THEORY AND MODELS OF LAMBDA CALCULUS: UNTYPED AND TYPED

Session 3: *Applications to Type Theory and Semantics*

Dana S. Scott
University Professor Emeritus
Carnegie Mellon University

Jeremy G. Siek
Associate Professor of Computer Science
Indiana University, Bloomington

Leap Workshop
LambdaConf 2018
Boulder, Colorado

Pairing and Relations

Recall. ***Pairing functions*** for sets in $\mathcal{P}(\mathbb{N})$ can be defined by these enumeration operators:

$$\mathbf{Pair}(\mathtt{X})(\mathtt{Y}) = \{2n \mid n \in \mathtt{X}\} \cup \{2m+1 \mid m \in \mathtt{Y}\}$$

$$\mathbf{Fst}(\mathtt{Z}) = \{n \mid 2n \in \mathtt{Z}\} \text{ and } \mathbf{Snd}(\mathtt{Z}) = \{m \mid 2m+1 \in \mathtt{Z}\}.$$

Note: **Under this definition we have** $\mathcal{P}(\mathbb{N}) = \mathcal{P}(\mathbb{N}) \times \mathcal{P}(\mathbb{N})$ **in the category of topological spaces. From time to time we may write** $\mathbf{Pair}(\mathtt{X})(\mathtt{Y}) = (\mathtt{X}, \mathtt{Y})$ **to save space.**

Convention. Every subset of $\mathcal{P}(\mathbb{N})$ can be regarded as a ***binary relation***, where for all $\mathcal{A} \subseteq \mathcal{P}(\mathbb{N})$ we write $\mathtt{X}\ \mathcal{A}\ \mathtt{Y}$ iff $\mathbf{Pair}(\mathtt{X})(\mathtt{Y}) \in \mathcal{A}$.

Partial Equivalences as Types

Definition. By a ***type*** over $\mathcal{P}(\mathbb{N})$ we shall understand a ***partial equivalence relation*** $\mathcal{A} \subseteq \mathcal{P}(\mathbb{N})$ where, for all $\mathtt{X},\mathtt{Y},\mathtt{Z} \in \mathcal{P}(\mathbb{N})$, we have

- $\mathtt{X}\ \mathcal{A}\ \mathtt{Y}$ implies $\mathtt{Y}\ \mathcal{A}\ \mathtt{X}$, and
- $\mathtt{X}\ \mathcal{A}\ \mathtt{Y}$ and $\mathtt{Y}\ \mathcal{A}\ \mathtt{Z}$ imply $\mathtt{X}\ \mathcal{A}\ \mathtt{Z}$.

We also write $\mathtt{X}:\mathcal{A}$ iff $\mathtt{X}\ \mathcal{A}\ \mathtt{X}$, and say that $\mathcal{A}$ ***types*** $\mathtt{X}$.

Note: Think of a type as a quotient space of a subspace of $\mathcal{P}(\mathbb{N})$. Taking quotients is a very common mathematical construction. It is, however, better NOT to pass to using equivalence classes as points in order to make it easier to employ our λ-calculus.

Embedding Spaces as Subspaces

Theorem. Every countably based T_0-space $\mathcal{X}$ is homeomorphic to a ***subspace*** of $\mathcal{P}(\mathbb{N})$.

Reference: P. Alexandroff, *Zur Theorie der topologischen Raume*, C.R. (Doklady) Acad. Sci. URSS, vol. 11 (1936), pp, 55-58.

Proof Sketch: Let a subbasis for the topology of $\mathcal{X}$ be $\{ \mathcal{O}_n \mid n \in \mathbb{N} \}$.

Define $\varepsilon : \mathcal{X} \to \mathcal{P}(\mathbb{N})$ by $\varepsilon(x) = \{ n \in \mathbb{N} \mid x \in \mathcal{O}_n \}$.

By the T_0-axiom, this mapping is one-one onto a subspace of $\mathcal{P}(\mathbb{N})$.

Check first that the ***inverse image*** of opens of $\mathcal{P}(\mathbb{N})$ are open in $\mathcal{X}$.

Notice next that $\varepsilon(\mathcal{O}_n) = \varepsilon(\mathcal{X}) \cap \{ S \in \mathcal{P}(\mathbb{N}) \mid n \in S \}$.

Hence, the ***image*** of a open of $\mathcal{X}$ is an open of the subspace.

Therefore, ε is a homeomorphism to a subspace. Q.E.D.

Important: Continuous functions ***between*** subspaces come from those of $\mathcal{P}(\mathbb{N})$.

