# 9- Algebraization of realizability

In the first parts of this thesis, we introduced several calculi for which we gave a Krivine realizability interpretation. Namely, in addition to Krivine's $\lambda_c$-calculus, we presented interpretations for the call-by-name, call-by-value and call-by-need $\lambda\mu\tilde{\mu}$-calculi, for $\mathrm{dL}_{\hat{\mathfrak{tp}}}$ and for $\mathrm{dLPA}^{\omega}$. Amongst others, we could cite Munch-Maccagnoni's interpretation for System L [126], Lepigre's interpretation for his call-by-value calculus with a semantical value restriction [108], or Jaber's interpretation of SECD machine code [80]. Since classical realizability interpretations provide powerful tools for computational analysis of programs, it naturally raises the question of knowing what is, in a calculus, the structure necessary to the definition of a classical realizability interpretation.

**The structures of classical realizability** Additionally, as we briefly mentioned in Section 3.5.3, the recent work of Krivine revealed impressive new perspectives in using realizability from a model-theoretic point of view. In [98] and [99], Krivine introduced the notion of *realizability algebras*, which constitute the classical counterpart of *partial combinatory algebras* for intuitionistic realizability. He showed how these structures allow for the construction of models of ZF. Relying on realizability algebras, he defined in particular a model in which neither the continuum hypothesis nor the axiom of choice are valid (see Section 3.5.3), bringing then new perspectives from a model-theoretic point of view.

Roughly speaking, a realizability algebra contains the minimal structure to be a suitable target for compiling the $\lambda_c$-calculus. It consists of three sets: a set of *terms* $\mathbf{\Lambda}$ (which contains a certain set of combinators[1]), a set of *stacks* $\mathbf{\Pi}$ and a set of *processes* $\mathbf{\Lambda} \star \mathbf{\Pi}$ together with a preorder relation $\succ$ on $\mathbf{\Lambda} \star \mathbf{\Pi}$. These elements are axiomatized in such a way that the relation $\succ$ behaves like the reduction of the abstract machine for the $\lambda_c$-calculus. Such a structure is indeed present in each of the cases we studied in this thesis.

**The structures of intuitionistic realizability** On the other hand, in the continuity of Kleene and Troelstra's tradition of intuitionistic realizability (see [159] for an historical overview), Hyland, Johnstone and Pitts introduced in the 1980s the notion of tripos [79, 135]. A major application of triposes is the effective topos $\mathscr{E}ff$, later introduced by Hyland in [78], which allows for an analysis of realizability in the general framework of toposes. Let us briefly outline the tripos underlying Kleene realizability. Recall that in Kleene realizability, a formula is realized by natural numbers (see Chapter 3). To each closed formula $\varphi$ we can then associate the set of its realizers $\{n \in \mathbb{N} : n \Vdash \varphi\}$, which belongs to $\mathcal{P}(\mathbb{N})$. This structure can be generalized to interpret a predicate $\varphi(x)$, where the free variable $x$ ranges over a set $X$, as a function from $X$ to $\mathcal{P}(\mathbb{N})$ which associates to each $x \in X$ the set $\{n \in \mathbb{N} : n \Vdash \varphi(x)\}$. For instance, given a set $X$, we can define the equality $=_X$ as the function:

$$=_X \colon (x,y) \in (X \times X) \mapsto \begin{cases} \mathbb{N} & \text{if } x = y \\ \emptyset & \text{otherwise} \end{cases}$$

[1]See [98] for the full definition. The key point is that the set of combinators is complete with respect to the $\lambda$-calculus and contains $\boldsymbol{cc}$.

Following the realizability interpretation, we can interpret predicate logic, for instance we define[2]:

$$(\varphi \to \psi)(x) \triangleq \{n \in \mathbb{N} : \forall m \in \varphi(x), n(m) \in \psi(x)\}\,.$$

This naturally induces an entailment relation $\vdash_X$ on predicates for each set $X$. Given $\varphi, \psi$ two predicates over $X$, we say that $\varphi \vdash_X \psi$ if there exists $n \in \mathbb{N}$ such that for all $x \in X$, $n$ realizes $(\varphi \to \psi)(x)$, that is to say:

$$\varphi \vdash_X \psi \quad \triangleq \quad \bigcap_{x \in X} (\varphi \to \psi)(x) \neq \emptyset\,.$$

