Fibrational views on typing, proving, and parsing

Noam Zeilberger, Ecole Polytechnique

special session on Categorical Type Theory
@ MFPS XXXVII (30 Aug - 3 Sep 2021)

1. Type systems as functors

Motivations

When looking at type theory via category theory, it is tempting to model type systems as categories.

A beautiful idea when it works...but sometimes it doesn't!

For example, it is problematic for "extrinsic" features like *subtyping* and *intersection types.*

More fundamentally, the category model collapses

terms = ***judgments*** = ***derivations***.

In type theory these are important distinctions!

Intrinsic vs extrinsic typing

J. C. Reynolds, "The Meaning of Types" (2000):

> There are two very different ways of giving denotational semantics to a programming language (or other formal language) with a nontrivial type system. ***In an intrinsic semantics, only phrases that satisfy typing judgements have meanings.*** Indeed, meanings are assigned to the typing judgements, rather than to the phrases themselves, so that a phrase that satisfies several judgements will have several meanings.
>
> In contrast, ***in an extrinsic semantics, the meaning of each phrase is the same as it would be in a untyped language,*** regardless of its typing properties. In this view, ***a typing judgement is an assertion that the meaning of a phrase possesses some property.***

An intrinsic bias

Modelling type systems by categories has a bias towards the intrinsic view of typing.

Indeed, every morphism of a category could be said to have an intrinsic type f : dom(f) → cod(f).

What would it mean for f to possess other types?

Subtyping and polymorphism

Certain typing rules are correspondingly difficult to interpret under a naive categorical reading:

$$\frac{\Gamma \vdash t : R \qquad R \leq S}{\Gamma \vdash t : S} \quad \textit{(subsumption)}$$

$$\frac{\Gamma \vdash t : R_1 \qquad \Gamma \vdash t : R_2}{\Gamma \vdash t : R_1 \wedge R_2} \quad \textit{(}\wedge\textit{-introduction)}$$

An extrinsic model

Some (e.g., Reynolds) have managed to construct intrinsic categorical models of subtyping and intersection types backed by non-trivial coherence theorems, whose goal is to show that *any two derivations of the same typing judgment are assigned the same meaning.*

We advocate a different approach, in a sense more naive, which takes the extrinsic view more seriously from a categorical perspective. In a word, the key is simply to replace categories by *functors.*

Functors as type systems

Every functor p : $\mathcal{D} \to \mathcal{T}$ induces an abstract type system:

* the "***terms***" are morphisms of $\mathcal{T}$;
* the "***types***" are objects of $\mathcal{D}$;
* the "***judgments***" are triples (R,f,S) of a term and a pair of types such that p(R) = dom(f) and p(S) = cod(f);
* the "***derivations***" of judgments (R,f,S) are morphisms α of $\mathcal{D}$ such that dom(α) = R, cod(α) = S, and p(α) = f.

Observe that:
* one term can be involved in many different judgments!
* one judgment can have many different derivations!

Type refinement systems

In fact, p : $\mathcal{D} \to \mathcal{T}$ induces an abstract ***type refinement*** system, in the sense that objects of $\mathcal{T}$ may also be considered as types, refined by objects of $\mathcal{D}$. Adapting notation from Pfenning et al., we write R $\sqsubset$ A to mean that p(R) = A.

Modelling type refinement systems this way as functors enables us to give first-class mathematical status to extrinsic concepts such as subtyping and intersection types.

Conversely, viewing functors as type systems allows us to import some at times helpful intuitions from logic.

Typing rules for a functor

Observe that the following *typing rules* are semantically valid for any functor p : $\mathcal{D} \to \mathcal{T}$ viewed as an abstract type (refinement) system.

$$\frac{(R,f,S) \quad (S,g,T)}{(R,fg,T)} \qquad \frac{S \sqsubset A}{(S,id_A,S)}$$

Validity of the rules reduces to functoriality of p.

Subtyping for a functor

Any functor p : $\mathcal{D} \to \mathcal{T}$ comes equipped with an abstract subtyping relation, defined on refinements of the same type. Given R ⊏ A and S ⊏ A, a **subtyping derivation** of R $\leq_A$ S is just a derivation of (R,id_A,S).

