Example: hypothetical syllogism

The rule that chains two implications. From the two sentences of the policy it derives the third that nobody wrote down: if Smith is a maintainer, he may publish. Show $\{p \to q,\ q \to r\} \vdash p \to r$.

LINE	FORMULA	JUSTIFICATION
	$p \to q,\ q \to r$	
1.	$q \to r$	hypothesis
2.	$(q \to r) \to p \to (q \to r)$	(I) $[q \to r,\ p]$
3.	$p \to (q \to r)$	MP 2, 1
4.	$(p \to (q \to r)) \to (p \to q) \to p \to r$	(II) $[p,\ q,\ r]$
5.	$(p \to q) \to p \to r$	MP 4, 3
6.	$p \to q$	hypothesis
7.	$p \to r$	MP 5, 6

2.4 · PROOF THEORY

Read the blocks, not the symbols

- **Lines 1–3.** Take the hypothesis $q \to r$ and weaken it, by (I), to $p \to (q \to r)$: a statement that says less, since it only promises $q \to r$ once p is given.
- **Lines 4–5.** Put that through (II), which turns "given p, q implies r" into "if p gives q then p gives r".
- **Line 7.** Fire the result on the other hypothesis.

Every Hilbert proof has this character: a few ideas, spelled out at a length the clerk can follow.

If you did Exercise 1.3, you have seen this list before, with programs on it: composition, applied to two functions, is the proof that chains two implications.

2.4 · PROOF THEORY

Hypothetical syllogism, with the programs on

Exercise 1.3, in the context $f : q \to r$, $g : p \to q$: the same seven lines, decorated by Theorem 2.13.

LINE	FORMULA	JUSTIFICATION	PROGRAM
1.	$q \to r$	hypothesis	f
2.	$(q \to r) \to p \to (q \to r)$	(I)	K
3.	$p \to (q \to r)$	MP 2, 1	$\mathsf{K}\, f$
4.	$(p \to q \to r) \to (p \to q) \to p \to r$	(II)	S
5.	$(p \to q) \to p \to r$	MP 4, 3	$\mathsf{S}\,(\mathsf{K}\, f)$
6.	$p \to q$	hypothesis	g
7.	$p \to r$	MP 5, 6	$\mathsf{S}\,(\mathsf{K}\, f)\, g$

Composition, applied to two functions, is the proof that chains two implications.

2.4 · PROOF THEORY

Example: the filed rule

By transitivity those seven lines never have to be written again. Hughes's rule is filed, and Monday's argument, five lines from the three things she was told, takes three:

LINE	FORMULA	JUSTIFICATION
	$p,\ p \to q,\ q \to r$	
1.	$p \to r$	hypothetical syllogism
2.	p	hypothesis
3.	r	MP 1, 2

Line 1 is not an axiom and not a hypothesis. It is justified by a fact about proofs.

If anyone objects, the seven lines can be spliced in and the objection goes away.

2.4 · PROOF THEORY

Example: chaining three

Or chain three implications instead of two: $\{p \to q,\ q \to r,\ r \to s\} \vdash p \to s$ is two uses of the filed rule.

LINE	FORMULA	JUSTIFICATION
	$p \to q,\ q \to r,\ r \to s$	
1.	$p \to r$	hypothetical syllogism
2.	$p \to s$	hypothetical syllogism, from 1 and $r \to s$

With the splices made, thirteen lines; each further implication adds six.

Monotonicity is what allows line 2: the rule was proved from two hypotheses, and here it is used among three.

SECTION 2.5

The Deduction Theorem

The clerk cannot suppose. He does not need to.

2.5 · THE DEDUCTION THEOREM

Nobody proves an implication this way

To prove $\varphi \to \psi$ a person *supposes* φ, reasons until ψ appears, and concludes $\varphi \to \psi$.

There is no way to do that inside a Hilbert proof, because "suppose" is not one of the three kinds of line.

I want to show you what that costs before I show you the way round it.

