

As Natural as 0, 1, 2

Philip Wadler

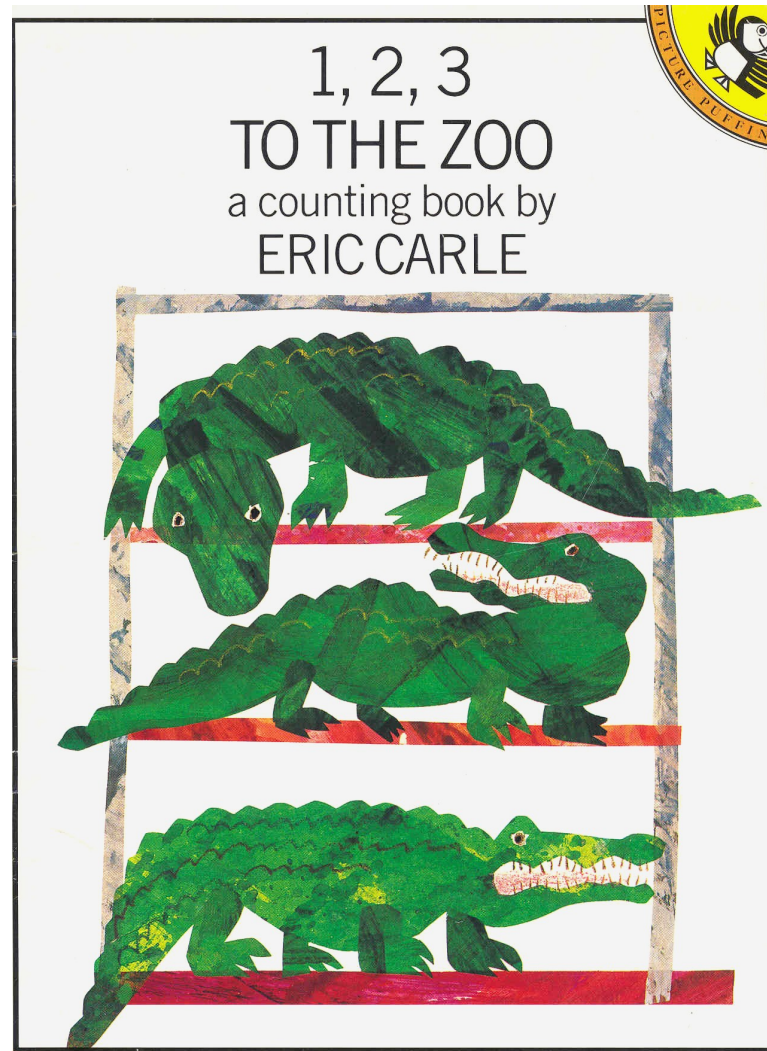
University of Edinburgh

wadler@inf.ed.ac.uk

Part 0

Counting starts at zero

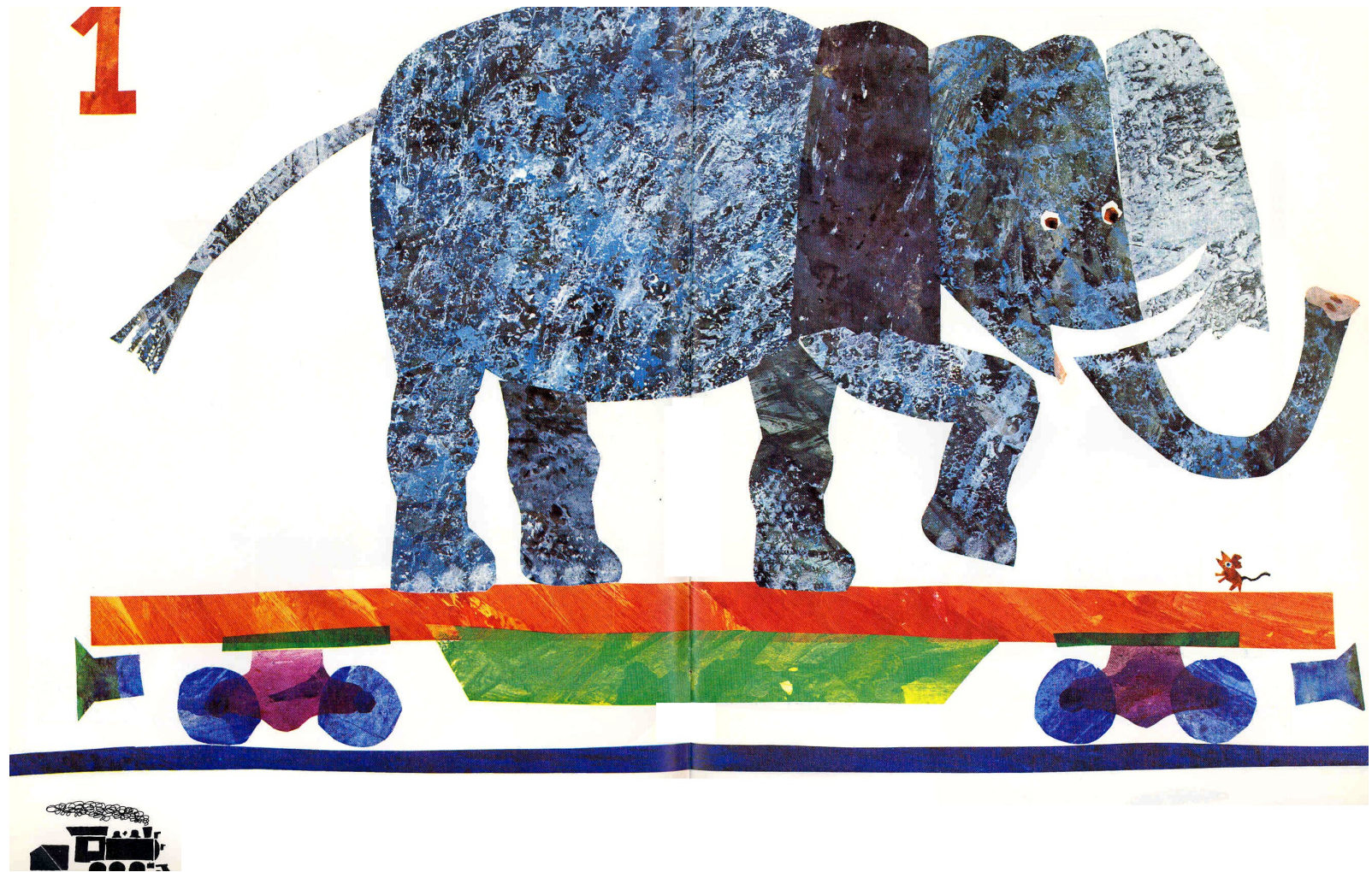
Carle



Carle



Carle

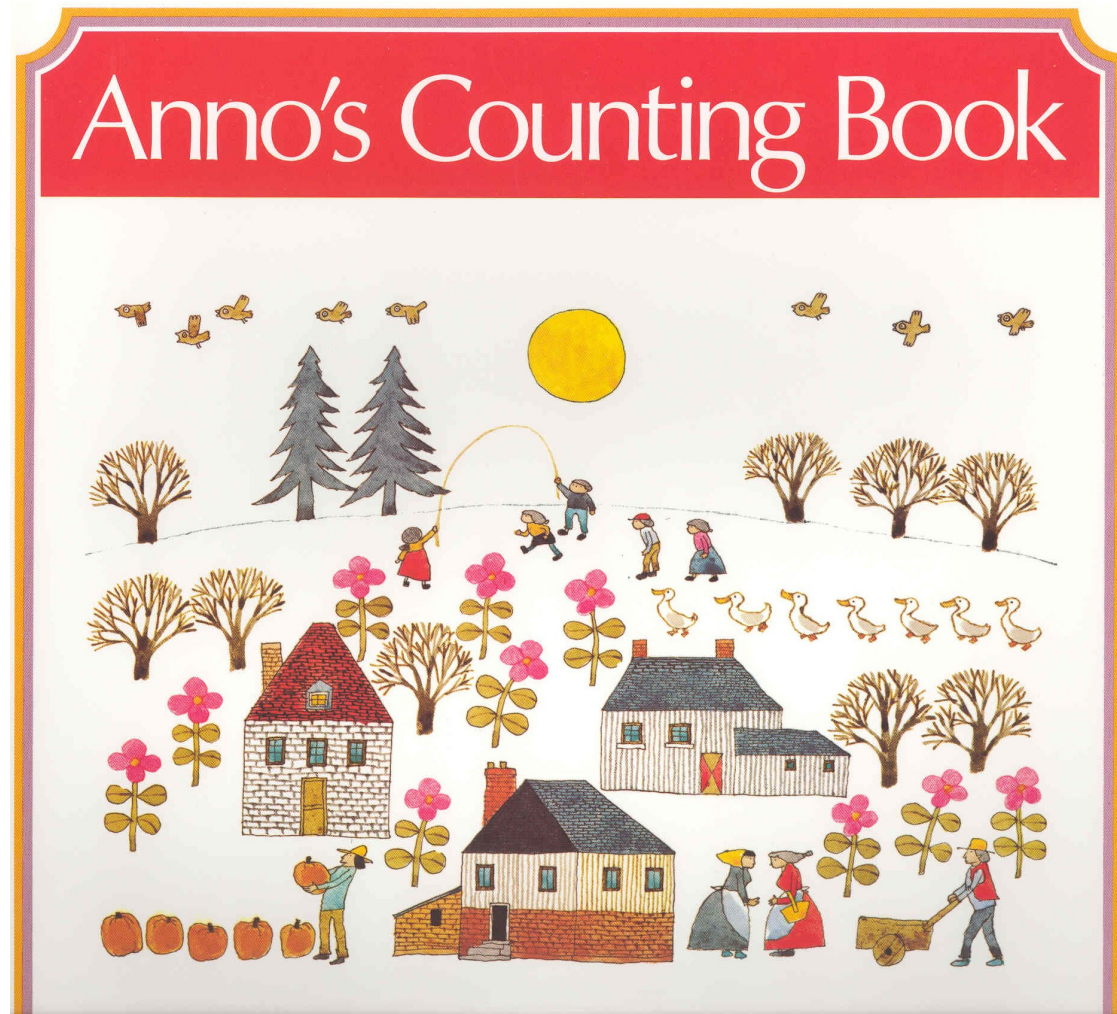


Carle

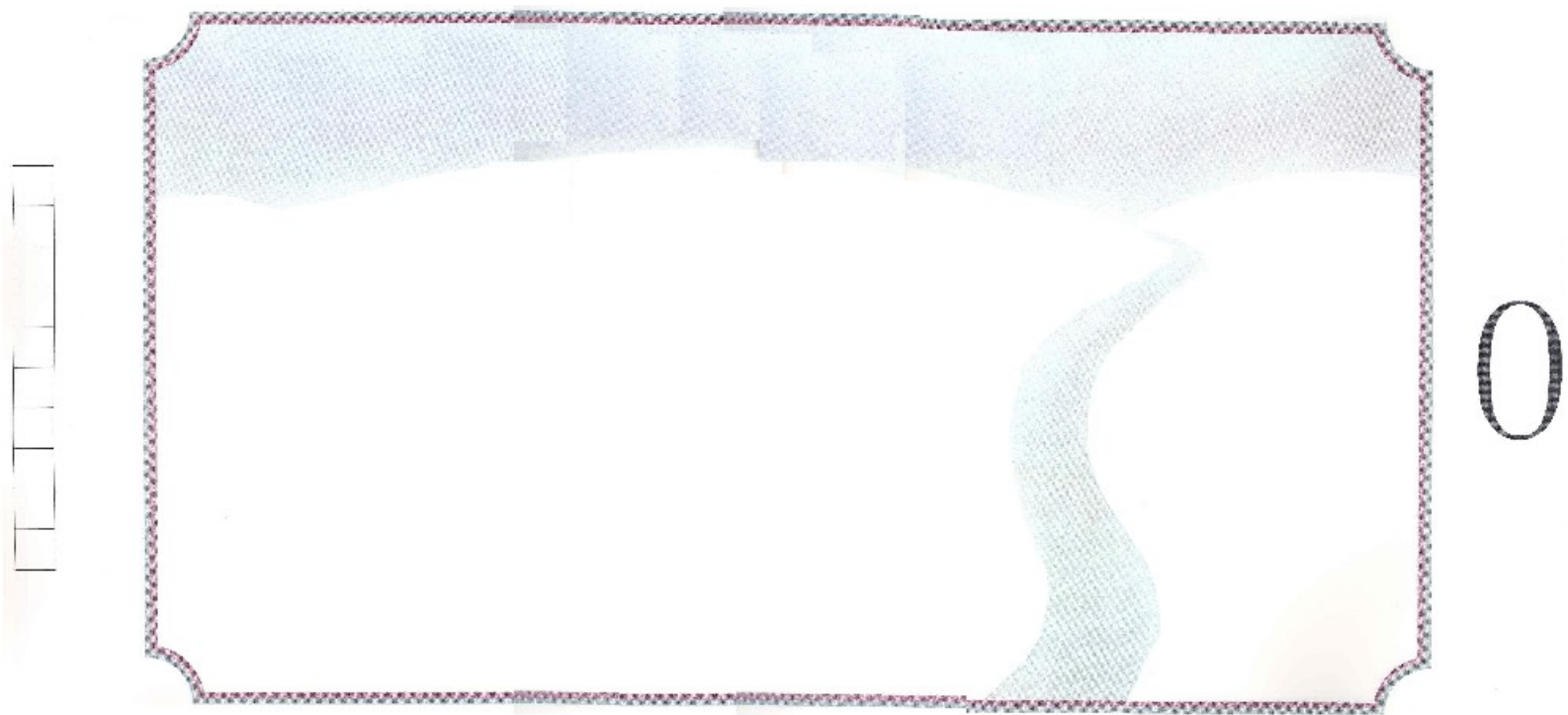
2



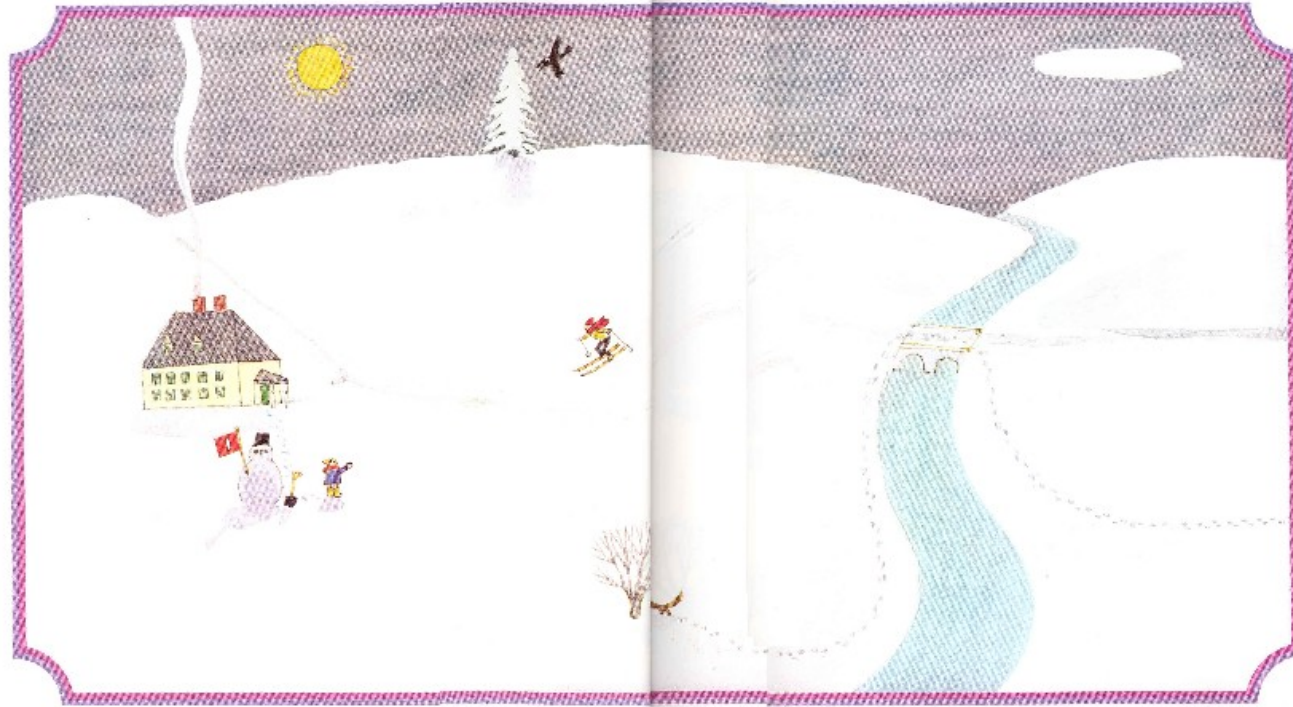
Anno



Anno



Anno



1

Anno



2