Observe that the covariant and contravariant subsumption rules are semantically valid under this interpretation.

$$\frac{(R,f,S) \quad S \leq_B S'}{(R,f,S')} \qquad \frac{R' \leq_A R \quad (R,f,S)}{(R',f,S)}$$

(where f : A → B)

Intersection types for a functor

A functor $p : \mathcal{D} \to \mathcal{T}$ has (binary) **intersection types** if for every pair of types $S_1 \sqsubset A$ and $S_2 \sqsubset A$ refining the same type, there is a type $S_1 \wedge S_2 \sqsubset A$ together with a pair of derivations $\pi_1 : S_1 \wedge S_2 \leq_A S_1$ and $\pi_2 : S_1 \wedge S_2 \leq_A S_1$, such that for any pair of derivations $\beta_1 : (R,f,S_1)$ and $\beta_2 : (R,f,S_2)$, there exists a unique derivation $\beta' : (R,f,S_1 \wedge S_2)$ with $\beta_i = \beta'\pi_i$ for $i \in \{1,2\}$.

[This generalizes the definition from fibred category theory of when a fibration $q : \mathcal{E} \to \mathcal{B}$ has "fibred finite products".]

$\mathrm{Der}(R,f,S_1 \wedge S_2) \cong \mathrm{Der}(R,f,S_1) \times \mathrm{Der}(R,f,S_2)$

Two powerful type constructors

[= "is a bifibration"]

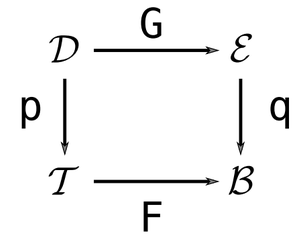
A functor p : $\mathcal{D} \to \mathcal{T}$ has ***push and pull types*** if:

* for every type R ⊏ A and term f : A → B, there is a type push(f,R) ⊏ B and a derivation α : (R,f,push(f,R)) such that for any derivation β : (R,fg,S) there is a unique derivation β' : (push(f,R),g,S) with β = αβ'.

* for every term g : A → B and type S ⊏ B, there is a type pull(g,S) ⊏ A and a derivation α : (pull(g,R),g,S) such that for any derivation β : (R,fg,S) there is a unique derivation β' : (R,f,pull(g,S)) with β = β'α.

Der(push(f,R),g,S) ≅ Der(R,fg,S) ≅ Der(R,f,pull(g,R))

Models



A model of a type refinement system p : $\mathcal{D}$ → $\mathcal{T}$ inside another type refinement system q : $\mathcal{E}$ → $\mathcal{B}$ is a pair of functors F : $\mathcal{T}$ → $\mathcal{B}$ and G : $\mathcal{D}$ → $\mathcal{E}$ such that Gq = pF.

For example, a model of p in q : Subset → Set interprets terms as functions, type refinements as subsets, and derivations as valid inclusions. A model in Psh → Cat interprets terms as functors, type refinements as presheaves, and derivations as natural transformations.

A brief bibliography

Papers by M&Z:

* Type refinement and monoidal closed bifibrations, arXiv:1310.0263
* Functors are type refinement systems, POPL 2015
* A bifibrational reconstruction of Lawvere's presheaf hyperdoctrine, LICS 2016
* An Isbell duality theorem for type refinement systems, MSCS 28:6, 2017

See also OPLSS 2016 lecture notes by Z.

See also work by Mazza (habilitation, 2017) and Mazza-Pellissier-Vial (POPL 2018) on (non-idempotent) intersection type systems.

Outlook

I believe that modelling type systems as functors rather than as categories is a better starting point for understanding what type systems do mathematically.

But clearly it is just a starting point.

One of our original goals in developing this fibrational framework was to have a language for describing the machinery of type inference and proof search. This is still a long-term aim...

2. (Bi)fibrations of proofs

Categorical logic

It is well-recognized since Lawvere that existential and universal quantification in predicate logic implicitly involve a fibrational structure.

It is less widely appreciated that the connectives of linear logic may also be analyzed in fibrational terms.

