As Natural as 0, 1, 2

Philip Wadler
University of Edinburgh
wadler@inf.ed.ac.uk
Part 0

Counting starts at zero
Carle
Carle
Anno’s Counting Book

Anno
Anno
Zero appears in India, 7th century CE
Counting in Japanese

<table>
<thead>
<tr>
<th></th>
<th>lea</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>ichi</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ni</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>san</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>shi</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>go</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part 1

Aliens
How to talk to aliens
Independence Day
A universal programming language?
Part 2

Boolean algebra
George Boole (1815–1864)

AN INVESTIGATION OF
THE LAWS
OF THOUGHT
ON WHICH ARE FOUNDED
THE MATHEMATICAL
THEORIES OF LOGIC
AND PROBABILITIES
Boole 1847: Mathematical analysis of logic

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

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

\[ x (1 - x) = 0; \]  

(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)
We could write this inference perhaps as follows:

\[
\begin{array}{c}
\hline A \\
B \\
\hline B \\
\hline A.
\end{array}
\]

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:

\[ \vdash A \]
\[ \vdash B \]
\[ \vdash B \]
\[ \vdash A. \]

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

\[ \vdash B \supset A \]
\[ \vdash B \]
\[ \vdash A \]
Frege 1879 — quantification

It is clear also that from

\[ \frac{}{\Phi(a)} \]

we can derive

\[ \frac{\varepsilon}{\Phi(a)} \]

if \( A \) is an expression in which \( a \) does not occur and if \( a \) stands only in the argument places of \( \Phi(a) \).\(^\text{14}\) If \( \varepsilon \Phi(a) \) is denied, we must be able to specify a meaning for \( a \) such that \( \Phi(a) \) will be denied. If, therefore, \( \varepsilon \Phi(a) \) were to be denied and
Frege 1879 — quantification

It is clear also that from

\[ \vdash \Phi(a) \]

A

we can derive

\[ \vdash \Phi(a) \]

A

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 \[ \neg \neg \neg \Phi(a) \] is denied, we must be able to specify a meaning for a such that \( \Phi(a) \) will be denied. If, therefore, \[ \neg \neg \neg \Phi(a) \] were to be denied and

\[ \vdash A \supset \Phi(a) \]

\[ \vdash A \supset \forall a. \Phi(x) \]
Frege 1879

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

This judgment replaces the mode of inference Barborn when the minor premise, \(\theta(b)\), has a particular content.
Part 4

Gentzen’s Natural Deduction
Gerhard Gentzen (1909–1945)
Gentzen 1934: Natural Deduction

\[
\begin{array}{cccc}
\&-I & \&-E & \lor-I & \lor-E \\
\frac{A \quad B}{A \& B} & \frac{A \& B}{A} & \frac{A \lor B}{B} & \frac{[A] \quad [B]}{A \lor B \\ C} \\
\end{array}
\]

\[
\begin{array}{cccc}
\forall-I & \forall-E & \exists-I & \exists-E \\
\frac{\forall x \exists x}{\exists x} & \frac{\forall x \exists x}{\exists x} & \frac{\exists x}{\exists x} & \frac{[\exists x]}{\forall x \exists x \\ C} \\
\end{array}
\]

\[
\begin{array}{cccc}
\imp-I & \imp-E & \neg-I & \neg-E \\
\frac{[A]}{B} & \frac{A \quad A \imp B}{B} & \frac{[A]}{\imp} & \frac{A \quad \imp}{\neg} \\
\end{array}
\]

\[
\begin{array}{cccc}
\forall- \exists & \exists- \forall \\
\frac{\exists x \forall x}{\forall x} & \frac{\forall x \exists x}{\exists x} \\
\end{array}
\]
Gentzen 1934: Natural Deduction

\[
\begin{align*}
  & [A]^x \\
  & \cdot \\
  & \cdot \\
  & B \\
  & \Rightarrow I^x \\
  & A \Rightarrow B \\
  & A \Rightarrow B \\
  \end{align*}
\]

\[
\begin{align*}
  & A \Rightarrow B \\
  & A \Rightarrow B \\
  \Rightarrow E
\end{align*}
\]

\[
\begin{align*}
  & A \land B \\
  \land I \\
  & A \land B \\
  \land E_0 \\
  & A \land B \\
  \land E_1
\end{align*}
\]
A proof

\[
\frac{[B \& A]^z \quad \&-E_1}{A} \quad \frac{[B \& A]^z \quad \&-E_0}{B}
\]

\[\frac{A \& B \quad \&-I}{(B \& A) \supset (A \& B)}\]

