Categorical Models of Subtyping

Greta Coraglia
Dipartimento di Filosofia (LUCI Lab), Dipartimento di Eccellenza 2023-2027,
Università degli Studi di Milano, Italy

Jacopo Emmenegger
Dipartimento di Matematica (DIMA), Dipartimento di Eccellenza 2023-2027,
Università degli Studi di Genova, Italy

Abstract

Most categorical models for dependent types have traditionally been heavily *set* based: contexts form a category, and for each we have a set of types in said context – and for each type a set of terms of said type. This is the case for categories with families, categories with attributes, and natural models; in particular, all of them can be traced back to certain discrete Grothendieck fibrations. We extend this intuition to the case of general, not necessarily discrete, fibrations, so that over a given context one has not only a set but a *category* of types.

We argue that the added structure can be attributed to a notion of subtyping that shares many features with that of *coercive* subtyping, in the sense that it is the product of thinking about subtyping as an abbreviation mechanism: we say that a given type A' is a subtype of A if there is a unique coercion from A' to A. Whenever we need a term of type A, then, it suffices to have a term of type A', which we can "plug-in" into A.

For this version of subtyping we provide rules, coherences, and explicit models, and we compare and contrast it to coercive subtyping as introduced by Z. Luo and others. We conclude by suggesting how the tools we present can be employed in finding appropriate rules relating subtyping and certain type constructors.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Categorical semantics

Keywords and phrases dependent types, subtyping, coercive subtyping, categorical semantics, categories with families, monad

Digital Object Identifier 10.4230/LIPIcs.TYPES.2023.3

Related Version *Previous Version*: `https://arxiv.org/abs/2312.14600`

Funding *Greta Coraglia*: The research has been partially funded by the Project PRIN2020 "BRIO" (2020SSKZ7R) awarded by the Italian Ministry of University and Research.
Jacopo Emmenegger: The research has been partially funded by the the Italian Ministry of University and Research under the scheme "Young Researchers" with agreement number SOE_0000071.

Acknowledgements The first author would like to thank the University of Genoa, where some of this research was completed. We are deeply grateful to Francesco Dagnino, who suggested coercive subtyping in the first place, for his immense clarity of thought. This work benefitted from a lovely visit to LAMA at Université Savoie Mont Blanc in November 2023, and the first author is for it indebted to Tom Hirschowitz and Peio Borthelle, and to the whole group there. Finally, we would like to thank Luca Mesiti for encouraging we explore the monad underlying our structure.

Introduction

The notion of subtyping is often quite tricky because of its double nature: on the one hand, it is meant to represent a relation so simple that one would like to not be bothered to look too much into it but, on the other hand, programming languages do not do well with things that are left unsaid. In [27, §15.1], subtyping $A \leq B$ is defined as a relation between two types A and B such that, if a term of type A is provided, than it can be "safely" used in a context where B is expected, and the program should not falter if this happens. This is often called the *principle of safe substitution*, and is encoded via a new typing rule that goes by the name of *subsumption*.

$$(\text{Sub})\frac{\Gamma \vdash a : A \quad \Gamma \vdash A \leq B}{\Gamma \vdash a : B} \tag{1}$$

The problem with subsumption is that it threatens to break certain structures and properties such as canonicity or induction principles. Many variants of subtyping have been proposed, and one in particular has proved itself to be well-behaved with respect to such issues, and that is *coercive subtyping* [21].

In the present work we tackle the problem of subtyping from the point of view of its categorical semantics: we extend a known model to naturally include a notion of subtyping (Section 1), and show how this turns out to be in fact quite closely related to coercive subtyping (Section 2). We then study some examples and applications (Section 3), and conclude by observing some properties that that this new notion intrinsically shows (Section 4).

1 Categorical models of dependent types

The relation between type theory and category theory is one which has been widely studied, and has in the years produced a large variety of models and structures. The interest of the present work is in those that strongly employ objects and techniques coming from the theory of Grothendieck fibrations.

1.1 Grothendieck fibrations

Fibrations were introduced by A. Grothendieck [9] for purposes pertaining to algebraic geometry, but were soon found extremely useful to describe certain phenomena in logic [16]. We will recall key results and definitions when needed, but we refer to [3, Chapter 8] for a detailed introduction.

▶ **Definition 1** (Cartesian morphism). *Let $p\colon \mathcal{E} \to \mathcal{B}$ a functor and $s\colon B \to A$ a morphism in $\mathcal{E}$. We say that s is p-*cartesian *or* cartesian *over $\sigma\colon \Theta \to \Gamma$ if $p(s) = \sigma$ and for any other $r\colon C \to A$ and τ such that $p(r) = \sigma \circ \tau$ there is a unique $t\colon C \to B$ in $\mathcal{E}$ such that $p(t) = \tau$ and $s \circ t = r$.*

$$\begin{tikzcd}
C \arrow[dr, dashed] \arrow[drr, bend left, "r"] & & & \\
& B \arrow[r, "s"'] & A & \mathcal{E} \arrow[dd, "p"] \\
\Xi \arrow[dr, "\tau"'] \arrow[drr, bend left, "\sigma\circ\tau"] & & & \\
& \Theta \arrow[r, "\sigma"'] & \Gamma & \mathcal{B}
\end{tikzcd} \tag{2}$$

*We say that s is a (p-)*cartesian lifting *of σ, and it is often denoted $s_{A,\sigma}\colon \sigma^* A \to A$.*

▶ **Definition 2** (Fibration). *A functor $p\colon \mathcal{E} \to \mathcal{B}$ is a* fibration *if for all A in $\mathcal{E}$, each $\sigma : \Theta \to pA$ has a cartesian lifting. We also say that $\mathcal{E}$ is* fibered over $\mathcal{B}$ *or that $\mathcal{E}$ is* over $\mathcal{B}$. *Oftentimes $\mathcal{B}$ is called the* base category *and $\mathcal{E}$ the* total category *of p.*

From the point of view of the logic, we can consider $\mathcal{B}$ to be a category of contexts and substitutions/terms, and $\mathcal{E}$ to be a category of formulae/types, so that each formula is sent to its corresponding context: using fibrations in this way is very much in the spirit of classical categorical logic (see for instance [17, 24, 12, 23]). Asking that for each term and formula there exists a lifting, then, amounts to asking that substitution can be computed and that it indeed produces a formula in then new context. A classical example is that of the Lindenbaum-Tarski algebra for a given first-order theory (Section 3.1).

A particular feature of fibrations is that they induce a factorization system on their total category, because for any given $r\colon C \to A$, we can always instantiate the diagram in (2) as follows,

$$\begin{tikzcd}
C \arrow[d] \arrow[dr, bend left, "r"] & & \\
\sigma^* A \arrow[r, "s_{A,\sigma}"'] & A & \mathcal{E} \arrow[dd, "p"] \\
\Theta \arrow[d, "\mathrm{id}"'] \arrow[dr, bend left, "\sigma"] & & \\
\Theta \arrow[r, "\sigma"'] & \Gamma & \mathcal{B}
\end{tikzcd}$$

producing a factorization of r. We refer to [26] for a review of its main features. Maps in the left class of such factorization system, meaning maps that are sent to identities, are called *vertical*.

▶ **Proposition 3** (Vertical/cartesian factorization system). *Consider $p\colon \mathcal{E} \to \mathcal{B}$ a fibration. The classes of vertical and cartesian morphisms form an orthogonal factorization system on $\mathcal{E}$. It additionally has the following properties:*
1. *if g and gf are vertical, then so is f;*
2. *pullbacks of vertical maps along cartesian ones exist and are vertical.*

If cartesian maps can be thought of as substitutions, vertical maps relate formulae/types in the same context precisely in a way that "does nothing" to the underlying terms.

Not only are vertical maps part of a factorization system, but they can be shown to form a category: more precisely, for each context Γ in $\mathcal{B}$, one can define a category $\mathcal{E}_\Gamma$ having for objects those in $\mathcal{E}$ that are sent to Γ, and for morphisms those in $\mathcal{E}$ that are vertical over id_Γ. This intuition is part of one of the most meaningful results of the theory, again due to Grothendieck.

▶ **Theorem 4** ([9]). *There exists a 2-equivalence*

$$\mathbf{Fib}(\mathcal{B}) \cong \mathbf{Psd}[\mathcal{B}^{\mathrm{op}}, \mathbf{Cat}]$$

between the 2-category of fibrations (with base $\mathcal{B}$), functors preserving cartesian maps, and natural transformations, and that of contravariant pseudofunctors (from $\mathcal{B}$), pseudonatural transformations, and modifications.

From left to right, a fibration is sent to a (pseudo)functor that computes for each Γ its *fiber* $\mathcal{E}_\Gamma$. From right to left, it performs what is called the *Grothendieck construction* of a (pseudo)functor, which to an $F\colon \mathcal{B}^{\mathrm{op}} \rightsquigarrow \mathbf{Cat}$, maps the fibration $p\colon \int F \to \mathcal{B}$, where objects of $\int F$ are pairs (Γ, A) with Γ in $\mathcal{B}$ and A in $F(\Gamma)$.

