Mirifici Logarithmorum Canonis Descriptio

Peter Hancock

May 2016

MIRIFICI

Logarithmorum
Canonis deſcriptio,

Ejuſque uſus, in utraque
Trigonometria; ut etiam in
omni Logiſtica Mathematica,
Ampliſſimi, Facillimi, &
expeditiſſimi explicatio.

Authore ac Inventore,
IOANNE NEPERO,
Barone Merchiſtonii,
&c. Scoto.

EDINBURGI,
Ex officinâ ANDREÆ HART
Bibliopôlæ, CIↃ. DC. XIV.

Napier's Preface:

Since nothing is more tedious, fellow mathematicians, in the practice of the mathematical arts, than the great delays suffered in the tedium of lengthy multiplications and divisions, the finding of ratios, and in the extraction of square and cube roots and in which not only is there the time delay to be considered, but also the annoyance of the many slippery errors that can arise: I had therefore been turning over in my mind, by what sure and expeditious art, I might be able to improve upon these said difficulties. In the end after much thought, finally I have found an amazing way of shortening the proceedings, and perhaps the manner in which the method arose will be set out elsewhere: truly, concerning all these matters, there could be nothing more useful than the method that I have found.

. . .

Farewell.

Laws of logarithms

$$
\begin{aligned}
\log_a(b \times c) &= \log_a b + \log_a c \\
\log_a 1 &= 0 \\
\\
\log_a(b^c) &= (\log_a b) \times c \\
\log_a a &= 1
\end{aligned}
$$

Two monoids; $(\times, 1)$ (slippery!) *versus* $(+, 0)$ (expeditious!).

In the 'Descriptio', Napier's idea was (I think) to replace geometrical progression by arithmetical progression.

Interestingly, he never thought in terms of a 'base'. [1]

Anyway, he overlooked a few things ...

[1] But his was $(1/e)$ *acc.* Eli Maor: "e: the Story of a Number".

Böhm's laws

Baron J. Napier overlooked the logarithms of sums:

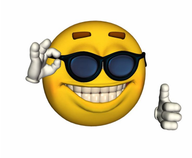
$$\begin{aligned} \log_x(\alpha + \beta) &= (,) + ((\log_x \alpha)^{(\wedge)} + (\log_x \beta)^{(\wedge)})^K \\ \log_x 0 &= 0^K \end{aligned}$$

Also, a more general form for the logarithm of a power:

$$\log_x(\alpha^\beta) = (\log_x \alpha) \times (,) + (\log_x \beta) \times (\wedge)$$

in the case when the 'base' x may occur in β.

I shall try to explain ...



An air of mystery

An air of straightforwardness

Arithmetical combinators

$$\begin{array}{lllllllll} A & a & b & c & d & \mapsto & b & c & (a\;c\;d) \\ M & a & b & c & & \mapsto & b & (a\;c) & \\ E & a & b & & & \mapsto & b & a & \\ N & a & b & & & \mapsto & b & & \end{array}$$

Infix notation:

$$\begin{aligned} a \wedge b &= b\,a \\ (a \times b)\,c &= (c \wedge a) \wedge b \\ (a + b)\,c &= (c \wedge a) \times (c \wedge b) \\ 0\,a\,b &= b \end{aligned}$$

or, writing arg^{fun} for application, 1 for identity:

$$\begin{array}{llll} c^{a \times b} = (c^a)^b & , & c^1 = c \\ c^{a+b} = c^a \times c^b & , & c^0 = 1 \end{array}$$

Essentially, Cantor's (Archimedes'?) Laws of Exponents.

'Ordinary' (boring) combinators

C	f	a	b	$\mapsto$	f	b	a		*swap*, `flip`, *interchange*, f^C
B	f	a	b	$\mapsto$	f	$(a\ b)$			*compose* $(\cdot)$
W	f	a		$\mapsto$	f	a	a		*duplicate*, `DUPL`, *contraction*
I	f			$\mapsto$	f				*identity*, `id`, `SKIP`, $=$ BCC
K	f	a		$\mapsto$	f				*weaken*, `const`, `POP`

$\{B, C\}$ linear, $\{B, C, K\}$ affine, $\{B, C, K, W\}$ general.
$\{E, M\}$ linear, $\{E, M, N\}$ affine, $\{A, M, E, N\}$ general.