\[\supset-I^z\]
Simplifying proofs

\[
\begin{array}{c}
[A]_x \\
\vdots \\
B \\
\overline{A \supset B} \quad \supset \text{-I}_x \\
A \supset B & A \\
\overline{B} \quad \supset \text{-E} \\
\Rightarrow & B \\
\vdots \\
A & B \\
\overline{A \& B} \quad \& \text{-I} \\
A \& B \\
\overline{A} \quad \& \text{-E}_0 \\
\Rightarrow & A \\
\end{array}
\]
Simplifying a proof

\[
\begin{align*}
&\frac{[B \& A]^z}{A} \quad \&-E_1 \\
&\frac{[B \& A]^z}{B} \quad \&-E_0 \\
&\frac{A \& B}{\&-I} \\
&\frac{(B \& A) \supset (A \& B)}{\supset-I^z} \\
&\frac{B \& A}{\&-I} \\
&\frac{B \& A}{\supset-E} \\
&\frac{A \& B}{\&-E} \\
\end{align*}
\]
Simplifying a proof

\[
\begin{align*}
\frac{[B \land A]^{z}}{A} & \quad \text{\&-E}_{1} \\
\frac{[B \land A]^{z}}{B} & \quad \text{\&-E}_{0} \\
\frac{(B \land A) \supset (A \land B)}{A \land B} & \quad \text{\&-I} \\
\frac{(B \land A) \supset (A \land B)}{B \land A} & \quad \text{\supset-I}^{z} \\
\frac{A \land B}{B \land A} & \quad \text{\&-I} \\
\frac{A \land B}{B \land A} & \quad \text{\&-I} \\
\frac{B \land A}{A} & \quad \text{\&-E}_{1} \\
\frac{B \land A}{B} & \quad \text{\&-E}_{0} \\
\frac{A \land B}{A \land B} & \quad \text{\&-I}
\end{align*}
\]
Simplifying a proof

\[ \frac{[B \land A]^z}{A} \quad \&\text{-}E_1 \]

\[ \frac{[B \land A]^z}{B} \quad \&\text{-}E_0 \]

\[ \frac{A \land B}{\quad \&\text{-}I} \]

\[ \frac{(B \land A) \supset (A \land B)}{\quad \supset\text{-}I^z} \]

\[ \frac{A \land B}{B \land A} \quad \supset\text{-}E \]

\[ \frac{[B]^y \quad [A]^x}{B \land A} \quad \&\text{-}I \]

\[ \frac{B \land A}{A} \quad \&\text{-}E_1 \]

\[ \frac{B \land A}{B} \quad \&\text{-}E_0 \]

\[ \frac{A \land B}{\quad \&\text{-}I} \]

\[ \frac{[B]^y \quad [A]^x}{B \land A} \quad \&\text{-}I \]

\[ \frac{B \land A}{A} \quad \&\text{-}E_1 \]

\[ \frac{B \land A}{B} \quad \&\text{-}E_0 \]

\[ \frac{A \land B}{\quad \&\text{-}I} \]
Part 5

Church’s Lambda Calculus
Alonzo Church (1903–1995)
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 $\lambda x[M]$; that is, if there is a formula $M$ such that $\lambda x[M]$ occurs in the given formula and the occurrence of $x$ in question is an occurrence in $\lambda x[M]$. 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
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

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

\[\Downarrow\]

\[\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle\]
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

\[\vdash [x : A]^x \quad \vdash u : B \quad \vdash \lambda x. u : A \supset B \quad \vdash s : A \supset B \quad \vdash t : A \quad \vdash st : B \]

\[\vdash s_0 : A \quad \vdash s_1 : B\]
A program

\[
\begin{align*}
&\quad \frac{[z : B \& A]^z}{z_1 : A} \quad \&\text{-E}_1 \\
\quad &\quad \frac{[z : B \& A]^z}{z_0 : B} \quad \&\text{-E}_0 \\
\quad &\quad \frac{\langle z_1, z_0 \rangle : A \& B}{\&\text{-I}} \\
\quad &\quad \frac{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)}{\supset\text{-I}^z}
\end{align*}
\]
Simplifying programs

\[
\begin{align*}
[x : A]^x & \quad : \\
\vdots & \\
\end{align*}
\]

\[
\begin{align*}
u : B & \quad \Rightarrow \quad \text{I}_x^x \\
\lambda x. u : A \supset B & \quad t : A \\
\hline
(\lambda x. u) t : B & \\
\vdash \text{-E} \quad \Rightarrow \quad u[t / x] : B
\end{align*}
\]

\[
\begin{align*}
\vdots & \\
t : A & \quad u : B \\
\hline
\langle t, u \rangle : A \& B & \quad \&-I \\
\vdash \&-E_0 \quad \Rightarrow \quad t : A
\end{align*}
\]
Simplifying a program