Relevant classes of fibrations are those that correspond to functors, which are called *split*, those whose fibers are preorders, which are called *faithful*, and those whose fibers are sets, which are called *discrete*.

1.2 Categories with families

▶ **Definition 5** (Category with families, [7]). *A* category with families (cwf) *is the data of*

- *a small category $\mathcal{B}$ with terminal object $\top$;*
- *a functor $Ty\colon \mathcal{B}^{\mathrm{op}} \to \mathbf{Set}$;*
- *a functor $Tm\colon \int Ty^{\mathrm{op}} \to \mathbf{Set}$*
- *for each Γ in $\mathcal{B}$ and A in $Ty(\Gamma)$ an object $\Gamma.A$ in $\mathcal{B}$, together with two projections $\mathsf{p}_A\colon \Gamma.A \to \Gamma$ and $\mathsf{v}_A \in Tm(\Gamma.A, Ty\,\mathsf{p}_A(A))$ such that for each $\sigma\colon \Theta \to \Gamma$ and $a \in Tm(Ty\sigma(A))$ there exists a unique morphism $\Theta \to \Gamma.A$ making the obvious triangles commute.*

In particular, $\mathrm{Ty}(\Gamma)$ is a *set*, the set of types in context Γ, and $\mathrm{Tm}(\Gamma, A)$ is again a set, the set of terms of type A in context Γ. The terminal object is meant to model the empty context, while the last condition is what provides context extension – since we are about to give a simpler, equivalent form of it, we do not dwell on it any longer.

Following Theorem 4, we can turn Definition 5 upside down. What we get is the (equivalent, see [2, Prop. 1.2]) notion of natural model.

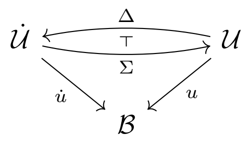
▶ **Definition 6** (Natural model, [2]). *A* natural model *is the data of*

- *a small category $\mathcal{B}$ with terminal object $\top$;*
- *a discrete fibration $u\colon \mathcal{U} \to \mathcal{B}$;*
- *a discrete fibration $\dot{u}\colon \dot{\mathcal{U}} \to \mathcal{B}$;*
- *a fibration morphism $\Sigma\colon \dot{u} \to u$ with a right adjoint functor.*

In this case u collects types and $\dot{u}$ collects terms, as both are fibered on contexts.

$$\begin{array}{cc} \Gamma \vdash A\,\mathtt{Type} & \Gamma \vdash a : A \\ u(A) = \Gamma & \dot{u}(a) = \Gamma, \Sigma(a) = A \\ u \text{ in } \mathbf{Fib}^{disc}(\mathcal{B}) & \dot{u} \text{ in } \mathbf{Fib}^{disc}(\mathcal{B}) \end{array}$$

The fibration morphism (*i.e.* a functor making the diagram below commute) Σ maps to each term its type, while Δ computes for each A what in Definition 5 is called v_A, meaning a variable in A "transported" to the context $\Gamma.A$.



$$\begin{tikzcd} \dot{\mathcal{U}} \arrow[rr, bend right, "\Sigma"'] \arrow[rr, phantom, "\top"] \arrow[dr, "\dot{u}"'] & & \mathcal{U} \arrow[ll, bend right, "\Delta"'] \arrow[dl, "u"] \\ & \mathcal{B} & \end{tikzcd} \qquad (\Sigma)\frac{\Gamma \vdash a : A}{\Gamma \vdash A\,\mathtt{Type}} \quad (\Delta)\frac{\Gamma \vdash A\,\mathtt{Type}}{\Gamma.A \vdash \mathsf{v}_A : A} \tag{3}$$

1.3 Generalized categories with families

Our intuition, now, is that we want to use the theory of fibrations to generalize categories with families (or natural models) to the case where fibers over a given context are no longer a set, but a (small) category. If we manage to do this swiftly, we will have found a model for dependent types that introduces and additional relation between types in the same context, so that this relation does imply absolutely no substitution. This leads us to the following definition.

▶ **Definition 7** (Generalized category with families, [5, 4]). *A* generalized category with families (gcwf) *is the data of*

- *a small category $\mathcal{B}$ (with terminal object $\top$)[1];*

[1] Existence of the terminal object is needed in case one wants to model the empty context – which one often wants to do. We only put its existence as conditional because we want to begin comparing gcwfs with comprehension categories, and they in turn do not require it.

- *a fibration $u\colon \mathcal{U} \to \mathcal{B}$;*
- *a fibration $\dot{u}\colon \dot{\mathcal{U}} \to \mathcal{B}$;*
- *a fibration morphism $\Sigma\colon \dot{u} \to u$ with a right adjoint functor, and unit and counit with cartesian components.*

Notice that the adjoint pair in Definition 7 is *not* fibered, as Δ does not make the desired triangle commute and unit and counit have cartesian components.

Of course a gcwf with discrete $u, \dot{u}$ is a regular cwf, so that a gcwf is simply a generalization of a well-known model. We choose the name as to remark that this new structure falls into a long tradition of models, which categories with families is perhaps one of the most prominent exponents of, but these could be very easily called "generalized natural models", or something entirely different (cf. [5]). We conclude this section with one last result relating gcwfs to other notable structures, namely comprehension categories [11], as to show that our path did not stray much away from known territory.

▶ **Theorem 8** ([6, 4]). *There is a biequivalence between (the 2-category of) gcwfs and (the 2-category of) comprehension categories.*

A generalized category with families, then, is precisely *as good a model as* a comprehension category. The purpose of introducing the new structure, though, lies in the clarity of its use, and we hope that the rest of the present work will be a witness to that.

▶ Remark 9. In our exposition we put aside two elements in the discussion on categorical models for dependent types. The first is of course the issue of coherence: since fibrations involve *pseudo*functors, equations for identities and, especially, associativity only hold up to vertical isomorphism, while it is usually preferable to have substitution "on the nose". If one so wishes, the reader is invited to only use *split* fibrations, meaning those that correspond to strict functors, and morphisms that preserve the splitting. A thorough discussion on the topic can be found in [30, §3].

Another sensitive topic is that of definitional equality. For the moment, having other purposes in mind, we settle for interpreting it as identity of objects (in the category over each context) and, therefore, rarely make explicit coherence rules involving it, as they are all trivial from our definitions. This is of course not unprecedented both for the literature on categorical semantics of type theory at large (cf. the largely influential [12] and [7]) and for semantics at large, as it lies in the path of the approach better known as *denotational semantics* (cf. [29]).

2 Vertical maps induce a notion of subtyping

The idea that vertical maps could nicely relate types in the same context is of course not new. The most closely related precedent to this is perhaps [25], followed by the notes in [32], where, quite radically, functors were interpreted to *be* type refinement systems. Our work is similar in spirit – though one should really be careful of the difference between "type refinement" and "subtyping" [32, §2.2] – but of course based on fibrations instead on functors, hence with a special focus on substitution.

2.1 Two new judgements

Now that we are all set, let us describe what it is that we can say in a gcwf, that was not already available in the discrete case. To the two judgements pertaining to types and terms, we add two new ones involving subtyping, and collect them all in Table 1. We read

Table 1 Judgements of the theory.

$\Gamma \vdash A\,\mathsf{Type}$	$\Gamma \vdash a : A$	$\Gamma \vdash A' \leq_f A$	$\Gamma \vdash a :_g A$
$u(A) = \Gamma$	$\dot{u}(a) = \Gamma, \Sigma(a) = A$	$f \colon A' \to A,\ u(f) = \mathrm{id}_\Gamma$	$g \colon \Sigma a \to A,\ u(g) = \mathrm{id}_\Gamma$

the new judgements, respectively, as *A' is a subtype of A, as witnessed by f* and *a is a term of type a subtype of A, as witnessed by g*. Recall that vertical maps enjoy some nice properties (Proposition 3), and their combination provides us with structural rules for the new judgements.

▶ **Proposition 10** (Structural subtyping rules). *The following rules are satisfied by a gcwf.*

$$\textit{(Sbsm)}\ \frac{\Gamma \vdash a :_g A' \qquad \Gamma \vdash A' \leq_f A}{\Gamma \vdash a :_{fg} A} \qquad \textit{(Trans)}\ \frac{\Gamma \vdash A' \leq_f A \qquad \Gamma \vdash A'' \leq_g A'}{\Gamma \vdash A'' \leq_{fg} A}$$

$$\textit{(Sbst)}\ \frac{\Gamma.A \vdash B' \leq_f B \qquad \Gamma \vdash a :_g A}{\Gamma \vdash B'[a] \leq_{\dot{u}(\Delta g \eta_a)^* f} B[a]} \qquad \textit{(Wkn)}\ \frac{\Gamma \vdash A' \leq_f A \qquad \Gamma \vdash B\,\mathit{Type}}{\Gamma.B \vdash A' \leq_{(u\epsilon_B)^* f} A}$$

Intuitively, Subsumption and Transitivity are both due to the fact that vertical maps compose to vertical maps, while Substitution and Weakening make use of the substitution structure due to the fibration part of the system. We refer to the appendix for further details.