To motivate this, let's review the connection between monoidal categories and representable multicategories, which (as noticed by Hermida) may in turn be seen as covariant fibrations of multicategories...

Monoidal categories

A category C equipped with:

- a bifunctor $\otimes : C \times C \to C$ and an object $I \in C$
- three natural isomorphisms

$$\alpha_{A,B,C} : (A \otimes B) \otimes C \xrightarrow{\simeq} A \otimes (B \otimes C)$$

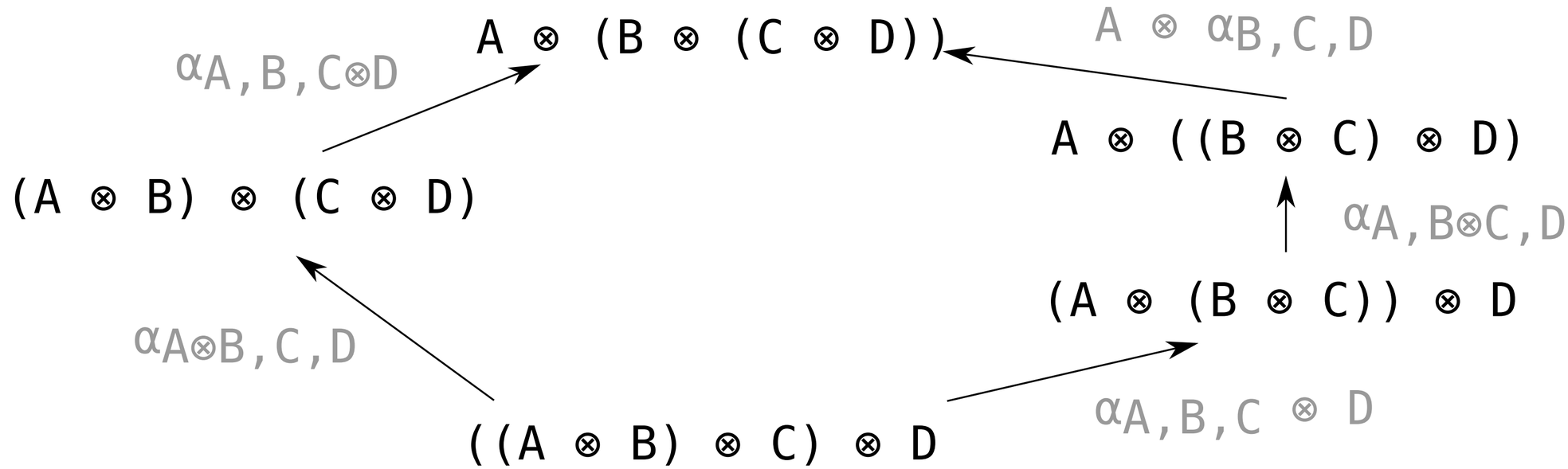
$$\lambda_A : I \otimes A \xrightarrow{\simeq} A$$

$$\rho_A : A \otimes I \xrightarrow{\simeq} A$$

- satisfying some coherence equations...

Monoidal categories

...such as the pentagon equation:



Thm (Mac Lane 1963, Kelly 1964): "all diagrams commute".

Multicategories

Recall that a ***multicategory*** (Lambek 1969) is like a category except that morphisms can have multiple inputs $f : A_1,...,A_n \to B$, and composition has the form of the *cut-rule* in intuitionistic linear logic

$$\frac{f \quad \Omega \to A \qquad g \quad \Gamma\ ,\ A\ ,\ \Delta \to C}{\Gamma\ ,\ \Omega\ ,\ \Delta \to C} = \mathrm{cut}_{\Gamma\text{-}\Delta}(f,g) = g \circ_i f \text{ where } i = |\Gamma|$$

subject to some equations omitted here.

Representable multicategories

A multicategory is said to be ***representable*** if for any list of objects Ω there is an object ⊗Ω and a multimap m_Ω : Ω → ⊗Ω such that for any multimap g : Γ,Ω,Δ → C there exists a unique multimap g' : Γ,⊗Ω,Δ → C with g = g'∘$_i$$m_\Omega$. Note this is equivalent to having ***tensors and unit object***, defined as above for Ω = A,B and Ω = ·.

