T. Scott (Ottawa). Lecture II

Natural Numbers & Lists in CCC's

A fund. data type in CS is $(\mathbb{N}, 0, S)$, where $S(n) = n+1$. $0 \in \mathbb{N}$. It satisfies (in Set)

(*)
$$
\begin{tikzcd}
1 \arrow[r, "0"] \arrow[dr, "a"'] & \mathbb{N} \arrow[r, "S"] \arrow[d, dotted] \arrow[dl, phantom, "\equiv"] \arrow[dr, phantom, "\equiv"] & \mathbb{N} \arrow[d, dotted, "I_{ah}"] \\
 & A \arrow[r, "h"] & A
\end{tikzcd}
$$

iterator

$$
\left.\begin{aligned} I_{ah}(0) &= a \\ I_{ah}(n+1) &= h(I_{ah}(n)) \end{aligned}\right\} \text{simple iteration}
$$

An NNO is an object N & arrows $1 \xrightarrow{0} N \xrightarrow{S} N$ s.t. $\forall A$ $\forall\, 1 \xrightarrow{a} A \xrightarrow{h} A$, $\exists!\, I_{ah} : N \to A$ s.t. (*)

Similarly, given $A \in |\mathcal{C}|$ of ccc $\mathcal{C}$, define list(A) as follows:

$nil : 1 \to list(A)$

$cons : A \times list(A) \to list(A)$

satisfying: $\forall\, 1 \xrightarrow{b} B$ $\forall\, A \times B \xrightarrow{h} B$, $\exists\, I_{bh} : list(A) \to B$ satisfying

$$I_{bh}\, nil = b$$

$$I_{bh}\, cons(a, w) = h(a, I_{bh}\, w)$$

in the associated λ-calculus ("internal lang").

For both cases above, using Cartesianness, we can add extra parameters to NNO's and lists.

A weak NNO is like an N.N.O. but we assume $\exists$ but not uniqueness of Φ_{ah}.

Similarly, we can define simply typed λ-calculus with weak NNO (iterators)

Types: $A = 1 \mid N \mid A_1 \times A_2 \mid A_2^{A_1}$

Terms: as before, but add constants $0 : N$, $S : N^N$ and $I_A : A^{A \times A^A \times N}$ satisfying eq^{n}s:

$$I_A\langle a, h, 0\rangle = a$$
$$I_A\langle a, h, Sn\rangle = h' I_A\langle a, h, n\rangle$$

where $h : A^A$, $a : A$, etc.

Uniqueness: usually specified by a rule: for any $f : N \to A$ s.t. $\left.\begin{array}{l} f0 = a \\ fS = hf \end{array}\right]$ then $f = I_{ah}$.

Thm (Lambek): In any ccc s.t. each type has a Mal'cev operator $\left(m_A : A^3 \to A \text{ s.t. } \begin{array}{l} mxyy = x \\ mxxy = y \end{array} \right)$ it is possible to eq^{n}lly specify being a (strong) NNO in terms of the $\{m_A\}$'s. In particular, this is true for the free ccc with weak NNO, $\mathcal{F}_{\mathcal{X}}$

(e.g. let $m_N\, xyz = (x+z) \dot{-} y$ then generate inductively $\{m_A\}_{A \text{ Type}}$)

The prim rec. f$^{\underline{n}}$s = smallest class of f$^{\underline{n}}$s $\mathcal{P}$ on $\mathbb{N}$ in <u>Set</u>

(i) containing $\lambda x.0$, S

(ii) containing $\pi^n_i = \lambda x_1 \dots x_n . x_i$

(iii) closed under general composition: $h(b_1, \dots, b_n) \in \mathcal{P}$ and $g_i(x_1, \dots, x_k) \in \mathcal{P} \Rightarrow h(g_1(\vec{x}), \dots, g_n(\vec{x})) \in \mathcal{P}$

(iv) closed under prim. recursion: $g(a)$, $h(n,m,a) \in \mathcal{P}$ then $f(n,a) \in \mathcal{P}$, where $f(0,a) = g(a)$, $f(0, n+1) = h(n, f(n,a), a)$

---

f$^{\underline{n}}$ $f : \mathbb{N}^k \to \mathbb{N}$ is <u>representable</u> (in ccc or λ-calc. with N) if $\exists$: $N^k \to N$ s.t. $F\langle \bar{n}_1, \dots, \bar{n}_k \rangle = \overline{f(n_1 \dots n_k)}$, where $\bar{n} = S^n 0$.

<u>Thm</u>: In free ccc with N,

① all prim. rec. f$^{\underline{n}}$s are representable

② The Ackermann f$^{\underline{n}}$ is representable

For a proof, see Lambek-Scott.