Otherwise the way round looks like a convenience, and it is a good deal more than that. The complaint is old: Łukasiewicz put it to his seminar in Warsaw in 1926, and by 1927 his student Jaśkowski had a logic in which the clerk may suppose. That is Chapter 3.

The cost

Section 2.4 proved hypothetical syllogism *from* its two implications. Suppose instead that we want it with nothing above the line, the implications part of the formula:

$$(q \to r) \to (p \to q) \to p \to r$$

It says the same thing, near enough. Its Hilbert proof is a different animal.

To keep it legible, abbreviate $\alpha := q \to r$, $\beta := p \to q \to r$, $\gamma := (p \to q) \to p \to r$. The goal is $\alpha \to \gamma$.

2.5 · THE DEDUCTION THEOREM

The proof from the schemas alone

No hypotheses. $\alpha := q \to r$, $\beta := p \to q \to r$, $\gamma := (p \to q) \to p \to r$.

LINE	FORMULA	JUSTIFICATION
	no hypotheses	
1.	$(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma$	(II) $[\alpha,\ \beta,\ \gamma]$
2.	$\beta \to \gamma$, that is, $(p \to q \to r) \to (p \to q) \to p \to r$	(II) $[p,\ q,\ r]$
3.	$(\beta \to \gamma) \to \alpha \to (\beta \to \gamma)$	(I) $[\beta \to \gamma,\ \alpha]$
4.	$\alpha \to \beta \to \gamma$	MP 3, 2
5.	$(\alpha \to \beta) \to \alpha \to \gamma$	MP 1, 4
6.	$\alpha \to \beta$, that is, $(q \to r) \to p \to (q \to r)$	(I) $[q \to r,\ p]$
7.	$\alpha \to \gamma$	MP 5, 6

Seven lines again, and each checks in a moment.

2.5 · THE DEDUCTION THEOREM

Line 1

With the abbreviations undone, line 1 is an instance of (II) with twenty letters and nineteen arrows:

$$((q \to r) \to (p \to q \to r) \to (p \to q) \to p \to r) \\ \to \ ((q \to r) \to (p \to q \to r)) \to (q \to r) \to (p \to q) \to p \to r$$

Nobody finds that by thinking about what implication means.

Compare the proof from hypotheses. There, $q \to r$ and $p \to q$ were simply written down as lines 1 and 6. Here they have been pushed *into the formula*, and line 1's only job is to carry out, under the hypothesis α, the modus ponens that line 5 performed with the hypotheses on the table. The proof with no hypotheses contains the proof from hypotheses, rebuilt one level up.

2.5 · THE DEDUCTION THEOREM

What one would want

PROVING FROM HYPOTHESES

Easy and natural: write the hypothesis down and use it.

PROVING AN IMPLICATION

Hard and unnatural: instances of (II) that grow with the formulas they carry.

A fact about proofs that turns the first into the second:

$$\text{if } \Gamma \cup \{\varphi\} \vdash \psi \text{ then } \Gamma \vdash \varphi \to \psi$$

Read as a licence: to prove $\varphi \to \psi$, suppose φ, reach ψ, and discharge. Exactly the move the clerk was not given.

2.5 · THE DEDUCTION THEOREM

Far too good?

Every implication in the chapter would then cost what it costs a person.

- The twenty-letter line: two modus ponens and three discharges.
- $p \to p$: one line and a discharge.
- Hypothetical syllogism as a theorem: the proof from hypotheses, its two hypotheses discharged in turn.

It looks far too good: nothing in (I), (II) and modus ponens mentions discharging anything.

And yet it holds, and for the best possible reason. There is an ***algorithm*** *that takes a proof of* ψ *from* Γ *and* φ *and produces, line by line, a proof of* $\varphi \to \psi$ *from* Γ *alone.*

The theorem says that the algorithm works, and the proof of the theorem is the algorithm.

2.5 · THE DEDUCTION THEOREM

Theorem 2.12: Deduction Theorem

For every set of formulas Γ and all formulas φ, ψ:

$$\Gamma \cup \{\varphi\} \vdash \psi \quad \Longleftrightarrow \quad \Gamma \vdash \varphi \to \psi$$

Right to left is one use of modus ponens. Left to right is the transformation.