The entailment relation $\vdash_X$ defines in fact a preorder on predicates. Moreover, the set of predicates equipped with this preorder $(\mathcal{P}(\mathbb{N})^X, \vdash_X)$ broadly defines a Heyting algebra[3]. Indeed, in addition to the arrow $\to$, the connectives $\wedge$, $\vee$ can be defined as in Kleene realizability. It is almost direct[4] to show that for any set $X$:

$$\chi \wedge \varphi \vdash_X \psi \quad \Leftrightarrow \quad \chi \vdash_X \varphi \to \psi$$

Given two sets $X, Y$, any function $f : X \to Y$ induces a function $f^*$ from $\mathcal{P}(\mathbb{N})^Y$ to $\mathcal{P}(\mathbb{N})^X$ by precomposing any $\varphi : Y \to \mathcal{P}(\mathbb{N})$ by $f$: $\varphi \circ f : X \to \mathcal{P}(\mathbb{N})$. In terms of logic, $f^*$ corresponds to the operation of reindexing the variables of a predicate $\varphi$ along $f$.

Before turning to a more formal introduction, the last logical notions we want to mention in this context are the quantifiers, whose presentation is due to Lawvere's work [105]. Consider the particular case of a projection $\pi : \Gamma \times X \to \Gamma$. It gives rise to a function $\pi^* : \mathcal{P}(\mathbb{N})^\Gamma \to \mathcal{P}(\mathbb{N})^{\Gamma \times X}$, which turns any predicate $\varphi$ on $\Gamma$ into a predicate $\pi^*(\varphi)$ on $\Gamma \times X$. On the contrary, since existential and universal quantifiers on $X$ bind a variable, they are defined as functions from $\mathcal{P}(\mathbb{N})^{\Gamma \times X} \to \mathcal{P}(\mathbb{N})^\Gamma$, in such a way[5] that the following equivalences hold for all $\varphi \in \mathcal{P}(\mathbb{N})^\Gamma$ and for all $\psi \in \mathcal{P}(\mathbb{N})^{\Gamma \times X}$:

$$\begin{array}{rclcrcl} \psi & \vdash_{\Gamma \times X} & \pi^*(\varphi) & \text{if and only if} & \exists_X(\psi) & \vdash_\Gamma & \varphi \\ \pi^*(\varphi) & \vdash_{\Gamma \times X} & \psi & \text{if and only if} & \varphi & \vdash_\Gamma & \forall_X(\psi) \end{array}$$

Up to this point, the structure we exhibited is called a *hyperdoctrine*, due to F. William Lawvere [105]. In broad terms, a hyperdoctrine is defined by a similar structure where the sets $\mathcal{P}(\mathbb{N})^X$ are generalized to arbitrary Heyting algebras $(H_X, \vdash_X)$. A *tripos*, as we will see, is a hyperdoctrine with the extra-assumption that there exists a set $\mathtt{Prop}$ (here $\mathcal{P}(\mathbb{N})$) of "propositions" and a generic "truth predicate" $\mathtt{tr} \in H_{\mathtt{Prop}}$ (here the identity function $\mathtt{id}_{\mathcal{P}(\mathbb{N})}$), such that for any predicate $\varphi$ in $H_X$, there exists a function $\chi_\varphi : X \to \mathtt{Prop}$ which verifies:

$$\varphi \dashv\vdash_X \chi_\varphi^*(\mathtt{tr})$$

Triposes, which were studied and defined by Andrew Pitts during his PhD thesis [135, 136], have been conducive to the categorical analysis of realizability.

**Towards a categorical presentation of classical realizability** For a long time, Krivine classical realizability and the categorical approach to realizability seemed to have no connections. The situation changed in the past ten years, notably thanks to Thomas Streicher who built an important bridge in [151]. After reformulating the Krivine's abstract machine of the $\lambda_c$-calculus as an *abstract Krivine*

[2] Remember that a natural number $n$ is identified with the $n^{\text{th}}$ recursive function of a fixed enumeration.

[3] Strictly speaking, it actually defines a Heyting prealgebra, that is to say a Heyting algebra whose career is a preorder (whitout the property of anti-symmetry) instead of a poset.

[4] In terms of recursive functions, the left-to-right implication is merely curryfication and vice-versa.