So, since Ackermann is not prim. recursive, we get a <u>proper</u> subset of total recursive f$^{\underline{n}}$s which <u>properly</u> <u>includes</u> prim. rec. f$^{\underline{n}}$s. It is the class of ϵ_0-recursive functions (= provably total f$^{\underline{n}}$s of first-order Peano arithmetic.)

More generally, Lambek studied free categories with structure gen. by proof theory & used Cut-Elimination of Gentzen to get normal forms of proofs (Cut-Elim is a version of <u>normalization</u>, weaker than SN, of proof terms)

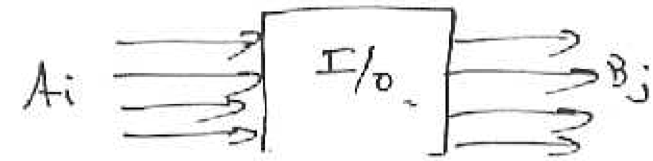
This work, more generally, uses Gentzen's <u>sequent</u> <u>calculus</u> (rather than natural ded$^{\underline{n}}$).

Sequents have form

$$\Gamma \vdash \Delta$$

where Γ, Δ lists of formulas.

We think of $\Gamma \vdash \Delta$ where $\Gamma = \{A_1, \ldots, A_m\}$, $\Delta = \{B_1, \ldots, B_n\}$ as an I/O Box



In general frameworks, we interpret categorically

$$I(\Gamma \vdash \Delta)$$

as arrows $\otimes\Gamma \longrightarrow \wp\Delta$ in appropriate $\otimes$, co-tensor categories (e.g. in linear logic)

e.g. Traditionally (Gentzen, Tarski) $\Gamma \vdash \Delta$ means $A_1 \wedge \cdots \wedge A_m \vdash B_1 \vee \cdots \vee B_n$.

Model Theory

Let $\mathcal{C}$ = some doctrine
(= category of structured categories & structure-preserving functors)

Suppose $\mathcal{F}_{\mathcal{X}}$ = a free category in $\mathcal{C}$. This means: $\forall \mathcal{A}$, $\forall$ interpretation of the generators $\mathcal{X} \longrightarrow |\mathcal{A}|$, $\exists!$ extension

$$\begin{tikzcd} \mathcal{F}_{\mathcal{X}} \arrow[r, "\exists! [\![\ ]\!]_I"] & \mathcal{A} \\ \mathcal{X} \arrow[ur, "I"'] & \end{tikzcd}$$

This is an $\mathcal{A}$-valued model of $\mathcal{F}_{\mathcal{X}}$

Thinking of $\mathcal{F}_{\mathcal{X}}$ as "syntax", the modelling is the interpretation $\mathcal{F}_{\mathcal{X}} \xrightarrow{[\![\ ]\!]_I} \mathcal{A}$.

A trivial model, e.g. would be $\mathcal{F}_{\mathcal{X}} \xrightarrow{id} \mathcal{F}_{\mathcal{X}}$, which does nothing!

We are interested in more "semantic" models $\mathcal{A}$ and modellings $[\![-]\!]$ preserving appropriate structure for the Doctrine in question.

A nice semantic category is **Set**. Is set-theoretic reasoning complete for simply typed λ-calculus? Well, Čubrić gave a generalizat^n of Friedman's Completeness Theorem (essentially extending Friedman to free ccc's with infinitely many indeterminates. This was highly non-trivial, using η-expansionary rewriting)

<u>Thm</u> (Čubrić). Let $\mathcal{F}_{\mathcal{X}}$ = free ccc gen by set $\mathcal{X}$ of atomic types. Then $\mathcal{F}_{\mathcal{X}}$ has a faithful represent^n into <u>Set</u>.

e.g. (Friedman). Let $\mathcal{X} = \{A\}$ (a single type). If A is interpreted as an ∞ set, $\vdash t_1 = t_2 : B$ in typed λ-calc iff $[\![t_1]\!] = [\![t_2]\!]$ (in Set).

<u>Full Completeness</u>

A recently important area is <u>full completeness</u> – modellings $\mathcal{F}_{\mathcal{X}} \xrightarrow{[\![\ ]\!]} \mathcal{A}$ which are full & faithful. In such models, maps $[\![A]\!] \xrightarrow{f} [\![B]\!]$ in $\mathcal{A}$ are the image of (unique) "proofs" $A \vdash^{\pi} B \in \mathcal{F}_{\mathcal{X}}$

This is studied by many workers in Linear Logic, e.g. w.r.t. games models, and many purely mathematical ones.

Moving now from propositional logics to more powerful logics, we can ask

- What is the meaning of (intuitionistic) first-order logic ?

- higher-order λ-calculi (proof terms for higher-order logics) ?

We begin with Girard's system $\mathcal{F}$ & $\mathcal{F}_\omega$ of polymorphic λ-calculi, then end with Lawvere's seminal notion of hyperdoctrine & quantifiers as adjoints, which influenced all of categorical logic.