Bracket abstraction

$$
\begin{array}{lcll}
[x] & a & = & K\ a \\
[x] & x & = & I \\
[x] & (a\ \beta) & = & B\ a\ ([x]\beta) \\
[x] & (\alpha\ b) & = & C\ ([x]\alpha)\ b \\
[x] & (\alpha\ \beta) & = & W\ (C\ B\ ([x]\alpha)\ (C\ B\ ([x]\beta)))
\end{array}
$$

Bracket abstraction

$$
\begin{array}{lccl}
[x] & a & = & K\ a \\
[x] & x & = & I \\
[x] & (a\ \beta) & = & B\ a\ ([x]\beta) \\
[x] & (\alpha\ b) & = & C\ ([x]\alpha)\ b \\
[x] & (\alpha\ \beta) & = & W\ (C\ B\ ([x]\alpha)\ (C\ B\ ([x]\beta)))
\end{array}
$$

You might want some explanation of the last bit!

Bracket abstraction

$$\begin{array}{llcl} [x] & a & = & K\ a \\ [x] & x & = & I \\ [x] & (a\ \beta) & = & B\ a\ ([x]\beta) \\ [x] & (\alpha\ b) & = & C\ ([x]\alpha)\ b \\ [x] & (\alpha\ \beta) & = & W\ (C\ B\ ([x]\alpha)\ (C\ B\ ([x]\beta))) \end{array}$$

You might want some explanation of the last bit!

$$\begin{array}{llcl} [x] & (\alpha\ \beta) & = & S\ ([x]\alpha)\ ([x]\beta) \\ & & = & W(S'\ ([x]\alpha)\ ([x]\beta)) \end{array}$$

Bracket abstraction

$$
\begin{array}{lccl}
[x] & a & = & K\ a \\
[x] & x & = & I \\
[x] & (a\ \beta) & = & B\ a\ ([x]\beta) \\
[x] & (\alpha\ b) & = & C\ ([x]\alpha)\ b \\
[x] & (\alpha\ \beta) & = & W\ (C\ B\ ([x]\alpha)\ (C\ B\ ([x]\beta)))
\end{array}
$$

You might want some explanation of the last bit!

$$
\begin{array}{lccl}
[x] & (\alpha\ \beta) & = & S\ ([x]\alpha)\ ([x]\beta) \\
 & & = & W(S'\ ([x]\alpha)\ ([x]\beta))
\end{array}
$$

$$
\begin{array}{lcl}
S'\ a\ b\ c_1\ c_2 & = & a\ c_1\ (b\ c_2) \\
S'\ a\ b\ c_1 & = & (a\ c_1) \cdot b \\
 & = & B\ (a\ c_1)\ b \\
 & = & C\ B\ b\ (a\ c_1) \\
S'\ a\ b & = & (C\ B\ b) \cdot a \\
 & = & B\ (C\ B\ b)\ a \\
 & = & C\ B\ a\ (C\ B\ b)
\end{array}
$$

Reducing BWICK to AMEN

$$
\begin{array}{llllllll|l}
C & a & b & c & = & a & c & b & b \wedge c \wedge a \\
 & & & & = & E & b & (a\ c) & (c \wedge a) \wedge (b\wedge) \\
C & a & b & & = & M & a & (E\ b) & a \times (b\wedge) \\
C & a & & & = & M & E & (M\ a) & (\wedge) \times (a\times) \\
C & & & & = & M & M & (M\ E) & (\times) \times ((\wedge)\times)
\end{array}
$$