[5] We let the reader check that in the general case of a function $f : X \to Y$, we can define the quantifiers by

$$\exists_f(\varphi)(y) \triangleq \textstyle\bigcup_{x \in X}(f(x) =_Y y \wedge \varphi(x)) \qquad\qquad \forall_f(\varphi)(y) \triangleq \textstyle\bigcap_{x \in X}(f(x) =_Y y \to \varphi(x))$$

*structure* (AKS), Streicher proved that from each AKS one may construct a *filtered ordered partial combinatory algebra* and a tripos. Later on, in a series of papers from 2013-2015 [44, 44, 45] Walter Ferrer Santos, Jonas Frey, Mauricio Guillermo, Octavio Malherbe and Alexandre Miquel developed the theory of *Krivine ordered combinatory algebras* ($^{\mathcal{K}}$OCA) for classical realizability. Their main purpose was to try to abstract as much as possible the essence of abstract Krivine structures, in order to get a structure which is as general as possible and which captures the necessary ingredients to generate Krivine models (*i.e.* triposes).

This part of the thesis is in line with this general purpose. In the next chapters, we will introduce the notion of *implicative algebras*, developed by Alexandre Miquel [121]. As we shall see, these are structures which encompass all the structure necessary to the definition of classical realizability models. In particular, the $\lambda_c$-calculus and the ordered combinatory algebras are definable within implicative algebras. In addition, they allow for simple criteria to determine whether the induced realizability tripos collapses to a forcing tripos. Based on the arrow connective, implicative algebras somewhat reflect the enriched lattice structure underlying Krivine realizability interpretation of logic. After introducing these structures, we will present the notion of *disjunctive algebras* and *conjunctive algebras*. Respectively based on the 'par' $\parr$ and the tensor $\otimes$ connectives together with a negation, these structures reflects the corresponding decompositions of the arrow in linear logic. As we will explain, these decompositions can be interpreted in terms of evaluation strategies: disjunctive algebras naturally arise from a call-by-name fragment of Munch-Maccagnoni's System L [126], while conjunctive algebras correspond to a call-by-value fragment of the same.

## 9.1 The underlying lattice structure

### 9.1.1 Classical realizability

Let us start by arguing that through the Curry-Howard interpretation of logic, and especially in realizability, there is an omnipresent lattice structure. This structure is reminiscent of the concept of subtyping, which makes concrete, in a programming language, a well-known fact in mathematics: if $f$ is a function whose domain is a set $X$ (say the set $\mathbb{R}$), and if $S$ is a subset of $X$ (say $\mathbb{N} \subset \mathbb{R}$), then $f$ can be restricted to a function $f_{|S}$ of domain $S$. Similarly, in object-oriented programming, if a program $p$ takes as input any object in a class $C$, if $D$ is a class which inherits of the structure of $C$, $p$ can be applied to any object in $D$. This idea is usually reflected in the theory of typed calculus by a *subtyping relation*, often denoted by $<:$, where $T <: U$ means that $T$ is more precise as a type. For instance, type systems including a subtyping relation (see [22] for instance) usually have the rules:

$$\frac{\Gamma \vdash p : T \quad T <: U}{\Gamma \vdash p : U}\ (\textsc{Sub}) \qquad\qquad \frac{U_1 <: T_1 \quad T_2 <: U_2}{T_1 \to T_2 <: U_1 \to U_2}\ (\textsc{S-Arr})$$

The first rule, called *subsumption*, says that we can always replace a type by a supertype. The second one expresses that the arrow is contravariant on its left-hand side and covariant on its right-hand side. To say it differently, if we think of $T <: U$ as "$T$ is more constrained than $U$ is", and consider the rule `nat` $<:$ `real`, a function of type `real` $\to$ `nat` is indeed more constrained than a function of type `real` $\to$ `real`, itself more constrained than the type `nat` $\to$ `real`. Besides, as suggested by the notation, the subtyping relation is reflexive and anti-symmetric, it thus induces a preorder on types.

This relation is implicit in classical realizability, in the sense that the subsumption rule is always adequate: if $A <: B$, for any pole, if $t \Vdash A$ then $t \Vdash B$ (see [144, Proposition 3.1.1]). In terms of truth values, this means that if $A <: B$, then $\|A\| \supseteq \|B\|$ (and hence $|A| \subseteq |B|$). We said that this relation was implicit, and indeed, even when the relation is not syntactically defined, given a pole $\perp\!\!\!\perp$ it is always possible to define a semantic notion of subtyping[6]:

**Subtyping** $\qquad A \leq_{\perp\!\!\!\perp} B \triangleq$ for all valuations $\rho$, $\|B\|_\rho \subseteq \|A\|_\rho$

In this case, the relation $\leq$ being induced from (reversed) set inclusions, it comes with a richer structure of complete lattice, where the meet $\wedge$ is defined as a union and the join $\vee$ as an intersection. Observe that in particular, this corresponds to the interpretation of universal quantifiers in classical realizability:

$$\|\forall x.A\|_\rho \triangleq \bigcup_{n\in\mathbb{N}} \|A\|_{\rho[x\mapsto n]} = \bigwedge\{\|A\|_{\rho[x\mapsto n]} : n \in \mathbb{N}\}$$

