Andrew Bacon, "A Theory of Structured Propositions"

Baiting Hou

28 October 2023

Background

Full λ-language

Given a signature Σ, the typed λ-languages $\mathcal{L}(\Sigma)$ is a plurality of sets of $\mathcal{L}^{\sigma}(\Sigma)$, which is the terms of type σ. We inductively define terms of type σ as follows:

Atomic x is a term of type σ if $x \in Var^{\sigma}$; c is a term of type σ if $c \in \Sigma^{\sigma}$

Application MN is a term of type τ if M is a term of type σ and N is a term of type $\sigma \to \tau$

Abstraction $\lambda x.M$ is a term of type $\sigma \to \tau$ if M is a term of type τ and $x \in Var^{\sigma}$,

Sometimes we don't want our language contains all λ-terms.[1]

[1] For example, consider the following theorem in **H** given the language in question contains all λ-terms: $\forall X \exists Y \forall zw(Yzw = Xwz)$

General λ-language

Relevant term:a term in which every bound variable occurs free at least once in the scope of the λ that binds it.

Affine term:a term in which every bound variable occurs free at most once in the scope of the λ that binds it.

Linear term:a term in which every bound variable occurs free exactly once in the scope of the λ that binds it.

Ordered term: a term in which every bound variable occurs free exactly once, and is the rightmost free variable in the scope of the λ that binds it.

These terms over a signature Σ can be defined inductively as follows:

- Any constant or variable is ordered.
- If $M : \sigma \to \tau$ and $N : \sigma$ are ordered terms, then so i s $MN : \tau$.

- If $M : \tau$ is ordered and $x : \sigma$ is the last element of $FV_s(M)$[2] and doesn't appear elsewhere in the sequence, then $\lambda x.M : \sigma \to \tau$ is ordered.

[2] Let $FV_s(M)$ be the operations from terms to sequences of free variables, which can be defined inductively.

Structural term:

- Any constant is structural.
- If $M : \sigma \to \tau$ and $x : \sigma$ is a variable, then so is $Mx : \tau$.
- If $M : \sigma \to \tau$ and $N : \sigma$ are ordered terms, then so is $MN : \tau$.
- If $M : \tau$ is structural and $x : \sigma$ is the last element of $FV_s(M)$ and doesn't appear elsewhere in the sequence, then $\lambda x.M : \sigma \to \tau$ is structural.

Curry Typing

Natural deduction for Curry Typing

$\dfrac{}{x{:}\sigma\vdash x{:}\sigma}$	**Identity**	$\dfrac{\Gamma\vdash M{:}\sigma\to\tau \qquad \Delta\vdash N{:}\sigma}{\Gamma,\Delta\vdash(MN){:}\tau}$	**Application**
$\dfrac{\Gamma,x{:}\sigma,y{:}\tau,\Delta\vdash M{:}\rho}{\Gamma,y{:}\tau,x{:}\sigma,\Delta\vdash M{:}\rho}$	**Exchange**	$\dfrac{\Gamma,x{:}\sigma\vdash M{:}\tau}{\Gamma\vdash\lambda x M{:}\sigma\to\tau}$	**Abstraction**
$\dfrac{\Gamma\vdash M{:}\tau}{x{:}\sigma,\Gamma\vdash M{:}\tau}$	**Weakening**	$\dfrac{\Gamma\vdash M{:}\sigma\to\tau}{\Gamma,x{:}\sigma\vdash Mx{:}\tau}$	**Concretion**
$\dfrac{\Gamma,x{:}\sigma,y{:}\sigma,\Delta\vdash M{:}\tau}{\Gamma,z{:}\sigma,\Delta\vdash M[z/x][z/y]{:}\tau}$	**Contraction**	$\dfrac{c\in\Sigma}{\vdash c{:}\sigma}$	**Constants**

Theorem 0.0.1
- *Relevant type theory is the result of dropping Weakening*
- *Affine type theory is the the result of dropping Contraction.*
- *Linear type theory is the result of dropping Weakening and Contraction.*
- *Ordered type theory is the result of dropping Weakening, Contraction and Exchange*
- *Structural type theory is the result of dropping Identity, Weakening, Contraction and Exchange.*
[3]

[3] From left to right can be proved by induction on the length of sequence of proof,from right to left can be proved by induction on the complexity of M

Theorem 0.0.2 *If $\Gamma \to P : \sigma$ is a derivable sequence in ND[] then it is derived uniquely.*

Syntactic Structured Theory of Reality

[4]

Propositions are structured in a way that is analogous to the way that sentences of a typed language are structured, properties and relations as predicates are structured, and so on.

