# *Propositions as Types*

Philip Wadler
University of Edinburgh

# Part I

# *Computability*

# *Euclid (325–265 BCE)*
# *Al Khwarizmi (780–850 CE)*

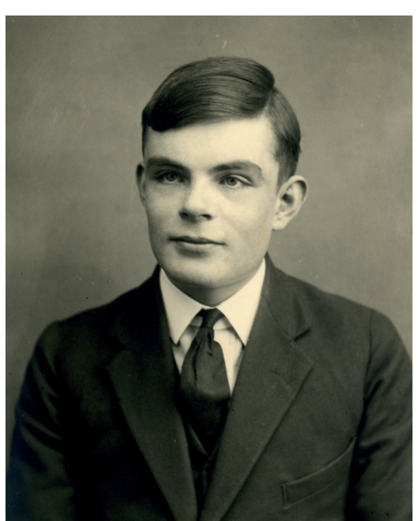

# *Algorithm, formalised*

Alonzo Church: *Lambda calculus*

An unsolvable problem of elementary number theory, *Bulletin the American Mathematical Society*, May 1935

Kurt Gödel: *Recursive functions*

Stephen Kleene, General recursive functions of natural numbers, *Bulletin the American Mathematical Society*, July 1935



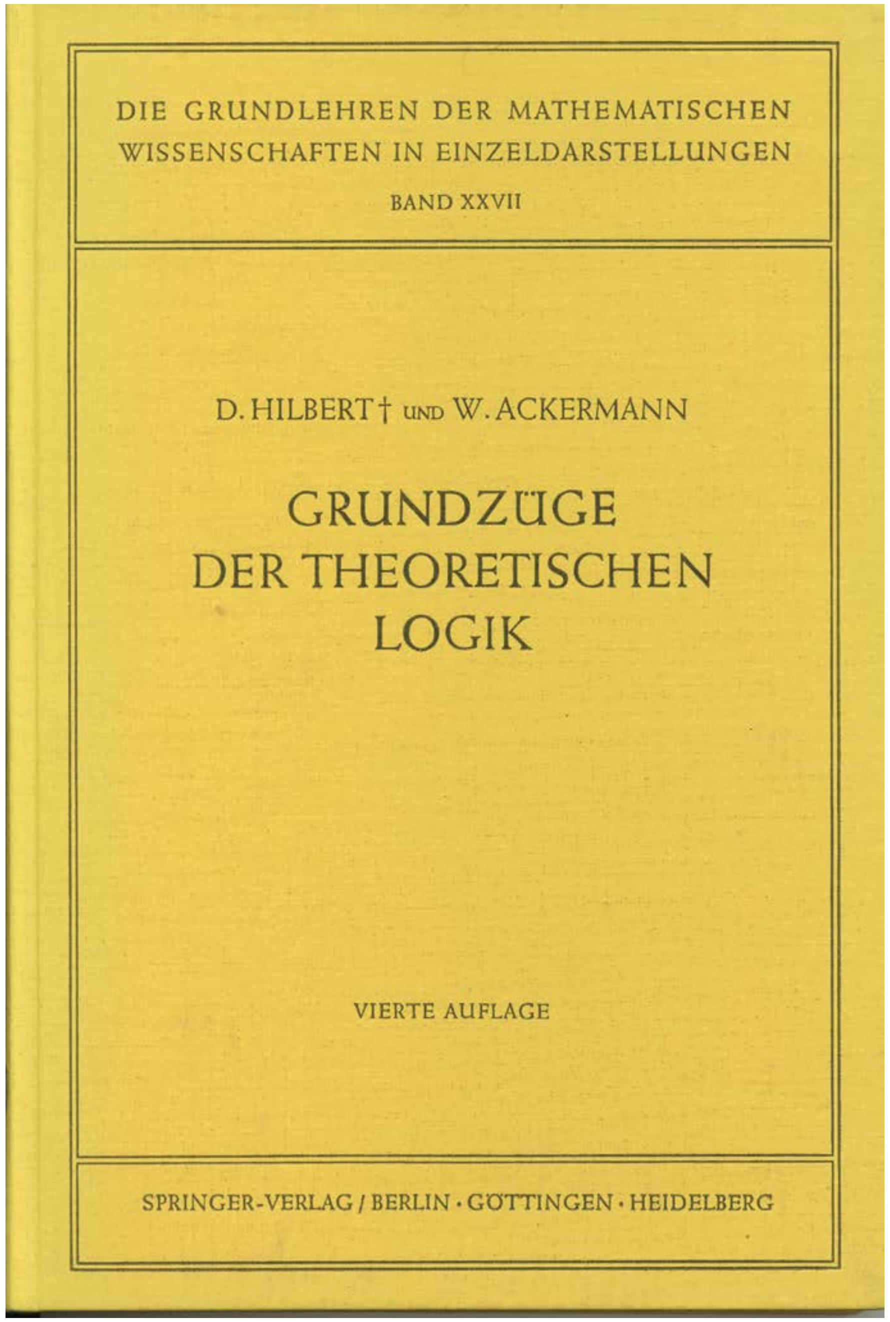
Alan M. Turing: *Turing machines*

On computable numbers, with an application to the *Entscheidungsproblem*, *Proceedings of the London Mathematical Society*, received 25 May 1936


HARRODS GREATEST SALE
157
Oxford Circus

# *David Hilbert (1862-1943)*

# *David Hilbert (1928)*
# *Entscheidungsproblem*



# *Kurt Gödel (1906-1978)*

# *Kurt Gödel (1930)*
# *Incompleteness*

42. $Ax(x) \equiv Z\text{-}Ax(x) \vee A\text{-}Ax(x) \vee L_1\text{-}Ax(x) \vee L_2\text{-}Ax(x) \vee R\text{-}Ax(x) \vee M\text{-}Ax(x)$,
$x$ is an AXIOM.