In this lattice structure, quantifiers are thus naturally defined as meets and joins, while the logical connectives $\wedge$ and $\vee$, in the case of realizability, are interpreted in terms of products and sums. To sum up, classical realizability then correspond to the following picture:

| **Realizability:** | $\forall = \bigwedge$ | $\wedge = \times$ | $\exists = \bigvee$ | $\vee = +$ |
|---|---|---|---|---|

### 9.1.2 Forcing

In turn, in the cases of semantics given by Heyting algebras (for intuitionistic logic) or Boolean algebras (for classical logic), quantifiers and connectives are both interpreted in terms of meets and joins. To put it differently, the universal quantifier is semantically defined as an infinite conjunction, while the existential one is defined as an infinite union. These cases are not different from Kripke semantics for intuitionistic logic or Cohen forcing in the case of classical logic.

Let us first examine the case of Kripke models to show that they induce Heyting algebras. Consider indeed a Kripke model $(\mathcal{W}, \leq, D, V)$ (see Chapter 1). Then let us denote by $\mathcal{U}$ the set of upward closed subsets of $\mathcal{W}$:

$$\mathcal{U} \triangleq \{U \subseteq \mathcal{W} : \forall v, w \in \mathcal{W}, v \in U \wedge v \leq w \Rightarrow w \in U\}$$

The intersection (resp. the union) of upward closed sets being itself upward closed, $(\mathcal{U}, \subseteq)$ defines a lattice structure, whose higher element $\top$ is $\mathcal{W}$. In fact, this structure is even a Heyting algebra, where for any sets $U, V \in \mathcal{U}$, the arrow is defined by:

$$U \to V \triangleq \{w \in \mathcal{W} : \forall v \in \mathcal{W}, w \leq v \wedge v \in U \Rightarrow v \in V\}$$

It is routine to check that $U \to V$ belongs to $\mathcal{U}$ and that it satisfies the properties of the implication operation in Heyting algebras[7]. Moreover, it can be shown[8] that the validity under Kripke semantics in the model $(\mathcal{W}, \leq, D, V)$ corresponds to the interpretation in the Heyting algebra $(\mathcal{U}, \subseteq)$:

$$[\![\varphi]\!]^{\mathcal{U}} = \{w \in \mathcal{W} : w \Vdash \varphi\}$$

and thus $\mathcal{U} \vDash \varphi$, that is to say $[\![\varphi]\!]^{\mathcal{U}} = \top$, if and only if $\forall w \in \mathcal{W}, w \Vdash \varphi$.

Regarding Cohen forcing, a very similar construction allows us to reduce it to the case of Boolean-valued models [14]. Loosely speaking, Cohen forcing is a construction which, starting from a ground model $\mathcal{M}$ of set theory and a poset $(P, \leq)$ of forcing conditions, defines a new model $\mathcal{M}[G]$ where $G$ is a generic filter on $P$. Without entering into the definition of $\mathcal{M}[G]$, we can briefly explain how the validity in $\mathcal{M}[G]$ can be understood in terms of Boolean algebras. First, any poset $(P, \leq_P)$ can be

[6]Note that this definition is specific to classical realizability, in the intuitionistic case, semantic subtyping $A <: B$ is defined as the inclusion $|A| \subseteq |B|$ of truth value. In the classical setting, semantic subtyping is thus defined as the reversed inclusion of falsity values $\|B\| \subseteq \|A\|$, which is a strictly stronger condition (in fact, the inclusion of truth value $|A| \subseteq |B|$ does not constitute a valid definition of subtyping in the classical case).

[7]Both direction of the equivalence $U \cap X \subseteq V \Leftrightarrow X \subseteq U \to V$ are simple exercises.

[8]See for instance [47] for a complete proof.

embedded by an order-preserving morphism to $RO(P)$ the complete Boolean algebra of regular open sets[9] of $P$. The embedding $e$ in question maps every forcing condition $p$ to the interior of the closure of the following open set:

$$O_p = \{q \in P : q \leq p\}.$$

Writing $B$ for the Boolean algebra $RO(P)$, the forcing relation can then be defined by:

$$p \Vdash \varphi \quad \triangleq \quad e(p) \leq [\![\varphi]\!]^{\mathcal{B}}$$

where $[\![\cdot]\!]^{\mathcal{B}}$ is the interpretation in the Boolean-valued model $\mathcal{M}^{\mathcal{B}}$. Finally, the validity of a formula $\varphi$ in $\mathcal{M}[G]$ is broadly[10] defined by the existence of a condition $p \in G$ which forces $\varphi$. The truth value under the forcing translation can thus be interpreted in terms of Boolean algebras.