"The perplexing subject of polymorphism"

C. Darwin, 1887

C. Strachey : late 1960's

Monomorphic Languages: Values & Vbls. have only one type.

Polymorphic Languages: Values & Vbls may range over many types

ad hoc
- Overloading
- Coercion

genuine
- parametric
- implicit
- inclusion (subtyping)

Parametric Polymorphism:

- Consider Sorting algorithms in PASCAL. Must declare types at beginning (even if going to use same algorithm for many types)

Polymorphism: "Uniform algorithms". Apply these to type-parameter A; get <u>Algorithm-at-A</u>.

"Parametric".

Generic Algorithms

<u>2nd Order</u> <u>Polymorphic</u> <u>λ-Calculus</u>

1972 1. Girard

2. Reynolds 1974

- 2nd Order Propositional Calculus

<u>Formulas</u> (= Types)

- (Propositional) Vbls: $\alpha, \beta, \ldots\ldots$
- $\sigma \Rightarrow \tau$, $\sigma \wedge \tau$
- $\forall \alpha\, A(\alpha)$, where $A(\alpha)$ formula

<u>e.g.</u>'s

$\forall \alpha . \alpha$

$\forall \alpha (\alpha \Rightarrow \alpha)$

$\forall \alpha (\alpha \wedge \beta \Rightarrow \alpha)$

<u>Proofs</u> (= <u>Terms</u>)

① $\wedge, \Rightarrow$: usual typed λ-calc.

② <u>$\forall$-introd.</u>

$$\begin{array}{c} C \\ \vdots \\ \underline{t_\alpha : B(\alpha)}\,{}^{(*)} \\ \Lambda\alpha . t_\alpha : \forall\alpha\, B(\alpha) \end{array}$$

(*) proviso

③ <u>$\forall$-Elimination</u> :

$$\begin{array}{c} \vdots \\ \underline{t : \forall\alpha\, B(\alpha)} \\ t\{p\} : B(p) \end{array} \qquad \text{any } p$$

<u>Rules</u> :

- $\wedge$: As for ordinary typed λ-
- β, η for both λ, Λ :

<u>1st order</u>: ① $\lambda x{:}A\, \varphi(x)\,{}^{\backprime}a = \varphi(a/x)$

② $\lambda x{:}A\,(f\,{}^{\backprime}x) = f \qquad x \notin FV(f)$

<u>2nd Order</u>: ③ $(\Lambda\alpha . t)\{\sigma\} = t(\sigma/\alpha)$

④ $\Lambda\alpha(t\{\alpha\}) = t \qquad \alpha \notin FV(t)$

③ says : there's a bijection

$$\frac{P \xrightarrow{\;t\;} A(\sigma)}{\overline{P \xrightarrow[\Lambda\alpha t]{} \forall\alpha\, A(\alpha)}} \qquad \begin{array}{c} S\{\sigma\} \\ \uparrow \\ S \end{array}$$

<u>PolyBoole</u> : $\forall\alpha(\alpha \Rightarrow (\alpha \Rightarrow \alpha))$

For each α, consider the "projections"

$T = \lambda x{:}\alpha\ \lambda y{:}\alpha\ .\ x$

$F = \lambda x{:}\alpha\ \lambda y{:}\alpha\ .\ y$

Now Λ-abstract over α.

In general, common data types (carriers of free algebras) are syntactically definable in Poly-λ : Bin-Nat, List-Nat, ...

But poly λ- has some strange features :

e.g. We can "type" $x'x$: (i.e. after erasing types, get $x'x$) :

Suppose $x : \forall\alpha(\alpha \Rightarrow \alpha)$

$x[B] : B \Rightarrow B$

Let $B = \forall\alpha(\alpha \Rightarrow \alpha)$

$x[B] : \underbrace{\forall\alpha(\alpha \Rightarrow \alpha)}_{x} \Rightarrow \forall\alpha(\alpha \Rightarrow \alpha)$

$(x[B])'x : B$

e.g. twice : $\forall\alpha((\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha))$

twice $\{\beta\}$: $(\beta \Rightarrow \beta) \rightarrow (\beta \Rightarrow \beta)$

$f \longmapsto f \circ f$

<u>Challenge</u> : type Twice 'Twice

Q: Can we type all untyped λ-terms ?

A : NO!

Thm (Girard '71) :

- Poly λ is SN & CR.
- Representable $f^{n}s$:

<u>Poly-Int</u> $\longrightarrow$ <u>Poly-Int</u> are Provably recursive (total) $f^{n}s$ of 2nd Order Peano Arith.

<u>Advantages : typed SN langs.</u>

(1) All programs halt & often embody their own termination proofs.

(2) Decidable eq^{n}al theory & type checking (correctness proofs)

(3) Elegant programming style

<u>Disadvantages</u> :

(1). Programs get larger

(2). Not universal : some recursive (total) $f^{n}s$ missing.