43. $Fl(x, y, z) \equiv y = z \text{ Imp } x \vee (Ev)[v \leqq x \,\&\, \text{Var}(v) \,\&\, x = v \text{ Gen } y]$,
$x$ is an IMMEDIATE CONSEQUENCE of $y$ and $z$.

44. $Bw(x) \equiv (n)\{0 < n \leqq l(x) \rightarrow Ax(n\ Gl\ x) \vee (Ep, q)[0 < p, q < n \,\&$
$Fl(n\ Gl\ x, p\ Gl\ x, q\ Gl\ x)]\} \,\&\, l(x) > 0$,
$x$ is a PROOF ARRAY (a finite sequence of FORMULAS, each of which is either an AXIOM or an IMMEDIATE CONSEQUENCE of two of the preceding FORMULAS.

45. $x\ B\ y \equiv Bw(x) \,\&\, [l(x)]\ Gl\ x = y$,
$x$ is a PROOF of the FORMULA $y$.

46. $\text{Bew}(x) \equiv (Ey)y\ B\ x$,
$x$ is a PROVABLE FORMULA. ($\text{Bew}(x)$ is the only one of the notions 1–46 of which we cannot assert that it is recursive.)

“This statement is not provable”


# *Alonzo Church (1903–1995)*

# *Alonzo Church (1936)*
# *Lambda Calculus*

**AN UNSOLVABLE PROBLEM OF ELEMENTARY NUMBER THEORY.**[1]

By ALONZO CHURCH.

---

The purpose of the present paper is to propose a definition of effective calculability[3] which is thought to correspond satisfactorily to the somewhat vague intuitive notion in terms of which problems of this class are often stated, and to show, by means of an example, that not every problem of this class is solvable.

We introduce at once the following infinite list of abbreviations,

$$1 \rightarrow \lambda ab \cdot a(b),$$
$$2 \rightarrow \lambda ab \cdot a(a(b)),$$
$$3 \rightarrow \lambda ab \cdot a(a(a(b))),$$

and so on, each positive integer in Arabic notation standing for a formula of the form $\lambda ab \cdot a(a(\cdots a(b) \cdots))$.

# *Alonzo Church (1932)*
# *λ-calculus*

$$
\begin{array}{rcl}
L, M, N & ::= & x \\
& | & (\lambda x.\, N) \\
& | & (L\ M)
\end{array}
$$

# *Kurt Gödel (1906-1978)*

# *Kurt Gödel (1936)*
# *Recursive Functions*

**General recursive functions of natural numbers[1]).**

Von

S. C. Kleene in Madison (Wis., U.S.A.).

The substitution

$$\varphi(x_1, \ldots, x_n) = \theta(\chi_1(x_1, \ldots, x_n), \ldots, \chi_m(x_1, \ldots, x_n)), \tag{1}$$

and the ordinary recursion with respect to one variable

$$\begin{aligned} \varphi(0, x_2, \ldots, x_n) &= \psi(x_2, \ldots, x_n) \\ \varphi(y+1, x_2, \ldots, x_n) &= \chi(y, \varphi(y, x_2, \ldots, x_n), x_2, \ldots, x_n), \end{aligned} \tag{2}$$

where $\theta, \chi_1, \ldots, \chi_m, \psi, \chi$ are given functions of natural numbers, are examples of the definition of a function $\varphi$ by equations which provide a step by step process for computing the value $\varphi(k_1, \ldots, k_n)$ for any given set $k_1, \ldots, k_n$ of natural numbers. It is known that there are other definitions of this sort, e. g. certain recursions with respect to two or more variables simultaneously, which cannot be reduced to a succession of substitutions and ordinary recursions[2]). Hence, a characterization of the notion of recursive definition in general, which would include all these cases, is desirable. A definition of general recursive function of natural numbers was suggested by Herbrand to Gödel, and was used by Gödel with an important modification in a series of lectures at Princeton in 1934. In this paper we offer several observations on general recursive functions, using essentially Gödel's form of the definition.