Subspaces and Closure Operators

Definition For ***subspaces*** $\mathcal{X} \subseteq \mathcal{P}(\mathbb{N})$, we write

$$[\mathcal{X}] = \{(X,X) \mid X \in \mathcal{X}\},$$

so that we may regard ***subspaces as types*.**

For ***closure operators*** C we can also write

$$[C] = \{(X,X) \mid C(X) = X\},$$

so that we may also regard ***closures as types*.**

Comment: These subspaces already give us a very wide variety of types, including many familiar examples (in view of the ***Embedding Theorems***). However, the topological spaces do not have all the best ***categorical*** and ***logical*** properties that are gained by quotients.

What we need to show is

that quotients work well with the λ-calulus.

The Category of Types

Definition. The ***exponentiation*** of types $\mathcal{A}, \mathcal{B} \subseteq \mathcal{P}(\mathbb{N})$

is defined as that relation where

$\mathtt{F}(\mathcal{A} \to \mathcal{B})\mathtt{G}$ iff $\forall \mathtt{X}, \mathtt{Y}.\ \mathtt{X}\ \mathcal{A}\ \mathtt{Y}$ implies $\mathtt{F}(\mathtt{X})\ \mathcal{B}\ \mathtt{G}(\mathtt{Y})$.

Exercise: Show $(\mathcal{A} \to \mathcal{B})$ is a partial equivalence relation.

Exercise: Show $\mathtt{F}:\mathcal{A} \to \mathcal{B}$ implies $\forall \mathtt{X}:\mathcal{A}.\ \mathtt{F}(\mathtt{X}):\mathcal{B}$.

Exercise: Show $(\lambda \mathtt{X}.\lambda \mathtt{Y}.\mathtt{X}) : \mathcal{A} \to (\mathcal{B} \to \mathcal{A})$ for any types $\mathcal{A}$ and $\mathcal{B}$.

Theorem: The types form a ***category*** expanding the category of subspaces.

Definition. For each type $\mathcal{A}$ the ***identity type*** on $\mathcal{A}$ is defined as that relation such that $\mathtt{Z}(\mathtt{X} \equiv_{\mathcal{A}} \mathtt{Y})\mathtt{W}$ iff $\mathtt{Z}\ \mathcal{A}\ \mathtt{X}\ \mathcal{A}\ \mathtt{Y}\ \mathcal{A}\ \mathtt{W}$.

Products and Sums of Types

Definition. The ***product*** of two types $\mathcal{A},\mathcal{B} \subseteq \mathcal{P}(\mathbb{N})$ is defined as that relation where

$$X(\mathcal{A}\times\mathcal{B})Y \text{ iff } \mathbf{Fst}(X)\mathcal{A}\,\mathbf{Fst}(Y) \text{ and } \mathbf{Snd}(X)\,\mathcal{B}\,\mathbf{Snd}(Y).$$

Exercise: The product of two types is again a type, and we have

$$X:(\mathcal{A}\times\mathcal{B}) \text{ iff } \mathbf{Fst}(X):\mathcal{A} \text{ and } \mathbf{Snd}(X):\mathcal{B}\ .$$

Definition. The ***sum*** of two types $\mathcal{A},\mathcal{B} \subseteq \mathcal{P}(\mathbb{N})$ is defined as that relation where $X(\mathcal{A}+\mathcal{B})Y$ iff

either $\exists X_0,Y_0[X_0\mathcal{A}Y_0\ \&\ X=(\{0\},X_0)\ \&\ Y=(\{0\},Y_0)]$

or $\exists X_1,Y_1[X_1\mathcal{B}Y_1\ \&\ X=(\{1\},X_1)\ \&\ Y=(\{1\},Y_1)]$.

Exercise: The sum of two types is again a type, and we have

$X:(\mathcal{A}+\mathcal{B})$ iff either $\mathbf{Fst}(X)=\{0\}\ \&\ \mathbf{Snd}(X):\mathcal{A}$

or $\mathbf{Fst}(X)=\{1\}\ \&\ \mathbf{Snd}(X):\mathcal{B}$.

Isomorphism of Types

Definition. Two types $\mathcal{A}, \mathcal{B} \subseteq \mathcal{P}(\mathbb{N})$ are ***isomorphic***, in symbols $\mathcal{A} \cong \mathcal{B}$, provided there are mappings $\mathtt{F}: \mathcal{A} \to \mathcal{B}$ and $\mathtt{G}: \mathcal{B} \to \mathcal{A}$ where

$$\forall \mathtt{X}{:}\mathcal{A}.\ \mathtt{X}\ \mathcal{A}\ \mathtt{G(F(X))} \text{ and } \forall \mathtt{Y}{:}\mathcal{B}.\ \mathtt{Y}\ \mathcal{B}\ \mathtt{F(G(Y))}.$$