(3). "Decidable" typechecking not feasibly computable

(4). Replace loops & fixpts by ∞ family of "iterators".

Martin-Löf Type Theory

Ultimate form of propositions-as-types : propositions = sets = types.

$\boxed{\text{Types}}$

- Formulas of first-order-logic

- $\boxed{\text{Terms}}$

- "proofs", as in Heyting Interpretat^n.

$\Sigma_{x:A} B(x)$ [= Sigma types]

$\langle a,b\rangle : \Sigma_{x:A} B(x)$ should mean:

$a \in A$ and $b \in B[a/x]$.

- $\Pi_{x:A} B(x)$ = [Π-types] = dependent products $\cong \{f \mid \forall a{:}A\ f(a) \in B[a/x]\}$

Note: Problem : terms in F.O.L. get substituted for variables in formulas. Here, proofs get substituted

Z Types & Terms are defined simultaneously in Martin-Löf type theory.

Simultaneously defines ("judgements").

A type

$A = B$, where A type, B type

$a : A$

$a = b : A$, where $a : A$, $b : A$

$\mathcal{B}$: base category of <u>kinds</u> (= <u>Sorts</u>, <u>orders</u>)

A <u>$\mathcal{B}$-indexed category</u> is a (pseudo-) functor

$$\mathcal{B}^{op} \xrightarrow{G} Cat, \text{ i.e.}$$

$$\frac{A \xrightarrow{f} B \in \mathcal{B}}{G(A) \xleftarrow[\substack{G(f) \\ \| \\ f^*}]{} G(B) \in Cat.}$$

s.t. $\left.\begin{array}{l} (1_A)^* \cong \mathrm{id}_{G(A)} \\ (fg)^* \cong g^* \cdot f^* \end{array}\right\}$ canon. iso's.

This means: natural isos

$$\frac{\exists_f(\varphi) \longrightarrow \psi}{\varphi \longrightarrow f^*\psi}, \quad \frac{f^*\psi \longrightarrow \varphi}{\psi \longrightarrow \forall_f(\varphi)}$$

<u>Examples</u>

(1) $\mathcal{B}$ = Sets, $G(A) = 2^A$

f^* = inverse image

$$\exists_f(\varphi) = f[\varphi] = \{y \in B \mid \exists_{x \in A}(x \in \varphi \wedge fx = y)\}$$

$$\forall_f(\varphi) = \{y \in B \mid \forall_{x \in A}(fx = y \to x \in \varphi)\}$$

(2) <u>Logic</u> : multisorted 1^{st} order intuitionistic theories

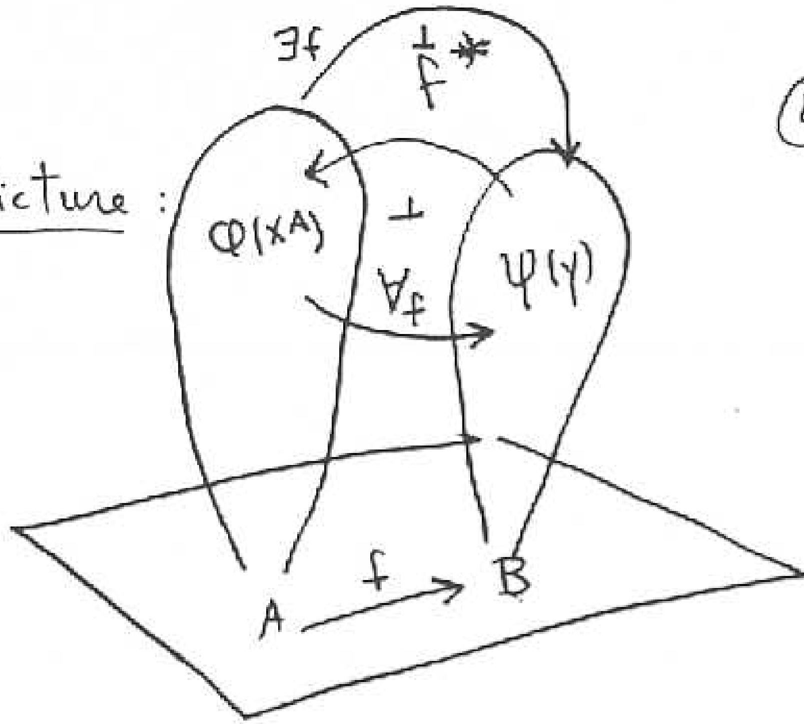
$\mathcal{B}$ = Sorts ; Maps = <u>terms</u>.

$G(A)$ = Lindenbaum alg. of formulas $\varphi(x^A)$.

with 1 free vbl x^A.

(4)

Picture :



f is a term of sort $A \to B$.