Reducing BWICK to AMEN

C	a	b	c	$=$	a	c	b		$b \wedge c \wedge a$
				$=$	E	b	$(a\ c)$		$(c \wedge a) \wedge (b\wedge)$
C	a	b		$=$	M	a	$(E\ b)$		$a \times (b\wedge)$
C	a			$=$	M	E	$(M\ a)$		$(\wedge) \times (a\times)$
C				$=$	M	M	$(M\ E)$		$(\times) \times ((\wedge)\times)$
B				$=$	C	M			$(\wedge) \times ((\times)\times)$

Reducing BWICK to AMEN

$$
\begin{array}{llll|l}
C\ a\ b\ c & = & a\ c\ b & & b \wedge c \wedge a \\
 & = & E\ b\ (a\ c) & & (c \wedge a) \wedge (b\wedge) \\
C\ a\ b & = & M\ a\ (E\ b) & & a \times (b\wedge) \\
C\ a & = & M\ E\ (M\ a) & & (\wedge) \times (a\times) \\
C & = & M\ M\ (M\ E) & & (\times) \times ((\wedge)\times) \\
B & = & C\ M & & (\wedge) \times ((\times)\times) \\
K & = & C\ N & & (\wedge) \times (0\times)
\end{array}
$$

Reducing BWICK to AMEN

C	a	b	c	$=$	a	c	b		$b \wedge c \wedge a$
				$=$	E	b	$(a\ c)$		$(c \wedge a) \wedge (b\wedge)$
C	a	b		$=$	M	a	$(E\ b)$		$a \times (b\wedge)$
C	a			$=$	M	E	$(M\ a)$		$(\wedge) \times (a\times)$
C				$=$	M	M	$(M\ E)$		$(\times) \times ((\wedge)\times)$
B				$=$	C	M			$(\wedge) \times ((\times)\times)$
K				$=$	C	N			$(\wedge) \times (0\times)$
I				$=$	N	N			$0 \wedge 0 \quad (alt.\ (\wedge) \times ((\wedge)\times))$

Reducing BWICK to AMEN

$$
\begin{array}{lllllll|l}
C & a & b & c & = a & c & b & b \wedge c \wedge a \\
 & & & & = E & b & (a\ c) & (c \wedge a) \wedge (b \wedge) \\
C & a & b & & = M & a & (E\ b) & a \times (b \wedge) \\
C & a & & & = M & E & (M\ a) & (\wedge) \times (a \times) \\
C & & & & = M & M & (M\ E) & (\times) \times ((\wedge) \times) \\
B & & & & = C & M & & (\wedge) \times ((\times) \times) \\
K & & & & = C & N & & (\wedge) \times (0 \times) \\
I & & & & = N & N & & 0 \wedge 0 \quad (alt.\ (\wedge) \times ((\wedge) \times)) \\
W & & & & = C & W^C & & (\wedge) \times (((\wedge) + (\wedge)) \times)
\end{array}
$$

Reducing BWICK to AMEN

C	a	b	c	$=$	a	c	b	$b \wedge c \wedge a$
				$=$	E	b	$(a\ c)$	$(c \wedge a) \wedge (b\wedge)$
C	a	b		$=$	M	a	$(E\ b)$	$a \times (b\wedge)$
C	a			$=$	M	E	$(M\ a)$	$(\wedge) \times (a\times)$
C				$=$	M	M	$(M\ E)$	$(\times) \times ((\wedge)\times)$
B				$=$	C	M		$(\wedge) \times ((\times)\times)$
K				$=$	C	N		$(\wedge) \times (0\times)$
I				$=$	N	N		$0 \wedge 0 \quad (alt.\ (\wedge) \times ((\wedge)\times))$
W				$=$	C	W^C		$(\wedge) \times (((\wedge) + (\wedge))\times)$
W^C	a	f		$=$	f	a	a	$a \wedge a \wedge f$
				$=$	E	a	$(E\ a\ f)$	$f \wedge ((a\wedge) \times (a\wedge))$
W^C				$=$	A	E	E	$(\wedge) + (\wedge)$

$$K = 0^C \qquad I = (\wedge)^C \qquad W = ((\wedge) + (\wedge))^C$$

Linear and affine logarithms

The following are easily verified, where $a, b, b_1, \ldots b_k$ are expressions that do not contain any occurrences of the variable x.