Note that a gcwf is also equipped with a notion of "sub-typing" for terms as well. However, the hom-set $\dot{\mathcal{U}}_\Gamma(a, b)$ of vertical arrows over Γ is in bijection with the set of vertical arrows $f \colon \Sigma a \to \Sigma b$ such that $(\dot{u}\Delta f)\dot{u}\eta_a = \dot{u}\eta_b$ (see [6, Lemma 3.16]). Therefore a term a is a sub-term of b precisely when the type of a is a sub-type of the type of b, and substituting b (*i.e.* reindexing along $\dot{u}\eta_b$) is the same as substituting a, modulo the fact that Σa is a sub-type of Σb.

2.2 Comparison with coercive subtyping

As we said in the introduction, it turns out that the calculus resulting in this generalization comes close to coercive subtyping. We hope to convince the reader that, despite all the technical differences, they actually entertain the same spirit.

> The basic idea of coercive subtyping is that subtyping is modelled as an abbreviation mechanism: A is a subtype of B, if there is a unique coercion c from A to B, written as $A <_c B$. Then, if a hole in a context requires an object of type B, it is legal to supply an object a of type A – it is equivalent to supplying the object $c(a)$. [22]

A coercion c from A to B is technically a term of the function type $A \to B$, hence computing $c(a)$ amounts to function application. Coercions are added "manually" to a given type theory, as they are *a priori* not part of the calculus, and the resulting system is shown both to be a conservative extension of the original one and to act well with respect to canonicity.

Let us now collect in a table the main similarities and differences between coercive subtyping and subtyping via vertical maps. We then will consider each point in detail.

coercive subtyping	*categorical subtyping*
$\Gamma \vdash f : A' \to A$	$f \colon A' \to A$ vertical over Γ
judgements added to the calculus	judgements "added" to the classical model
no witnesses for typing judgements	witnesses for typing judgements
f is unique	f is not necessarily unique
(Sbsm) via substitution	(Sbsm) via composition
satisfies (Trans)	satisfies (Trans)
satisfies (Sbst)	satisfies (Sbst)
satisfies (Wkn)	satisfies (Wkn)
satisfies congruence	satisfies congruence

The main technical difference is of course what coercions *are*, as on one side they are function terms, on the other they are (particular) morphisms in the total category. In some sense this difference is unavoidable: if one starts from the traditional syntax of a type theory, syntactic objects are all that is available, while if one looks at a very general model as if it *was* a syntax (and one would have some merit in doing so, see Section 3.1), then they have more objects at hand. In both cases, though, subtyping is a notion that is somehow independent of the calculus, because in one case it is added by selecting a choice of witnesses (and adding one new judgement for each), while on the other it is literally *orthogonal* to the rest of the structure.

While we will deal with the matter of uniqueness of coercions in Section 2.3, a difference that is unbridgeable is that of dealing with typing judgements: in coercive subtyping, a term can indeed have more than one type, so that a typing judgments is in some sense ambiguous, while in subtyping with vertical maps one can always unambiguously know *by which means* a term is of a certain type, but for this it pays the price of having a whole new set of annotated judgements.

Finally, validity with respect to rules appearing in the table above is common to both systems – by "congruence", in particular, we mean congruence of the subtyping relation with respect to definitional equality, see Remark 9. It should be remarked that rules appearing on the "coercive subtyping" side are not all rules required, for example, in [22], but they are in a sense the structural ones, as the others aim to discipline either how coercions (as terms of function type) interact with the system, while in our case that is granted due to the fact that they come a categorical setting, which automatically ensures a certain degree of "coherence" (*e.g.* $\leq$ is automatically reflexive, transitive, and functorial).

2.3 On uniqueness of coercions

As we suggested in presenting this new perspective on subtyping, uniqueness of coercions is not really an issue. In particular, we have a whole theory of fibrations whose fibers are preorders, we actually call them *faithful* fibrations (cf. Section 1.1).

Not only that, but we can show that, given a gcwf, if its type fibration is faithful, then so is its term fibration.

▶ **Proposition 11.** *Let* $(u, \dot{u}, \Sigma \dashv \Delta)$ *be a gwcf. Then* Σ *is a faithful functor.*

▶ **Corollary 12.** *Let* $(u, \dot{u}, \Sigma \dashv \Delta)$ *be a gwcf. If* u *is faithful, then so is* $\dot{u}$*.*

In this sense, faithfulness, hence uniqueness of coercions, can indeed be modeled by simply looking at faithful fibrations, and such a burden is in fact not laid on the choice of the fibration collecting terms.

Table 2 Judgements of the theory – faithful case.

$$\Gamma \vdash A\,\texttt{Type} \qquad \Gamma \vdash a : A \qquad \Gamma \vdash A' \leq A \qquad \Gamma \vdash a :_{\leq} A$$

From the point of view of the syntax, then, judgements in Table 1 only have one witness and become as in Table 2, with $a :_{\leq} A$ denoting that $\Sigma a \leq A$, and rules in Proposition 10 become the (possibly) more familiar following.

$$(\text{Sbsm})\ \frac{\Gamma \vdash a :_{\leq} A' \qquad \Gamma \vdash A' \leq A}{\Gamma \vdash a :_{\leq} A} \qquad (\text{Trans})\ \frac{\Gamma \vdash A' \leq A \qquad \Gamma \vdash A'' \leq A'}{\Gamma \vdash A'' \leq A}$$

$$(\text{Sbst})\ \frac{\Gamma.A \vdash B' \leq B \qquad \Gamma \vdash a :_{\leq} A}{\Gamma \vdash B'[a] \leq B[a]} \qquad (\text{Wkn})\ \frac{\Gamma \vdash A' \leq A \qquad \Gamma \vdash B\,\texttt{Type}}{\Gamma.B \vdash A' \leq A}$$

▶ Remark 13 (A case against uniqueness). Though it is a key feature of coercive subtyping, avoiding uniqueness might have its merits. Consider for example the case of sum types: it seems like one should have two different witnesses for the judgement $\Gamma \vdash A \leq A + A$, one per coproduct injection.

2.4 Comparison with related variants of type theory

Directed type theory

Extending the paradigm connecting types, ω-groupoids, and homotopy theory (cf. [31]), to the directed case – meaning involving ω-categories and directed homotopy theory, one encounters the notion of directed type theory. The underlying intuition wishes to add, for each pair of terms of the same type (and substitutions), a new *asymmetric* "identity" type of transformations from one to the other, possibly stopping at some given height/iteration [19]. Though vertical in some sense, this notion of directed-ness is considered only for terms, while in our case the main focus is for types (though one could regard types as terms of a given universe, in DTT transformations are introduced for *all* pairs of terms). Being based on (ω-)categories, they share some properties of morphisms, such as composition and identity, but our vertical maps are a lot simpler, and in no way higher-dimensional – though the monad in Theorem 25 suggests possible extensions in this direction.

Observational type theory

Observational type theory was introduced in [1] to combine the nice computational features of intensional type theory, such as termination of reductions, and propositional equality of extensional type theory, so that two functions are equal if they are equal point-wise, or "if all observations about them agree" [1, §1]. The fact that conversion rules allow to pass implicitly between definitionally equal types is extended to a mechanism that allows to pass implicitly between *provably* equal types. This process is explicit and its agents are called *coercions*. Coercions for a OTT work in a way that is much similar to our vertical maps, but their being originated by proofs of equality of types ensures that for a given coercion, one can always find one associated to it and going in the opposite direction.

Practical subtyping

Another relevant syntactic approach is that of "practical subtyping" as introduced in [18]. There, the new subtyping judgements are *ternary* relations as $t \in A \subset B$, which are encoded as sorts of implications: "if t is a term of type A, then it is a term of type B". In particular, this notion of subtyping most notably (and "practically") describes subtyping from the perspective of terms, and not types, as in our case. In particular, subtyping between types, meaning intrinsic judgements such as $A \subseteq B$, are encoded by means of choice operators, such as Hilbert's ϵ [18, p. 26].

We can compare our categorical subtyping with the practical one as follows: again, consider the case where all fibrations involved are faithful (cf. Section 2.3), so that there is at most *one* subtyping judgement for each pair of types (or for a type and a term). In particular, judgements in [18, p. 22] seem to match our structural ones, when one replaces "$t : A$" with our "$t :_{\leq} A$". The treatment of type constructors, too, seems quite keen in spirit to ours, but we postpone an in-depth analysis of said comparison to future work.

3 Examples and applications

3.1 Gcwfs from Lindenbaum-Tarski

Consider the Lindenbaum-Tarski algebra of a given first-order theory $\mathcal{T}$ in a language $\mathcal{L}$. We take **ctx** to be the category where

- objects are lists of distinct variables $x = (x_1, \dots, x_n)$,
- arrows are lists of substitution for variables, meaning $[t_1/y_1, \dots, t_m/y_m] = [t/y] : x \to y$, with t_j's being $\mathcal{L}$-terms that are built on variables $x_1, \dots, x_n$,

and composition is defined by simultaneous substitution. The product of two lists x and y is a list w with length the sum of the lengths of x and y, and projections on x and y are substitutions with, respectively, the first n and the last m variables in w. Categorically and in the sense of [16], this is the free Lawvere theory on the language $\mathcal{L}$.