Zero appears in India, 7th century CE



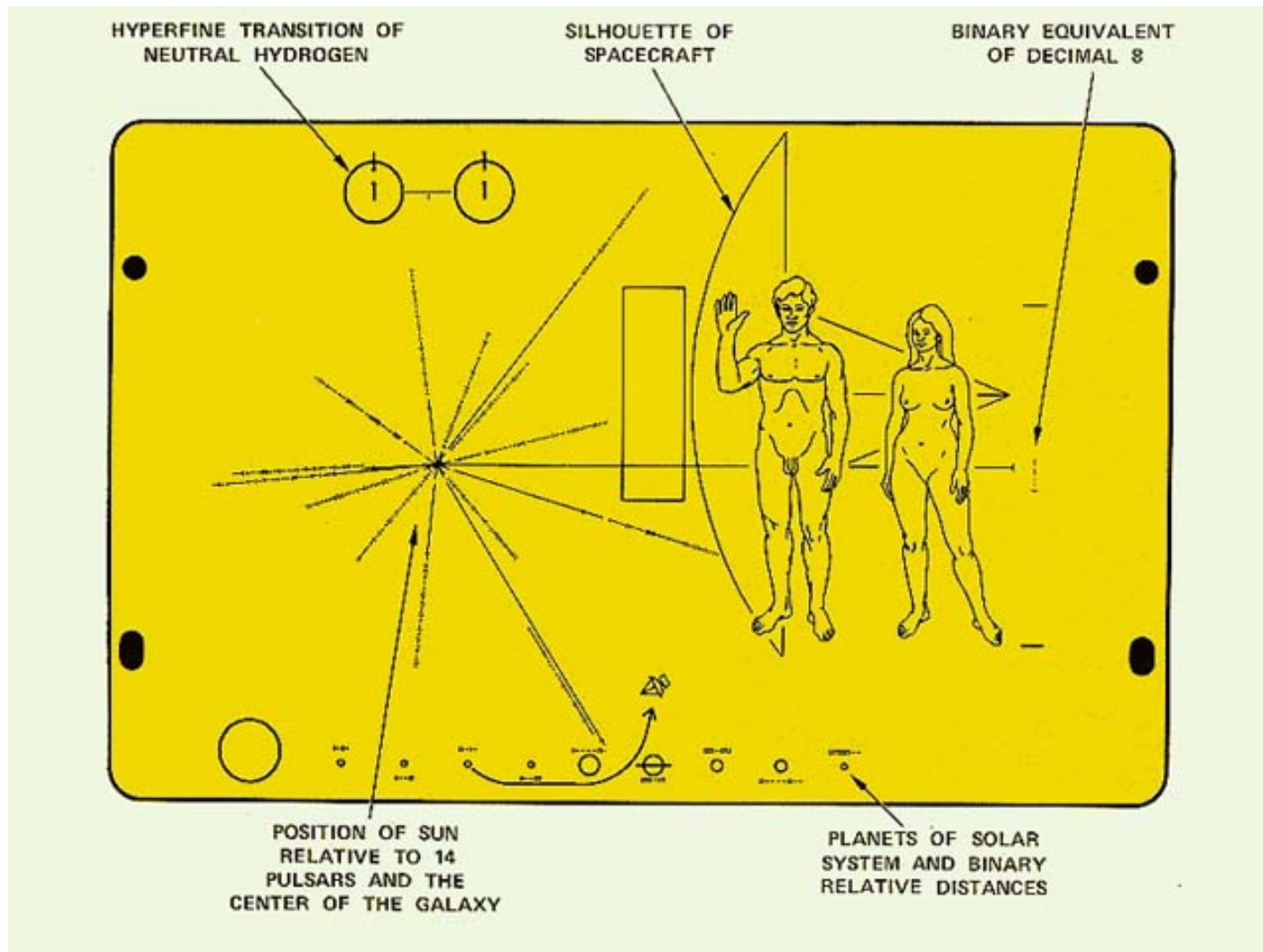
Counting in Japanese

0	leā	零	零
1	īchi	一	一
2	nī	二	二
3	san	三	三
4	shi	四	四
5	gō	五	五

Part 1

Aliens

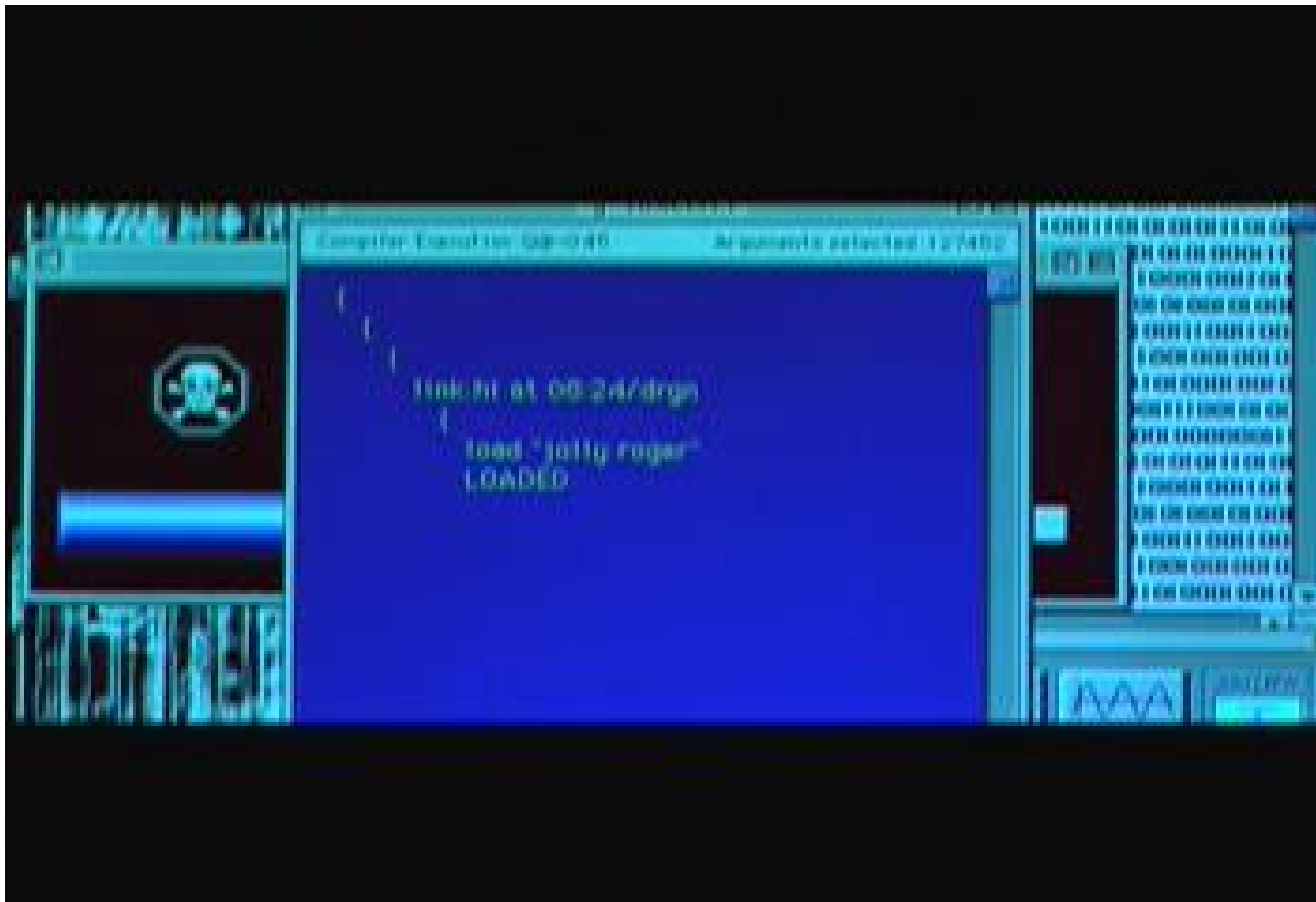
How to talk to aliens



Independence Day



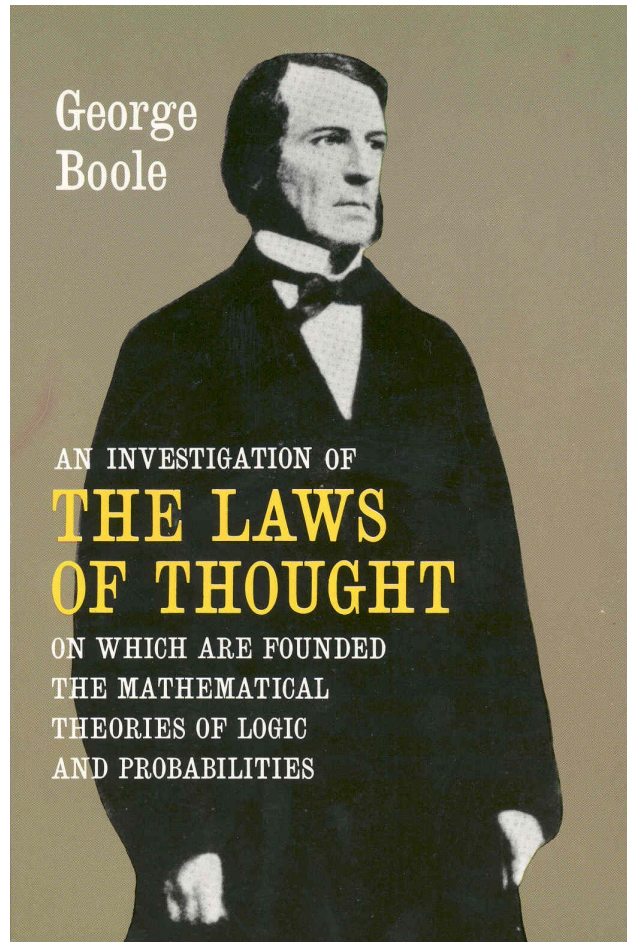
A universal programming language?



Part 2

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 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