$$\begin{aligned}
\log_x x &= 1 \\
\log_x (a\,x\,b) &= \log_x (b \wedge x \wedge a) \\
&= \log_x ((x \wedge a) \wedge (b^{\wedge})) \\
&= a \times (b^{\wedge}) \\
\log_x (a\,x\,b_1 \,..\, b_k) &= a \times (b_1^{\wedge}) \times .. \times (b_k^{\wedge}) \\
\log_x (a\,x\,x) &= ((\wedge) + (\wedge)) \times (a^{\wedge}) \\
\log_x a &= 0 \times (a^{\wedge}) = a^K
\end{aligned}$$

General linear logarithms can be put in the form:

$$\prod_{i=1}^{n} (a_i \times (\prod_{j=1}^{k_i} (b_{i,j}^{\wedge})))$$

Logarithms of exponentials

The trick is to use a standard pairing combinator (,).

$$(,)\, a\, b\, c = (a, b)\, c = c\, a\, b$$

so we can express exponentiation generally as a sum of constant powers.

$$\begin{array}{lll} a\, b\, c & = a \wedge (b, c) & \text{hence} \\ a\, b & = (a^{\wedge}) \cdot (b^{(,)}) & \text{ie.} \\ b \wedge a & = (b^{(,)}) \times (a^{\wedge}) & \end{array}$$

now (α-convert and) take logs. (We already know how to compute logarithms of constant powers.)

$$\log_x(\alpha^\beta) \quad = (\log_x \alpha) \times (,) + (\log_x \beta) \times (\wedge)$$

Logarithms of sums

The trick here is to use currying (easily expressible arithmetically) to switch between two arguments for a function and one

$$\text{curry}\, f\ x\ y = f(x, y)$$

In fact, $\text{curry}\,(a^{\wedge}) = a$, so that $(\wedge) \times \text{curry} = 1$. They $((\wedge)$ and curry) are (near-semi?) reciprocal.

$$\log_x(\alpha + \beta) \quad = \text{curry}((\log_x \alpha)^{(\wedge)} + (\log_x \beta)^{(\wedge)})$$

The proof is (I'm afraid) a longish, but straightforward verification. Next page if anyone wants to see it.

Verification of formula for log-of-sum

$$
\begin{array}{ll}
\text{curry}((\log_x \alpha)^{(\wedge)} + (\log_x \beta)^{\wedge})\, x\, y & = \\
((\log_x \alpha)^{(\wedge)} + (\log_x \beta)^{\wedge})\,(x, y) & = \\
(x, y) \wedge (\log_x \alpha) \wedge (\wedge) \times (x, y) \wedge (\log_x \beta) \wedge (\wedge) & = \\
(\log_x \alpha) \wedge (x, y) \times (\log_x \beta)(x, y) & = \\
(\log_x \alpha)\, x\, y \times (\log_x \beta)\, x\, y & = \\
y \wedge ((x \wedge \log_x \alpha) + (x \wedge \log_x \beta))
\end{array}
$$

Now apply the ζ rule of ‘exponentiality’.

Postlude: algebra

Use: 'ζ-rule': where x not free in a, b,

$$\frac{a\ x = b\ x}{a = b}$$

Near-semi-ri(n)g. Like arithmetic of transfinite ordinals (BUT without $1^\alpha = 1$, or $0 \times \alpha = 0$).

$$\begin{aligned}
&\alpha \times 1 = \alpha = 1 \times \alpha \\
&\alpha \times (\beta \times \gamma) = (\alpha \times \beta) \times \gamma \\
&\alpha + 0 = \alpha = 0 + \alpha \\
&\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma \\
&\alpha \times (\beta + \gamma) = \alpha \times \beta + \alpha \times \gamma \\
&\alpha \times 0 = 0
\end{aligned}$$

