Informatics 1

Functional Programming Lecture 18

Friday 21 November 2014

Logic, Proof and Programs

Don Sannella

University of Edinburgh

Tutorials

Last tutorials next week, usual time/place

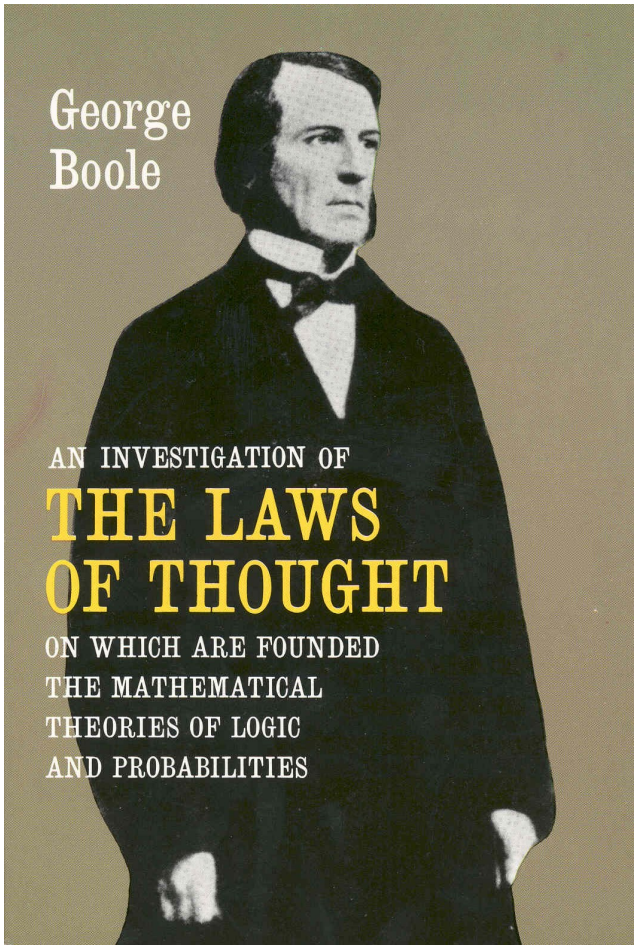
Revision tutorial next week:
Wednesday 2–3pm in AT 5.05

Revision tutorials after next week:
Wednesday 2–3pm in AT 5.05 on 3 Dec
Wednesday 2–3pm in AT 5.05 on 10 Dec

Part 0

Boolean algebra

George Boole (1815–1864)



Boole 1847: Mathematical analysis of logic

The primary canonical forms already determined for the expression of Propositions, are

All Xs are Ys,	$x(1-y)=0$,	....A.
No Xs are Ys,	$xy=0$,	....E.
Some Xs are Ys,	$v=xy$,	....I.
Some Xs are not Ys,	$v=x(1-y)$	....O.

On examining these, we perceive that E and I are symmetrical with respect to x and y, so that x being changed into y, and y into x, the equations remain unchanged. Hence E and I may be interpreted into

No Ys are Xs,
Some Ys are Xs,

respectively. Thus we have the known rule of the Logicians, that particular
26|27 affirmative and universal negative Propositions admit of simple conversion. |

Boole 1854: Laws of Thought

Proposition IV.

That axiom of metaphysicians which is termed the principle of contradiction, and which affirms that it is impossible for any being to possess a quality, and at the same time not to possess it, is a consequence of the fundamental law of thought, whose expression is $x^2 = x$.