Exercises: Prove these ***algebraic laws*** for all types $\mathcal{A}, \mathcal{B}, \mathcal{C}$:

$$(\mathcal{A} \times \mathcal{B}) \cong (\mathcal{B} \times \mathcal{A}) \text{ and } (\mathcal{A} + \mathcal{B}) \cong (\mathcal{B} + \mathcal{A}),$$

$$((\mathcal{A} \times \mathcal{B}) \times \mathcal{C}) \cong (\mathcal{A} \times (\mathcal{B} \times \mathcal{C})) \text{ and } ((\mathcal{A} + \mathcal{B}) + \mathcal{C}) \cong (\mathcal{A} + (\mathcal{B} + \mathcal{C})),$$

$$(\mathcal{A} \times (\mathcal{B} + \mathcal{C})) \cong ((\mathcal{A} \times \mathcal{B}) + (\mathcal{A} \times \mathcal{C})) \text{ and } ((\mathcal{A} \times \mathcal{B}) \to \mathcal{C}) \cong (\mathcal{A} \to (\mathcal{B} \to \mathcal{C})),$$

$$(\mathcal{A} \to (\mathcal{B} \times \mathcal{C})) \cong ((\mathcal{A} \to \mathcal{B}) \times (\mathcal{A} \to \mathcal{C})) \text{ and } ((\mathcal{A} + \mathcal{B}) \to \mathcal{C}) \cong ((\mathcal{A} \to \mathcal{C}) \times (\mathcal{B} \to \mathcal{C})).$$

Note: Types do form a (bi) cartesian closed category — whereas the topological category of subspaces does not.

Dependent Products

Definition. Let $\mathcal{T}$ be the class of all types. For each $\mathcal{A} \in \mathcal{T}$, an $\mathcal{A}$***-indexed family of types*** is a function $\mathcal{B}: \mathcal{P}(\mathbb{N}) \to \mathcal{T}$, such that $\forall x_0, x_1.\ x_0\ \mathcal{A}\ x_1$ implies $\mathcal{B}(x_0) = \mathcal{B}(x_1)$.

In words: ***Equivalent*** parameters produce ***equivalent*** types.

Definition. The ***dependent product*** of an $\mathcal{A}$-indexed family of types, $\mathcal{B}$, is this equivalence relation:

$$F_0 (\prod X{:}\mathcal{A}.\mathcal{B}(X)) F_1 \text{ iff}$$

$$\forall x_0, x_1.\ x_0\ \mathcal{A}\ x_1 \text{ implies } F_0(x_0)\ \mathcal{B}(x_0)\ F_1(x_1).$$

Exercise: Show that the dependent product of types is again a type.

Exercise: $(\mathcal{A} \to \mathcal{B}) = \prod X{:}\mathcal{A}.\mathcal{B}$.

Dependent Sums

Definition. The ***dependent sum*** of an $\mathcal{A}$-indexed family of types, $\mathcal{B}$, is this equivalence relation:

$$Z_0(\sum X{:}\mathcal{A}.\mathcal{B}(X))Z_1 \text{ iff}$$

$$\exists X_0,Y_0,X_1,Y_1[X_0\mathcal{A}X_1 \;\&\; Y_0\mathcal{B}(X_0)Y_1 \;\&\; Z_0 = (X_0,Y_0) \;\&\; Z_1 = (X_1,Y_1)]$$

Exercise: Show that the dependent sums of types is again a type.

Exercise: $(\mathcal{A}\times\mathcal{B}) = \sum X{:}\mathcal{A}.\mathcal{B}$.

Note: To be able to compound sums and products of families of types, we need to take care of the ways types depend on their parameters.

Systems of Dependent Types

Definition. We say that $\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D}$ together form a ***system of dependent types*** iff

- $\forall X_0, X_1.\,[\,X_0 \,\mathcal{A}\, X_1 \Rightarrow \mathcal{B}(X_0) = \mathcal{B}(X_1)\,]$, and
- $\forall X_0, X_1, Y_0, Y_1.\,[\,X_0 \,\mathcal{A}\, X_1 \;\&\; Y_0 \,\mathcal{B}(X_0)\, Y_1 \Rightarrow \mathcal{C}(X_0, Y_0) = \mathcal{C}(X_1, Y_1)\,]$, and
- $\forall X_0, X_1, Y_0, Y_1, Z_0, Z_1.\,[\,X_0 \,\mathcal{A}\, X_1 \;\&\; Y_0 \,\mathcal{B}(X_0)\, Y_1 \;\&\; Z_0 \,\mathcal{C}(X_0, Y_0)\, Z_1 \Rightarrow \mathcal{D}(X_0, Y_0, Z_0) = \mathcal{D}(X_1, Y_1, Z_1)\,]$,

provided that $\mathcal{A} \in \mathcal{T}$, and $\mathcal{B},\mathcal{C},\mathcal{D}$ are functions on $\mathcal{P}(\mathbb{N})$ to $\mathcal{T}$ of the indicated number of arguments.