$f^*(\psi(y^B)) = \psi(f(x^A)/y)$
(substitution)

$\forall_f(\varphi) = \forall_{x^A}(fx = y^B \longrightarrow \varphi(x))$

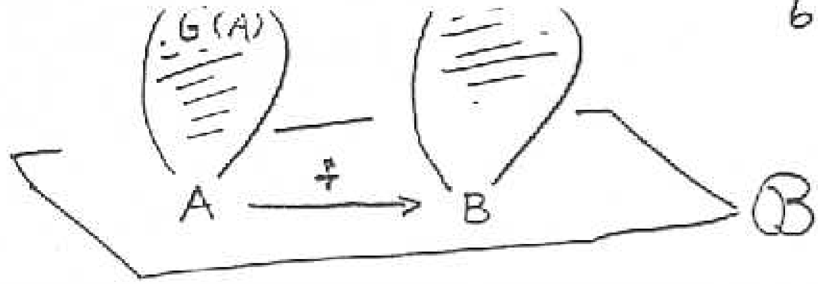
$\exists_f(\varphi) = \exists_{x^A}(fx = y^B \wedge \varphi(x))$

<u>Aside</u> : Ordinary quantification = quantification along proj^{n}s

if $A \times B \xrightarrow{\pi} A$ then :

$\forall_\pi \psi(\langle x, y\rangle) = \forall_{y \in B} \psi(\langle x, y \rangle)$.

<u>Generalization</u> (Lawvere): In logic, let $G(A)$ not just be heyting algebra, but C.C.C. : <u>objects</u> = formulas $\varphi(x^A)$ <u>arrows</u> = <u>proofs</u> <u>or derivations in</u> <u>f.o.l.</u> (Seely '77 : thesis)



A *hyperdoctrine* (Lawvere, '69) is an indexed category $\mathcal{B}^{op} \xrightarrow{G} CCC$, with:

(i) $\mathcal{B}$ cartesian

(ii) $f^* : G(B) \longrightarrow G(A)$ is a CCC - functor

(iii) adjoints $\exists_f \dashv f^* \dashv \forall_f$

(iv) "Frobenius Reciprocity" $\approx$ logical law: if $x \notin B$,
" $\exists x\,(\varphi(x) \wedge B) \Leftrightarrow \exists x\, \varphi(x) \wedge B$ "

(v) Beck-Chevalley . . .

Seely's PL - Categories

Hyperdoctrines $G : \mathcal{B}^{op} \longrightarrow CCC$ satisfying:

(1) $\mathcal{B}$: Cartesian category of "kinds", with object $\Omega \in \mathcal{B}$ & closed under exponentiation of form Ω^A, $A \in \mathcal{B}$.

(2) $\mathcal{B}^{op} \xrightarrow{G} CCC \xrightarrow{ob} Sets$
$\cong \mathcal{B}(-, \Omega)$.

(3) Indexed adjoints : $\forall C \in \mathcal{B}$,

$$G(-\times C) \underset{\Sigma_c}{\overset{\Pi_c}{\leftrightarrows}} G(-) \quad (p^* : G(-) \to G(-\times C))$$

$\Sigma_c \dashv p^* \dashv \Pi_C$, where $p_A : A \times C \to A$

Elaboration of (2), (3)

- Each fiber $G(A)$ is CCC and objects $(G(A)) \cong \mathrm{Hom}_{\mathcal{B}}(A,\Omega)$

(Recall: objects $G(A)$ = "types over A" = terms $\sigma(x^A) \in \Omega$, linguistically, $\cong$ arrows $A \xrightarrow{\sigma} \Omega$.)

- Ω has internal CCC structure inducing (pointwise) CCC structure on $G(A)$'s:

$\Omega^2 \xrightarrow{\wedge} \Omega$, $\Omega^2 \xrightarrow{\supset} \Omega$ s.t.

$G(A)$ is CCC, where

if $\alpha, \beta \in G(A) \cong \mathrm{Hom}_{\mathcal{B}}(A,\Omega)$,

$\alpha \wedge \beta = A \xrightarrow{\langle \alpha, \beta \rangle} \Omega^2 \xrightarrow{\wedge} \Omega$

$\alpha \supset \beta = \cdots\cdots\ \Omega^2 \xrightarrow{\supset} \Omega$

etc.

- Ω has internal structure

$\Omega^C \underset{\exists_C}{\overset{\forall_C}{\rightrightarrows}} \Omega \in \mathcal{B}$ defining adjoints. We want, for all A,

$$\boxed{\mathrm{Hom}_{G(A\times C)}(p^* h, \varphi) \cong \mathrm{Hom}_{G(A)}(h, \Pi_C(\varphi))}$$

that is,

$$\frac{p^* h \to \varphi \quad \text{in } G(A\times C)}{h \to \Pi_C(\varphi) \quad \text{in } G(A)}$$

where $h: A \to \Omega$, $\varphi: A\times C \to \Omega$.