Let us now attempt to formulate a formal theory that captures this informal picture of reality.

[4] We will restrict attention to languages that are only based on relational types theory.

PREDICATE ARGUMENT STRUCTURE(PAS) [5]
$\forall XY \forall \vec{z}\vec{w}(X\vec{z} = Y\vec{w} \to X = Y \wedge z_i = w_i)$ [6]

Problems with PAS: explicit counterexamples (converses, reflexizations, vacuous abstraction) and implicit counterexamples (Russel-Myhill).

Note that PAS is only a commitment of syntactic theory concerning the structure of reality theory. So perhaps we should be looking for different models of the structure of reality.

[5] Captures the following informal idea: if two sentences in subject-predicate form express the same proposition, then they have the same syntactic structure,with corresponding syntactic constituents expressing the same entities.
[6] It is schematic in the type.

Pictorial Structured Theory of Reality

The account of propositional structure posited by pictorial theory are generally more faithfully represented diagrammatically than by expressions in some language.

Relational Diagrams

The diagram for type t	a grey box
The diagram for type e	a grey circle
a hole of type σ	taking a diagram of type σ and swithing grey and white

COMPLICATION If you have a diagram **M** which has holes of shape σ_1, ..., σ_k, ..., σ_n in that order, and another diagram N that has holes of shape τ_1, ..., τ_i, ..., τ_j, where τ_{i+1}, ..., τ_j are the types of the holes in a relation of type τ_k, then you can plug N into Ms kth hole, greying out the holes corresponding to τ_{i+1}, ..., τ_j to form a relation, $(MN)^i_{k_1}$, with holes of type σ_1, ..., σ_{k-1}, τ_1, ..., τ_i, τ_{k+1}, ..., τ_n in that order.

RELATIONAL DIAGRAM THEORY: The relational diagram theory is the smallest set containing the simple diagram closed under complications.

TRANSLATION FROM RELATIONAL DIAGRAMS TO $lambda$-TERMS

- Each simple diagram c of type σ may be associated with the constant $c : \sigma$ it is tagged by.

- If **M** and **N** are relational diagrams associated with λ-terms M and N, then the complication of diagrams $(\mathbf{MN})^m_n$ may be associated with the λ-term (i.e. $\lambda\vec{x}\vec{y}.M\vec{x}(N\vec{y})$ where $\vec{x} = x_1, ..., x_m$ and $\vec{y} = y_1, ..., y_n$ are sequences of variables of the appropriate type).

Inductive definition of relational diagram theory

- **c** is a relational diagram for any constant c.
- You may plug a variable diagram or relational diagram into the leftmost hole of a relation diagram, provided it fits.If **R** is a relational diagram and x a variable diagram that fits into the leftmost hole of R and doesn't appear in R then $(Rx)^0_0$ is a relational diagram;If **R** is a relational diagram and a a relational diagram that fits into the leftmost hole of **R** then $(\mathbf{Ra})^0_0$ is a relational diagram.
- If **R** is a relational diagram, then the result of punching out the rightmost variable diagram in **R** is a relational diagram.

Translation from λ-terms to relational diagram

- $c^d = \mathbf{c}$
- $(Mx)^d = (M^d x)^0_0$ (provided x does not occur in M).
- $(MN)^d = (M^d N^d)^0_0$
- $(\lambda x.M)^d$ =the result of punching a hole where x occurs in M^d (provided $x$occurs once and is the rightmost occurrence of a variable in M).

Theorem 0.0.3 (Soundness for relational diagram with respect to $\beta\eta$) *If P and Q are immediately β-equivalent or η-equivalent then **P**=**Q**.*

Theorem 0.0.4 (Completeness for relational diagrams with respect to $\beta\eta$) *For any structuralλ-terms P and Q , if **P**=**Q** then P and Q are $\beta\eta$ equivalent.]*

Theorize structure in the object language

Primitives

It seems that notions like *definability* or *simple* are not obviously notions we can simply define in the pure signature of higher-order language.We could instead add these notions to higher-order language and subject them to axioms that constrain their behaviour in the ways we'd expect:

(metaphysical) definability: $A_1, ..., A_n \rhd C$ means C is metaphysically definable from $A1, ..., An$.