Exercise: Show under the above assumptions on the system $\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D}$, we will always have $\prod X{:}\mathcal{A}\,.\sum Y{:}\mathcal{B}(X)\,.\prod Z{:}\mathcal{C}(X,Y).\ \mathcal{D}(X,Y,Z) \in \mathcal{T}$.

Polymorphic Types

Theorem. The class $\mathcal{T}$ of all types is a ***complete lattice***, because it is closed under ***arbitrary intersections***.

Exercise: Show that $\boldsymbol{\lambda}\mathtt{X}.\boldsymbol{\lambda}\mathtt{Y}.(\mathtt{X},\mathtt{Y}) : \bigcap_{\mathcal{A},\mathcal{B}} (\mathcal{A} \rightarrow (\mathcal{B} \rightarrow (\mathcal{A} \times \mathcal{B})))$

Definition. The ***Scott numerals*** (1963) in $\boldsymbol{\lambda}$-calculus are:

$\underline{\mathbf{0}} = \boldsymbol{\lambda}\mathtt{X}.\boldsymbol{\lambda}\mathtt{F}.\mathtt{X}$, $\underline{\mathbf{1}} = \boldsymbol{\lambda}\mathtt{X}.\boldsymbol{\lambda}\mathtt{F}.\mathtt{F}(\underline{\mathbf{0}})$, $\underline{\mathbf{2}} = \boldsymbol{\lambda}\mathtt{X}.\boldsymbol{\lambda}\mathtt{F}.\mathtt{F}(\underline{\mathbf{1}})$, etc., and

$\underline{\mathbf{succ}} = \boldsymbol{\lambda}\mathtt{Y}.\boldsymbol{\lambda}\mathtt{X}.\boldsymbol{\lambda}\mathtt{F}.\mathtt{F}(\mathtt{Y})$, and

$\underline{\mathbf{pred}} = \boldsymbol{\lambda}\mathtt{Y}.\mathtt{Y}(\underline{\mathbf{0}})(\boldsymbol{\lambda}\mathtt{X}.\mathtt{X})$.

Exercise: Show $\mathcal{Scott} = \bigcap_{\mathcal{A}} (\mathcal{A} \rightarrow ((\mathcal{Scott} \rightarrow \mathcal{A}) \rightarrow \mathcal{A}))$ types these numerals.

Exercise: Any ***monotone*** $\Phi : \mathcal{T} \rightarrow \mathcal{T}$ has a ***least & greatest fixed point***.

Propositions as Types

Definition. Every type $\mathcal{P} \in \mathcal{T}$ can be regarded as a ***proposition***, where ***asserting*** (or ***proving*** $\mathcal{P}$) means finding ***evidence*** $\mathtt{E}:\mathcal{P}$.

Exercise: Given $\mathtt{F}:(\mathcal{A} \to (\mathcal{A} \to \mathcal{A}))$, then explain why asserting

$$\prod \mathtt{X}:\mathcal{A}.\prod \mathtt{Y}:\mathcal{A}.\prod \mathtt{Z}:\mathcal{A}.\ \mathtt{F(X)(F(Y)(Z))} \equiv_{\mathcal{A}} \mathtt{F(F(X)(Y))(Z)}$$

is the same as asserting that $\mathtt{F}$ is an ***associative binary operation.***

Conventions: Under this interpretation of logic,

asserting $(\mathcal{P} \times \mathcal{Q})$ means asserting a ***conjunction***,

asserting $(\mathcal{P} + \mathcal{Q})$ means asserting a ***disjunction***,

asserting $(\mathcal{P} \to \mathcal{Q})$ means asserting an ***implication***,

asserting $(\prod \mathtt{X}:\mathcal{A}.\mathcal{B}(\mathtt{X}))$ means asserting a ***universal quantification***, and

asserting $(\sum \mathtt{X}:\mathcal{A}.\mathcal{B}(\mathtt{X}))$ means asserting an ***existential quantification***.

Some Conclusions

- Enumeration operators over $\mathcal{P}(\mathbb{N})$ model λ-calculus and are characterized by a simple topology.
- The large category of types over $\mathcal{P}(\mathbb{N})$ inherits much topology.
- λ-calculus over $\mathcal{P}(\mathbb{N})$ plus the arithmetic combinators provides a basic notion of computability.
- The category of types over $\mathcal{P}(\mathbb{N})$ thus also inherits aspects of computability.
- Polymorphism for types then gives an abstract foundation for defining inductive and co-inductive data structures.
- Propositions-as-types then will enforce using constructive logic.

The model can in this way function as a laboratory for exploring these ideas in a very concrete fashion.