Thm (Lambek 1969, Hermida 2000): "monoidal categories and representable multicategories are equivalent".

Hom(Γ,⊗Ω,Δ;C) ≅ Hom(Γ,Ω,Δ;C)
Hom(Γ,A⊗B,Δ;C) ≅ Hom(Γ,A,B,Δ;C)
Hom(Γ,I,Δ;C) ≅ Hom(Γ,Δ;C)

Fibrations of multicategories

The notion of (bi)fibration may be extended to functors of multicategories in different ways. Hermida's (2004) definition of a ***covariant fibration of multicategories*** $p : \mathcal{E} \to \mathcal{B}$ may be expressed as follows, in the (suitably adapted) language of type refinement systems:

* for any list of types $\Omega = (R_1 \sqsubset A_1,...,R_n \sqsubset A_n)$ and term $f : A_1,...,A_n \to B$, there is a type $push(f,\Omega) \sqsubset B$ and a derivation $\alpha : (\Omega,f,push(f,\Omega))$ such that for any derivation $\beta : ((\Gamma,\Omega,\Delta),g \circ_i f,S)$ there is a unique derivation $\beta' : ((\Gamma,push(f,R),\Delta),g,S)$ with $\beta = \beta' \circ_i \alpha$.

$Der((\Gamma,push(f,\Omega),\Delta),g,S) \cong Der((\Gamma,\Omega,\Delta),g \circ_i f,S)$

Fibrations of multicategories

Let $\mathbb{1}$ be the ***terminal multicategory*** defined as having a single object * and with a single morphism $n : *^n \to *$ for every $n \in \mathbb{N}$. Observe that for any multicategory $\mathcal{M}$ there is a unique functor of multicategories $! : \mathcal{M} \to \mathbb{1}$.

Proposition (Hermida): $\mathcal{M}$ is representable iff $! : \mathcal{M} \to \mathbb{1}$ is a covariant fibration.

Proof: take $\otimes\Omega := \mathrm{push}(n,\Omega)$ where $n = |\Omega|$.

(covariant) (of multicategories)

$\therefore$ monoidal categories $\cong$ fibrations over $\mathbb{1}$!

Bifibrations of proofs

The story extends to classical MLL: by introducing an appropriate notion of bifibration of polycategories, one obtains an equivalence between *-autonomous categories and bifibrations over the terminal polycategory. For details, see Nicolas Blanco's talk from MFPS 2020 and our joint paper.

In another direction, one can replace $\mathbb{1}$ by something more interesting to express a richer algebra of contexts. A programme for encoding substructural and modal logics this way based on bifibrations of 2-multicategories was proposed by Licata, Riley, and Shulman, see their paper at FSCD 2017.

3. Parsing as a lifting problem

(work-in-progress w/PAM)

An old analogy: parsing ~ typing

ON THE CALCULUS OF SYNTACTIC TYPES[1] (1961)

BY

JOACHIM LAMBEK

In classical physics it was possible to decide whether an equation was grammatically correct by comparing the "dimensions" of the two sides of the equation. These dimensions formed an abelian group with three generators L, M and T, admitting fractional exponents.[2]

One may ask whether it is similarly possible to assign "grammatical types" to the words of English in such a way that the grammatical correctness of a sentence can be determined by a computation on these types. As long as "John loves Jane" fails to imply "Jane loves John" one cannot expect these types to form an abelian group. Probably they should not form a group at all.

First steps

Any context-free grammar G over an alphabet Σ can be encoded by a functor of multicategories $\mathcal{D}$[G] → $\mathcal{W}$[Σ]:

* $\mathcal{W}$[Σ] is the ***operad of spliced words***, a one-object multicategory with an operation w_0-w_1-...-w_n : $*^n$ → * for every (n+1)-tuple of words in Σ*.

* $\mathcal{D}$[G] is the ***freely generated multicategory*** over the non-terminals with an operation r : R_1,...,R_n → R for every production r in G of the form R → $w_0R_1w_1...R_nw_n$.