Postlude: algebra

Use: 'ζ-rule': where x not free in a, b,

$$\frac{a\,x = b\,x}{a = b}$$

Near-semi-ri(n)g. Like arithmetic of transfinite ordinals (BUT without $1^\alpha = 1$, or $0 \times \alpha = 0$).

$$\begin{aligned}
&\alpha \times 1 = \alpha = 1 \times \alpha \\
&\alpha \times (\beta \times \gamma) = (\alpha \times \beta) \times \gamma \\
&\alpha + 0 = \alpha = 0 + \alpha \\
&\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma \\
&\alpha \times (\beta + \gamma) = \alpha \times \beta + \alpha \times \gamma \\
&\alpha \times 0 = 0
\end{aligned}$$

Postlude: algebra

Use: 'ζ-rule': where x not free in a, b,

$$\frac{a\,x = b\,x}{a = b}$$

Near-semi-ri(n)g. Like arithmetic of transfinite ordinals (BUT without $1^\alpha = 1$, or $0 \times \alpha = 0$).

$$\begin{aligned} \alpha \times 1 &= \alpha = 1 \times \alpha \\ \alpha \times (\beta \times \gamma) &= (\alpha \times \beta) \times \gamma \\ \alpha + 0 &= \alpha = 0 + \alpha \\ \alpha + (\beta + \gamma) &= (\alpha + \beta) + \gamma \\ \alpha \times (\beta + \gamma) &= \alpha \times \beta + \alpha \times \gamma \\ \alpha \times 0 &= 0 \end{aligned}$$

C. Böhm "Combinatory foundation of functional programming" published ACM 1982

"Notion of zero" – closed under 0, $(+)$, $(a\times)$, $(\times a)$

eg c^K. (*An "ideal" ?*)

. . . Farewell

> *"Indeed I await the judgment and censure of the learned men . . . perhaps rashly, to be examined in the light of envious disparagement."*

John Napier, Baron of Merchiston, *Descriptio*, CH II.

Example: (,), the pairing combinator

$$
\begin{aligned}
(,)\, a\, b\, c &= c\, a\, b & &= b \wedge a \wedge c \\
(,)\, a\, b &= (b^{(\wedge)}) \cdot (a^{(\wedge)}) & &= (a^{(\wedge)}) \times (b^{(\wedge)}) \\
(,)\, a &= ((a^{(\wedge)})^{(\times)}) \cdot (\wedge) & &= (\wedge) \times ((a^{(\wedge)})^{(\times)}) \\
(,) &= ((\wedge)^{(\times)}) \cdot ((\wedge) \times (\times)) & &= (\wedge) \times (\times) \times (\wedge)^{(\times)}
\end{aligned}
$$

Example: curry, the currying combinator

$$\begin{aligned}
\mathsf{curry}\, f\, x\, y &= f((,)\, x\, y) \\
\mathsf{curry}\, f\, x &= f \cdot ((,)\, x) \\
\mathsf{curry}\, f &= (f^{(\cdot)}) \cdot (,) &&= (,) \times (f^{(\cdot)}) \\
\mathsf{curry} &= (,)^{(\times)} \cdot (\cdot) &&= (\wedge) \times (\times)^{(\times)} \times (,)^{(\times)}
\end{aligned}$$

Example: curry, the currying combinator

$$
\begin{aligned}
\text{curry}\, f\, x\, y &= f\,((,)\, x\, y) \\
\text{curry}\, f\, x &= f \cdot ((,)\, x) \\
\text{curry}\, f &= (f^{(\cdot)}) \cdot (,) &&= (,) \times (f^{(\cdot)}) \\
\text{curry} &= (,)^{(\times)} \cdot (\cdot) &&= (\wedge) \times (\times)^{(\times)} \times (,)^{(\times)}
\end{aligned}
$$

There are other definitions. One such is curry $f = (,) + f^K$, from which we get curry $= K \times (,)^{(+)}$.