One can define the functor $LT_{\mathcal{T}} : \mathbf{ctx}^{\mathrm{op}} \to \mathbf{InfSL}$ so that to each list x, the category $LT_{\mathcal{T}}(x)$ has for objects equivalence classes of well-formed formulae in $\mathcal{L}$ with free variables at most those that are in x, and with respect to the equivalence relation induced by reciprocal deducibility in $\mathcal{T}$, $\phi \dashv\vdash_{\mathcal{T}} \phi'$. Notice that this makes our treatment *proof-irrelevant*. Maps in $LT_{\mathcal{T}}(x)$ are provable consequences in $\mathcal{T}$. Composition is given by the cut rule of the calculus, and identities are tautologies. Since substitution preserves provability, $LT_{\mathcal{T}}$ can be suitably extended to a functor, and its correspondent under the Grothendieck construction Theorem 4 is a faithful fibration $p\colon \int LT_{\mathcal{T}} \to \mathbf{ctx}$.

Such a construction was first introduced with the name of *(hyper)doctrine* in [17] and is thoroughly explained in [14] and [23]. We here show that it underlies the structure of a gcwf where terms are entailments.

▶ **Example 14.** Call $\mathcal{E}$ the category of p-vertical maps and commutative squares, which is again fibered over **ctx** – call e this fibration, $\mathsf{Cod}\colon \mathcal{E} \to \int LT_{\mathcal{T}}$ the codomain functor, Diag the functor mapping each formula to its identity. The triple $(p, e, \mathsf{Cod} \dashv \mathsf{Diag})$ is a gcwf.

This gcwf is in fact *not* a cwf. Here, types are formulae and terms are (unique) witnesses to entailment, meaning triples $x; \phi \vdash \psi$ where ϕ and ψ are both formulae in the fiber over x. The underlying notion of subtyping actually coincides with terms.

▶ Remark 15. This is an example of a more general instance, which will thoroughly be discussed in Section 4.

3.2 Gcwfs from topos theory

Let $\mathcal{E}$ be an elementary topos and $\top \colon 1 \to \Omega$ its subobject classifier, then consider

$$\begin{tikzcd}
\mathcal{E}_{/1} \arrow[rr, bend right, "\Sigma_\top"'] \arrow[dr, "\sim"'] & & \mathcal{E}_{/\Omega} \arrow[ll, bend right, "\Delta_\top"'] \arrow[dl, "\mathrm{dom}"] \\
& \mathcal{E} &
\end{tikzcd}$$

with $\Sigma_\top = \top \circ \text{-}$ and $\Delta_\top(\phi \colon X \to \Omega) = (\text{canonical p.b. of } \phi \text{ along } \top)$.

▶ **Example 16.** The triple $(\sim, \mathrm{dom}, \Sigma_\top \dashv \Delta_\top)$ is a gcwf.

Compatibly with the Mitchell-Bénabou interpretation, types are (proof irrelevant) propositions, $\Delta_\top$ computes the comprehension $\{x \mid \phi(x)\}$, and so on – we refer to [13, Part D], [15, Chap. II] for an extensive treatment of the topic, and to [20] for a quick overview. In this case, the fibrations involved are discrete, hence this is really is a cwf, and there is no subtyping.

A topos is quite a general structure, so let us break down a couple of examples for it. The prototypical example of an elementary topos is **Set**, the categories of sets and functions, with subobject classifier $\top \colon 1 \to 2$, $* \mapsto 1$, which classifies subsets via their characteristic function.

$$\begin{tikzcd}
A_\phi \arrow[r] \arrow[d, hook, "i_\phi"'] \arrow[dr, phantom, "\lrcorner", very near start] & 1 \arrow[d, "\top"] \\
A \arrow[r, "\phi"'] & 2
\end{tikzcd}$$

In this case terms are sets, types are functions $\phi \colon A \to 2$, which are equivalently subsets of A – and the reason why we usually call them *propositions*. The context for each proposition ϕ is its domain set A. Let us now break down rules as in (3),

$$(\Sigma_\top)\frac{A \vdash A}{A \vdash i_{\mathrm{id}_A}\ \texttt{Type}} \qquad (\Delta_\top)\frac{A \vdash \phi\,\texttt{Type}}{A_\phi \vdash A_\phi} \tag{4}$$

where we use the fact that $\Delta_\top(\phi) = \mathrm{dom} i_\phi = \{a \mid \phi(a)\} \subseteq A$ by definition. Notice how, again, Δ does not make the triangle commute, as, in principle, $A \neq A_\phi$.

Let us now look at a more complicated case, namely that of **Eff**, the effective topos [10, 28], whose objects are sets A with an $\mathbb{N}$-valued equality predicate $=_A \in \mathcal{P}(\mathbb{N})^{A \times A}$, and whose morphisms are $\mathcal{P}(\mathbb{N})$-valued functional relations: we think of each subset of $\mathbb{N}$ as the *realizers* (in the sense of Kleene) for a given proposition. One can show that **Eff** is a topos with subobject classifier $\top \colon (\{*\}, \mathbb{N}) \to (\mathcal{P}(\mathbb{N}), \leftrightarrow)$, where $A \leftrightarrow B := (A \to B) \wedge (B \to A)$ is a pair of recursive functions mimicking bi-implication [10, §1], and $\top(*, A) = [A \leftrightarrow \mathbb{N}]$. In this case, terms are sets with an $\mathbb{N}$-valued equality predicate, and types are subobjects of them, which can be shown to be sort of subsets with a compatible equality predicates and an additional "membership" unary predicate compatible with it.

3.3 Gcwfs from pullbacks

For $\mathcal{C}$ with pullbacks we can define $\mathcal{C}^{\to}$ the category of arrows in $\mathcal{C}$ and $\mathbf{Sec}(\mathcal{C})$ of sections of arrows in $\mathcal{C}$, meaning of pairs (s, f) with $f, s \in \mathcal{C}^{\to}$ and $f \circ s = \mathrm{id}$. Therefore a type in context A is a map $f \colon B \to A$ and a term of type f is one of its sections, s. For a given type

f (map) context extension can be performed using what is usually called its “kernel pair” construction, meaning computing the following pullback,

$$\begin{tikzcd}
B \arrow[dr, dashed, "! d(f)"] \arrow[drr, bend left, "\mathrm{id}"] \arrow[ddr, bend right, "\mathrm{id}"'] & & \\
& K_f \arrow[r] \arrow[d, "f^+"] \arrow[dr, phantom, "\lrcorner", very near start] & B \arrow[d, "f"] \\
& B \arrow[r, "f"'] & A
\end{tikzcd}$$

which we can interpret as the universal f-induced congruence on B. One can always compute the section $B \to K_f$, which can be extended to a functor $K\colon \mathcal{C}^{\to} \to \mathbf{Sec}(\mathcal{C})$.

▶ **Example 17.** Call $U\colon \mathbf{Sec}(\mathcal{C}) \to \mathcal{C}^{\to}$ the functor mapping each pair (s, f), with s section of f, to f. The triple $(\mathrm{cod}, \mathrm{cod}U, U \dashv K)$ is a gcwf.

Actually, by Theorem 8, the same can be said for any $\mathcal{D}$ subcategory of $\mathcal{C}^{\to}$ closed for pullbacks, for example we can consider monomorphisms, and their sections. Notice that this is *not* a cwf, as vertical maps are commutative triangles.

Let us give a couple of examples of this case, as well. We look at **Set** again, for a function $f\colon B \to A$, we find that K_f is the set of pairs of elements of B with the same image through f, and $B \to K_f$ is the diagonal. We again try to break down rules as in (3), and get the following.

$$(U)\frac{A \vdash f\colon B \leftrightarrows A\colon s}{A \vdash f\colon B \to A\,\mathtt{Type}} \qquad (K)\frac{A \vdash f\colon B \to A\,\mathtt{Type}}{B \vdash f^+\colon K_f \leftrightarrows B\colon d(f)} \tag{5}$$

Again, it is important that the right adjoint K, the one performing context extension, does not make the triangle commute, as B is not (necessarily) equal to A. Let us now look at subtyping judgements: morphisms in $\mathcal{C}^{\to}$ are commutative squares, vertical ones are squares such that the codomain is the identity, hence commutative triangles. The whole set of structural judgements available in this model (cf. Section 2.1) is then the following.

$$A \vdash f\,\mathtt{Type} \quad \begin{tikzcd} \mathrm{dom} f \arrow[d, "f"] \\ A \end{tikzcd} \qquad A \vdash s : f \quad \begin{tikzcd} \mathrm{dom} f \arrow[d, shift left, "f"] \\ A \arrow[u, shift left, "s"] \end{tikzcd} \qquad A \vdash f' \leq_h f \quad \begin{tikzcd} \mathrm{dom} f' \arrow[rr, "h"] \arrow[dr, "f'"'] & & \mathrm{dom} f \arrow[dl, "f"] \\ & A & \end{tikzcd} \qquad A \vdash s :_k f \quad \begin{tikzcd} \mathrm{dom} f' \arrow[rr, "k"] \arrow[dr, shift left, "f'"] & & \mathrm{dom} f \arrow[dl, "f"] \\ & A \arrow[ul, shift left, "s"] & \end{tikzcd}$$

Of course one could consider a great multitude of other categories, such as **Pos**, **Top**, $\mathbf{Vect}_K$, **Grp**, all topoi, and many others.