# *Alan Turing (1912-1954)*

# *Alan Turing (1936)*
# *Turing Machine*

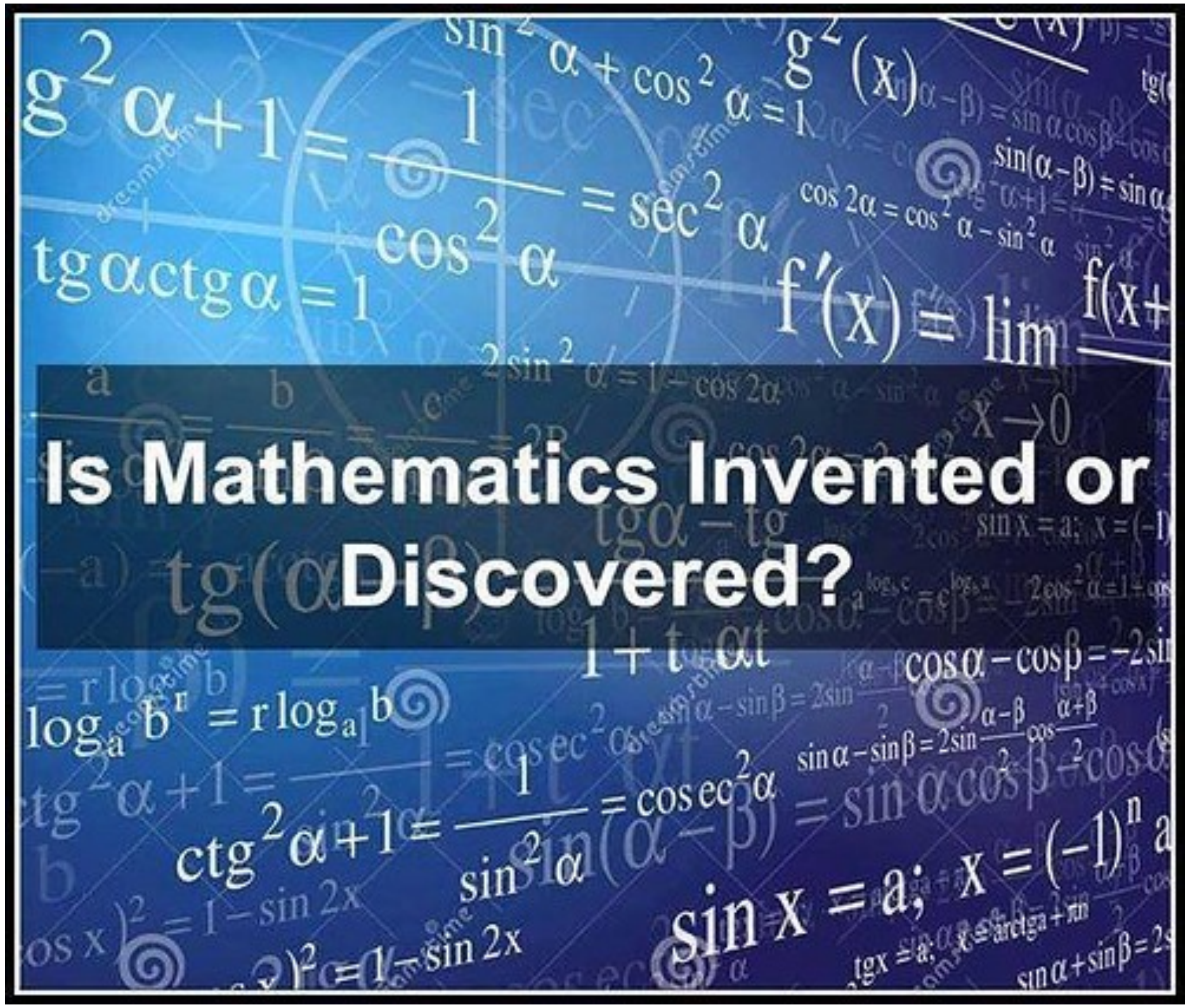
ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means.

In §§ 9, 10 I give some arguments with the intention of showing that the computable numbers include all numbers which could naturally be regarded as computable. In particular, I show that certain large classes of numbers are computable. They include, for instance, the real parts of all algebraic numbers, the real parts of the zeros of the Bessel functions, the numbers $\pi$, $e$, etc. The computable numbers do not, however, include all definable numbers, and an example is given of a definable number which is not computable.


Is Mathematics Invented or Discovered?

Kurt Gödel, 28

David Hilbert, 68

Alan Turing, 23

Alonzo Church, 33

Kurt Gödel, 30

Part II

# *Propositions as Types*

# *Gerhard Gentzen (1909-1945)*

# Gerhard Gentzen (1935)
# Natural Deduction

&–*I*

$$\frac{\mathfrak{A} \qquad \mathfrak{B}}{\mathfrak{A} \,\&\, \mathfrak{B}}$$

&–*E*

$$\frac{\mathfrak{A} \,\&\, \mathfrak{B}}{\mathfrak{A}} \qquad \frac{\mathfrak{A} \,\&\, \mathfrak{B}}{\mathfrak{B}}$$

∨–*I*

$$\frac{\mathfrak{A}}{\mathfrak{A} \lor \mathfrak{B}} \qquad \frac{\mathfrak{B}}{\mathfrak{A} \lor \mathfrak{B}}$$

∨–*E*

$$\frac{\mathfrak{A} \lor \mathfrak{B} \quad \begin{array}{c}[\mathfrak{A}]\\ \mathfrak{C}\end{array} \quad \begin{array}{c}[\mathfrak{B}]\\ \mathfrak{C}\end{array}}{\mathfrak{C}}$$