Define

$\Pi_C(\varphi) = A \xrightarrow{\lambda(\varphi)} \Omega^C \xrightarrow{\forall_C} \Omega$.

$p^* h = A \times C \xrightarrow{p} A \xrightarrow{h} \Omega$.

$\Sigma_C(\varphi) = A \xrightarrow{\lambda(\varphi)} \Omega^C \xrightarrow{\exists_C} \Omega$

(10)

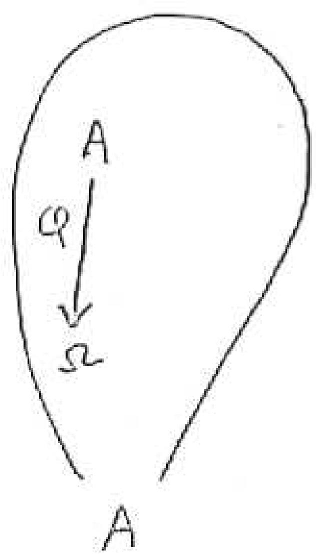
Motivation:

"type" = term $\varphi(\alpha^A) \in \Omega$
where $\alpha \in A$ is a variable

$$\cong A \xrightarrow{\varphi} \Omega$$

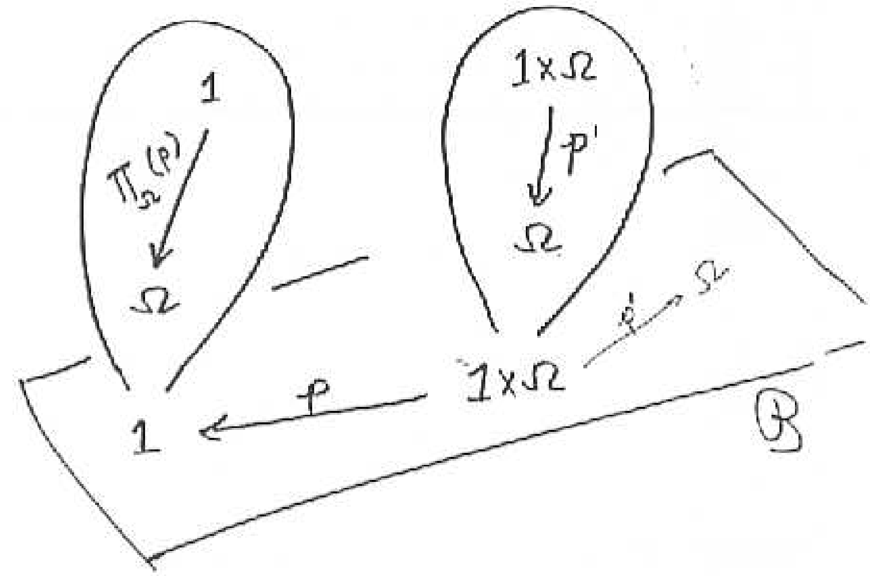
$\therefore$ from fiber: CCC of "types"

$G(A) =$



(11)

Example: $\forall \alpha . (\alpha)$



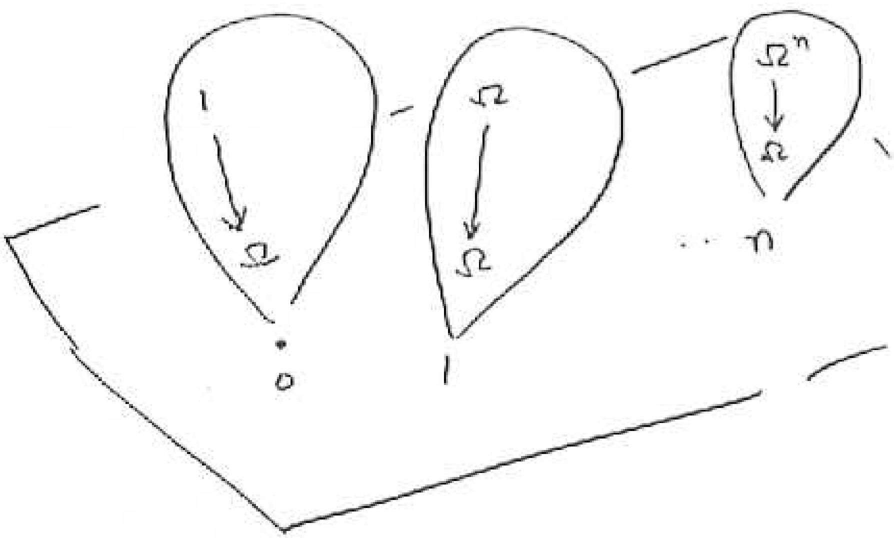
$$\cong \frac{\dfrac{\Omega \xrightarrow{id} \Omega}{1\times\Omega \xrightarrow{p'} \Omega}}{1 \xrightarrow{\lambda(p)} \Omega^{\Omega}} \xrightarrow{\forall_\Omega} \Omega = \Pi_\Omega(p).$$

<u>Remark on 2nd order Poly-λ</u>:

A formula (= term $\varphi \in \Omega$) looks like $\varphi(\alpha_1, \dots, \alpha_n) \in \Omega$ where $\alpha_i \in \Omega$ are free variables.