3.4 On type constructors

We begin by considering function types of the form $\mathsf{Fun}(A, B)$, where A and B are in the same context (in particular, B does not depend on A). In this case, given a subtype A' of A and a subtype B of B' (note the different order), one would like to conclude that $\mathsf{Fun}(A, B)$ is a subtype of $\mathsf{Fun}(A', B')$. In our proposed framework, this can be expressed by requiring an action of Fun on vertical arrows, which is contravariant in the first component.

In the case of a discrete gcwf (that is, in the fibrational formulation of a natural model/cwf) the existence of function types can be expressed by requiring the existence of functors Fun and abs making the right-hand square below a pullback [2]

$$\begin{tikzcd}
\mathcal{U}_{\dot{u}\Delta}\times_{\dot{u}}\dot{\mathcal{U}} \arrow[d, "\mathrm{id}\times\Sigma"'] & \mathrm{W}^*(\mathcal{U}_{\dot{u}\Delta}\times_{\dot{u}}\dot{\mathcal{U}}) \arrow[l] \arrow[r, "\mathsf{abs}"] \arrow[d] & \dot{\mathcal{U}} \arrow[d, "\Sigma"] \\
\mathcal{U}_{\dot{u}\Delta}\times_{u}\mathcal{U} & \mathcal{U}_{u}\times_{u}\mathcal{U} \arrow[l, "\mathrm{W}"'] \arrow[r, "\mathsf{Fun}"] & \mathcal{U}
\end{tikzcd}$$

where the left-hand square is a pullback. The lower functor Fun simply takes two types A and B in the same context Γ to the type $\mathsf{Fun}(A, B)$. The functor W is weakening of the second type with the first one: it takes a pair (A, B) of two types A and B in the same context Γ to the pair $(A, (u\epsilon_A)^*B)$ where $(u\epsilon_A)^*B$ is the type B weakened to the context $\Gamma.A$. As the left-hand square is a pullback, the domain of the functor abs consists of pairs (A, b) of a type A in context Γ and a term b of the weakened type $(u\epsilon_A)^*B$ in context $\Gamma.A$. The functor abs maps such a pair to the term $\mathsf{abs}(A, b)$. Commutativity of the square ensures that the type of $\mathsf{abs}(A, b)$ is $\mathsf{Fun}(A, B)$. As it is not relevant for our discussion, we refer to [2] for details on how the pullback property of the right-hand square validates both the elimination rule and the η-rule.

To extend this setting to include subtyping in the form of vertical arrows, we need to take into account that the action of Fun (and abs) should be contravariant *only on the vertical arrows* of the first component. Contravariant actions of functors on some category $\mathcal{B}$ are rendered by considering functors on the opposite category $\mathcal{B}^{\mathrm{op}}$, but here we want to take the opposite only of vertical arrows, otherwise substitution, *i.e.* the action of Fun on cartesian arrows, would go in the wrong direction. We need to consider what is known as the *fiberwise opposite* $u^{\mathrm{o}} \colon \mathcal{U}^{\mathrm{o}} \to \mathcal{B}$ of u, see [30, Section 5]. The construction of u^{o} uses the orthogonal factorisation system on $\mathcal{U}$ given by vertical and cartesian arrows to construct a bicategory on the same objects of $\mathcal{U}$ whose 1-cells $B \to B'$ are spans $B \leftarrow \bar{B}' \rightarrowtail B'$ of a vertical arrow $B \leftarrow \bar{B}'$ and a cartesian one $\bar{B}' \rightarrowtail B'$. 2-cells are morphisms of spans, that is, arrows between the vertices making the two triangles commute. It turns out that 2-cells are unique (given two spans) and always invertible. The category $\mathcal{U}^{\mathrm{o}}$ induced by identifying two isomorphic 1-cells is again fibered over $\mathcal{B}$, where an arrow $[g, \bar{\sigma}]$ is cartesian (over $\sigma = \dot{u}\bar{\sigma}$) if g is invertible (equivalently, if there is a representative of the form $(\mathrm{id}, \bar{\sigma}')$). It is easy to see that, for every Γ, there is an isomorphism $(\mathcal{U}^{\mathrm{o}})_\Gamma \cong (\mathcal{U}_\Gamma)^{\mathrm{op}}$ natural in Γ. In fact, it is well-known (see *e.g.* [26, Theorem 3.7]) that the fibered category u^{o} is equivalent (over $\mathcal{B}$) to the fibered category $\mathrm{F}((-)^{\mathrm{op}} \circ \mathrm{P}(u))$, where $\mathrm{P} \colon \mathbf{Fib}(\mathcal{B}) \to \mathbf{Psd}[\mathcal{B}^{\mathrm{op}}, \mathbf{Cat}]$ and $\mathrm{F} \colon \mathbf{Psd}[\mathcal{B}^{\mathrm{op}}, \mathbf{Cat}] \to \mathbf{Fib}(\mathcal{B})$ are the two 2-functors realizing the 2-equivalence in Theorem 4.

▶ Remark 18. If we start with a fibration equipped with a (split) cleavage, then it is easy to see that u^{o} can also be equipped with a (split) cleavage. Indeed, we can pick $(A, \sigma) \mapsto [\mathrm{id}, s(A, \sigma)]$ as cleavage for u^{o}, where s is a cleavage for u. If s is split, then the above choice is clearly normal, and $(\mathrm{id}, s(A, \sigma \circ \rho))$ is a representative for $[\mathrm{id}, s(s_0(A, \sigma), \rho)] \circ [\mathrm{id}, s(A, \sigma)]$. Furthermore, a cleavage s for u provides us with a choice of a span $(g, s(A, \sigma))$ in each equivalence class. These choices compose (*i.e.* form a category) as soon as the cleavage is split. Therefore, in the case of a split fibration u, an arrow $A \to B$ in $\mathcal{U}^{\mathrm{o}}$ over some σ in $\mathcal{B}$ is actually just a vertical arrow $s_0(B, \sigma) \to A$.

▶ **Proposition 19.** *Let $(u, \dot{u}, \Sigma, \Delta)$ be a gcwf over $\mathcal{B}$ equipped with two morphisms of fibrations* Fun *and* abs *making the right hand square below a pullback over $\mathcal{B}$.*

$$\begin{tikzcd}
\mathcal{U}^{\circ}{}_{\dot{u}^{\circ}\Delta^{\circ}}\!\times_{\dot{u}} \dot{\mathcal{U}} \arrow[d, "\mathrm{id}\times\Sigma"'] & \mathrm{W}^{*}(\mathcal{U}^{\circ}{}_{\dot{u}^{\circ}\Delta^{\circ}}\!\times_{\dot{u}} \dot{\mathcal{U}}) \arrow[l] \arrow[r, "\mathsf{abs}"] \arrow[d] & \dot{\mathcal{U}} \arrow[d, "\Sigma"] \\
\mathcal{U}^{\circ}{}_{\dot{u}^{\circ}\Delta^{\circ}}\!\times_{u} \mathcal{U} & \mathcal{U}^{\circ}{}_{u^{\circ}}\!\times_{u} \mathcal{U} \arrow[l, "\mathrm{W}"'] \arrow[r, "\mathsf{Fun}"] & \mathcal{U}
\end{tikzcd}$$

Then, in addition to the usual formation, introduction, elimination and computation rules for function types, the following rules are satisfied.

$$\frac{\Gamma \vdash A' \leq_f A \qquad \Gamma \vdash B \leq_g B'}{\Gamma \vdash \mathsf{Fun}(A,B) \leq_{\mathsf{Fun}(f,g)} \mathsf{Fun}(A',B')} \qquad\qquad \frac{\Gamma.A \vdash b :_f B}{\Gamma \vdash \mathsf{abs}(A,b) :_{\mathsf{Fun}(\mathrm{id}_A,f)} \mathsf{Fun}(A,B)}$$

In order to extend Proposition 19 to the case of dependent function types, *i.e.* Π-types, we need to take into account two changes in variance. One is due, as in the non-dependent case, to the fact that Π acts contravariantly on the first argument. The second change in variance happens because the second argument depends on the first one, but the subtyping relation on second arguments goes in the opposite direction of the subtyping between on first arguments. These two changes in variance can be described as follows. Consider first a pullback of Σ as in the left-hand diagram below. Then take its fiberwise opposite, resulting in the morphism of fibrations in the middle below. Finally, compose that triangle with the fibration $u\colon \mathcal{U} \to \mathcal{B}$ and take again the fiberwise opposite of the result, obtaining the morphism of fibrations $\Sigma_{\sqcap}$ on the right-hand below, from $\dot{u}_{\sqcap} := (u \circ \dot{u}_1^{\circ})^{\circ}$ to $u_{\sqcap} := (u \circ u_1^{\circ})^{\circ}$.