Let us write this equation in the form

$$x - x^2 = 0,$$

whence we have

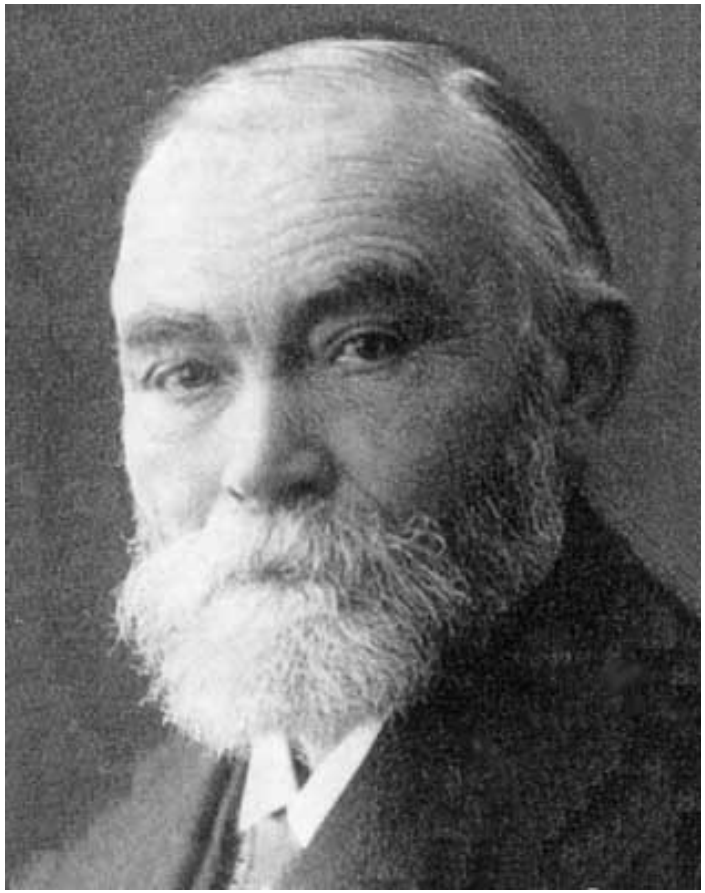
$$x\,(1 - x) = 0\,; \qquad (1)$$

both these transformations being justified by the axiomatic laws of combination and transposition (II. 13). Let us, for simplicity

Part 1

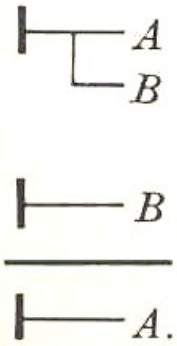
Frege's *Begriffsschrift*

Gottlob Frege (1848–1925)



Frege 1879 — *imp-elem a.k.a. modus ponens*

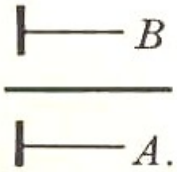
We could write this inference perhaps as follows:



This would become awkward if long expressions were to take the places of A and B, since each of them would have to be written twice. That is why I use the following

Frege 1879 — *imp-elem a.k.a. modus ponens*

We could write this inference perhaps as follows:

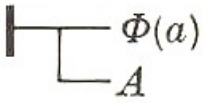


This would become awkward if long expressions were to take the places of A and B, since each of them would have to be written twice. That is why I use the following

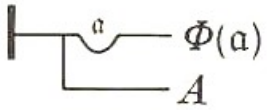
$$\frac{B \to A \qquad B}{A}$$

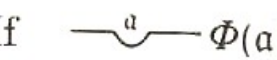
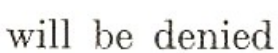
Frege 1879 — quantification

It is clear also that from



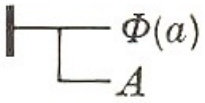
we can derive



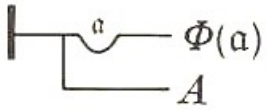
if A is an expression in which a does not occur and if a stands only in the argument places of $\Phi(a)$.[14] If  is denied, we must be able to specify a meaning for a such that $\Phi(a)$ will be denied. If, therefore,  were to be denied and

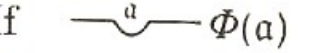
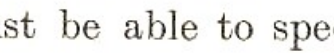
Frege 1879 — quantification

It is clear also that from



we can derive

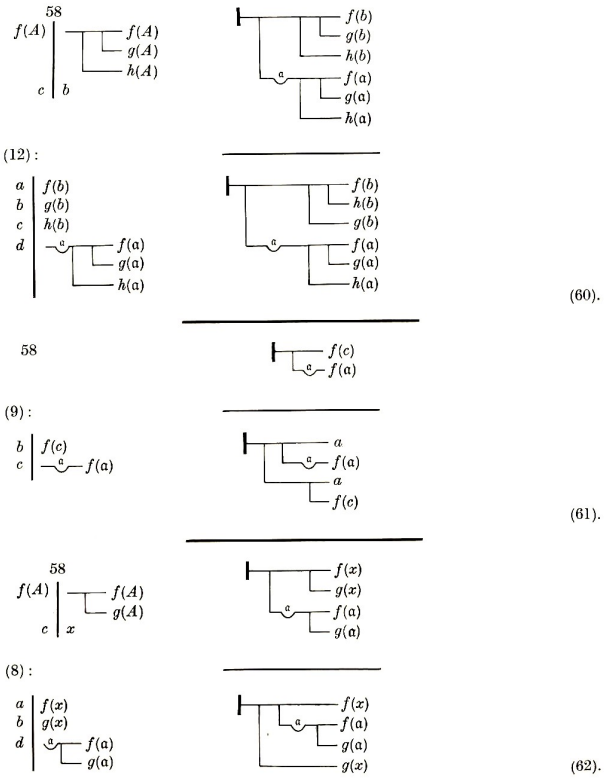


if A is an expression in which a does not occur and if a stands only in the argument places of $\Phi(a)$.[14] If  is denied, we must be able to specify a meaning for a such that $\Phi(a)$ will be denied. If, therefore,  were to be denied and