* $\mathcal{D}$[G] → $\mathcal{W}$[Σ] sends R ↦ * and r ↦ (w_0-w_1-...-w_n).

(cf. RFC Walters, "A note on context-free languages", JPAA, 1989)

CFGs as type refinement systems

The functor $\mathcal{D}[G] \to \mathcal{W}[\Sigma]$ is not anything like a bifibration. Still, we can use the fibrational language of type refinement systems to analyze meaningful questions about the grammar, such as:

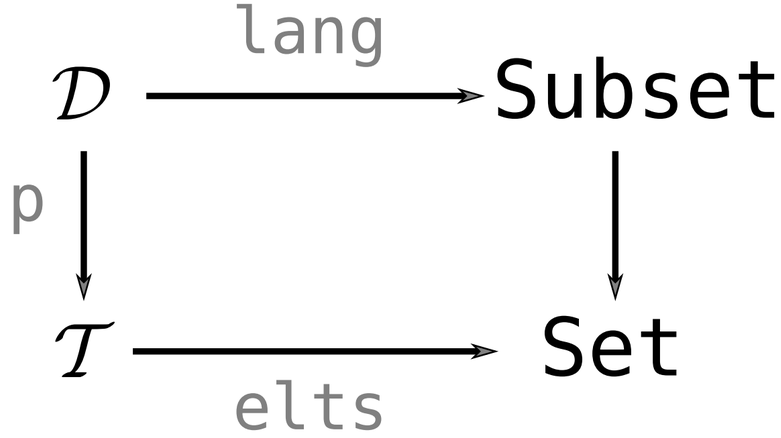
1. What is the language associated to a non-terminal?

2. What is the parse matrix associated to a word?

The language of a grammar

(of multicategories)

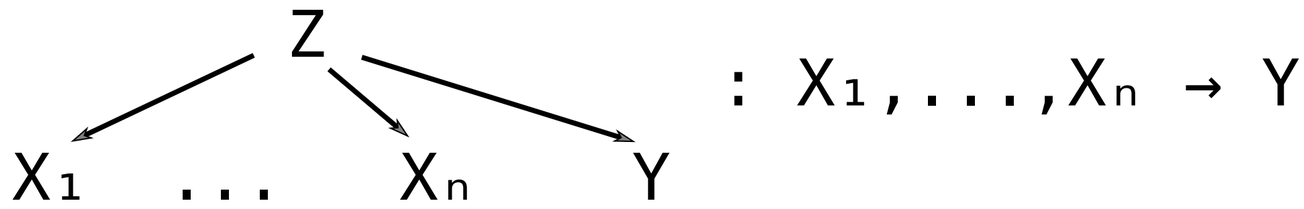
Any functor p : $\mathcal{D}$ → $\mathcal{T}$ has a canonical model in Subset → Set:



where elts sends every type A in $\mathcal{T}$ to the set of constants { c | c : A }, and lang sends every type R ⊏ A in $\mathcal{D}$ to the subset { p(α) | α : R } ⊆ elts(A). For $\mathcal{D}$[G] → $\mathcal{W}$[Σ], this recovers the language lang(R) ⊆ Σ* of each non-terminal.

The parse matrix of a grammar

Any functor p : $\mathcal{D}$ → $\mathcal{T}$ may be faithfully encoded by a *lax* functor ∂p : $\mathcal{T}$ → Span(Set) into the (weak) multicategory whose objects are sets and whose multimaps are multispans:



(Going from ∂p to p is a variant "Grothendieck construction".)

For $\mathcal{D}$[G] → $\mathcal{W}$[Σ], ∂p sends * to the set of non-terminals, and any word w to its set of parse trees equipped with the root-labelling function. (This generalizes to spliced words.)

4. Conclusions

Typing, proving, parsing...

Fibrational views on deductive systems:

* Naturally represented as functors p : $\mathcal{D} \to \mathcal{T}$

* Bifibrational connectives allow for expressing rich logical and algebraic structures...

* ...but not everything is a fibration!

* There are a wealth of techniques for type inference, proof search, parsing, etc., that deserve deeper analysis and are a potential source of beautiful mathematics.