$$\begin{tikzcd}
\dot{\mathcal{U}} \arrow[dr, "\Sigma"] \arrow[dd, "\dot{u}"'] & & \dot{\mathcal{U}}_1 \arrow[ll] \arrow[dr, "\Sigma_1"] \arrow[dd, "\dot{u}_1"' near end] & \\
& \mathcal{U} \arrow[dl, "u"] & & \mathcal{U}_1 \arrow[ll] \arrow[dl, "u_1"] \\
\mathcal{B} & & \mathcal{U} \arrow[ll, "\dot{u}\Delta"] &
\end{tikzcd}
\qquad
\begin{tikzcd}
\dot{\mathcal{U}}_1^{\circ} \arrow[dr, "\Sigma_1^{\circ}"] \arrow[dd, "\dot{u}_1^{\circ}"'] & \\
& \mathcal{U}_1^{\circ} \arrow[dl, "u_1^{\circ}"] \\
\mathcal{U} &
\end{tikzcd}
\qquad
\begin{tikzcd}
\dot{\mathcal{U}}_{\sqcap} \arrow[dr, "\Sigma_{\sqcap}"] \arrow[dd, "\dot{u}_{\sqcap}"'] & \\
& \mathcal{U}_{\sqcap} \arrow[dl, "u_{\sqcap}"] \\
\mathcal{B} &
\end{tikzcd}
\tag{6}$$

Unfolding the constructions involved, we see that an arrow $(f,[g,\bar{f}])\colon (A,B) \to (A',B')$ in $\mathcal{U}_{\sqcap}$ vertical over Γ consists of an arrow $f\colon A' \to A$ vertical over Γ together with a 1-cell $[g,\bar{f}]\colon B' \to B$ in $\mathcal{U}_1^{\circ}$ with $g\colon \bar{B} \to B'$ vertical over $\Gamma.A' = \dot{u}\Delta A'$ and $\bar{f}\colon \bar{B} \to B$ cartesian over $\dot{u}\Delta f\colon \Gamma.A' \to \Gamma.A$. Similarly, an arrow $(A,b) \to (A',b')$ in $\dot{\mathcal{U}}_{\sqcap}$ vertical over Γ consists of an arrow $f\colon A' \to A$ as before, together with a 1-cell $[\dot{g},\bar{f}]$ in $\dot{\mathcal{U}}_1^{\circ}$ with $\dot{g}\colon \bar{b} \to b'$ vertical over $\Gamma.A'$ and $\bar{f}\colon \bar{b} \to b$ cartesian over $\dot{u}\Delta f$. The following result follows.

▶ **Proposition 20.** *Let $(u,\dot{u},\Sigma,\Delta)$ be a gcwf equipped with two morphisms of fibrations* $\sqcap$ *and* abs *making the square below a pullback over $\mathcal{B}$.*

$$\begin{tikzcd}
\dot{\mathcal{U}}_{\sqcap} \arrow[r, "\mathsf{abs}"] \arrow[d, "\Sigma_{\sqcap}"'] & \dot{\mathcal{U}} \arrow[d, "\Sigma"] \\
\mathcal{U}_{\sqcap} \arrow[r, "\sqcap"] & \mathcal{U}
\end{tikzcd}$$

Then, in addition to the usual formation, introduction, elimination and computation rules for dependent function types, as well as the η-rule, the following rules are satisfied.

$$\frac{\Gamma \vdash A' \leq_f A \qquad \Gamma.A' \vdash B[\dot{u}\Delta f] \leq_g B'}{\Gamma \vdash \sqcap(A,B) \leq_{\sqcap(f,g)} \sqcap(A',B')} \qquad\qquad \frac{\Gamma.A \vdash b :_g B}{\Gamma \vdash \mathsf{abs}(A,b) :_{\sqcap(\mathrm{id}_A,g)} \sqcap(A,B)}$$

A type constructor like the dependent sum (*i.e.* Σ-types), on the other hand, requires no change in variance. Therefore its formulation for a gcwf is a straightforward generalization of the discrete case. The fibration classifying the premises is what is called u_1 in diagram (6) above: the objects, as expected, are the same of u_Π, but the vertical arrows $(A, B) \to (A', B')$ now are just pairs of a vertical arrow $f\colon A \to A'$ and $g\colon B \to B'$ over $\dot{u}\Delta f$. The construction of the fibration $\dot{u}_\Sigma$ classifying the premises of the introduction rule is slightly more involved, and we refer to [5, Example 3.7.4] for a complete description.

▶ **Proposition 21.** *Let $(u, \dot{u}, \Sigma, \Delta)$ be a gcwf equipped with two morphisms of fibrations Σ and* pair *making the square below a pullback over $\mathcal{B}$.*

$$\begin{tikzcd}
\dot{\mathcal{U}}_\Sigma \arrow[r, "\mathsf{pair}"] \arrow[d, "\Sigma_\Sigma"'] & \dot{\mathcal{U}} \arrow[d, "\Sigma"] \\
\mathcal{U}_1 \arrow[r, "\Sigma"] & \mathcal{U}
\end{tikzcd}$$

Then, in addition to the usual formation, introduction, elimination and computation rules for dependent sum types, as well as the η-rule, the following rules are satisfied.

$$\frac{\Gamma \vdash A \leq_f A' \qquad \Gamma.A \vdash B \leq_g B'[\dot{u}\Delta f]}{\Gamma \vdash \Sigma(A, B) \leq_{\Sigma(f,g)} \Sigma(A', B')} \qquad \frac{\Gamma \vdash a :_f A \qquad \Gamma.A \vdash b :_g B[a]}{\Gamma \vdash \mathsf{pair}(a, b) :_{\Sigma(f,g)} \Sigma(A, B)}$$

In writing the rules above in Propositions 20 and 21, we have written the action of reindexing as if the fibrations involved were split. This allows us to simplify notation in the rules, which would otherwise look quite cumbersome. In the case of non-split fibrations, a judgement like $\Gamma.A \vdash B \leq_g B'[\dot{u}\Delta f]$ in a premise should be read as "for all pairs of a cartesian $\bar{f}\colon \bar{B}' \to B'$ over $\dot{u}\Delta f$ and a vertical $g\colon B \to \bar{B}'$", and similarly for the other judgements. In particular, in the non-split case, the cartesian arrow $\bar{f}$ should appear in the conclusion as well, as one of the arguments of $\Sigma(f, g)$.

4 The "subtyping" monad

We now deepen our intuition for both the properties and the features of this new construction. In particular, the process of *taking into consideration* vertical maps turns out to amount to the result of the action of a particular monad. Grothendieck fibrations are known to be strongly linked to certain (co)monads, to the point of them being classified as pseudo-algebras of a given one (cf. [8] for a recent review and extension). We here present a simple monad that has the nice feature of collecting vertical maps, and show that it actually produces gcwfs out of gcwfs.

This section is perhaps best intended not as a *development*, but more of an additional perspective of our theory of subtyping: though it does not provide new results, it puts its emerging in context, and can hopefully be used in future works to combine different notions with this one – for example, what would a distributive law combining the monad here and that producing split fibrations [30, §3] mean, and what would it entail?

We start from the endofunctor $(\mathrm{Id}_{\_}/\mathrm{Id}_{\_})\colon \mathbf{Fib}(\mathcal{B}) \to \mathbf{Fib}(\mathcal{B})$ computing for each fibration $p\colon \mathcal{E} \to \mathcal{B}$ the *comma object* $(\mathrm{Id}_p/\mathrm{Id}_p)$, meaning the following (very boring) 2-limit below.

$$\begin{tikzcd}
(\mathrm{Id}_p/\mathrm{Id}_p) \arrow[r] \arrow[d] & p \arrow[d, "\mathrm{Id}_p"] \arrow[dl, Rightarrow, shorten=3mm] \\
p \arrow[r, "\mathrm{Id}_p"'] & p
\end{tikzcd}$$

To make the reding a bit smoother, we denote $(\mathrm{Id}_p/\mathrm{Id}_p)$ by (p/p). Unpacking this categorical construction, one finds that the new fibration $(p/p)\colon \mathcal{E}\downarrow^v \mathcal{E} \to \mathcal{B}$ has in the domain triples (f, A', A) such that $f\colon A' \to A$ is a p-vertical map, while morphisms in $\mathcal{E}\downarrow^v \mathcal{E}$ are simply commutative squares.

▶ **Lemma 22.** *There is a monad on* $\mathbf{Fib}(\mathcal{B})$ *with endofunctor* $(\mathrm{Id}_{-}/\mathrm{Id}_{-})$.

Its associated unit has components $\eta\colon p \to Tp$ sending each object to its identity, while the multiplication $\mu_p\colon TTp \to Tp$ is fiber-wise composition – as objects in the total category of TTp are squares of vertical maps, and vertical maps compose.

This simple construction we can actually extend to a gcwf, meaning we can use it to build out of a gcwf with type fibration u, a new gcwf with type fibration Tu.