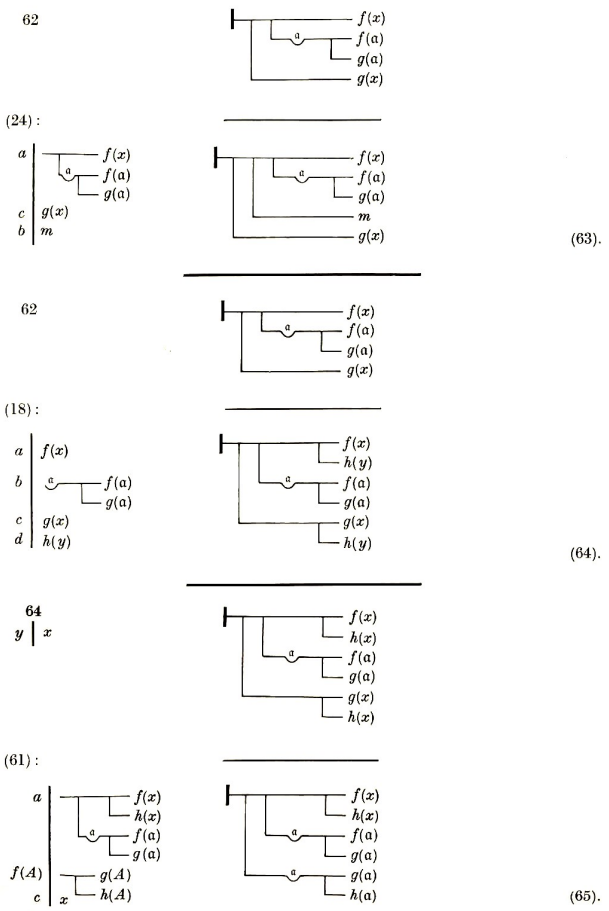
$$\frac{A \to \Phi(a)}{A \to \forall a.\, \Phi(a)}$$ a not free in A

Frege 1879

We see how this judgment replaces one mode of inference, namely, Felapton or Fesapo, between which we do not distinguish here since no subject has been singled out.



This judgment replaces the mode of inference Barbara when the minor premiss, $g(x)$, has a particular content.