The theorem is Jacques Herbrand's, from a doctoral thesis finished in 1929, when he was twenty-two. A year later he was dead, climbing in the Écrins.

2.5 · THE DEDUCTION THEOREM

Right to left

Suppose $\Gamma \vdash \varphi \to \psi$. Show $\Gamma \cup \{\varphi\} \vdash \psi$.

1. Take a proof of $\varphi \to \psi$ from Γ. It is also a proof from $\Gamma \cup \{\varphi\}$, by monotonicity.
2. Add the line φ, justified as a hypothesis: φ is a member of $\Gamma \cup \{\varphi\}$.
3. Add the line ψ, by modus ponens on $\varphi \to \psi$ and φ.

The easy direction: use the implication.

2.5 · THE DEDUCTION THEOREM

Left to right: the plan

Let $\varphi_1, \ldots, \varphi_n = \psi$ be a proof of ψ from $\Gamma \cup \{\varphi\}$. Go down it one line at a time. For each line φ_k, write out a proof of $\varphi \to \varphi_k$ from Γ *alone*, using the proofs already written for the earlier lines. At the last line, $\varphi_n = \psi$, we have a proof of $\varphi \to \psi$.

INDUCTION ON k

Hypothesis: for every $j < k$, a proof of $\varphi \to \varphi_j$ from Γ has been written.

To show: a proof of $\varphi \to \varphi_k$ from Γ, built on top of those.

THREE CASES

φ_k is an axiom or a member of Γ; or φ_k is φ itself; or φ_k follows by modus ponens.

An axiom and a member of Γ are handled alike.

2.5 · THE DEDUCTION THEOREM

Case 1: an axiom or a member of Γ

Then φ_k is a legitimate line of a proof from Γ. Write:

LINE	FORMULA	JUSTIFICATION
Γ		
a.	φ_k	as before
b.	$\varphi_k \to \varphi \to \varphi_k$	(I) $[\varphi_k,\ \varphi]$
c.	$\varphi \to \varphi_k$	MP b, a

Three lines. A line we did not need φ for stays true when we suppose φ as well; (I) says exactly that.

"Write access lets you publish" was told. So: if Smith is a maintainer, write access lets you publish.

Case 2: φ_k is φ itself

Then $\varphi \to \varphi_k$ is $\varphi \to \varphi$, and the five-line proof of Lemma 2.7, with φ for p, proves it from no hypotheses at all, so in particular from Γ.

LINE	FORMULA	JUSTIFICATION
	Γ	
a.	$(\varphi \to (\varphi \to \varphi) \to \varphi) \to (\varphi \to (\varphi \to \varphi)) \to \varphi \to \varphi$	(II) $[\varphi,\ \varphi \to \varphi,\ \varphi]$
b.	$\varphi \to (\varphi \to \varphi) \to \varphi$	(I) $[\varphi,\ \varphi \to \varphi]$
c.	$(\varphi \to (\varphi \to \varphi)) \to \varphi \to \varphi$	MP a, b
d.	$\varphi \to (\varphi \to \varphi)$	(I) $[\varphi,\ \varphi]$
e.	$\varphi \to \varphi$	MP c, d

"Suppose Smith is a maintainer; then he is." This is where the five-line proof was waiting to be used.

2.5 · THE DEDUCTION THEOREM

Case 3: modus ponens

Then there are earlier lines φ_j and $\varphi_i = \varphi_j \to \varphi_k$. By the induction hypothesis, proofs from Γ of $\varphi \to \varphi_j$ and of $\varphi \to \varphi_j \to \varphi_k$ have already been written.

LINE	FORMULA	JUSTIFICATION
	Γ	
a.	$(\varphi \to \varphi_j \to \varphi_k) \to (\varphi \to \varphi_j) \to \varphi \to \varphi_k$	(II) $[\varphi,\ \varphi_j,\ \varphi_k]$
b.	$(\varphi \to \varphi_j) \to \varphi \to \varphi_k$	MP a, $(\varphi \to \varphi_j \to \varphi_k)$
c.	$\varphi \to \varphi_k$	MP b, $(\varphi \to \varphi_j)$

Three lines, on top of the proofs already written. Only this case uses the induction hypothesis.