▶ **Proposition 23.** *Let* $(u, \dot{u}, \Sigma \dashv \Delta)$ *a gcwf. Then there are adjoint functors* $\overline{\Sigma} \dashv \overline{\Delta}$ *such that* $(u/u, \Sigma/u, \overline{\Sigma} \dashv \overline{\Delta})$ *is a gcwf.*

If we look at the basic judgments this second gcwf is describing, we will find precisely the two introduced in Section 2.1, so that we can reformulate our old perspective:

$$\begin{array}{cccc} \Gamma \vdash A\,\mathsf{Type} & \Gamma \vdash a : A & \Gamma \vdash A' \leq_f A & \Gamma \vdash a :_g A \\ u(A) = \Gamma & \dot{u}(a) = \Gamma, \Sigma(a) = A & f\colon A' \to A,\, u(f) = \mathrm{id}_\Gamma & g\colon \Sigma a \to A,\, u(g) = \mathrm{id}_\Gamma \end{array}$$

into the newer following notation.

$$\begin{array}{cccc} \Gamma \vdash A\,\mathsf{Type} & \Gamma \vdash a : A & \Gamma \vdash A' \leq_f A & \Gamma \vdash a :_g A \\ u(A) = \Gamma & \dot{u}(a) = \Gamma, \Sigma(a) = A & (u/u)(f, A', A) = \Gamma & (\Sigma/u)(g, a, A) = \Gamma,\ p_2 \circ \overline{\Sigma}(g, a, A) = A \end{array}$$

Notice now the symmetry of the first and the second two judgments: the two new ones can be regarded as "classifiers" for new types and terms themselves, in a way such that it is always possible to recover the original theory – one simply has to look at those vertical maps that, in particular, are identities.

▶ **Definition 24** (The category of gcwfs)**.** *A* gcwf morphism $(u, \dot{u}, \Sigma \dashv \Delta) \to (v, \dot{v}, \Sigma' \dashv \Delta')$ *is the data of a pair* $(H, \dot{H})$*, with* $H\colon u \to v$*,* $\dot{H}\colon \dot{u} \to \dot{v}$ *fibration morphisms such that* $H\Sigma = \Sigma'\dot{H}$*. We denote* $\mathbf{GCwF}$ *the category of gcwfs and gcwf morphisms, and* $\mathbf{GCwF}(\mathcal{B})$ *its subcategory with fixed context category* $\mathcal{B}$*.*

Again, it should be noted that gcwfs can be equipped with more interesting (and higher dimensional) structure [4, §3.2], but for the purposes of the present paper we are not interested in that.

▶ **Theorem 25.** *The monad in Lemma 22 can be lifted to a monad over* $\mathbf{GCwF}(\mathcal{B})$*.*

If a monad is an object for abstract computation, this one in particular computes "subtyping judgements". An interesting question is now to check what objects does iterating this computation produce: the first iteration collects vertical maps, and morphisms are squares; in particular, vertical morphisms are those squares having vertical (with respect to the original type fibration) components, therefore the second iteration produced the fibration of said "all vertical" squares; here morphisms are cubes, and the vertical ones are those where all "connecting" maps are vertical (again, with respect to the original type fibration), and so on.

5 Open problems

Of course the applications to constructors can be taken much further than what is done in Section 3.4, and considering more complex ones can lead to a new extension of the present theory. Our intention was to introduce the "vertical maps" perspective, so that whoever gets interested in categorical models for type theory is motivated to consider the benefit of including non-discrete fibrations – usually at very little additional cost.

We conclude this exposition with a take that was suggested to the first author by F. Dagnino, and which we believe might be of some interest: as a fibration can be regarded as an internal category in the category of discrete fibrations, it seems like the process of going from a theory *without* a notion of subtyping to a theory *with* subtyping is close to that of considering a category internal to another one. In this sense, our work is perhaps not best interpreted as a generalization of some known models, but as a structure internal to them.

References

1 Thorsten Altenkirch and Conor McBride. Towards Observational Type Theory. Manuscript, available at `https://www.cs.nott.ac.uk/~psztxa/publ/ott-conf.pdf`, 2006.

2 Steve Awodey. Natural models of homotopy type theory. *Mathematical Structures in Computer Science*, 28(2):241–286, 2018. `doi:10.1017/S0960129516000268`.

3 Francis Borceux. *Handbook of Categorical Algebra*, volume 2 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1994. `doi:10.1017/CBO9780511525865`.

4 Greta Coraglia. *Categorical structures for deduction.* PhD thesis, Università degli Studi di Genova, 2023. URL: `https://hdl.handle.net/11567/1134175`.

5 Greta Coraglia and Ivan Di Liberti. Context, judgement, deduction, 2021. `arXiv:2111.09438`.

6 Greta Coraglia and Jacopo Emmenegger. A 2-categorical analysis of context comprehension, 2024. `arXiv:2403.03085`.

7 Peter Dybjer. Internal type theory. In Stefano Berardi and Mario Coppo, editors, *Types for Proofs and Programs*, pages 120–134, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg. `doi:10.1007/3-540-61780-9_66`.

8 Jacopo Emmenegger, Luca Mesiti, Giuseppe Rosolini, and Thomas Streicher. A comonad for Grothendieck fibrations. *Theory and Applications of Categories*, 40:371–389, 2023.

9 Alexander Grothendieck. Categoriés fibrées et descente (Exposé VI). *Revêtements étales et groupe fondamental - SGA1*, 1960-61.

10 J. Martin E. Hyland. The Effective Topos. In A.S. Troelstra and D. van Dalen, editors, *The L. E. J. Brouwer Centenary Symposium*, volume 110 of *Studies in Logic and the Foundations of Mathematics*, pages 165–216. Elsevier, 1982. `doi:10.1016/S0049-237X(09)70129-6`.

11 Bart Jacobs. Comprehension categories and the semantics of type dependency. *Theoretical Computer Science*, 107(2):169–207, 1993. `doi:10.1016/0304-3975(93)90169-T`.

12 Bart Jacobs. *Categorical logic and type theory.* Elsevier, 1999.

13 Peter T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium: Volume 1.* Oxford University Press, 2002.

14 Anders Kock and Gonzalo E. Reyes. Doctrines in categorical logic. In *Studies in Logic and the Foundations of Mathematics*, volume 90, pages 283–313. Elsevier, 1977. `doi:10.1016/S0049-237X(08)71104-2`.

15 Joachim Lambek and Philip J. Scott. *Introduction to Higher Order Categorical Logic*, volume 7 of *Cambridge Studies in Advanced Mathematics.* Cambridge University Press, 1986.

16 F. William Lawvere. *Functorial Semantics of Algebraic Theories and Some Algebraic Problems in the Context of Functorial Semantics of Algebraic Theories.* PhD thesis, Columbia University, 1963. Available as Reprints in Theory and Applications of Categories, No. 5 (2004) pp 1-121 at `http://www.tac.mta.ca/tac/reprints/articles/5/tr5abs.html`.

17 F.W. Lawvere. Equality in hyperdoctrines and comprehension schema as an adjoint functor. In A. Heller, editor, *Proc. New York Symposium on Application of Categorical Algebra*, pages 1–14. Amer.Math.Soc., 1970.

18 Rodolphe Lepigre and Christophe Raffalli. Practical subtyping for Curry-style languages. *ACM Trans. Program. Lang. Syst.*, 41(1):5:1–5:58, 2019. `doi:10.1145/3285955`.

19 Daniel R. Licata and Robert Harper. 2-dimensional directed type theory. *Electronic Notes in Theoretical Computer Science*, 276:263–289, 2011. Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics (MFPS XXVII). `doi:10.1016/j.entcs.2011.09.026`.

20 Zhen Lin Low. Logic in a topos, 2013. Manuscript, available at `http://zll22.user.srcf.net/talks/2013-01-24-InternalLogic.pdf`.

21 Zhaohui Luo. Coercive subtyping. *Journal of Logic and Computation*, 9(1):105–130, 1999. `doi:10.1093/logcom/9.1.105`.

22 Zhaohui Luo, Sergei Soloviev, and Tao Xue. Coercive subtyping: Theory and implementation. *Information and Computation*, 223:18–42, 2013. `doi:10.1016/j.ic.2012.10.020`.

23 Maria E. Maietti and Giuseppe Rosolini. Quotient completion for the foundation of constructive mathematics. *Logica Universalis*, 7(3):371–402, 2013. `doi:10.1007/s11787-013-0080-2`.

24 Michael Makkai. The fibrational formulation of intuitionistic predicate logic I: completeness according to Gödel, Kripke, and Läuchli, part 2. *Notre Dame J. Formal Log.*, 34:471–498, 1993. `doi:10.1305/ndjfl/1093633902`.

25 Paul-André Melliès and Noam Zeilberger. Functors are type refinement systems. *SIGPLAN Not.*, 50(1):3–16, January 2015. `doi:10.1145/2775051.2676970`.

26 David Jaz Myers. Cartesian factorization systems and Grothendieck fibrations, 2020. `arXiv:2006.14022`.

27 Benjamin C. Pierce. *Types and Programming Languages.* MIT Press, 2002.

28 Edmund Robinson and Giuseppe Rosolini. Colimit completions and the effective topos. *The Journal of Symbolic Logic*, 55(2):678–699, 1990. `doi:10.2307/2274658`.