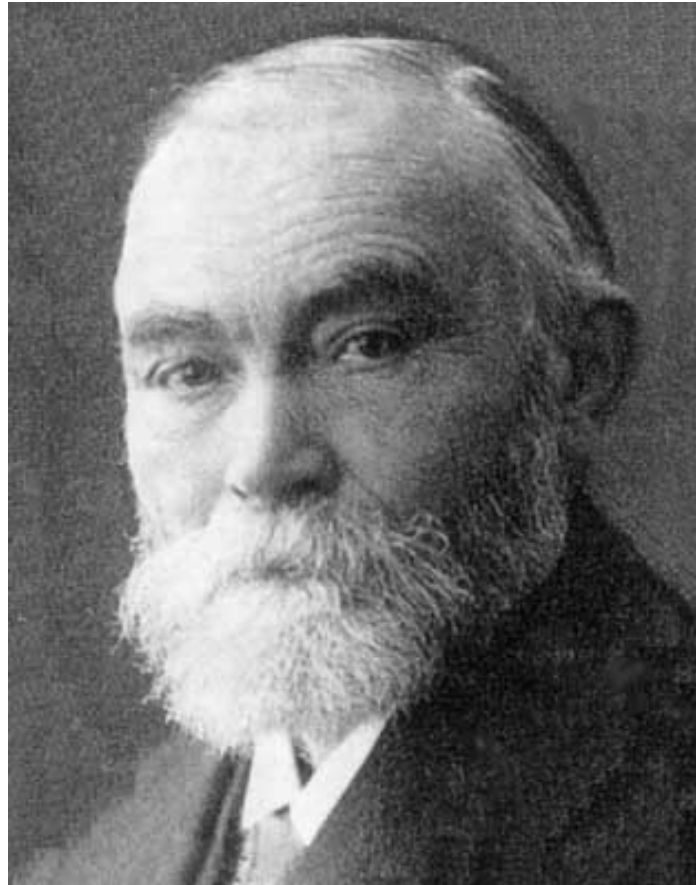
$$x(1 - x) = 0; \tag{1}$$

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

Part 3

Frege's *Begriffsschrift*

Gotlob Frege (1848–1925)



Frege 1879 — *modus ponens*

We could write this inference perhaps as follows :

$$\begin{array}{l} \vdash A \\ \quad \vdash B \end{array}$$
$$\vdash B$$

$$\vdash A.$$

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 — *modus ponens*

We could write this inference perhaps as follows :

$$\begin{array}{l} \vdash A \\ \quad \vdash B \end{array}$$
$$\vdash B$$

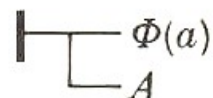
$$\vdash A.$$

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

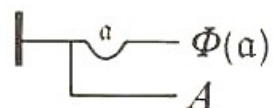
$$\frac{\vdash B \supset A \quad \vdash B}{\vdash A}$$