Concatenating the pieces in order gives a proof from Γ whose last line is $\varphi \to \varphi_n$, that is, $\varphi \to \psi$.

2.5 · THE DEDUCTION THEOREM

Demonstration: the input

A silly goal, since $p \to q$ was told, but a short run that shows every case. $\Gamma = \{p \to q\}$, the supposition is $\varphi = p$, and the given proof of q from $\Gamma \cup \{p\}$ is:

LINE	FORMULA	JUSTIFICATION	
	$p \to q,\ p$		
1.	$p \to q$	hypothesis	a member of Γ: case 1
2.	p	hypothesis	the supposition itself: case 2
3.	q	MP 1, 2	case 3

The output will be a proof of $p \to q$ from Γ alone. Three lines in, one block of output for each.

2.5 · THE DEDUCTION THEOREM

Demonstration: line 1, case 1

$\varphi_1 = p \to q$ is a member of Γ. Write it, weaken it by (I), fire.

LINE	FORMULA	JUSTIFICATION
	$p \to q$	
1.	$p \to q$	hypothesis
2.	$(p \to q) \to p \to (p \to q)$	(I) $[p \to q,\ p]$
3.	$p \to (p \to q)$	MP 2, 1

Three output lines, ending in $p \to \varphi_1$.

2.5 · THE DEDUCTION THEOREM

Demonstration: line 2, case 2

$\varphi_2 = p$ is the supposition itself. Copy in the five-line proof of $p \to p$.

LINE	FORMULA	JUSTIFICATION
	$p \to q$	
4.	$(p \to (p \to p) \to p) \to (p \to (p \to p)) \to p \to p$	(II) $[p,\ p \to p,\ p]$
5.	$p \to (p \to p) \to p$	(I) $[p,\ p \to p]$
6.	$(p \to (p \to p)) \to p \to p$	MP 4, 5
7.	$p \to (p \to p)$	(I) $[p,\ p]$
8.	$p \to p$	MP 6, 7

Five output lines, ending in $p \to \varphi_2$.

2.5 · THE DEDUCTION THEOREM

Demonstration: line 3, case 3

$\varphi_3 = q$ came by modus ponens from lines 1 and 2. We hold $p \to (p \to q)$ at output line 3 and $p \to p$ at output line 8. Distribute by (II), fire twice.

LINE	FORMULA	JUSTIFICATION
	$p \to q$	
9.	$(p \to p \to q) \to (p \to p) \to p \to q$	(II) $[p,\ p,\ q]$
10.	$(p \to p) \to p \to q$	MP 9, 3
11.	$p \to q$	MP 10, 8

Eleven lines, ending in $p \to \varphi_3$, that is, $p \to q$: a proof from $\{p \to q\}$ alone.

Three lines in, eleven out. Each discharge, carried out, roughly triples the proof it is applied to.

2.5 · THE DEDUCTION THEOREM

Why these two shapes?

Look at what the proof used. It answers the question put off in Section 2.1.

CASE	MET	USED
1	a line that did not depend on the supposition, and weakened it to one that mentions the supposition anyway	(I)
3	a modus ponens, and distributed the supposition over its two premises	(II)
2	the supposition itself	the five lines: one (II), two (I)s

The two shapes are not self-evident truths about if. *They are exactly, and only, what it takes to push a supposition into a formula, one line at a time.*

2.5 · THE DEDUCTION THEOREM

Seen from the computing end

Chapter 1 said that K throws an argument away and S copies one, and that this is why the two of them did the work of every variable. It promised the proof for this chapter. Here it is.

CASE 1 IS K

A line that does not use the parameter is wrapped to ignore it.

CASE 3 IS S

An application whose halves both use the parameter has it copied to each.

CASE 2 IS SKK

The identity, for the parameter itself.