However curry is linear, and should not need any additive apparatus like K or $(+)$.

An example: 'linear' S

$$
\begin{array}{lclcl}
S' \; a \; b \; c_1 \; c_2 & = & a \; c_1 \; (b \; c_2) & & \\
S' \; a \; b \; c_1 & = & a \; c_1 \; \cdot \; b & = & b \times (a \; c_1) \\
S' \; a \; b & = & (b\times) \cdot a & = & a \times (b\times) \\
S' \; a & = & (a\times) \cdot (\times) & = & (\times) \times (a\times) \\
S' & = & ((\times)\times) \cdot (\times) & = & (\times) \times ((\times)\times)
\end{array}
$$

Weirdly, $C = S'(\wedge)$, $S' = S'(\times)$.

An example: 'linear' S

$$\begin{array}{lclcl}
S' \; a \; b \; c_1 \; c_2 & = & a \; c_1 \; (b \; c_2) & & \\
S' \; a \; b \; c_1 & = & a \; c_1 \; \cdot \; b & = & b \times (a \; c_1) \\
S' \; a \; b & = & (b\times) \cdot a & = & a \times (b\times) \\
S' \; a & = & (a\times) \cdot (\times) & = & (\times) \times (a\times) \\
S' & = & ((\times)\times) \cdot (\times) & = & (\times) \times ((\times)\times)
\end{array}$$

Weirdly, $C = S'(\wedge)$, $S' = S'(\times)$.
$S \; a \; b = (S' \; a \; b)^W = (a \times (b\times)) \wedge ((\wedge) + (\wedge))^C$

An example: 'linear' S

$$
\begin{array}{lllll}
S' \; a \; b \; c_1 \; c_2 & = & a \; c_1 \; (b \; c_2) & & \\
S' \; a \; b \; c_1 & = & a \; c_1 \; \cdot \; b & = & b \times (a \; c_1) \\
S' \; a \; b & = & (b\times) \cdot a & = & a \times (b\times) \\
S' \; a & = & (a\times) \cdot (\times) & = & (\times) \times (a\times) \\
S' & = & ((\times)\times) \cdot (\times) & = & (\times) \times ((\times)\times)
\end{array}
$$

Weirdly, $C = S'(\wedge)$, $S' = S'(\times)$.

$S \; a \; b = (S' \; a \; b)^W = (a \times (b\times)) \wedge ((\wedge) + (\wedge))^C$

$S = (\times) \times ((\times)\times) \times (\times) \times ((\wedge) + (\wedge)) \wedge ((\times) \times ((\wedge)\times) \times (\wedge))$

More examples: fixpoint combinators

$\mathsf{sap} = (\wedge)^W, \quad$ (or 1^W for that matter)

More examples: fixpoint combinators

$\mathsf{sap} = (\wedge)^W$, (or 1^W for that matter)

$$\begin{aligned} Y_{\mathsf{Curry}} &= \lambda f.\,(\lambda x.\, f(x\, x))(\lambda x.\, f(x\, x)) \\ &= \lambda f.\,\mathsf{sap}(f \cdot \mathsf{sap}) \\ &= \lambda f.\,\mathsf{sap}(\mathsf{sap} \times f) \\ &= (\mathsf{sap}\times) \times \mathsf{sap} \\ &= \mathsf{sap} \wedge ((\times) + 1) \end{aligned}$$

More examples: fixpoint combinators

$\mathsf{sap} = (\wedge)^W$, (or 1^W for that matter)

$$\begin{aligned} Y_{\mathsf{Curry}} &= \lambda f.\,(\lambda x.\,f(x\;x))(\lambda x.\,f(x\;x))) \\ &= \lambda f.\,\mathsf{sap}(f \cdot \mathsf{sap}) \\ &= \lambda f.\,\mathsf{sap}(\mathsf{sap} \times f) \\ &= (\mathsf{sap}\times) \times \mathsf{sap} \\ &= \mathsf{sap} \wedge ((\times) + 1) \end{aligned}$$