Frege in modern notation

$$\frac{B \to A \qquad B}{A}$$

$$A \to (B \to A)$$

$$(C \to (B \to A)) \to ((C \to B) \to (C \to A))$$

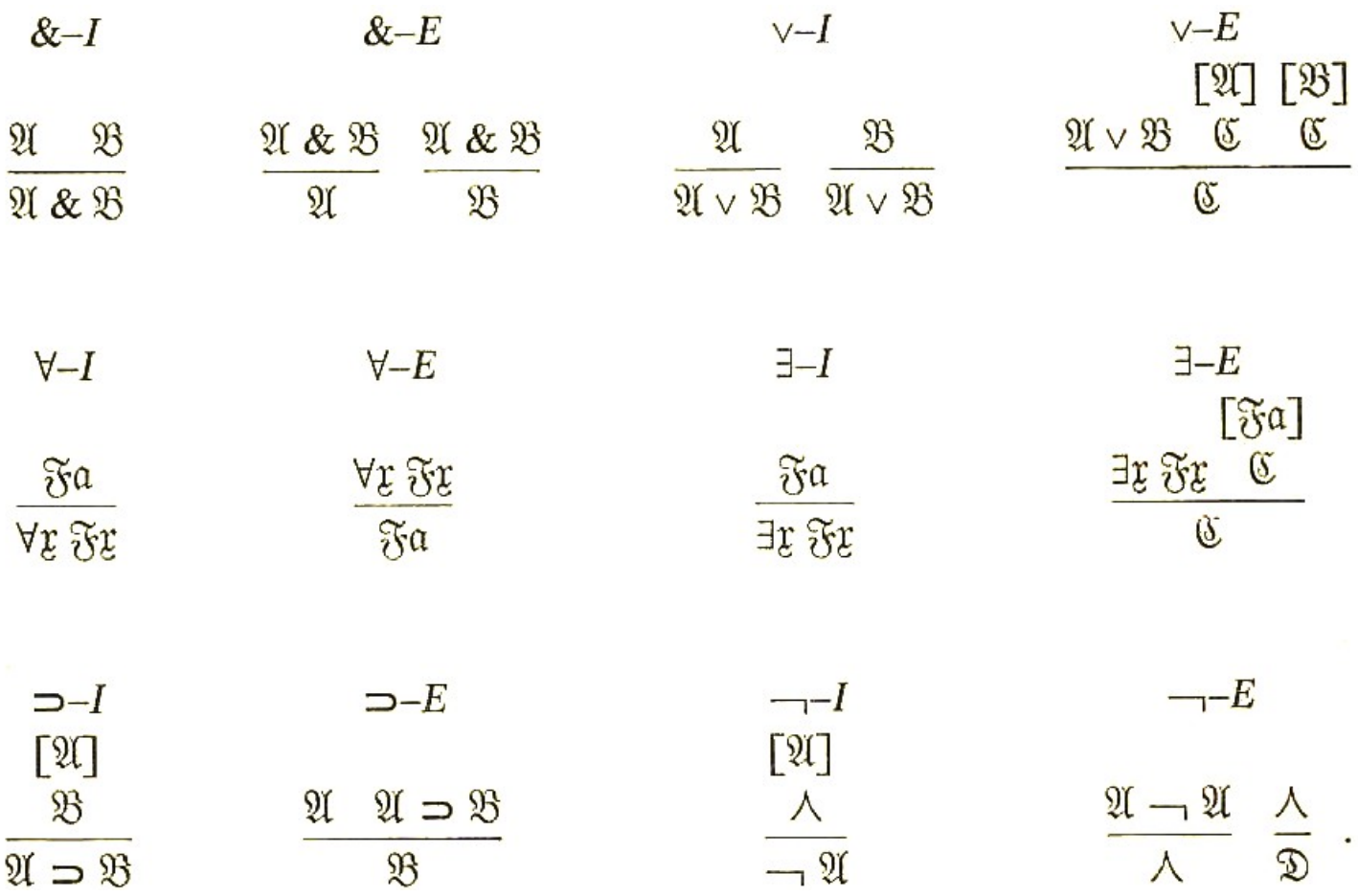
$$(C \to (B \to A)) \to (B \to (C \to A))$$

Part 2

Gentzen's Natural Deduction

Gerhard Gentzen (1909–1945)

Gentzen 1934: Natural Deduction



Gentzen 1934: Natural Deduction

$$\dfrac{\begin{array}{c}[A]^x\\ \vdots\\ B\end{array}}{A \to B}\ \to\text{-I}^x \qquad \dfrac{A \to B \qquad A}{B}\ \to\text{-E}$$

$$\dfrac{A \qquad B}{A \wedge B}\ \wedge\text{-I} \qquad \dfrac{A \wedge B}{A}\ \wedge\text{-E}_0 \qquad \dfrac{A \wedge B}{B}\ \wedge\text{-E}_1$$

A proof

$$\dfrac{\dfrac{\dfrac{[B \wedge A]^z}{A}\,\wedge\text{-E}_1 \qquad \dfrac{[B \wedge A]^z}{B}\,\wedge\text{-E}_0}{A \wedge B}\,\wedge\text{-I}}{(B \wedge A) \to (A \wedge B)}\,\to\text{-I}^z$$

Simplifying proofs

$$\dfrac{\dfrac{\dfrac{[A]^x}{\vdots}}{\dfrac{B}{A \to B}\,{\to}\text{-I}^x} \qquad \dfrac{\vdots}{A}}{B}\,{\to}\text{-E} \quad \Rightarrow \quad \dfrac{\vdots}{B}$$

$$\dfrac{\dfrac{\dfrac{\vdots}{A} \qquad \dfrac{\vdots}{B}}{A \wedge B}\,{\wedge}\text{-I}}{A}\,{\wedge}\text{-E}_0 \quad \Rightarrow \quad \dfrac{\vdots}{A}$$

Simplifying a proof

$$\dfrac{\dfrac{\dfrac{\dfrac{[B \wedge A]^z}{A}\,\wedge\text{-E}_1 \qquad \dfrac{[B \wedge A]^z}{B}\,\wedge\text{-E}_0}{A \wedge B}\,\wedge\text{-I}}{(B \wedge A) \to (A \wedge B)}\,\to\text{-I}^z \qquad \dfrac{[B]^y \quad [A]^x}{B \wedge A}\,\wedge\text{-I}}{A \wedge B}\,\to\text{-E}$$