For these reasons, we can say that the interpretation of connectives and quantifications in intuitionistic (Kripke) and classical (Cohen) forcing amount to their interpretations in Heyting and Boolean algebras, respectively. This situation can be summarized by:

| **Forcing:** | $\forall = \wedge = \curlywedge$ | $\exists = \vee = \curlyvee$ |
|---|---|---|

In this sense, the realizability interpretation is therefore, *a priori*, more general than the forcing one.

## 9.2 A types-as-programs interpretation

Let us put the focus back on the lattice structure in realizability, and more specifically to the subtyping relation. Given a fixed pole $⫫$, the semantic definition of the subtyping relation that we gave is equivalent to:

$$A \leq_{⫫} B \text{ if for all } t, \text{ whenever } t \Vdash A \text{ then } t \Vdash B$$

Formulas are thus ordered according to their truth values, which are set of realizers. Loosely speaking, we are identifying formulas with their realizers. On the other hand, many semantics allows us to associate terms with their principal types. For instance, the identity $I = \lambda x.x$ can be identified to its principal type $\forall X.X \to X$; doing so, the fact that $I \Vdash \texttt{nat} \to \texttt{nat}$ can be read as $\forall X.X \to X \leq \texttt{nat} \to \texttt{nat}$ at the level of formulas. Identifying terms with their principal type allows us to associate to each realizer the truth value of its principal types (to which it belongs). In other words, it corresponds to the following informal inclusion:

$$\text{Realizers} \subseteq \text{Truth values}$$

But what can be said about the reverse inclusion? In order to consider truth values as realizers we should be able to lift the operations of $\lambda$-abstraction and application at the level of truth values. As we shall see in the next chapters, this is in fact perfectly feasible in simple algebraic structures, called *implicative structures*. In these structures, that we present in Chapter 10, truth values can be regarded as generalized realizers and manipulated as such. In particular, it suggests that the previous inclusion of realizers into truth values could actually be turned into an equality:

$$\text{Realizers} = \text{Truth values}$$

An important feature of implicative structures is thus that they allow to formalize this identification. In particular, any truth value $a$ will be identified with the realizer whose principal type is $a$ itself. Implicative structures are complete lattices equipped with a binary operation $a \to b$ verifying properties

[9] For the order topology. Regular open sets are open sets which are equal to the interior of their closure.

[10] To be more accurate, a formula $\phi(x_1, \dots x_n)$ is valid in $\mathcal{M}[G]$ if there exists a condition $p$ in $G$ which forces $\phi(\underline{x}_1, \dots \underline{x}_n)$ where $\underline{x}_i$ is a name in $\mathcal{M}^B$ for $x_i$. We really do not want to formally introduced forcing here, an introduction in terms of Boolean-valued model is given in [14].

coming from the logical implication. As we will see, they indeed allow us to interpret both the formulas and the terms in the same structure. For instance, the ordering relation $a \leq b$ will encompass different intuitions depending on whether we regard $a$ and $b$ as formulas or as terms. Namely, $a \leq b$ will be given the following meanings:

- the formula $a$ is a subtype of the formula $b$;
- the term $a$ is a realizer of the formula $b$;
- the realizer $a$ is more defined than the realizer $b$.

The last item correspond to the intuition that if $a$ is a realizer of all the formulas of which $b$ is a realizer, $a$ is more precise than $b$, or more powerful as a realizer. Therefore, $a$ and $b$ should be ordered.

In terms of the Curry-Howard correspondence, this means that not only do we identify types with formulas and proofs with programs, but we also identify types and programs. Visually, this corresponds to the following situation:

$$\begin{tikzcd}
\text{Types} \arrow[r, equal] \arrow[d, equal] & \text{Formulas} \arrow[d, equal] \\
\lambda\text{-terms} \arrow[r, equal] & \text{Proofs}
\end{tikzcd}$$

which is to be compared with the corresponding diagram in Section 2.3.