Purity: $Pure_\sigma Q$ means Q is a 'constituentless entity', or at least, metaphysically definable from nothing.

Now we present some principles govern these notions in a λ-language $\mathcal{J}(\Sigma)$:

Pure Combinators $Pure_\sigma$Q whenever Q is a $\mathcal{J}$-combinator

Pure Complication $\forall X_\sigma y_\tau (Pure_\sigma X \wedge Pure_\tau y \rightarrow Pure_\rho (Xy)^n_m)$

Only Pure Combinators $\neg Pure_\sigma M$ whenever M is a closed term in β-normal form containing constants.[7]

Perhaps we can define metaphysical definability from purity as follows:

$\rhd_{\vec{\sigma}\tau} := \lambda \vec{x} y \exists_{\vec{\sigma} \to \tau} X(PureX \wedge X\vec{x} = y)$[8]

Or conversely:

$Pure_\sigma := \lambda x. \rhd_\sigma x$

Some principles governing metaphysical definability:[9]:

Reflexivity $A \rhd A$

Cut $(\Gamma \rhd A) \wedge (\Delta, A, \Delta' \rhd B) \rightarrow (\Delta, \Gamma, \Delta' \rhd B)$

Weakening $\Gamma, \Delta \rhd B \rightarrow \Gamma, A, \Delta \rhd B$

Exchange $\Gamma, A, B, \Delta \rhd C \rightarrow \Gamma, B, A, \Delta \rhd C$

Contraction $\Gamma, A, A, \Delta \rhd B \rightarrow \Gamma, A, \Delta \rhd B$

[7] Note that in languages with vacuous abstraction we need to stipulate that M cannot be β-reduced.

[8] Roughly speaking, it caputres the intuition that for every M with constants, we can keep λ-abstracting it until it become a combinator

[9] Note that **Reflexivity**is not inconsistent with the existence of pure entities. It just follows that pure entities can both be defined from nothing and from themselves.

Framework

We work in the structural language $\mathcal{S}(\Sigma)$.[10,11]

So far, we seem to have a structured view corresponding to every choice of:

- Signature Σ representing the metaphysically simple entities.
- Signature Π representing the pure constituentless entities.
- General λ-language $\mathcal{J}(\Pi \cup \Sigma)$, containing all and only terms representing the entities that can be created using the simple entities using legitimate modes of combination.
- Logic L, closed under the rule of substitution for constants in Σ, representing when two terms correspond to the same entity in reality.

$\mathcal{J}(\Pi \cup \Sigma)/L$Structuralism : (Informally)Reality is structured like the language $\mathcal{J}(\Pi \cup \Sigma)$ quotiented by the identities in L. It can be axiomatized with the following rules:

[10] Σ should include the full signature of logical constants instead of relying on metalinguistic abbreviation

[11] Remember in the correspondence between structural λ-language and relational diagram, we associate constants with simple diagrams. Thus we should assume that $S(\Sigma)$ is a logically perfect language,in the sense that the constants in Σ denote distinct simple entities, and every simple entity is expressed by some constant.

L-Identity $A =_\sigma B$ iff $A, B : \sigma$ are closed terms and $A =_\sigma B \in L$
$\mathcal{J}(\Pi \cup \Sigma)$**-Completeness**[12] $F[d_1/\mathbf{a}], F[d_2/\mathbf{a}], ... \vdash \forall_\sigma F\check{\mathbf{a}}$[13]
The inconsistency result:

Theorem 0.0.5 (Russel-Myhill) *$\mathcal{J}(\Pi \cup \Sigma)/\beta\eta$-structuralism is inconsistent.*

Thus many structural views that restrict or relax along either of these dimensions are left open:

- General λ-languages that don't have combinators.
- More liberal λ-languages (say, with combinators) but a stronger notion of equivalence than $\beta\eta$.

The consistency result:

Theorem 0.0.6 *Our substrctural Curry type systems are consistent.*

[12] Captures the idea that $\mathcal{J}(\Pi \cup \Sigma)$ denotes every entity .

[13] Where d_1, d_2... enumerate all the closed terms in $\mathcal{J}(\Pi \cup \Sigma)$.