∀–*I*

$$\frac{\mathfrak{F}\mathfrak{a}}{\forall \mathfrak{x}\, \mathfrak{F}\mathfrak{x}}$$

∀–*E*

$$\frac{\forall \mathfrak{x}\, \mathfrak{F}\mathfrak{x}}{\mathfrak{F}\mathfrak{a}}$$

∃–*I*

$$\frac{\mathfrak{F}\mathfrak{a}}{\exists \mathfrak{x}\, \mathfrak{F}\mathfrak{x}}$$

∃–*E*

$$\frac{\exists \mathfrak{x}\, \mathfrak{F}\mathfrak{x} \quad \begin{array}{c}[\mathfrak{F}\mathfrak{a}]\\ \mathfrak{C}\end{array}}{\mathfrak{C}}$$

⊃–*I*

$$\frac{\begin{array}{c}[\mathfrak{A}]\\ \mathfrak{B}\end{array}}{\mathfrak{A} \supset \mathfrak{B}}$$

⊃–*E*

$$\frac{\mathfrak{A} \qquad \mathfrak{A} \supset \mathfrak{B}}{\mathfrak{B}}$$

¬–*I*

$$\frac{\begin{array}{c}[\mathfrak{A}]\\ \curlywedge\end{array}}{\neg\, \mathfrak{A}}$$

¬–*E*

$$\frac{\mathfrak{A} \quad \neg\, \mathfrak{A}}{\curlywedge} \qquad \frac{\curlywedge}{\mathfrak{D}} \;.$$

# *Gerhard Genesen (1935)*
# *Natural Deduction*

$$\dfrac{\begin{matrix}[A]^x \\ \vdots \\ B\end{matrix}}{A \supset B}\;\supset\text{-I}^x \qquad \dfrac{A \supset B \qquad A}{B}\;\supset\text{-E}$$

$$\dfrac{A \qquad B}{A \,\&\, B}\;\&\text{-I} \qquad \dfrac{A \,\&\, B}{A}\;\&\text{-E}_1 \qquad \dfrac{A \,\&\, B}{B}\;\&\text{-E}_2$$

# *A proof*

$$\dfrac{\dfrac{[B \mathbin{\&} A]^z}{A}\,\&\text{-E}_2 \qquad \dfrac{[B \mathbin{\&} A]^z}{B}\,\&\text{-E}_1}{\dfrac{A \mathbin{\&} B}{(B \mathbin{\&} A) \supset (A \mathbin{\&} B)}\,\supset\text{-I}^z}\,\&\text{-I}$$

# *Simplifying proofs*

$$\dfrac{\dfrac{\begin{matrix}[A]^x\\ \vdots\\ B\end{matrix}}{A \supset B}\ \supset\text{-}\mathbf{I}^{x} \qquad \begin{matrix}\vdots\\ A\end{matrix}}{B}\ \supset\text{-}\mathbf{E}$$

# *Simplifying proofs*

$$\dfrac{\dfrac{\begin{matrix}\vdots \\ A\end{matrix} \qquad \begin{matrix}\vdots \\ B\end{matrix}}{A \& B}\ \&\text{-I}}{A}\ \&\text{-E}_1$$

# *Simplifying a proof*

$$\cfrac{\cfrac{\cfrac{\cfrac{[B\&A]^z}{A}\,\&\text{-E}_2 \qquad \cfrac{[B\&A]^z}{B}\,\&\text{-E}_1}{A\&B}\,\&\text{-I}}{(B\&A)\supset(A\&B)}\,\supset\text{-I}^z \qquad \cfrac{\begin{matrix}\vdots \\ B\end{matrix} \qquad \begin{matrix}\vdots \\ A\end{matrix}}{B\&A}\,\&\text{-I}}{A\&B}\,\supset\text{-E}$$

# *Simplifying a proof*

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ B\end{array} \qquad \begin{array}{c}\vdots\\ A\end{array}}{B\&A}\,\&\text{-I}}{A}\,\&\text{-E}_2 \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\\ B\end{array} \qquad \begin{array}{c}\vdots\\ A\end{array}}{B\&A}\,\&\text{-I}}{B}\,\&\text{-E}_1}{A\&B}\,\&\text{-I}$$

# *Alonzo Church (1903–1995)*

# *Alonzo Church (1932)*
# *Typed λ-calculus*

$$\dfrac{\begin{matrix}[x : A]^x \\ \vdots \\ N : B\end{matrix}}{\lambda x.\, N : A \supset B}\ \supset\text{-I}^x \qquad \dfrac{L : A \supset B \qquad M : A}{L\, M : B}\ \supset\text{-E}$$

$$\dfrac{M : A \qquad N : B}{(M,\, N) : A \mathbin{\&} B}\ \&\text{-I} \qquad \dfrac{L : A \mathbin{\&} B}{\pi_1 L : A}\ \&\text{-E}_1 \qquad \dfrac{L : A \mathbin{\&} B}{\pi_2 L : B}\ \&\text{-E}_2$$