Run the construction with the programs left on and it compiles a definition with a parameter into K *and* S*. "Why did two combinators suffice?" and "why these two axioms?" are one question.*

2.5 · THE DEDUCTION THEOREM

Demonstration: with the programs on

The run again, in the context $f : p \to q$, with each output line's program beside it. The input was the body of the definition $g\,x := f\,x$.

LINES	CASE	PROGRAM
1–3	case 1	f, K, K f
4–8	case 2	S K K
9	case 3	S
10		S (K f)
11		S (K f) (S K K)

THE OUTPUT

S (K f) (S K K) : $p \to q$

The parameter x is gone. No variable is left but f.

2.5 · THE DEDUCTION THEOREM

The missing rule

To type a definition $f\,x := M$, a checker wants to say: suppose x has type A; if on that supposition M has type B, then f has type $A \to B$.

$$\Gamma, x : A \vdash M : B \implies \Gamma \vdash f : A \to B$$

With the programs rubbed out: if $\Gamma \cup \{\varphi\} \vdash \psi$ then $\Gamma \vdash \varphi \to \psi$. That *is* the Deduction Theorem, left to right.

That is the rule for parameters that Chapter 1 did not have. It was never a rule. It was a theorem.

And a machine has run it: Turner's compiler for SASL (1979) is the theorem's construction, and SKIM (Cambridge, 1979–80) was a processor whose instruction set was S, K and I.

Example: hypothetical syllogism, by supposing

Suppose write access lets you publish, maintainers have write access, and Smith is a maintainer. Then he may publish.

LINE	FORMULA	JUSTIFICATION
	$q \to r,\ p \to q,\ p$	
1.	$q \to r$	hypothesis
2.	$p \to q$	hypothesis
3.	p	hypothesis
4.	q	MP 2, 3
5.	r	MP 1, 4

- Discharge p: $\{q \to r,\ p \to q\} \vdash p \to r$.
- Discharge $p \to q$: $\{q \to r\} \vdash (p \to q) \to p \to r$.
- Discharge $q \to r$: $\vdash (q \to r) \to (p \to q) \to p \to r$.

So the clerk may suppose after all. The twenty-letter line is what the third discharge writes.

2.5 · THE DEDUCTION THEOREM

Finishing Section 2.3

Three boxes ended with a supposition stuck above the line. “Discharge φ”, the Deduction Theorem cited as a fact, brings it down.

EXAMPLE	DISCHARGE	AND GET
2.10	$r \wedge a$	$\vdash r \wedge a \to a \wedge r$
2.11	first box: s, then q, then the rule; second box: $q \wedge s$, then the rule	$\vdash (q \wedge s \to r) \to q \to s \to r$, and back
2.12	p, giving $p \to \bot$, that is $\neg p$; then $\neg q$; then $p \to q$	$\vdash (p \to q) \to \neg q \to \neg p$

Each discharge is one line of thought and, carried out, some dozens of lines of proof. With the theorem in hand a bigger proof can be planned in English, with lemmas spliced in by transitivity.

Example 2.24: distributing $\wedge$ over $\vee$

The signing-key rule: people with write access who are on the release rota or the security rota may use the key. Hughes would rather have two simple rules, and wants them to cover everyone the one did. w "write access", e "release rota", c "security rota"; δ for $(w \wedge e) \vee (w \wedge c)$, the thing wanted.

$$\vdash w \wedge (e \vee c) \to \delta$$

In words: take someone with write access on one rota or the other. See which. Either way, they meet one of the two simple rules.

The two cases are two short lemmas.

2.5 · THE DEDUCTION THEOREM

Example 2.24: the first lemma

From write access and the release rota, conclude δ.

LINE	FORMULA	JUSTIFICATION
	$w,\ e$	
1.	w	hypothesis
2.	e	hypothesis
3.	$w \to e \to w \wedge e$	(pair) $[w,\ e]$
4.	$e \to w \wedge e$	MP 3, 1
5.	$w \wedge e$	MP 4, 2
6.	$w \wedge e \to \delta$	(left) $[w \wedge e,\ w \wedge c]$
7.	δ	MP 6, 5

Discharge e: $\{w\} \vdash e \to \delta$. The same seven lines with c for e, and (right) for (left), give $\{w\} \vdash c \to \delta$.