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 $\overset{a}{\smile} \Phi(a)$ is denied, we must be able to specify a meaning for a such that $\Phi(a)$ will be denied. If, therefore, $\overset{a}{\smile} \Phi(a)$ were to be denied and

Frege 1879 — quantification

It is clear also that from

$$\vdash \begin{array}{l} \Phi(a) \\ \quad \vdash \\ \quad A \end{array}$$

we can derive

$$\vdash \begin{array}{l} \overset{a}{\underbrace{\quad}} \Phi(a) \\ \quad \vdash \\ \quad A \end{array}$$

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 $\overset{a}{\underbrace{\quad}} \Phi(a)$ is denied, we must be able to specify a meaning for a such that $\Phi(a)$ will be denied. If, therefore, $\overset{a}{\underbrace{\quad}} \Phi(a)$ were to be denied and

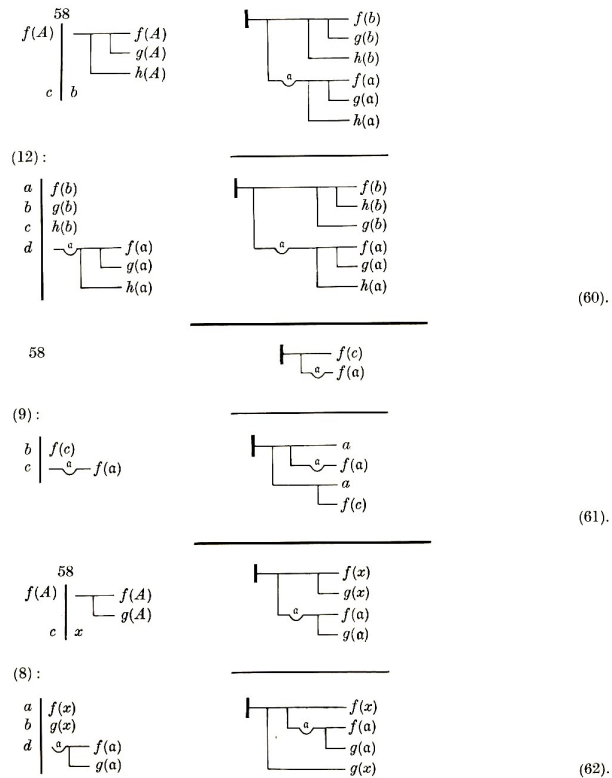
$$\frac{\vdash A \supset \Phi(a)}{\vdash A \supset \forall a. \Phi(x)}$$

Frege 1879

52

FREGE

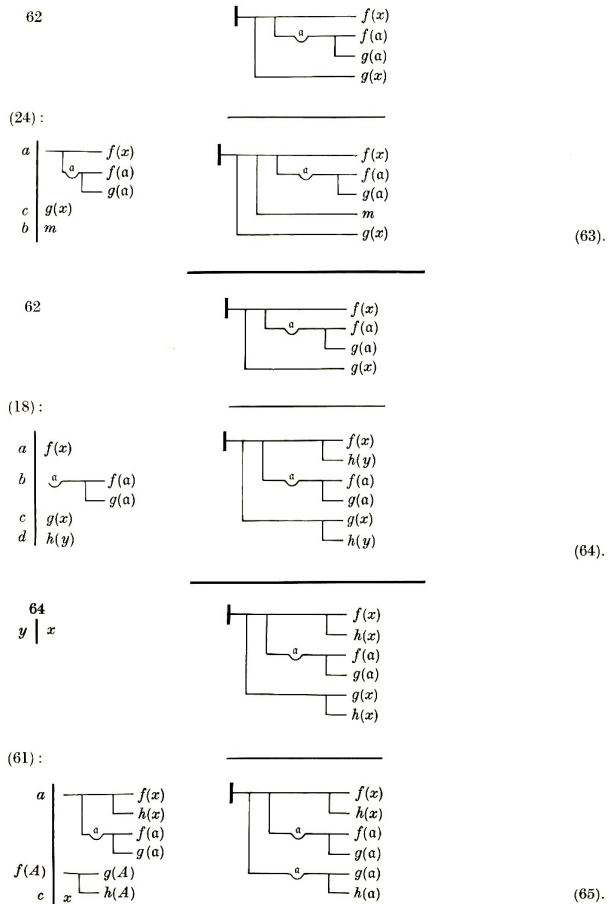
We see how this judgment replaces one mode of inference, namely, Felapton or Fesapo, between which we do not distinguish here since no subject has been singled out.



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

BEGRIFFSSCHRIFT

53



Part 4

Gentzen's Natural Deduction

Gerhard Gentzen (1909–1945)



Gentzen 1934: Natural Deduction

$\&-I$ $\frac{\mathcal{A} \quad \mathcal{B}}{\mathcal{A} \& \mathcal{B}}$	$\&-E$ $\frac{\mathcal{A} \& \mathcal{B} \quad \mathcal{A} \& \mathcal{B}}{\mathcal{A} \quad \mathcal{B}}$	$\vee-I$ $\frac{\mathcal{A} \quad \mathcal{B}}{\mathcal{A} \vee \mathcal{B} \quad \mathcal{A} \vee \mathcal{B}}$	$\vee-E$ $\frac{\mathcal{A} \vee \mathcal{B} \quad \begin{array}{c} [\mathcal{A}] \\ \mathcal{C} \end{array} \quad \begin{array}{c} [\mathcal{B}] \\ \mathcal{C} \end{array}}{\mathcal{C}}$
$\forall-I$ $\frac{\mathcal{F}a}{\forall x \mathcal{F}x}$	$\forall-E$ $\frac{\forall x \mathcal{F}x}{\mathcal{F}a}$	$\exists-I$ $\frac{\mathcal{F}a}{\exists x \mathcal{F}x}$	$\exists-E$ $\frac{\exists x \mathcal{F}x \quad \begin{array}{c} [\mathcal{F}a] \\ \mathcal{C} \end{array}}{\mathcal{C}}$
$\supset-I$ $\frac{\begin{array}{c} [\mathcal{A}] \\ \mathcal{B} \end{array}}{\mathcal{A} \supset \mathcal{B}}$	$\supset-E$ $\frac{\mathcal{A} \quad \mathcal{A} \supset \mathcal{B}}{\mathcal{B}}$	$\neg\neg-I$ $\frac{\begin{array}{c} [\mathcal{A}] \\ \wedge \end{array}}{\neg \mathcal{A}}$	$\neg\neg-E$ $\frac{\mathcal{A} \neg \mathcal{A} \quad \frac{\wedge}{\mathcal{D}}}{\wedge}$

Gentzen 1934: Natural Deduction

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

$$\frac{A \quad B}{A \& B} \&\text{-I}$$

$$\frac{A \& B}{A} \&\text{-E}_0$$

$$\frac{A \& B}{B} \&\text{-E}_1$$

A proof

$$\frac{\frac{[B \& A]^z}{A} \&-E_1 \quad \frac{[B \& A]^z}{B} \&-E_0}{A \& B} \&-I$$
$$\frac{A \& B}{(B \& A) \supset (A \& B)} \supset-I^z$$