# *A program*

$$\dfrac{\dfrac{[z : B \mathbin{\&} A]^z}{\pi_2 z : A}\ \&\text{-E}_2 \qquad \dfrac{[z : B \mathbin{\&} A]^z}{\pi_1 z : B}\ \&\text{-E}_1}{\dfrac{(\pi_2 z,\ \pi_1 z) : A \mathbin{\&} B}{\lambda z.\,(\pi_2 z,\ \pi_1 z) : (B \mathbin{\&} A) \supset (A \mathbin{\&} B)}\ \supset\text{-I}^z}\ \&\text{-I}$$

# *Evaluating programs*

$$\dfrac{\dfrac{\begin{array}{c}[z : A]^z \\ \vdots \\ N : B\end{array}}{\lambda z.N : A \supset B}\supset\text{-}\mathbf{I}^z \qquad \begin{array}{c}\vdots \\ M : A\end{array}}{(\lambda z.N)M : B}\supset\text{-}\mathbf{E}$$

# *Evaluating programs*

$$\dfrac{\dfrac{\begin{array}{c}\vdots \\ M : A\end{array} \qquad \begin{array}{c}\vdots \\ N : B\end{array}}{(M, N) : A \& B}\ \&\text{-I}}{\pi_1(M, N) : A}\ \&\text{-E}_1$$

# *Alan Turing (1942)*

AN EARLY PROOF OF NORMALIZATION
BY A.M. TURING

R.O. Gandy

*Mathematical Institute, 24-29 St. Giles,*
*Oxford OX1 3LB, UK*

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

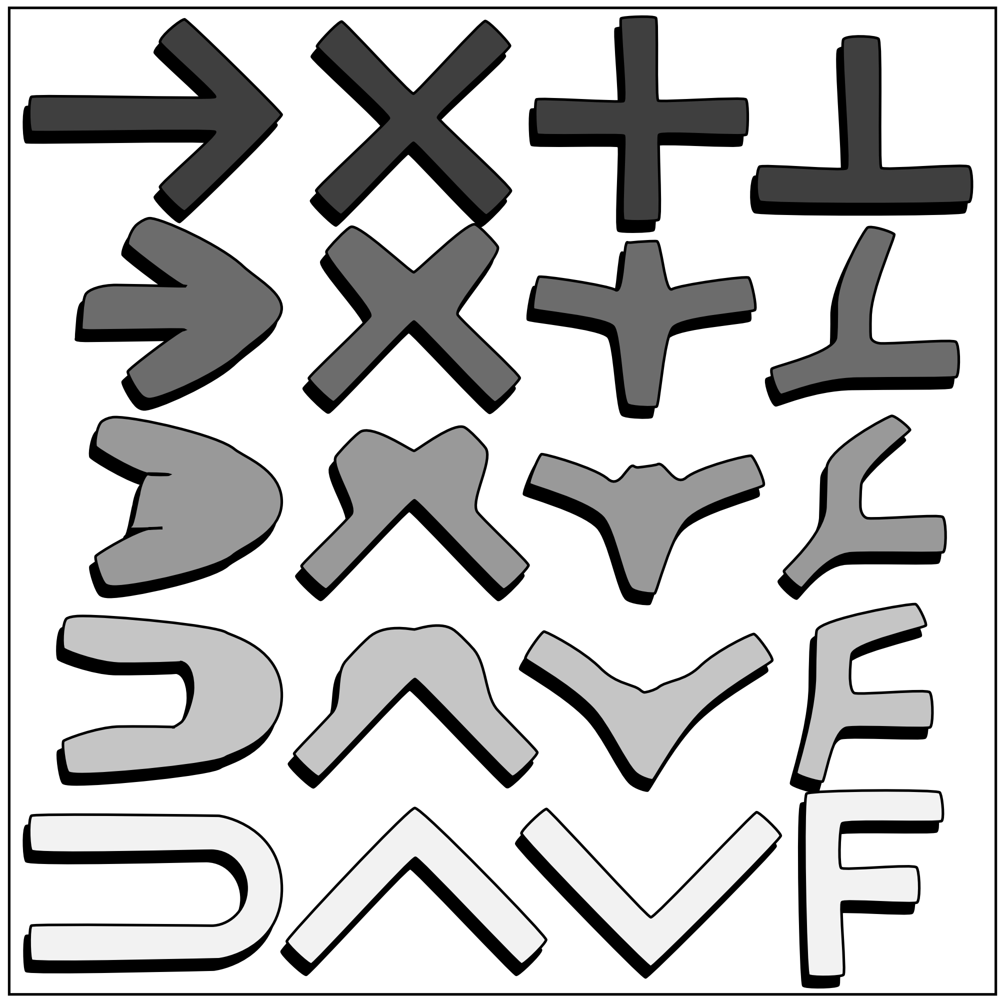
In the extract printed below, Turing shows that every formula of Church's simple type theory has a normal form. The extract is the first page of an unpublished (and incomplete) typescript entitled 'Some theorems about Church's system'. (Turing left his manuscripts to me; they are deposited in the library of King's College, Cambridge). An account of this system was published by Church in 'A formulation of the simple theory of types' (J. Symbolic Logic 5 (1940), pp. 56-68).