2.5 · THE DEDUCTION THEOREM

Example 2.24: the main proof

From the one rule's condition: take the two halves; use the two lemmas for the two cases of the second half.

LINE	FORMULA	JUSTIFICATION
	$w \wedge (e \vee c)$	
1.	$w \wedge (e \vee c)$	hypothesis
2.	$w \wedge (e \vee c) \to w$	(fst) $[w,\ e \vee c]$
3.	w	MP 2, 1
4.	$w \wedge (e \vee c) \to e \vee c$	(snd) $[w,\ e \vee c]$
5.	$e \vee c$	MP 4, 1
6.	$e \to \delta$	first lemma, by transitivity from line 3
7.	$c \to \delta$	second lemma, likewise
8.	$(e \to \delta) \to (c \to \delta) \to e \vee c \to \delta$	(either) $[e,\ c,\ \delta]$
9.	$(c \to \delta) \to e \vee c \to \delta$	MP 8, 6
10.	$e \vee c \to \delta$	MP 9, 7
11.	δ	MP 10, 5

Discharge $w \wedge (e \vee c)$, and the theorem is proved. Lines 6 and 7 stand for the lemmas, spliced in by transitivity.

2.5 · THE DEDUCTION THEOREM

Why not by hand

Every discharge is a use of the Deduction Theorem, and each one, carried out, roughly triples the proof it is applied to.

The lemmas of Example 2.24 become some twenty lines each, the main proof with both spliced in about sixty, and its final discharge triples that. From the schemas alone: well over a hundred lines, every one of them checkable and almost none of them findable. Nobody writes it.

The Deduction Theorem is what keeps this chapter's proofs short. In a compiler, the same theorem is what keeps your programs short.

You write the function, with its parameter; the compiler, running the construction with the programs left on the lines, writes the combinators. Chapter 3 makes this exact.

SUMMARY

Chapter 2

- A **proof** is a box of formulas, the hypotheses above the line, each line an axiom, a hypothesis, or modus ponens from earlier lines. Checking one is pattern matching: the clerk needs no understanding.
- A **typing is a proof** with the programs written beside the lines (Theorem 2.13): K and S are the two schemas, application is modus ponens, a variable in the context is a hypothesis. The compiler that refused Jones was refusing a bad proof.
- The rule the compiler needed for a parameter is the **Deduction Theorem**, $\Gamma \cup \{\varphi\} \vdash \psi$ iff $\Gamma \vdash \varphi \to \psi$, whose proof is the algorithm that compiles the parameter away.
- When the language grew to **and, or and not**, seven schemas were added, each the type of a function you know, and neither theorem changed.

How did the compiler know? *It checked a proof.*

LOOKING AHEAD

Chapter 3

We have been compiling definitions away because that is all the four rules could type. But the Deduction Theorem says the case for parameters is admissible: adding it as a rule would let nothing new be typed.

So add it, and let the parameter stay: the λ put back, and the logic whose proofs are programs.

Hughes's loose end waits longer. Her policy says *every* maintainer; her letters, and her types, speak only of Smith. *Every* is Chapter 4's.

CHAPTER 3

Łukasiewicz's complaint answered: a logic in which the clerk may suppose. And, or, not get rules of the same kind.

READING

Sources

- Open Logic Project, chapter 12, *Axiomatic derivations.* Its remark that the schemas were chosen for the Deduction Theorem is the one this chapter turns into a compiler.
- Curry and Feys (1958); Howard (1980); Wadler (2015). Where K, S meet (I), (II) in print; the modern form; the whole story. Turner (1979). The construction as a compiler.
- Herbrand (1930); Jaśkowski (1934). The theorem; the complaint behind Chapter 3.
- Abadi et al. (1993); Appel and Felten (1999); Necula (1997). Access control as logic; a request that carries its proof; a program that does.

Thank you.