Because we consider formulas as realizers, any formula will be at least realized by itself. In particular, the lowest formula $\perp$ is realized. While this can be dazzling at first sight, it merely reflects that implicative structures do not come with an intrinsic criterion of consistency. To this purpose, we will introduce the notion of *separator*, which is similar to the usual notion of filter for Boolean algebras. *Implicative algebras* will be defined as implicative structures equipped with a separator. As we shall see, they capture the algebraic essence of classical realizability models. In particular, we will embed both the $\lambda_c$-calculus and its type system in such a way that the adequacy is preserved. Furthermore, we will see that they give rise to the usual realizability triposes, and that they provide us with simple criteria to determine whether the induced triposes collapse to forcing triposes. Implicative algebras therefore appear to be the adequate algebraic structure to study classical realizability and the models it induces.

## 9.3 Organization of the third part

Above all, we shall warn the reader that the very concept of implicative algebras—as well as the different results that we present about it—in this manuscript are not ours. They are due to Alexandre Miquel, who have been giving numerous talks on the topic [121], but they are unpublished for the time being. In particular, the next chapter should not be taken as a scientific contribution peculiar to this thesis, even our presentation of the subject is deeply influenced by Miquel's own presentation. Our only contribution about implicative algebras is the Coq formalization that we will mention in the next chapter.

First, we recall in the next section some definitions of basic algebraic structures and some vocabulary from category theory that are used in the sequel. Next, in the last section of this chapter, we present the algebraic structures prior this work which are used in the study of realizability from a categorical point of view. This last section is intended to be a brief survey of the work of Streicher [151] and Ferrer, Frey, Guillermo, Malherbe and Miquel [45] on the topic. This will naturally lead us to the definitions of implicative algebras in the following chapter.

Chapter 10 is then devoted to the presentation of implicative algebras. We first introduce the notion of implicative structures and give a few examples. Next, we show how to embed both the $\lambda_c$-calculus and its second-order type system while proving the adequacy of the embedding. We then introduce the notion of separators and implicative algebras, and show how they induce realizability triposes.

In Chapter 11, we present a similar structure which is based on the decomposition of the arrow $a \to b$ as $\neg a \vee b$. We first give a computational account for this decomposition in a fragment of Munch-Maccagnoni's system L, and explain how it is related to the choice of a call-by-name evaluation strategy for the $\lambda$-calculus. We then introduce the notion of disjunctive algebras, which we relate to the implicative ones. Similarly, we present in Chapter 12 a structure based on the decomposition of the arrow $a \to b$ as $\neg(a \wedge \neg b)$ and follow the same process towards the definition of conjunctive algebras.

This part of the thesis is supported by a Coq development[11], in which most of the results are proved. My motivation for this development was twofold. First, I should confess that I started it as an (amusing) exercise to better understand implicative algebras. Because I was probably the first in the position of checking Miquel's definitions and results, I thought that the best way to do it might be to formalize everything. Second, insofar as implicative algebras aim, on a long-term perspective, at providing a foundational ground for the algebraic analysis of realizability models, a Coq formalization also seemed to be a good way of laying the foundations of these structures.

## 9.4 Categories and algebraic structures

### 9.4.1 Lattices

We recall some definitions and properties about lattices. Since the proofs are very standard, we omit them and refer the reader to the Coq formalization if needed.

**Definition 9.1** (Lattice)**.** A *lattice* is a partially ordered set $(\mathcal{L}, \leq)$ such that that any pair of elements $a, b \in \mathcal{L}$ admits:

1.• a greatest lower bound, which we write $a \wedge b$;
2.• a lowest upper bound, which we write $a \vee b$.

⌟

In order to interpret the quantifications, we will pay attention to arbitrary meets and joins, hence to complete lattices:

**Definition• 9.2.** A lattice $\mathcal{L}$, is said to be *meet-complete* (resp. *join-complete*) if any subset $A \subseteq \mathcal{L}$ admits a greatest lower bound (resp. lowest upper bound), written $\bigwedge_{a \in A} a$ or simply $\bigwedge A$ (resp. $\bigvee_{a \in A} a$ and $\bigvee A$). It is said to be *complete* if it is both meet- and join-complete. ⌟

The following theorem states that any meet-complete lattice is also join-complete and vice-versa:

**Theorem• 9.3.** *If $\mathcal{L}$ is a meet-complete lattice, then $\mathcal{L}$ is a complete lattice with the join operation defined by:*

$$\bigvee_{a \in A} a \triangleq \bigwedge_{a \in ub(A)} a$$

*where $ub(A)$ is the set of upper-bounds of $A$. The converse direction is similar.*

Any complete lattice has a lowest and a highest element, which we write $\bot$ and $\top$:

**Proposition 9.4.** *In any complete lattice $\mathcal{L}$, the following holds:*

[11] The source of the Coq development can be browsed or downloaded from here• [122]. We use the bullet to denote the statements which are formalized in the development. In the electronic version of the manuscript, these statements are given with an hyperlink pointing directly to their Coq counterpart.