$\therefore \varphi \leftrightarrows \Omega^n \xrightarrow{\varphi} \Omega \in ob(G(\Omega^{\Omega^n}))$

$\therefore$ Let base $\mathbb{B} = \{0, 1, 2, \dots\}$



$Ob(G(n))$ = formulas with n free prop$^{\underline{n}}$al variables.

<u>Theorem</u> (Seely): There is an equivalence of categories

$$\boxed{\text{PL CAT}} \underset{G}{\overset{F}{\rightleftarrows}} \boxed{\text{PL Theories}}$$

between the <u>category</u> of PL Cats & <u>category</u> of PL theories (with appropriate morphisms).

<u>Corollary</u>: Soundness & Completeness

<u>Proof</u>: By construction, $G(\mathcal{L})$ = "PL-CAT generated by $\mathcal{L}$" = term-model of $\mathcal{L}$, considered as a PL-cat. Let $\mathcal{L}_0$ = <u>pure</u> (= free) language. Then

$G(\mathcal{L}_0)$ = initial PL-CAT. The result follows by general arguments.

<u>Examples</u>:

- $P\omega$
- Girard's Q.d.'s
- HEO - models.

---

e.g. <u>HEO - models</u>:

Let M be a model of untyped λ-calculus, say a C-monoid. (Let $m \cdot n$ in M = $ev\langle m,n\rangle$;

$Per(M)$ = symm., transitive rel^{ns} on M
= partitions of a subset of M.

<u>$Per(M)$ is a category</u>: define $\sigma \xrightarrow{t} \tau$ to be $t \in M$ s.t. $\forall a,b \; (a \underset{\sigma}{=} b \Rightarrow t.a \underset{\tau}{=} t.b)$.

<u>$Per(M)$ is a CCC</u>: IF $\sigma, \tau \in Per(M)$, define

$$\langle m,n\rangle \underset{\sigma\times\tau}{=\!=} \langle m',n'\rangle \text{ iff } \begin{cases} m \underset{\sigma}{=} m' \; \& \\ n \underset{\tau}{=} n' . \end{cases}$$

$$m \underset{\sigma\supset\tau}{=\!=} n \quad \text{iff} \quad m,n : \sigma \to \tau$$
$$\underline{\text{and}} \quad \forall i,j \in M \; (i \underset{\sigma}{=} j \Longrightarrow m \cdot i \underset{\tau}{=} n \cdot j) .$$

Let $\Omega = Per(M)$. Form the base category $\mathcal{B}$ as follows:

$\mathcal{B}$ = full subcat. of Sets generated by $1, \Omega$ under $- \times -$ and Ω^{-}.

Define $G : \mathcal{B}^{op} \longrightarrow CCC$

<u>objects</u> $G(A) = \mathcal{B}(A, \Omega)$
$= \underline{all}$ f^ns $A \longrightarrow \Omega$.

<u>Maps in $G(A)$</u>: if $\sigma, \tau : A \to \Omega$

$t : \sigma \longrightarrow \tau$ must, $\forall a \in A$, be a map $t_a : \sigma(a) \to \tau(a) \in Per(M)$.

$\therefore t = \{ \sigma(a) \xrightarrow{t_a} \tau(a) \mid a \in A \}$.

A <u>morphism</u> in $G(A)$ is a <u>constant</u> <u>family</u>: i.e. all components $t_a : \sigma(a) \to \tau(a) = m$, fn some $m \in M$.

$\therefore \forall a \in A. \ (x \equiv_{\sigma(a)} y \Rightarrow m \cdot x \equiv_{\tau(a)} m \cdot y)$.

$\Omega^C \xrightarrow{\forall_C} \Omega$: if $\sigma : C \to \Omega$ is an arbitrary <u>Set</u> - f^n, $\forall_C(\sigma) = \bigcap_{a \in C} \sigma(a)$

---

e.g. $C = \Omega$; $\sigma = id_\Omega : \Omega \to \Omega$

$\therefore \forall_\Omega(\sigma) = \bigcap_{a \in \Omega} \sigma(a) = \bigcap_{a \in PER} a = \phi$

$\therefore \forall_\alpha(\alpha) = \forall_\Omega(id) = \phi$.