Simplifying a proof

$$\dfrac{\dfrac{\dfrac{\dfrac{[B \wedge A]^z}{A}\ \wedge\text{-E}_1 \qquad \dfrac{[B \wedge A]^z}{B}\ \wedge\text{-E}_0}{A \wedge B}\ \wedge\text{-I}}{(B \wedge A) \to (A \wedge B)}\ \to\text{-I}^z \qquad \dfrac{[B]^y \quad [A]^x}{B \wedge A}\ \wedge\text{-I}}{A \wedge B}\ \to\text{-E}$$

$$\Downarrow$$

$$\dfrac{\dfrac{\dfrac{[B]^y \quad [A]^x}{B \wedge A}\ \wedge\text{-I}}{A}\ \wedge\text{-E}_1 \qquad \dfrac{\dfrac{[B]^y \quad [A]^x}{B \wedge A}\ \wedge\text{-I}}{B}\ \wedge\text{-E}_0}{A \wedge B}\ \wedge\text{-I}$$

Simplifying a proof

$$\dfrac{\dfrac{\dfrac{\dfrac{[B \wedge A]^z}{A}\;\wedge\text{-E}_1 \qquad \dfrac{[B \wedge A]^z}{B}\;\wedge\text{-E}_0}{A \wedge B}\;\wedge\text{-I}}{(B \wedge A) \rightarrow (A \wedge B)}\;\rightarrow\text{-I}^z \qquad \dfrac{[B]^y \quad [A]^x}{B \wedge A}\;\wedge\text{-I}}{A \wedge B}\;\rightarrow\text{-E}$$

$$\Downarrow$$

$$\dfrac{\dfrac{\dfrac{[B]^y \quad [A]^x}{B \wedge A}\;\wedge\text{-I}}{A}\;\wedge\text{-E}_1 \qquad \dfrac{\dfrac{[B]^y \quad [A]^x}{B \wedge A}\;\wedge\text{-I}}{B}\;\wedge\text{-E}_0}{A \wedge B}\;\wedge\text{-I}$$

$$\Downarrow$$

$$\dfrac{[A]^x \quad [B]^y}{A \wedge B}\;\wedge\text{-I}$$

Part 3

Church's Lambda Calculus

Alonzo Church (1903–1995)

Church 1932: Lambda Calculus

An occurrence of a variable **x** in a given formula is called an occurrence of **x** as a *bound variable* in the given formula if it is an occurrence of **x** in a part of the formula of the form λ**x**[**M**]; that is, if there is a formula **M** such that λ**x**[**M**] occurs in the given formula and the occurrence of **x** in question is an occurrence in λ**x**[**M**]. All other occurrences of a variable in a formula are called occurrences as a *free variable*.

A formula is said to be *well-formed* if it is a variable, or if it is one

Reduction rules

$$(\lambda x.\, u)\, t \quad \Rightarrow \quad u[t/x]$$

$$\mathrm{fst}\ (t, u) \quad \Rightarrow \quad t$$

$$\mathrm{snd}\ (t, u) \quad \Rightarrow \quad u$$

Simplifying a term

$$(\lambda z.\,(\mathrm{snd}\ z, \mathrm{fst}\ z))\ (y, x)$$

Simplifying a term

$$(\lambda z.\,(\text{snd}\ z, \text{fst}\ z))\ (y, x)$$

$$\Downarrow$$

$$(\text{snd}\ (y, x), \text{fst}\ (y, x))$$

Simplifying a term

$$(\lambda z.\,(\text{snd}\ z, \text{fst}\ z))\ (y, x)$$

$$\Downarrow$$

$$(\text{snd}\ (y, x), \text{fst}\ (y, x))$$

$$\Downarrow$$

$$(x, y)$$

Church 1940: Typed Lambda Calculus

$$\frac{\begin{array}{c}[x : A]^x \\ \vdots \\ u : B\end{array}}{\lambda x.\, u : A \to B}\ \to\text{-I}^x \qquad \frac{s : A \to B \qquad t : A}{s\ t : B}\ \to\text{-E}$$

$$\frac{t : A \qquad u : B}{(t, u) : A \wedge B}\ \wedge\text{-I} \qquad \frac{s : A \wedge B}{\text{fst}\ s : A}\ \wedge\text{-E}_0 \qquad \frac{s : A \wedge B}{\text{snd}\ s : B}\ \wedge\text{-E}_1$$

A program

$$\dfrac{\dfrac{\dfrac{[z : B \wedge A]^z}{\text{snd } z : A}\,\wedge\text{-E}_1 \qquad \dfrac{[z : B \wedge A]^z}{\text{fst } z : B}\,\wedge\text{-E}_0}{(\text{snd } z, \text{fst } z) : A \wedge B}\,\wedge\text{-I}}{\lambda z.\,(\text{snd } z, \text{fst } z) : (B \wedge A) \to (A \wedge B)}\,\to\text{-I}^z$$