*1.*• $\top = \bigwedge \emptyset = \bigvee \mathcal{L}$ *2.*• $\bot = \bigvee \emptyset = \bigwedge \mathcal{L}$

Finally, we recall that reversing the order of a (complete) lattice still gives a (complete) lattice where meet and join are exchanged:

**Proposition• 9.5.** *If $(\mathcal{L}, \leq)$ is a complete lattice, then $(\mathcal{L}, \triangleleft)$ where $a \triangleleft b \triangleq b \leq a$ is a complete lattice.*

### 9.4.2 Boolean algebras

We recall the definition and some key properties of Boolean algebras.

**Definition• 9.6.** A *Boolean algebra* is a quadruple $(\mathcal{B}, \leq, \bot, \top)$ such that:

- $(\mathcal{B}, \leq, \vee, \wedge)$ is a bounded lattice, $\top$ being the upper bound of $\mathcal{B}$ and $\bot$ its lower bound
- $\mathcal{B}$ is distributive, in the sense that:

$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c) \qquad a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c) \qquad (\forall a, b, c \in \mathcal{B})$$

- every element $a \in \mathcal{B}$ has a complement, which we write $\neg a$, in the sense that:

$$a \vee \neg a = \top \qquad a \wedge \neg a = \bot \qquad (\forall a \in \mathcal{B})$$

A Boolean algebra is said to be *complete* if it is complete as a lattice. ⌟

We state some properties of Boolean algebras, in particular the commutation of the negation with the other internal laws:

**Proposition 9.7.** *If $\mathcal{B}$ is a complete Boolean algebra, the following hold:*

*1.*• $b = \neg a$ *if and only if* $(a \vee b = \top)$ *and* $(a \wedge b = \bot)$ $(\forall a, b \in \mathcal{B})$

*2.*• $\neg\neg a = a$ $(\forall a \in \mathcal{B})$

*3.*• $\neg(a \vee b) = (\neg a) \wedge (\neg b)$ *and* $\neg(a \wedge b) = (\neg a) \vee (\neg b)$ $(\forall a, b \in \mathcal{B})$

Finally, we recall the commutation of the negation with arbitrary joins and meets in complete Boolean algebras:

**Theorem 9.8.** *If $\mathcal{B}$ is a complete Boolean algebra, then the following holds for any $A \subseteq \mathcal{B}$:*

*1.*• $\neg \bigwedge\{a : a \in A\} = \bigvee\{\neg a : a \in A\}$ *2.*• $\neg \bigvee\{a : a \in A\} = \bigwedge\{\neg a : a \in A\}$

All these commutations can be interpreted in terms of logical commutation in Boolean-valued models. The first ones indicate that the internal logic of Boolean-valued models (and in particular of forcing models) has an involutive negation and that De Morgan's laws are satisfied. The former theorem indicate that negation commutes with quantifiers as follows:

$$\neg\forall = \exists\neg \qquad \neg\exists = \forall\neg$$

These equalities will not hold in general in implicative algebras. Better, they will precisely characterize the collapse of the induced realizability triposes to forcing ones. In this sense, these commutations show that implicative algebras are a strict refinement of Boolean algebras. As such, they also are the sign that implicative algebras might provide us with models which are *a priori* more general than Boolean-valued models.

### 9.4.3 Categories

We briefly introduce some standard notions of category theory in order to further define the notions of hyperdoctrine and tripos.

**Definition 9.9.** A *category* $C$ is given by a class of *objects* together with a class of *morphisms* $C(a,b)$ for each pair $a,b \in C$ of objects, as well as:

- an associative composition of morphism, which is written $g \circ f$ for all $f \in C(a,b), g \in C(b,c)$,
- a morphism $\mathtt{id}_a \in C(a,a)$ (identity) for each $a \in C$, such that:

$$\forall f \in C(a,b), f \circ \mathtt{id}_a = \mathtt{id}_b \circ f = f$$

┘

The property required for the identity and the associativity of the composition can be expressed in terms of diagrams, by requiring that the following diagrams commute[12]:

$$\begin{tikzcd} a \arrow[r, "f"] \arrow[d, "\mathtt{id}_a"'] \arrow[dr, "f"] & b \arrow[d, "\mathtt{id}_b"] \\ a \arrow[r, "f"'] & b \end{tikzcd} \qquad \begin{tikzcd} & b \arrow[d, "g"] \arrow[dr, "h \circ g"] & \\ a \arrow[ur, "f"] \arrow[dr, "g \circ f"'] & & d \\ & c \arrow[ur, "h"'] & \end{tikzcd}$$