Simplifying proofs

$$\frac{\frac{\begin{array}{c} [A]^x \\ \vdots \\ B \end{array}}{A \supset B} \supset\text{-I}^x \quad \begin{array}{c} \vdots \\ A \end{array}}{B} \supset\text{-E} \Rightarrow \begin{array}{c} \vdots \\ A \\ \vdots \\ B \end{array}$$

$$\frac{\frac{\begin{array}{c} \vdots \\ A \end{array} \quad \begin{array}{c} \vdots \\ B \end{array}}{A \& B} \&\text{-I} \quad \begin{array}{c} \vdots \\ A \end{array}}{A} \&\text{-E}_0 \Rightarrow \begin{array}{c} \vdots \\ A \end{array}$$

Simplifying a proof

$$\frac{\frac{\frac{[B \& A]^z}{A} \&-E_1 \quad \frac{[B \& A]^z}{B} \&-E_0}{A \& B} \&-I \quad \frac{[B]^y \quad [A]^x}{B \& A} \&-I}{(B \& A) \supset (A \& B) \supset-I^z \quad B \& A \supset-E} \supset-E$$
$$\frac{}{A \& B}$$

Simplifying a proof

$$\begin{array}{c}
 \frac{[B \& A]^z}{A} \&-E_1 \quad \frac{[B \& A]^z}{B} \&-E_0 \\
 \hline
 A \& B \quad \&-I \\
 \hline
 (B \& A) \supset (A \& B) \quad \supset-I^z \\
 \hline
 \frac{\quad \quad \quad \quad \quad \frac{[B]^y \quad [A]^x}{B \& A} \&-I}{A \& B} \supset-E
 \end{array}$$

\Downarrow

$$\begin{array}{c}
 \frac{[B]^y \quad [A]^x}{B \& A} \&-I \quad \frac{[B]^y \quad [A]^x}{B \& A} \&-I \\
 \frac{\frac{B \& A}{A} \&-E_1 \quad \frac{B \& A}{B} \&-E_0}{A \& B} \&-I
 \end{array}$$

Simplifying a proof

$$\begin{array}{c}
 \frac{[B \& A]^z}{A} \&-E_1 \quad \frac{[B \& A]^z}{B} \&-E_0 \\
 \hline
 A \& B \quad \&-I \\
 \hline
 (B \& A) \supset (A \& B) \quad \supset-I^z \\
 \hline
 \frac{A \& B \quad \frac{[B]^y \quad [A]^x}{B \& A} \&-I}{A \& B} \supset-E \\
 \\
 \Downarrow \\
 \frac{\frac{[B]^y \quad [A]^x}{B \& A} \&-I \quad \frac{[B]^y \quad [A]^x}{B \& A} \&-I}{\frac{B \& A}{A} \&-E_1 \quad \frac{B \& A}{B} \&-E_0} \&-I \\
 \\
 \Downarrow \\
 \frac{[A]^x \quad [B]^y}{A \& B} \&-I
 \end{array}$$

Part 5

Church's Lambda Calculus

Alonzo Church (1903–1995)



Church 1932: Lambda Calculus

An occurrence of a variable \mathbf{x} in a given formula is called an occurrence of \mathbf{x} as a *bound variable* in the given formula if it is an occurrence of \mathbf{x} in a part of the formula of the form $\lambda \mathbf{x}[\mathbf{M}]$; that is, if there is a formula \mathbf{M} such that $\lambda \mathbf{x}[\mathbf{M}]$ occurs in the given formula and the occurrence of \mathbf{x} in question is an occurrence in $\lambda \mathbf{x}[\mathbf{M}]$. 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

Lambda



Reduction rules

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

$$\langle t, u \rangle_0 \Rightarrow t$$

$$\langle t, u \rangle_1 \Rightarrow u$$

Simplifying a term

$$(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle$$

Simplifying a term

$$(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle$$

↓

$$\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle$$

Simplifying a term

$$(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle$$

\Downarrow

$$\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle$$

\Downarrow

$$\langle x, y \rangle$$

Church 1940: Typed Lambda Calculus

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

$$\frac{t : A \quad u : B}{\langle t, u \rangle : A \& B} \&\text{-I} \qquad \frac{s : A \& B}{s_0 : A} \&\text{-E}_0 \qquad \frac{s : A \& B}{s_1 : B} \&\text{-E}_1$$

A program

$$\frac{\frac{[z : B \& A]^z}{z_1 : A} \&-E_1 \quad \frac{[z : B \& A]^z}{z_0 : B} \&-E_0}{\langle z_1, z_0 \rangle : A \& B} \&-I}{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)} \supset-I^z$$

Simplifying programs

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

$$\frac{\frac{\begin{array}{c} \vdots \\ t : A \end{array} \quad \begin{array}{c} \vdots \\ u : B \end{array}}{\langle t, u \rangle : A \& B} \&\text{-I}}{\frac{\quad}{\langle t, u \rangle_0 : A} \&\text{-E}_0} \Rightarrow \begin{array}{c} \vdots \\ t : A \end{array}$$

Simplifying a program

$$\begin{array}{c}
 \frac{[z : B \& A]^z}{z_1 : A} \&-E_1 \quad \frac{[z : B \& A]^z}{z_0 : B} \&-E_0 \\
 \frac{\quad}{\langle z_1, z_0 \rangle : A \& B} \&-I \\
 \frac{\quad}{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)} \supset-I^z \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I \\
 \frac{\quad}{(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B} \supset-E
 \end{array}$$

Simplifying a program

$$\frac{\frac{[z : B \& A]^z}{z_1 : A} \&-E_1 \quad \frac{[z : B \& A]^z}{z_0 : B} \&-E_0}{\langle z_1, z_0 \rangle : A \& B} \&-I \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I}{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)} \supset-I^z \quad \supset-E}$$

$$(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B$$

↓

$$\frac{\frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I}{\langle y, x \rangle_1 : A} \&-E_1 \quad \frac{\langle y, x \rangle : B \& A}{\langle y, x \rangle_0 : B} \&-E_0}{\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle : A \& B} \&-I$$

Simplifying a program

$$\frac{\frac{[z : B \& A]^z}{z_1 : A} \&-E_1 \quad \frac{[z : B \& A]^z}{z_0 : B} \&-E_0}{\langle z_1, z_0 \rangle : A \& B} \&-I \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I}{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)} \supset-I^z \quad \supset-E}$$