There's bijection:

$$\frac{p^* h \xrightarrow{t} \varphi}{h \longrightarrow \Pi_c(\varphi)} \equiv \frac{h(p(a,c)) \xrightarrow{\text{all } a,c} \varphi(a,c)}{h(a) \xrightarrow{\text{all } a} \bigcap_{c \in C} \varphi(a,c)}$$

(easily checked on - constant families)

Consider $\mathrm{Per}(\mathbb{N})$, where $m \cdot n = \{m\}(n)$. This is C.C.C. in Sets. But it is also an <u>internal</u> category object in the realizability topos $\mathcal{R}$

<u>Moggi</u>: $\mathcal{R} \models$ "$\mathrm{Per}(\mathbb{N})$ is complete CCC"

Classically false.

<u>Thm</u> (Pitts): H.O. Poly^{λ} is complete w.r.t. topos models $(\bar{\mathcal{R}}, \mho)$, where $\bar{\mathcal{R}} \models$ "$\mho$ is complete CCC".

<u>Method</u>: Start with Seely cat. $G: \mathcal{B}^{o} \to$ C.C.C. Construct an

ordinary category $\mathcal{G}r(G)$ from it (Grothendieck). Look at Yoneda embedding

$$\mathcal{G}r(G) \xhookrightarrow[\text{Yoneda}]{} \underbrace{[\mathcal{G}r(G)^{o}, \mathrm{Sets}]}_{\mathcal{E}}$$

$\mathcal{E}$ is topos with internally complete C.C.C.

Girard's Qd's (Girard; also LaMarche).

A Qd is a family $X \subseteq \mathcal{P}(|X|)$ s.t.
(1) $X \neq \emptyset$
(2) $a \in X$ & $b \subseteq a \Rightarrow b \in X$
(3) X closed under directed unions

(∴ singletons $\in X$).

These are <u>posets</u>, ∴ categories.

<u>Stable Maps</u> = functors f preserving p.b.'s & filtered $\varinjlim$.

∴ (1) order preserving
(2) $a \cup b \in X \Rightarrow f(a \cap b) = fa \cap fb$
(3) $f(\bigcup_{i \in I} a_i) = \bigcup_{i \in I} f(a_i)$, I dir.

Berry order on stable maps:

$f \preceq g$ iff $\exists$ Cartesian natural trans: $f \dashrightarrow g$.

iff $a \subseteq b \Rightarrow f(b) \cap g(a) = f(a)$

∴ $\preceq \subseteq$ ptwise order.

<u>Thm</u> (Girard): $(\mathrm{Stab}(X,Y), \preceq)$ $\cong$ the poset of a Qd.

Let Stab = Qd's & stable maps

<u>Thm</u>: Stab is CCC

A <u>Qd-morphism</u>, say $f: X \to Y$, = inclusion $|X| \xhookrightarrow{f} |Y|$ s.t. $a \in X$ iff $f(a) \in Y$ for all finite $a \subseteq |X|$ (∴ for all $a \subseteq |X|$)

(21)

Let $\underline{Qd}$ = category of Qd's & Qd-morphisms.

$\underline{Qd}$ has p.b's and filtered $\varinjlim$.

There are functors

$$Qd \xrightarrow{(\ )^+} \mathrm{Stab}, \quad Qd \xrightarrow{(\ )^-} \mathrm{Stab}$$

$$X \xrightarrow{f} Y \longmapsto X \underset{f^-}{\overset{f^+}{\rightleftarrows}} Y$$

f^+ = direct image
f^- = inverse image

<u>Facts</u> (Girard): There are functors

$$Qd^2 \xrightarrow{\wedge} Qd$$
$$Qd^2 \xrightarrow{\supset} Qd$$

(Covariant !!)

$$\begin{array}{ccc} Qd \times Qd & \xrightarrow{\times} & \mathcal{S} \\ (X, Y) & & X \times Y \\ \Big\downarrow (f,g) & \longmapsto & \Big\downarrow f^+ \times g^+ \\ (X', Y') & & X' \times Y' \end{array}$$

$\in Qd.$

$$\begin{array}{ccc} Qd \times Qd & \xrightarrow{\supset} & \mathcal{S} \\ (X, Y) & & (X \Rightarrow Y) \\ \Big\downarrow (f,g) & \longmapsto & \Big\downarrow \lambda s.(g^+ \cdot s \cdot f^-) \\ (X', Y') & & (X' \Rightarrow Y') \end{array}$$

$\in Qd.$

So Qd internally has CCC structure. Let $\Omega = Qd$.
Look at powers Ω^n, etc.

$\mathbb{B} = \{0, 1, 2, \ldots\}$ (powers of Qd). Form Seely PL-Cat

$\mathbb{B}^{op} \xrightarrow{G} CCC$.

$G(n)$: <u>Objects</u> : <u>Entire functors</u> $Qd^n \xrightarrow{F} Qd$

(Entire = pres. p.b.'s & filtered $\varinjlim$).

<u>arrows</u>: $F \xrightarrow{t} G$ map s.t. $t_X : FX \to GX$ satisfying