# *Evaluating a program*

$$
\dfrac{\dfrac{\dfrac{\dfrac{z : B\&A}{\pi_2 z : A}\ \&\text{-E}_2 \quad \dfrac{z : B\&A}{\pi_1 z : B}\ \&\text{-E}_1}{(\pi_2 z, \pi_1 z) : A\&B}\ \&\text{-I}}{\lambda z.(\pi_2 z, \pi_1 z) : (B\&A) \supset (A\&B)}\ \supset\text{-I}^z \quad \dfrac{\begin{array}{c}\vdots\\ N : B\end{array} \quad \begin{array}{c}\vdots\\ M : A\end{array}}{(N, M) : B\&A}\ \&\text{-I}}{(\lambda z.(\pi_2 z, \pi_1 z))(N, M) : A\&B}\ \supset\text{-E}
$$

# *Evaluating a program*

$$
\dfrac{
\dfrac{\dfrac{\dfrac{\vdots}{N : B} \qquad \dfrac{\vdots}{M : A}}{(N, M) : B \& A}\ \&\text{-I}}{\pi_2(N, M) : A}\ \&\text{-E}_2
\qquad
\dfrac{\dfrac{\dfrac{\vdots}{N : B} \qquad \dfrac{\vdots}{M : A}}{(N, M) : B \& A}\ \&\text{-I}}{\pi_1(N, M) : B}\ \&\text{-E}_1
}{(\pi_2(N, M), \pi_1(N, M)) : A \& B}\ \&\text{-I}
$$



LC'90

*The Curry-Howard homeomorphism*

# *Haskell Curry (1900-1982)*
# *William Howard (1926-)*


# *William 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.

# *Curry-Howard correspondence*

| | | |
|---|---|---|
| propositions | as | types |
| proofs | as | programs |
| normalisation of proofs | as | evaluation of programs |

# *Curry-Howard correspondence*

| | | |
|---|---|---|
| Natural Deduction<br>Gentzen (1935) | ⟷ | Typed Lambda Calculus<br>Church (1940) |
| Type Schemes<br>Hindley (1969) | ⟷ | ML Type System<br>Milner (1975) |
| System F<br>Girard (1972) | ⟷ | Polymorphic Lambda Calculus<br>Reynolds (1974) |
| Modal Logic<br>Lewis (1910) | ⟷ | Monads (state, exceptions)<br>Kleisli (1965), Moggi (1987) |
| Classical-Intuitionistic Embedding<br>Gödel (1933) | ⟷ | Continuation Passing Style<br>Reynolds (1972) |
| Linear Logic<br>Girard (1987) | ⟷ | Session Types<br>Honda (1993) |

# *Functional Languages*

- Lisp (McCarthy, 1960)
- Iswim (Landin, 1966)
- Scheme (Steele and Sussman, 1975)
- ML (Milner, Gordon, Wadsworth, 1979)
- Haskell (Hudak, Hughes, Peyton Jones, and Wadler, 1987)
- O'Caml (Leroy, 1996)
- Erlang (Armstrong, Virding, Williams, 1996)
- Scala (Odersky, 2004)
- F# (Syme, 2006)
- Clojure (Hickey, 2007)
- Elm (Czaplicki, 2012)