$$(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B$$

↓

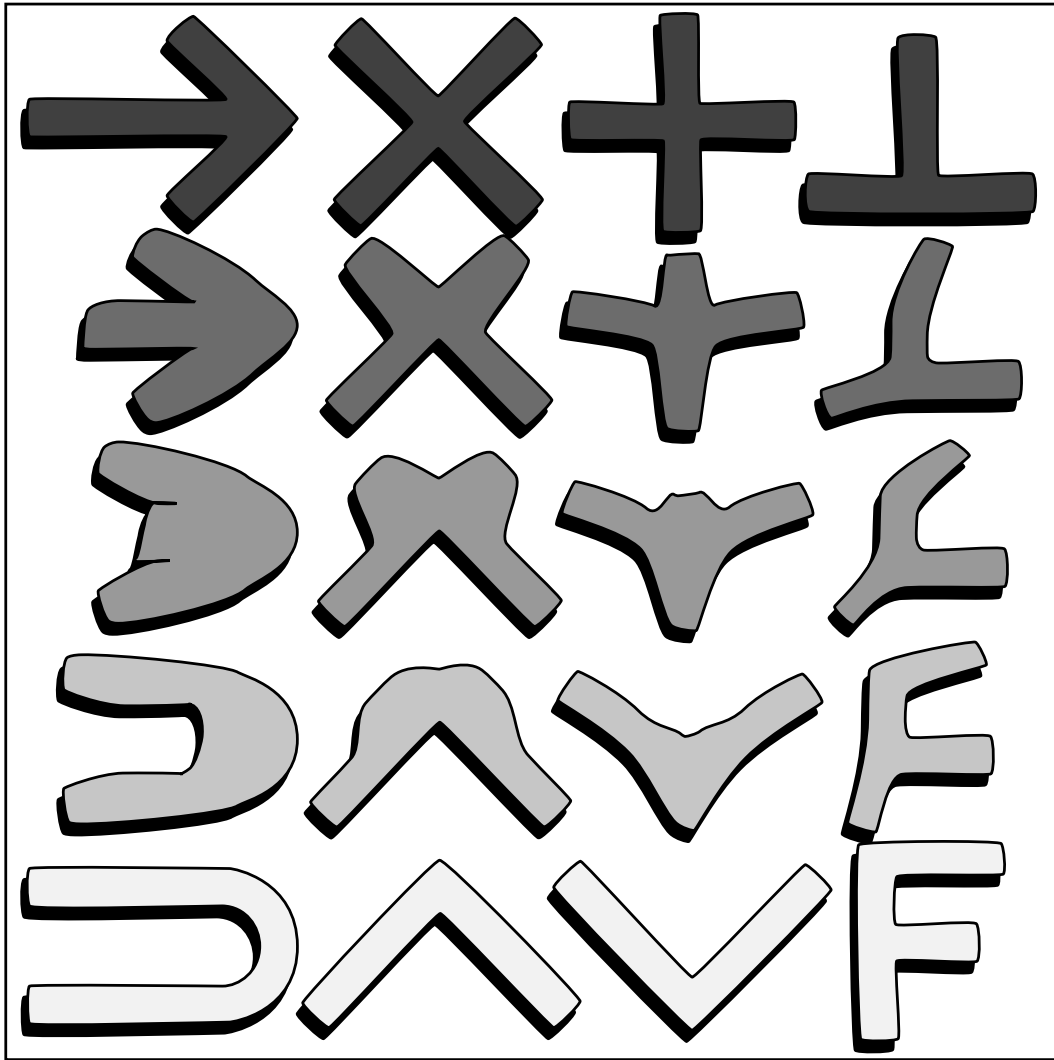
$$\frac{\frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \&-I}{\langle y, x \rangle_1 : A} \&-E_1 \quad \frac{\langle y, x \rangle : B \& A}{\langle y, x \rangle_0 : B} \&-E_0}{\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle : A \& B} \&-I$$

↓

$$\frac{[x : A]^x \quad [y : B]^y}{\langle x, y \rangle : A \& B} \&-I$$

Part 6

The Curry-Howard Isomorphism



LC'90

The Curry-Howard homeomorphism

Haskell Curry (1900–1982) / William Howard



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 $\Gamma \rightarrow \beta$, 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
 $\alpha \rightarrow \alpha$

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

(1.3)
$$\frac{\Gamma \rightarrow \alpha \quad \Delta \rightarrow \alpha \supset \beta}{\Gamma, \Delta \rightarrow \beta}$$

(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, \dots 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 7

Programs and Proofs

Programs

- [Lisp](#) (McCarthy, 1960)
- [Iswim](#) (Landin, 1966)
- [Scheme](#) (Steele and Sussman, 1975)
- [ML](#) (Milner, Gordon, Wadsworth, 1979)
- [Hope](#) (Burstall, MacQueen, Sannella, 1980)
- [Miranda](#) (Turner, 1985)
- [Haskell](#) (Hudak, Peyton Jones, and Wadler, 1987)
- [O'Caml](#) (Leroy, 1996)
- [Links](#) (Wadler et al, 2005)

Proofs

- [Automath](#) (de Bruijn, 1970)
- [Type Theory](#) (Martin Löf, 1975)
- [ML/LCF](#) (Milner, Gordon, and Wadsworth, 1979)
- [HOL](#) (Gordon and Melham, 1988)
- [CoQ](#) (Huet and Coquand, 1988)
- [Isabelle](#) (Paulson, 1993)

Proofs/Programs

- Hindley/Milner (1969/1975)
- Girard/Reynolds (1972/1975)
- Linear Logic/Syntactic Control of Interference (1987/1978)
- Classical Logic/Continuation-Passing Style (1990)
- And dual to Or/Call-by-value dual to Call-by-name (2000)