Simplifying programs

$$\dfrac{\dfrac{\begin{array}{c}[x : A]^x \\ \vdots \\ u : B\end{array}}{\lambda x.\, u : A \to B}\ {\to}\text{-I}^x \qquad \begin{array}{c}\vdots \\ t : A\end{array}}{(\lambda x.\, u)\ t : B}\ {\to}\text{-E} \quad \Rightarrow \quad \begin{array}{c}\vdots \\ t : A \\ \vdots \\ u[t/x] : B\end{array}$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots \\ t : A\end{array} \qquad \begin{array}{c}\vdots \\ u : B\end{array}}{(t, u) : A \wedge B}\ {\wedge}\text{-I}}{\text{fst}\ (t, u) : A}\ {\wedge}\text{-E}_0 \quad \Rightarrow \quad \begin{array}{c}\vdots \\ t : A\end{array}$$

Simplifying a program

$$\dfrac{\dfrac{\dfrac{\dfrac{[z : B \wedge A]^z}{\text{snd } z : A}\wedge\text{-E}_1 \qquad \dfrac{[z : B \wedge A]^z}{\text{fst } z : B}\wedge\text{-E}_0}{(\text{snd } z, \text{fst } z) : A \wedge B}\wedge\text{-I}}{\lambda z.\,(\text{snd } z, \text{fst } z) : (B \wedge A) \to (A \wedge B)}\to\text{-I}^z \qquad \dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{(\lambda z.\,(\text{snd } z, \text{fst } z))\,(y, x) : A \wedge B}\to\text{-E}$$

Simplifying a program

$$\dfrac{\dfrac{\dfrac{\dfrac{[z : B \wedge A]^z}{\text{snd } z : A}\wedge\text{-E}_1 \qquad \dfrac{[z : B \wedge A]^z}{\text{fst } z : B}\wedge\text{-E}_0}{(\text{snd } z, \text{fst } z) : A \wedge B}\wedge\text{-I}}{\lambda z.\,(\text{snd } z, \text{fst } z) : (B \wedge A) \to (A \wedge B)}\to\text{-I}^z \qquad \dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{(\lambda z.\,(\text{snd } z, \text{fst } z))\,(y, x) : A \wedge B}\to\text{-E}$$

$$\Downarrow$$

$$\dfrac{\dfrac{\dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{\text{snd } (y, x) : A}\wedge\text{-E}_1 \qquad \dfrac{\dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{\text{fst } (y, x) : B}\wedge\text{-E}_0}{(\text{snd } (y, x), \text{fst } (y, x)) : A \wedge B}\wedge\text{-I}$$

Simplifying a program

$$\dfrac{\dfrac{\dfrac{\dfrac{[z : B \wedge A]^z}{\text{snd } z : A}\wedge\text{-E}_1 \qquad \dfrac{[z : B \wedge A]^z}{\text{fst } z : B}\wedge\text{-E}_0}{(\text{snd } z, \text{fst } z) : A \wedge B}\wedge\text{-I}}{\lambda z.\,(\text{snd } z, \text{fst } z) : (B \wedge A) \to (A \wedge B)}\to\text{-I}^z \qquad \dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{(\lambda z.\,(\text{snd } z, \text{fst } z))\,(y, x) : A \wedge B}\to\text{-E}$$

$$\Downarrow$$

$$\dfrac{\dfrac{\dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{\text{snd } (y, x) : A}\wedge\text{-E}_1 \qquad \dfrac{\dfrac{[y : B]^y \quad [x : A]^x}{(y, x) : B \wedge A}\wedge\text{-I}}{\text{fst } (y, x) : B}\wedge\text{-E}_0}{(\text{snd } (y, x), \text{fst } (y, x)) : A \wedge B}\wedge\text{-I}$$

$$\Downarrow$$

$$\dfrac{[x : A]^x \quad [y : B]^y}{(x, y) : A \wedge B}\wedge\text{-I}$$

Part 4

The Curry-Howard Isomorphism

Haskell Curry (1900–1982) / William Howard (1926–)

Howard 1980

THE FORMULAE-AS-TYPES NOTION OF CONSTRUCTION

W. A. Howard

Department of Mathematics, University of Illinois at Chicago Circle, Chicago, Illinois 60680, U.S.A.

Dedicated to H. B. Curry on the occasion of his 80th birthday.

The following consists of notes which were privately circulated in 1969. Since they have been referred to a few times in the literature, it seems worth while to publish them. They have been rearranged for easier reading, and some inessential corrections have been made.