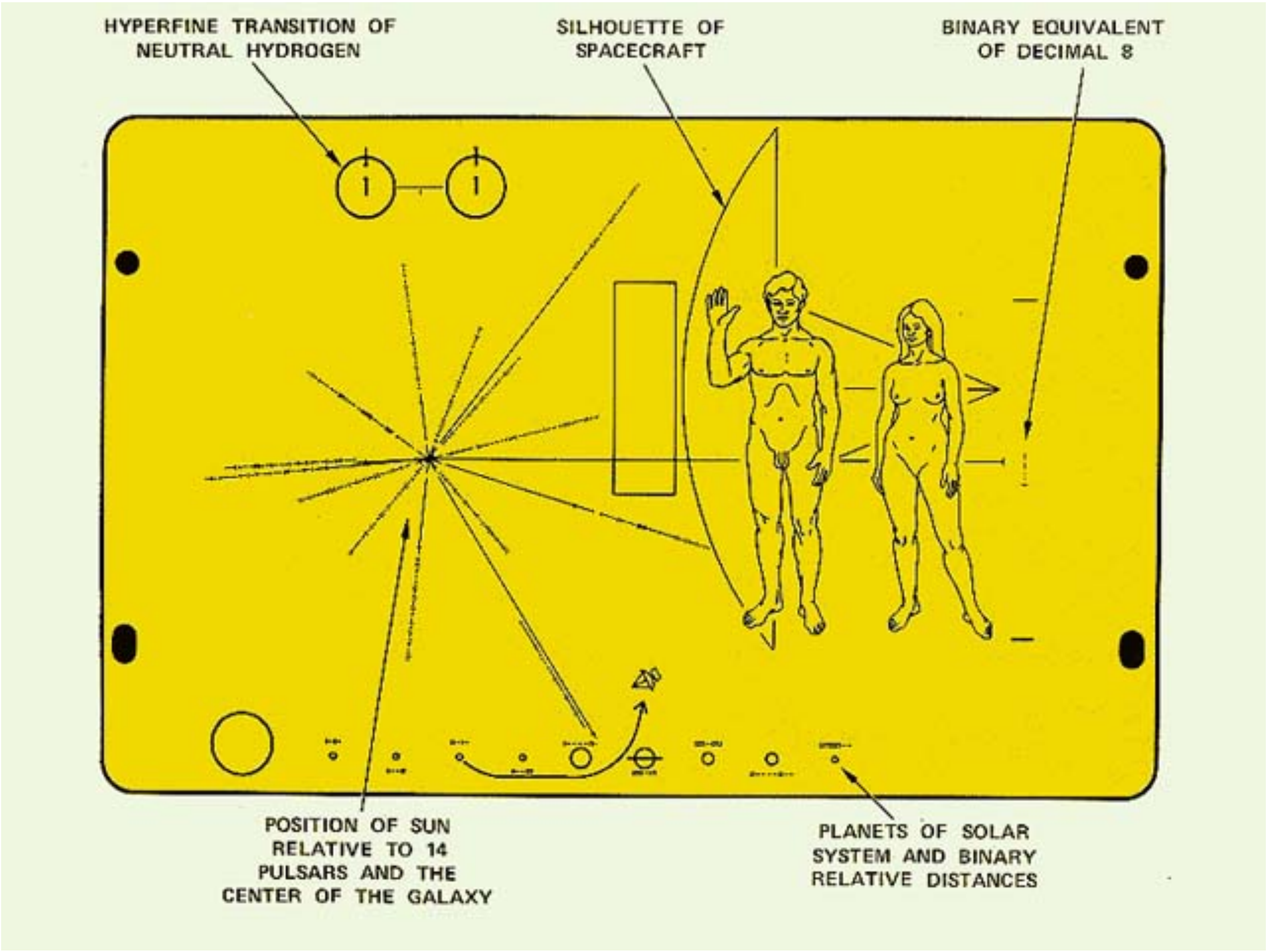
# *Proof Assistants*

- Automath (de Bruijn, 1970)
- Type Theory (Martin Löf, 1975)
- Mizar (Trybulec, 1975)
- ML/LCF (Milner, Gordon, and Wadsworth, 1979)
- NuPrl (Constable, 1985)
- HOL (Gordon and Melham, 1988)
- Coq (Huet and Coquand, 1988)
- Isabelle (Paulson, 1993)
- Epigram (McBride and McKinna, 2004)
- Agda (Norell, 2005)
- Lean (de Moura, 2013)