$Y_{\mathsf{Turing}} = T \wedge T = T \wedge \mathsf{sap}$ where $T\;x\;y = y(x\;x\;y)$.

$$\begin{aligned} T^C\;y\;x &= y(\mathsf{sap}\;x\;y) &&= y(\mathsf{sap}^C\;y\;x) &&= (y \cdot (\mathsf{sap}^C\;y))x \\ T^C\;y &= (y \wedge \mathsf{sap}^C) \times y &&= y \wedge (\mathsf{sap}^C + 1) \\ T &= (\mathsf{sap}^C + 1)^C \end{aligned}$$

More examples: fixpoint combinators

$\text{sap} = (\wedge)^W$, (or 1^W for that matter)

$$\begin{aligned} Y_{\text{Curry}} &= \lambda f.\,(\lambda x.\,f(x\;x))(\lambda x.\,f(x\;x)) \\ &= \lambda f.\,\text{sap}(f \cdot \text{sap}) \\ &= \lambda f.\,\text{sap}(\text{sap} \times f) \\ &= (\text{sap}\times) \times \text{sap} \\ &= \text{sap} \wedge ((\times) + 1) \end{aligned}$$

$Y_{\text{Turing}} = T \wedge T = T \wedge \text{sap}$ where $T\;x\;y = y(x\;x\;y)$.

$$\begin{aligned} T^C\;y\;x &= y(\text{sap}\;x\;y) &&= y(\text{sap}^C\;y\;x) &&= (y \cdot (\text{sap}^C\;y))x \\ T^C\;y &= (y \wedge \text{sap}^C) \times y &&= y \wedge (\text{sap}^C + 1) \\ T &= (\text{sap}^C + 1)^C \end{aligned}$$

So $Y_{\text{Turing}} = (\text{sap}^C + 1)^{C \times \text{sap}}$.

Notation (Sorry!)

Operators

$$
\begin{aligned}
A &\rightarrow (+) \\
M &\rightarrow (\times) \\
E &\rightarrow (\wedge) \\
N &\rightarrow \ 0
\end{aligned}
$$

Section notation

$$
\begin{array}{ccccccccc}
(\wedge)\ a\ b & = & (a\wedge)\ b & = & a \wedge b & = & b \wedge a \wedge (\wedge) & = & b \wedge (a\wedge) \\
(\times)\ a\ b & = & (a\times)\ b & = & a \times b & = & b \wedge a \wedge (\times) & = & b \wedge (a\times) \\
(+)\ a\ b & = & (a+)\ b & = & a + b & = & b \wedge a \wedge (+) & = & b \wedge (a+)
\end{array}
$$

Laws of exponents

$$\begin{array}{rcl|rcl} \alpha^{\beta\times\gamma} & = & (\alpha^{\beta})^{\gamma} & \alpha^{1} & = & \alpha \\ \alpha^{\beta+\gamma} & = & \alpha^{\beta}\times\alpha^{\gamma} & \alpha^{0} & = & 1 \end{array}$$

Laws of exponents

$$\begin{array}{rcl|rcl} \alpha^{\beta\times\gamma} & = & (\alpha^{\beta})^{\gamma} & \alpha^{1} & = & \alpha \\ \alpha^{\beta+\gamma} & = & \alpha^{\beta}\times\alpha^{\gamma} & \alpha^{0} & = & 1 \end{array}$$

$$\begin{array}{rcl|rcl} \alpha\wedge(\beta\times\gamma) & = & (\alpha\wedge\beta)\wedge\gamma & \alpha\wedge 1 & = & \alpha \\ \alpha\wedge(\beta+\gamma) & = & (\alpha\wedge\beta)\times(\alpha\wedge\gamma) & \alpha\wedge 0 & = & 1 \end{array}$$