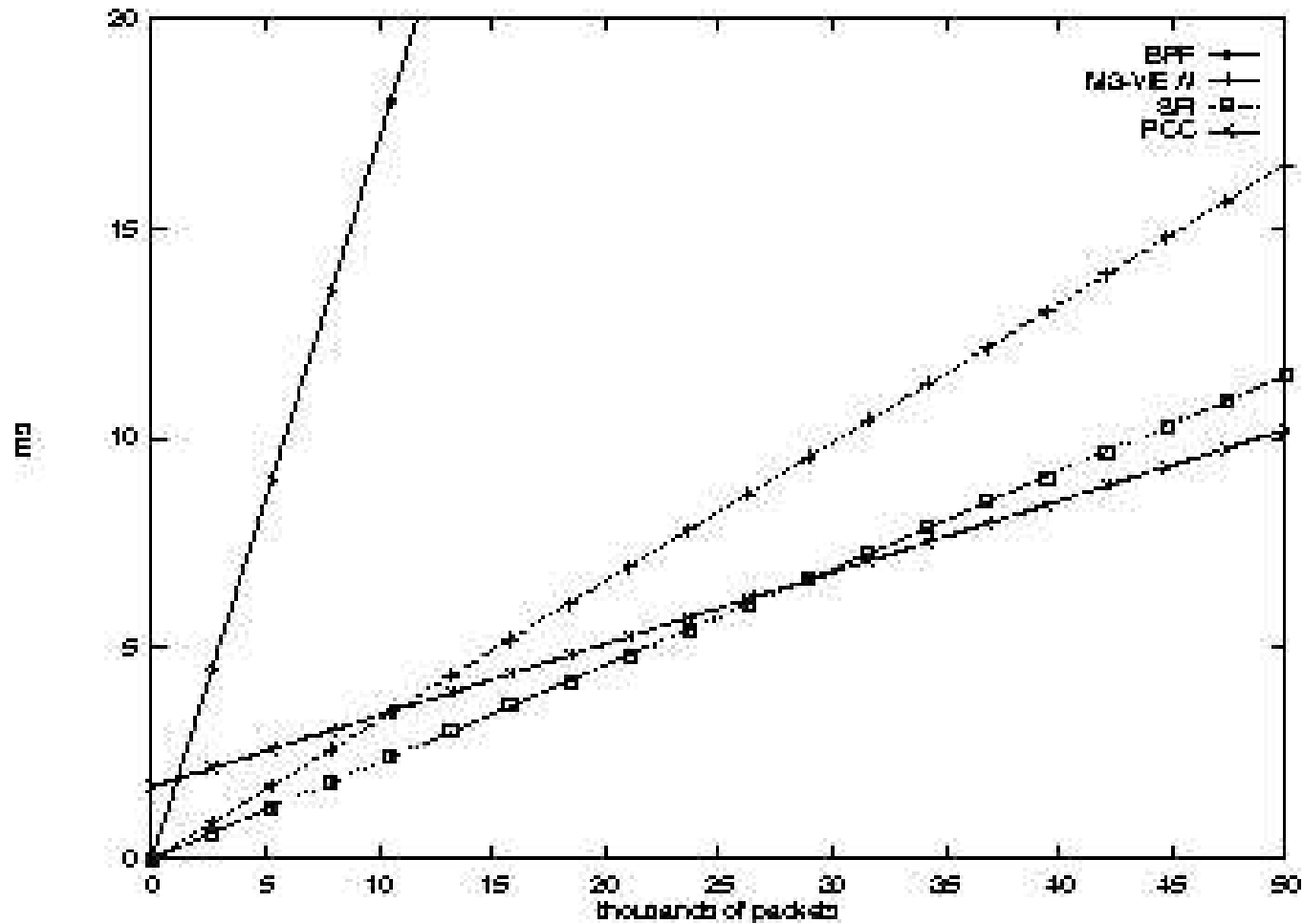
Part 8

Programs and Proofs on the Web

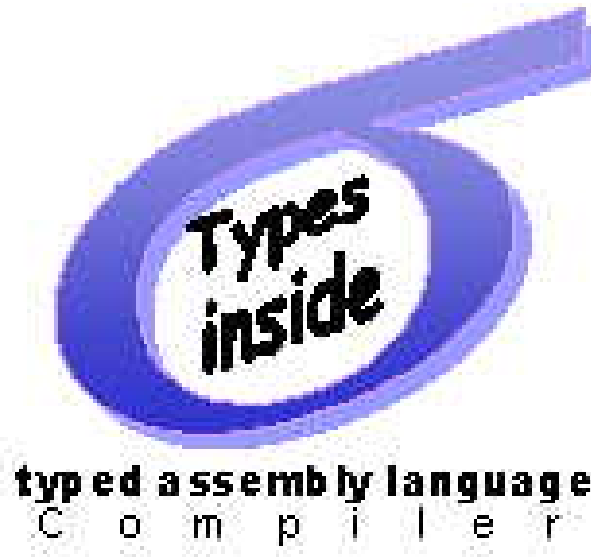
Java (Gosling, Joy, and Steele, 1996)



Proof-Carrying Code (Necula and Lee, 1996)



Typed Assembly Language (Morrisett et al 1998)



Typed Assembly Language (Morrisett et al 1998)



What do you want to type check today?

XQuery



XQuery 1.0: An XML Query Language

W3C Working Draft 16 August 2002

This version:

<http://www.w3.org/TR/2002/WD-xquery-20020816/>

Latest version:

<http://www.w3.org/TR/xquery/>

Previous versions:

<http://www.w3.org/TR/2002/WD-xquery-20020430/>

<http://www.w3.org/TR/2001/WD-xquery-20011220/>

<http://www.w3.org/TR/2001/WD-xquery-20010607/>

Editors:

Scott Boag (XSL WG), IBM Research <scott_boag@us.ibm.com>

Don Chamberlin (XML Query WG), IBM Almaden Research Center <chamberlin@almaden.ibm.com>

Mary F. Fernandez (XML Query WG), AT&T Labs <mff@research.att.com>

Daniela Florescu (XML Query WG), XQRL <dana@xqrl.com>

Jonathan Robie (XML Query WG), DataDirect Technologies <jonathan.robie@datadirect-technologies.com>

Jérôme Siméon (XML Query WG), Bell Labs, Lucent Technologies <simeon@research.bell-labs.com>

[Copyright](#) © 2002 W3C[®] (MIT, INRIA, Keio), All Rights Reserved. W3C [liability](#), [trademark](#), [document use](#), and [software licensing](#) rules apply.

XQuery - Microsoft

XML Query Language Demo

Microsoft .net **New XQuery Prototype released!**

A new version of Microsoft's XQuery Prototype was released on December 9, 2002. This version is based on the August 15th draft of the W3C XQuery specification.

Documentation

- What's New
- Readme
- Known Issues

Specifications

- XQuery Syntax Draft
- XQuery Functions & Operators
- XQuery Use Cases

W3C Use Cases

- XMP_Cases
- TREE_Cases
- SEQ_Cases
- SGML_Cases
- NS_Cases

Download

[XQuery Demo \(Dec 20, 2001 Draft\)](#)

Feedback

[Click here to send us feedback.](#)

Introduction

Welcome to Microsoft's XQuery Demo. This demo was designed with the August 15th, 2002 version of the XQuery working draft.

Instructions:

1. Either select one of the example "W3C Use Cases" in the pane to the left or type your own query in the text box below.
2. Click the "Execute Query" Button to generate results. (Note: Results will be displayed in a new window)

Please refer to the [Readme](#) for more information.

Query Expression

```
<code>< bib >
{
  for $b in document("http://www.bn.com/bib.xml")/bib/book
  where $b/publisher = "Addison-Wesley" and $b/@year > 1991
  return
  < book year="{ $b/@year }">
    { $b/title }
  < /book >
}
< /bib >
```

XQuery - Oracle

The screenshot shows the Oracle Technology Network (OTN) website. The page title is "Oracle XQuery Prototype Querying XML the XQuery way". The page is dated "January 2003". The main content describes a prototype implementation of the XQuery language with Oracle extensions, available as a technical preview release (RELEASE_0.2_030121). It includes a jarfile, Java docs, and a command-line utility (JXQI). The page also lists prerequisites: Oracle XDK, JDK 1.2.2_07, Oracle XSU, and a JDBC driver. A sidebar on the left contains navigation links for Resources and Services. A sidebar on the right provides download instructions and related documents.

ORACLE Technology NETWORK

Oracle.com Products Oracle Store Downloads My Profile Search

(Click Here to register for a free OTN Web account)

Home Technology Centers **Resources** Services

Oracle Technology Network > Sample Code > XML > XML DB >

Need Assistance? Ask the Oracle Concierge

Resources

- Products
- DBA
- Downloads
- Documentation
- Sample Code
- Security Alerts
- Oracle Magazine

Services

- Technology Tracks
- Discussion Forums
- Skills Marketplace
- Training & Support

Oracle Developer Software

Oracle XQuery Prototype Querying XML the XQuery way

January 2003

This download is a prototype implementation of the evolving XQuery language, with Oracle extensions. The release version is RELEASE_0.2_030121. This is a technical preview release.

The download contains a jarfile with the Oracle XQuery prototype (in a jarfile), and java docs. Once installed, you can use the Java API (JXQI), or the command-line utility to test the prototype.

The Readme includes simple installation and setup instructions.

Prerequisites :

- [Oracle XDK](#)
- [JDK 1.2.2_07](#)
- [Oracle XSU](#): if you want to access the database
- [JDBC driver](#): if you want to access the database

What's new :

Changes to previous release (RELEASE_0.1_020210, March 2002)

Download the XQuery Prototype

The download includes installation instructions, Java jar file, and instructions for use.

- [Download Oracle XQuery prototype \(for windows\)](#)
- [Download Oracle XQuery prototype \(for Unix\)](#)
- [Read the Readme](#)

Related Documents

- [JXQI : a Java API to the](#)

Part 9

Conclusion

Frege

Undone by Russell's Paradox

Church and Curry

Attended 1982 Conference on
Lisp and Functional Programming

Gentzen

“He once confided in me that he was really quite contented since now he had at last time to think about a consistency proof for analysis.”

Frege

Undone by Russell's Paradox

Church and Curry

Attended 1982 Conference on
Lisp and Functional Programming

Gentzen

“He once confided in me that he was really quite contented since now he had at last time to think about a consistency proof for analysis.”

Died in prison, 4 August 1945

Special thanks to

Martina Sharp, Avaya Labs
for scanning all the pictures

Adam and Leora Wadler
for loaning me their books