Howard 1980

1. Formulation of the sequent calculus

Let $P(\supset)$ denote positive implicational propositional logic. The prime formulae of $P(\supset)$ are propositional variables. If α and β are formulae, so is $\alpha \supset \beta$. A *sequent* has the form $\Gamma \to \beta$, where Γ is a (possibly empty) finite sequence of formulae and β is a formula. The axioms and rules of inference of $P(\supset)$ are as follows.

(1.1) Axioms: all sequents of the form $\alpha \to \alpha$

(1.2) $$\frac{\Gamma, \alpha \to \beta}{\Gamma \to \alpha \supset \beta}$$

(1.3) $$\frac{\Gamma \to \alpha \qquad \Delta \to \alpha \supset \beta}{\Gamma, \Delta \to \beta}$$

(1.4) Thinning, permutation and contraction rules

Howard 1980

2. Type symbols, terms and constructions

By a type symbol is meant a formula of $P(\supset)$. We will consider a λ-formalism in which each term has a type symbol α as a superscript (which we may not always write); the term is said to be of type α. The rules of term formation are as follows.

(2.1) Variables X^{α}, Y^{β},... are terms

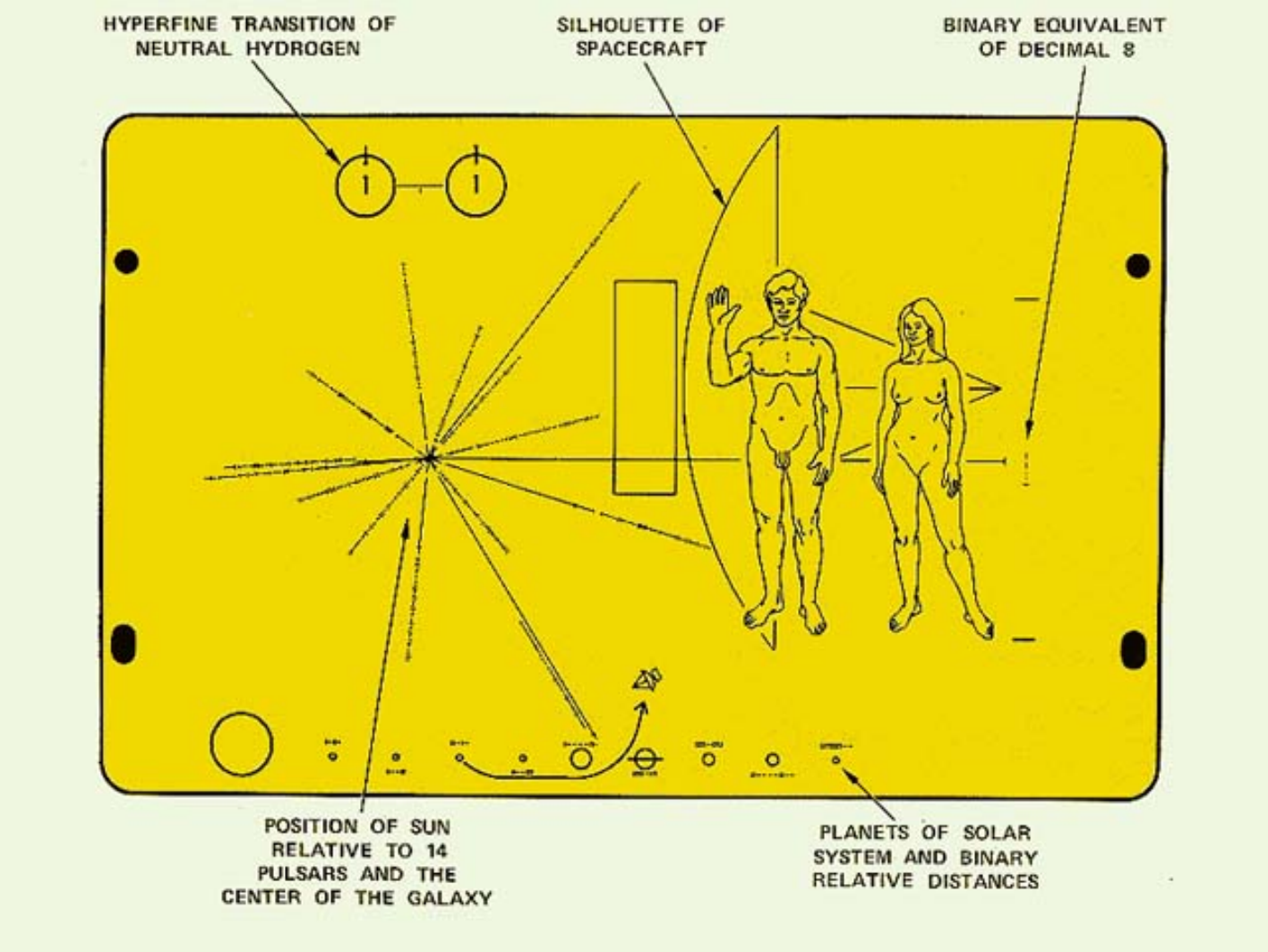
(2.2) λ-abstraction: from F^{β} get $(\lambda X^{\alpha}.F^{\beta})^{\alpha \supset \beta}$.