\[
\begin{align*}
[z : B \& A]^z 
& \quad \&-E_1 \quad \frac{z_1 : A}{z_1 : A} \\
& \\
[z : B \& A]^z 
& \quad \&-E_0 \quad \frac{z_0 : B}{z_0 : B} \\
& \\
& \quad \&-I \quad \frac{\langle z_1, z_0 \rangle : A \& B}{\langle z_1, z_0 \rangle : A \& B} \\
& \\
& \quad \exists-I^z \quad \frac{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)}{(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B} \\
& \\
& \quad \&-I \quad \frac{[y : B]^y \quad [x : A]^x}{\langle y, x \rangle : B \& A} \\
& \quad \exists-E \quad \frac{(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B}{(\lambda z. \langle z_1, z_0 \rangle) \langle y, x \rangle : A \& B}
\end{align*}
\]
Simplifying a program

\[
\begin{align*}
\frac{[z : B \& A]^z}{z_1 : A} & \quad \& - E_1 \\
\frac{[z : B \& A]^z}{z_0 : B} & \quad \& - E_0 \\
\hline
\frac{\langle z_1, z_0 \rangle : A \& B}{\lambda z. \langle z_1, z_0 \rangle : (B \& A) \supset (A \& B)} & \quad \& - I \\
\hline
\frac{\langle \lambda z. \langle z_1, z_0 \rangle \rangle \langle y, x \rangle : A \& B}{\supset - E} & \\
\hline
\frac{[y : B]^y [x : A]^x}{\langle y, x \rangle : B \& A} & \quad \& - I \\
\hline
\frac{\langle y, x \rangle : B \& A}{\langle y, x \rangle_1 : A} & \quad \& - E_1 \\
\frac{\langle y, x \rangle : B \& A}{\langle y, x \rangle_0 : B} & \quad \& - E_0 \\
\hline
\frac{\langle \langle y, x \rangle_1, \langle y, x \rangle_0 \rangle : A \& B}{\& - I}
\end{align*}
\]
Simplifying a program

\[
\begin{align*}
[z : B \& A]^z & \quad \&-E_1 \\
\quad z_1 : A & \quad \&-E_0 \\
\left< z_1, z_0 \right> : A \& B & \quad \&-I \\
\lambda z. \left< z_1, z_0 \right> : (B \& A) \supset (A \& B) & \quad \supset-I^z \\
\left( \lambda z. \left< z_1, z_0 \right> \right) \left< y, x \right> : A \& B & \quad \supset-E
\end{align*}
\]

\[
\begin{align*}
[y : B]^y \quad [x : A]^x & \quad \&-I \\
\left< y, x \right> : B \& A & \quad \&-I \\
\left< y, x \right>_1 : A & \quad \&-E_1 \\
\left< \left< y, x \right>_1, \left< y, x \right>_0 \right> : A \& B & \quad \&-I \\
\left< x, y \right> : A \& B & \quad \&-I
\end{align*}
\]
Part 6

The Curry-Howard Isomorphism
The Curry-Howard homeomorphism
Haskell Curry (1900–1982) / William Howard
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 worthwhile to publish them. They have been rearranged for easier reading, and some inessential corrections have been made.
1. Formulation of the sequent calculus

Let \( \mathsf{P(C)} \) denote positive implicational propositional logic. The prime formulae of \( \mathsf{P(C)} \) are propositional variables. If \( \alpha \) and \( \beta \) are formulae, so is \( \alpha \rightarrow \beta \). A sequent has the form \( \Gamma \rightarrow \beta \), where \( \Gamma \) is a (possibly empty) finite sequence of formulae and \( \beta \) is a formula. The axioms and rules of inference of \( \mathsf{P(C)} \) are as follows.

(1.1) Axioms: all sequents of the form 
\[ \alpha \rightarrow \alpha \]

(1.2) \[
\frac{\Gamma, \alpha \rightarrow \beta}{\Gamma \rightarrow \alpha \rightarrow \beta}
\]

(1.3) \[
\frac{\Gamma \rightarrow \alpha}{\Gamma, \Lambda \rightarrow \alpha \rightarrow \beta}
\]

(1.4) Thinning, permutation and contraction rules
2. Type symbols, terms and constructions

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

(2.1) Variables $X^\alpha, Y^\beta, \ldots$ are terms

(2.2) $\lambda$-abstraction: from $F^\beta$ get

$$\lambda X^\alpha.F^\beta \alpha \supset \beta .$$

(2.3) Application: from $G^\alpha \supset \beta$ and $H^\alpha$

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

- 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?
XML Query Language Demo

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.

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

```xml
<bib>
   where $b/publisher = "Addison-Wesley" and $b/@year > 1991
   return <book year="{$b/@year}">
     <title>{$b/title}</title>
   </book>
  </for>
</bib>
```
Oracle XQuery Prototype

Quering 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_020216, March 2002)
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