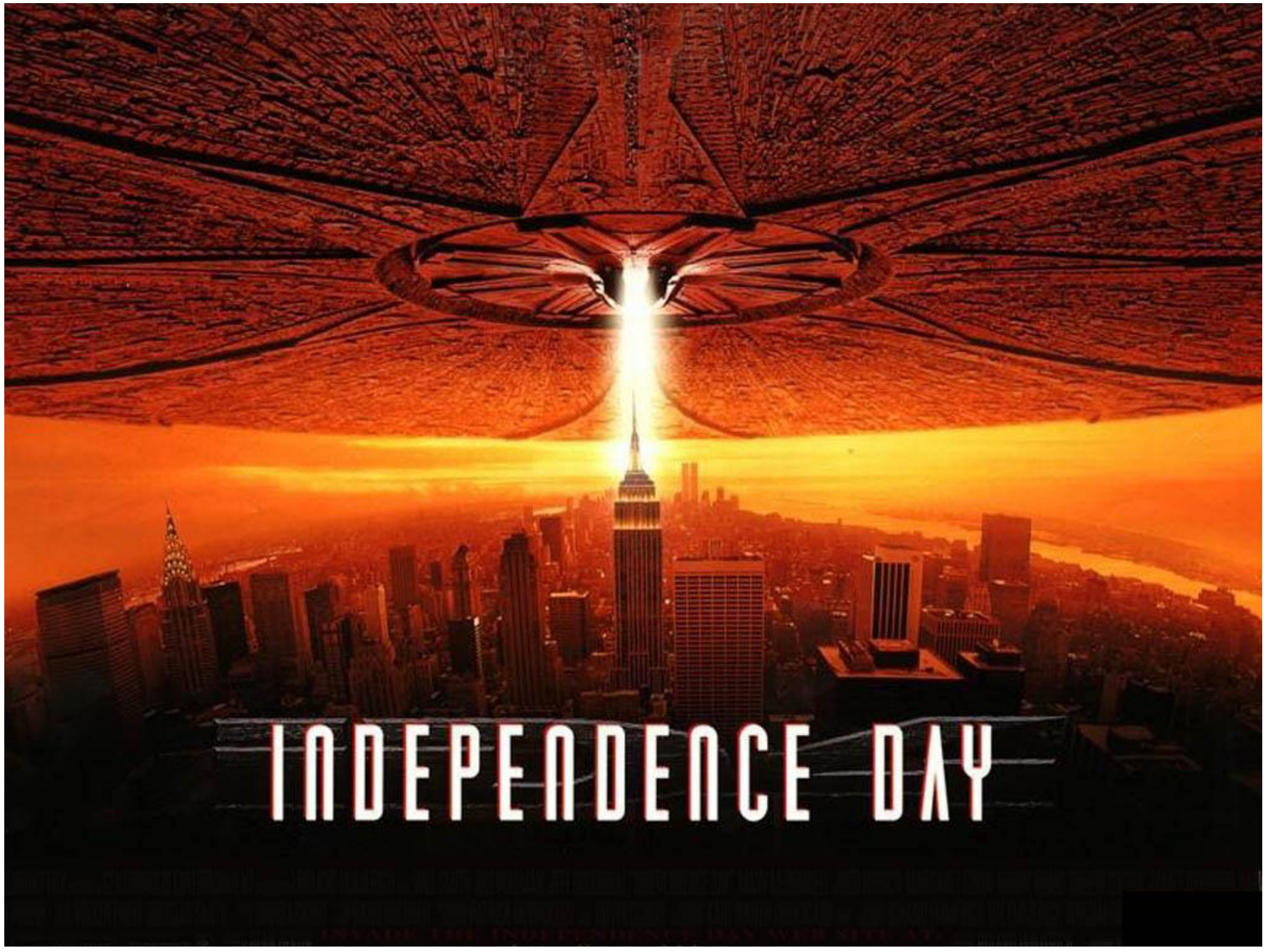
# Part III: Conclusion

# *Philosophy*

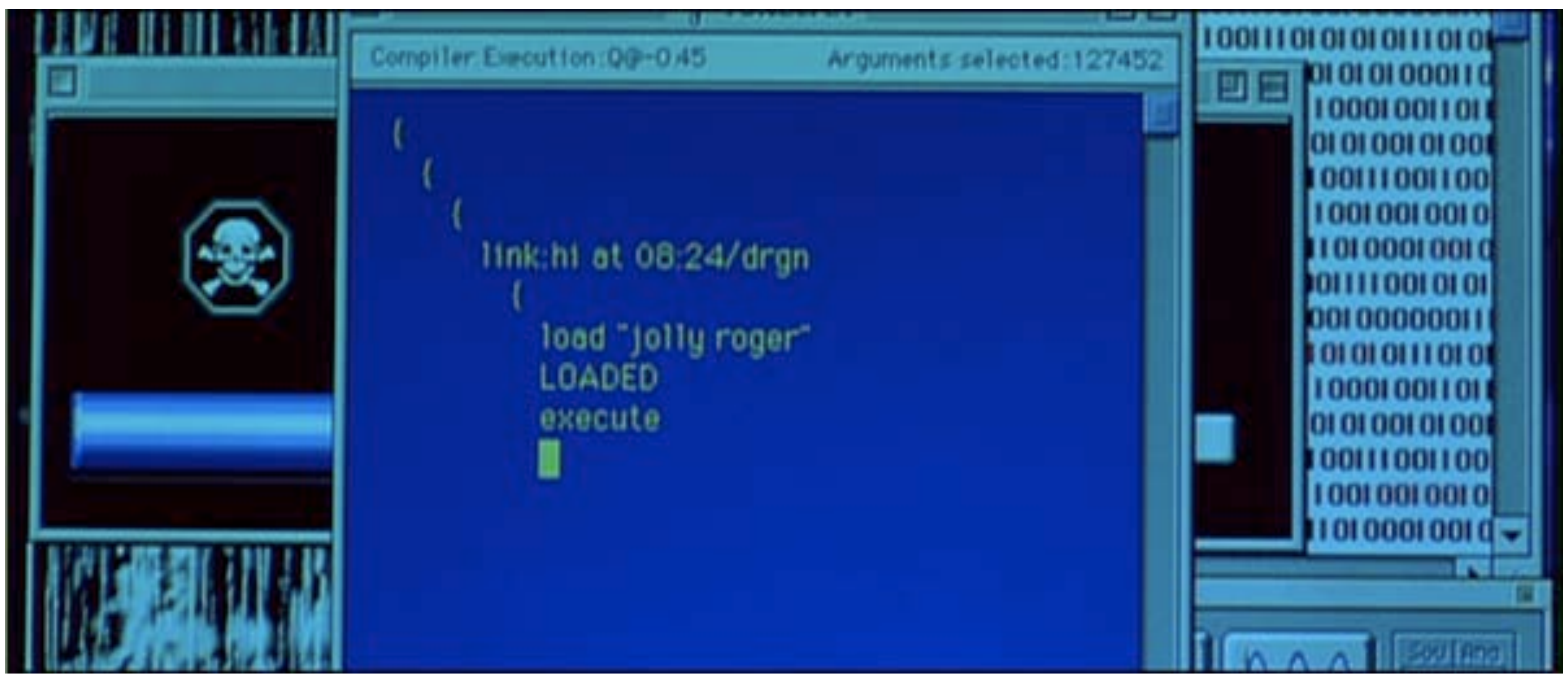
# *Let's talk to aliens!*



# *Independence Day*



# *A universal programming language?*



# *Multiverses*

# Lambda is Omniversal