(2.3) Application: from $G^{\alpha \supset \beta}$ and H^{α} get $(G^{\alpha \supset \beta} H^{\alpha})^{\beta}$.

Part 5

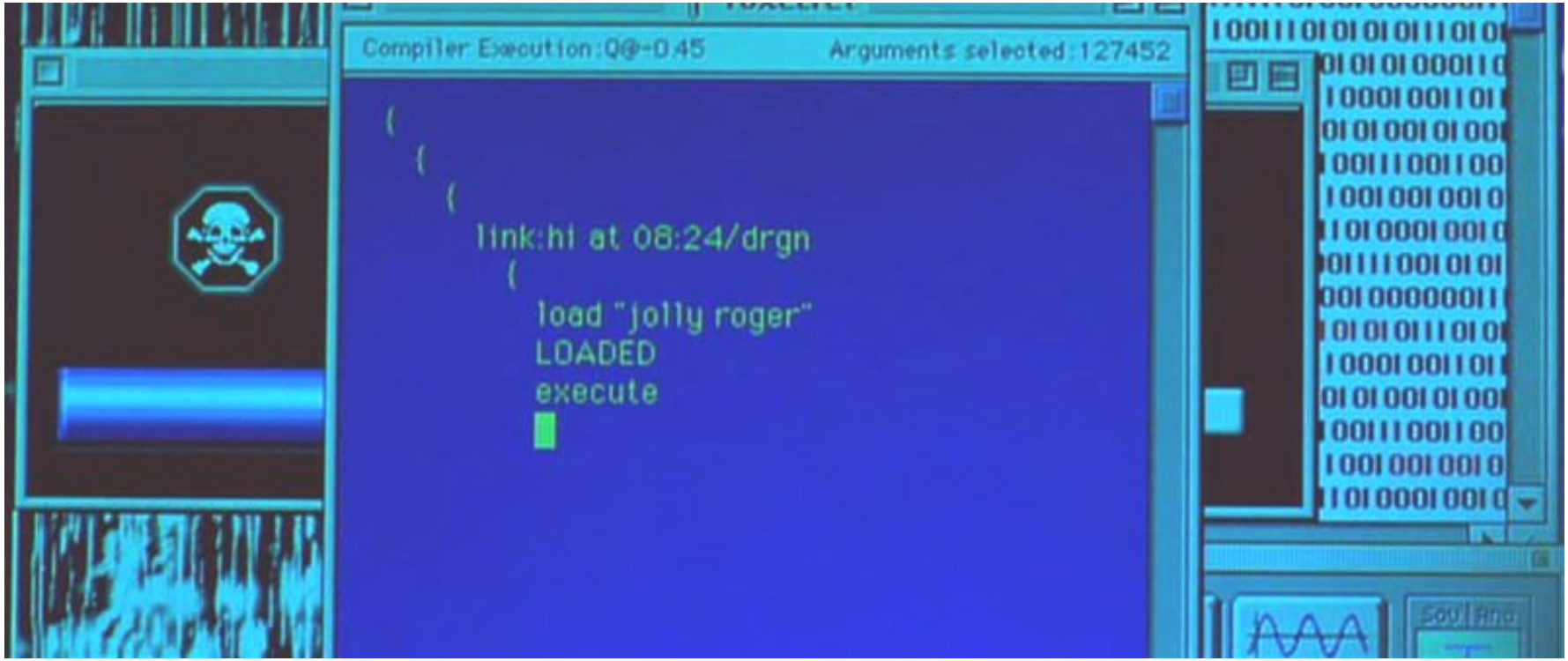
Aliens

How to talk to aliens



Independence Day

A universal programming language?



Special thanks to:

Karoliina Lehtinen, the tutors and demonstrators,
Alistair Hill, Claire Edminson and Paul Anderson

for making the course run

Phil Wadler, Phil Scott, the Haskell community

for the content

You

for listening, for questions, and for the future