Laws of exponents

$$
\begin{array}{rcl|rcl}
\alpha^{\beta\times\gamma} & = & (\alpha^{\beta})^{\gamma} & \alpha^{1} & = & \alpha \\
\alpha^{\beta+\gamma} & = & \alpha^{\beta}\times\alpha^{\gamma} & \alpha^{0} & = & 1
\end{array}
$$

$$
\begin{array}{rcl|rcl}
\alpha\wedge(\beta\times\gamma) & = & (\alpha\wedge\beta)\wedge\gamma & \alpha\wedge 1 & = & \alpha \\
\alpha\wedge(\beta+\gamma) & = & (\alpha\wedge\beta)\times(\alpha\wedge\gamma) & \alpha\wedge 0 & = & 1
\end{array}
$$

$$
\mathsf{arg}\wedge\mathsf{fun} \quad = \quad \mathsf{fun}(\mathsf{arg})
$$

Laws of exponents

$$\begin{array}{rcl|rcl} \alpha^{\beta\times\gamma} & = & (\alpha^{\beta})^{\gamma} & \alpha^{1} & = & \alpha \\ \alpha^{\beta+\gamma} & = & \alpha^{\beta}\times\alpha^{\gamma} & \alpha^{0} & = & 1 \end{array}$$

$$\begin{array}{rcl|rcl} \alpha\wedge(\beta\times\gamma) & = & (\alpha\wedge\beta)\wedge\gamma & \alpha\wedge 1 & = & \alpha \\ \alpha\wedge(\beta+\gamma) & = & (\alpha\wedge\beta)\times(\alpha\wedge\gamma) & \alpha\wedge 0 & = & 1 \end{array}$$

$$\mathsf{arg}\wedge\mathsf{fun} \quad = \quad \mathsf{fun}(\mathsf{arg})$$

$$\begin{array}{rcl|rcl} \beta\times\gamma & = & \lambda\,\alpha.\,(\alpha\wedge\beta)\wedge\gamma & 1 & = & \lambda\,\alpha.\,\alpha \\ \beta+\gamma & = & \lambda\,\alpha.\,(\alpha\wedge\beta)\times(\alpha\wedge\gamma) & 0 & = & \lambda\,\alpha.\,1 \end{array}$$

Exponents, λogarithms

$$
\begin{array}{ll}
\alpha \wedge (\beta + \gamma) = (\alpha \wedge \beta) \times (\alpha \wedge \gamma) & \alpha \wedge 0 = 1 \\
\alpha \wedge (\beta \times \gamma) = (\alpha \wedge \beta) \wedge \gamma & \alpha \wedge 1 = \alpha \\
\log_\alpha (B \times C) = \log_\alpha B + \log_\alpha C & \log_\alpha 1 = 0 \\
\log_\alpha (B \wedge \gamma) = (\log_\alpha B) \times \gamma & \log_\alpha \alpha = 1
\end{array}
$$

C. Böhm: "Un modèle arithmétique des termes de la logique combinatoire."
in "Lambda Calcul et Sémantique Formelle des Langages de Programmation – Actes de la Sixième Ecole de Printemps d'Informatique Theorique, La Châtre, 1978".

Oh, mega!

$$R(b, a) = (a, b(0, a), b(1, b(0, a)), \ldots, c_{n+1} = b(n, c_n), \ldots)$$

$$\Pi(b) = (1, b(0), b(0) \times b(1), \ldots, c_{n+1} = b(0) \times \ldots \times b(n), \ldots)$$

$$\Sigma(b) = (0, b(0), b(0) + b(1), \ldots, c_{n+1} = b(0) + \ldots + b(n), \ldots)$$

$$R = \Sigma(\mathrm{hd}, \mathrm{tl} \times \mathrm{hd}, \mathrm{tl}^2 \times \mathrm{hd}, \ldots)$$

$$\omega = \Sigma(1, 1, \ldots) = \Sigma(1!)$$

$$f^\omega = (1, f, f^2, f^3, \ldots)$$