29 Dana Scott and Christopher Strachey. Toward a mathematical semantics for computer languages. Technical Report PRG06, OUCL, August 1971. Available at `https://www.cs.ox.ac.uk/publications/publication3723-abstract.html`.

30 Thomas Streicher. Fibred categories à la Jean Bénabou, 2022. `arXiv:1801.02927`.

31 Benno van den Berg and Richard Garner. Types are weak ω-groupoids. *Proceedings of the London Mathematical Society*, 102(2):370–394, October 2010. `doi:10.1112/plms/pdq026`.

32 Noam Zeilberger. Principles of type refinement, 2016. Notes of lectures at the Oregon Programming Language Summer School 2016, available at `http://noamz.org/oplss16/refinements-notes.pdf`.

A Proofs

Proof of Proposition 10. The first two rules, Subsumption *(Sbsm)* and Transitivity *(Trans)*, simply follow from composition of vertical arrows. Substitution *(Sbst)* and Weakening *(Wkn)* are a bit trickier and make use of, precisely, the substitutional part of the structure, namely it being a fibration. We begin with Weakening, as it is a bit easier: consider that the premise of the rule is given by maps and objects as below.

$$\begin{tikzcd}
 & A' \arrow[d, "f"] & \\
 & A & \mathcal{U} \arrow[d, "u"] \\
\Gamma.B \arrow[r, "u\epsilon_B"] & \Gamma & \mathcal{B}
\end{tikzcd}$$

Since B is a type in context Γ, we can compute the counit $\epsilon\colon \Sigma\Delta B \to B$, which we know by (3) that acts as weakening, in particular producing the context extension $u\epsilon_B\colon \Gamma.B \to \Gamma$. In the fiber over Γ, not only do we have B, but we also have a vertical map $f\colon A' \to A$. Applying the reindexing $(u\epsilon_B)^*$ to the three of them yields a new vertical map over $\Gamma.B$, hence the judgement $\Gamma.B \vdash (u\epsilon_B)^*A' \leq_{(u\epsilon_B)^*f} (u\epsilon_B)^*A$. In Proposition 10 we have actually written A for $(u\epsilon_B)^*A$, at it is customary for weakening rules – and as, in fact, it appears in (3). All in all, the Weakening rule only tells us that subtyping is preserved through weakening.

Now let us get to Substitution. In order to prove it, we first need to recall how $B[a]$ is computed in a traditional natural model (cf. [2, §2.1]): for given $\Gamma \vdash a : A$ and $\Gamma.A \vdash B$, $B[a]$ is the result of the reindexing B along the (image of the) unit η_a. (This is not precisely how the original paper presents it, though it is equivalent to that: see [4, §2.3.4] for a detailed discussion on the matter.) In our setting, the premises of *(Sbst)* start from the following.

$$\begin{tikzcd}
\Sigma a \arrow[r] \arrow[d] & \Sigma\Delta\Sigma a \arrow[rr] \arrow[dr] & & \Sigma a \arrow[d, "g"] & \\
A \arrow[rr, dashed] & & \Sigma\Delta A \arrow[r] & A & \mathcal{U} \arrow[d, "u"] \\
\Gamma \arrow[r, "\dot{u}\eta_a"] & \Gamma.\Sigma a \arrow[r, "\dot{u}\Delta g"] & \Gamma.A \arrow[r, "u\epsilon_{\Sigma a}"] & \Gamma & \mathcal{B}
\end{tikzcd}$$

Notice that, since g is vertical, and the top horizontal compositions in the top diagram is an identity due to triangle identities of $\Sigma \dashv \Delta$, by cartesianness of the counit $\epsilon_A \colon \Sigma\Delta A \to A$ there is a unique map making the top diagram commute. Now, each "external" side in the top diagram is mapped to the identity through u, meaning we can repeat Awodey's argument, and the required vertical arrow is the result of reindexing along the (image of the) unit – post-composed with $\dot{u}\Delta g$. Notice that a is only "allowed" to interact with B, B' through g, which in turn does nothing to contexts, so that we are in our right to write, as customary, $B[a]$ for $(\dot{u}(\Delta g \circ \eta_a))^* B$. ◀

▶ Remark 26. It might not seem like it, but this last proof makes heavy use of the universal property of the comma object (u/u), and it could be in fact entirely proved using that profusely. We point the interested reader to [4, §3.5] for the corresponding alternative proof.

Proof of Proposition 11. Each component of the unit in a gcwf is a monic arrow. Indeed, let $f, g \colon a \to b$ in $\dot{\mathcal{U}}$ be such that $\eta_b f = \eta_b g$. It follows that

$$\dot{u}f = (u\epsilon_{\Sigma b})(\dot{u}\eta_b)(\dot{u}f) = (u\epsilon_{\Sigma b})(\dot{u}\eta_b)(\dot{u}g) = \dot{u}g$$

and, in turn, that $f = g$ since η_b is cartesian. The left adjoint Σ is then faithful. ◀

Proofs of Propositions 19–21. This is straightforward, by reading the action of the morphisms involved (Fun, Π, abs, Σ, pair) on vertical arrows and unfolding the definitions of the sub-typing judgements as in Table 1. ◀

Proof of Proposition 23. We begin by describing what the fibration of terms (Σ/u) does: it simply collects pairs (g, a, A) such that $g \colon \Sigma a \to A$ is a u-vertical map, and sends them to their underlying context. Its vertical maps are pairs of vertical maps (respectively $\dot{u}$- and u-vertical) fitting in appropriate squares, and its cartesian maps are pairs of cartesian maps (again, respectively, with reference to $\dot{u}$ and u).

The required typing functor, then, is simply $\overline{\Sigma}$ sending each triple (g, a, A) to $(g, \Sigma a, A)$, which is cartesian because Σ is. Describing its right adjoint requires a little more effort: for a triple (f, A', A) in (u/u) one considers its image through Δ, then its vertical-cartesian factorization system (cf. Proposition 3), then the image of the vertical portion of it through Σ to get back to $\mathcal{U}$.

$$\begin{tikzcd}
A' \arrow[d, "f"] \\
A
\end{tikzcd}
\qquad
\begin{tikzcd}
\Delta A' \arrow[d, "(\Delta f)^v"'] \arrow[dr, "\Delta f"] & \\
a_f \arrow[r, "(\Delta f)_c"'] & \Delta A
\end{tikzcd}
\qquad
\begin{tikzcd}
\Sigma\Delta A' \arrow[rr, "\epsilon_{A'}"] \arrow[d, "\overline{\Delta}f = \Sigma(\Delta f)^v"'] \arrow[dr, "\Sigma\Delta f"] & & A' \arrow[d, "f"] \\
\Sigma a_f \arrow[r, "\Sigma(\Delta f)_c"'] & \Sigma\Delta A \arrow[r, "\epsilon_A"'] & A
\end{tikzcd}$$

Cartesianness of $(\Delta f)_c$ can be used to provide functoriality. Let us now show how the two functors form an adjoint pair by means of the universal property of its counit, which is actually depicted on the right square above. For any triple (g, b, B) in (Σ/u) and morphism $(h, k)\colon \overline{\Sigma}(g, b, B) \to (f, A', A)$ we must show that there is a unique pair (m, n) such that $\overline{\Sigma}(m, n)\colon \overline{\Sigma}(g, b, B) \to \overline{\Sigma}\overline{\Delta}(f, A', A)$ makes the obvious triangle commute. The universal property of the counit ϵ'_A yields a unique $m\colon b \to \Delta A'$ so that $h = \epsilon_{A'} \circ m$,

$$\begin{tikzcd}
\Sigma b \arrow[d, "g"'] \arrow[dr, dashed, "\Sigma(!)"'] \arrow[drrr, bend left=10, "h"] & & & \\
B \arrow[drrr, "k"] & \Sigma\Delta A' \arrow[rr, "\epsilon_{A'}"'] \arrow[d, "\overline{\Delta}f=\Sigma(\Delta f)^v"'] \arrow[dr, "\Sigma\Delta f"'] & & A' \arrow[d, "f"] \\
 & \Sigma a_f \arrow[r, "\Sigma(\Delta f)_c"'] & \Sigma\Delta A \arrow[r, "\epsilon_A"'] & A
\end{tikzcd}$$

$$\begin{tikzcd}
\Theta \arrow[dr, "u\Sigma(!)"'] \arrow[drrr, bend left=10] & & & \\
 & \Gamma.A' \arrow[r] & \Gamma.A \arrow[r] & \Gamma
\end{tikzcd}$$

hence a map filling the bottom diagram to a commuting triangle. By cartesianness of $\epsilon_A \circ \Sigma(\Delta f)_c$ there is a unique map $n\colon B \to \Sigma a_f$ making the "other" triangle commute as well. ◀

Proof of Theorem 25. Let us denote (T, η, μ) the desired monad. On objects it of course acts as $T(u, \dot{u}, \Sigma \vdash \Delta) = ((u/u), (\Sigma/u), \overline{\Sigma} \dashv \overline{\Delta})$ as in Proposition 23. The image of a morphism $(H, \dot{H})$ is actually simply the action of H, and it provides a commutative square because $H\Sigma = \Sigma'\dot{H}$. Unit is again induced by identity, and multiplication by composition. ◀