In the sequel, we will often express properties by means of diagrams. Most of the algebraic structures that we mentioned until here can be regarded as particular categories with extra structure. The class of a given structure (say the Boolean algebras, the lattices) also form a category in general, whose morphisms are the structure-preserving functions. For instance, the following structures are categories:

- **Set**, the category of sets, whose objects are sets and whose morphisms are the functions between sets;
- **Poset**, the category whose objects are posets and whose morphisms are order-preserving functions;
- any poset $(P, \leq)$ can be regarded as a category whose objects are its elements, and where there is morphism between two objects $x$ and $y$ when $x \leq y$;
- **Lat**, the category of lattices, is formed with lattices as objects and functions preserving the meet $\wedge$ and the join $\vee$ as morphisms;
- any lattice $(\mathcal{L}, \leq)$ can be considered in itself as a category;
- etc.

We recall some standard definitions about objects and morphisms:

**Definition 9.10.** Let $C$ be a category:

- A morphism $f : a \to b$ is said to be *invertible* if there exists a morphism $g : b \to a$ such that $g \circ f = \mathtt{id}_a$ et $f \circ g = \mathtt{id}_b$
- $a$ and $b$ are said to be *isomorphic* if there exists $f \in C(a,b)$ invertible
- an object $t$ is said to be *terminal* if $\forall a \in C, \exists! f : a \to t$
- an object $i$ is said to be *initial* if $\forall a \in C, \exists! f : i \to a$

┘

[12]That is to say that if we take an element of the object $a$, the images we will obtain by two paths leading to the same object will be equal.

**Definition 9.11** (Dual category). Let $\mathcal{C}$ and $\mathcal{D}$ be two categories. We define:

- $\mathcal{C}^{op}$ the *dual category* of $\mathcal{C}$ as being the category with the same objects in which morphisms and the composition are reversed: $\mathcal{C}^{op}(a,b) = \mathcal{C}(b,a)$, $f \circ_{\mathcal{C}^{op}} g = g \circ_{\mathcal{C}} f$
- $\mathcal{C} \times \mathcal{D}$ the *product category* of $\mathcal{C}$ and $\mathcal{D}$, whose objects are pairs of objects $(c \in \mathcal{C}, d \in \mathcal{D})$, and whose morphisms are pairs of morphisms, identities pairs of identities and where the composition is defined componentwise. $\lrcorner$

### 9.4.4 Functors

The notion of (covariant) functor is a natural generalization of the usual notion of morphism:

**Definition 9.12** (Functor). Let $\mathcal{C}$ and $\mathcal{D}$ be two categories. A *covariant functor* $F$ from $\mathcal{C}$ to $\mathcal{D}$ is a correspondence that maps each object $a$ of $\mathcal{C}$ to an object $F(a)$ of $\mathcal{D}$, and each morphism $f$ in $\mathcal{C}(a,b)$ to a morphism $F(f)$ in $\mathcal{D}(F(a),F(b))$ for all $a,b \in \mathcal{C}$, which preserves:

- the identity: $\forall a \in \mathcal{C}, F(\mathtt{id}_a) = \mathtt{id}_{F(a)}$
- the composition: $\forall f \in \mathcal{C}(a,b), g \in \mathcal{C}(b,c), F(g \circ f) = F(g) \circ F(f)$

$\lrcorner$

**Example 9.13.** For instance, we can define the powerset functor $\mathcal{P} : \mathbf{Set} \to \mathbf{Set}$ which constructs the subsets of a set:

$$\mathcal{P} : \begin{cases} x \mapsto \mathcal{P}(x) \\ (f : x \to y) \mapsto \mathcal{P}f : \begin{cases} \mathcal{P}(x) \to \mathcal{P}(y) \\ s \mapsto f(s) \end{cases} \end{cases}$$

$\lrcorner$

The composition of functors is defined canonically. An *isomorphism of categories* is as a functor which is bijective both on objects and on morphisms (or equivalently as a functor which is invertible for the composition of functors). This allows us to define **Cat**, the category whose objects are categories and whose morphisms are functors.

The previous definition can be extended to the notion of *contravariant functors*, which reverse morphisms and the composition:

**Definition 9.14** (Contravariant functor). A *contravariant functor* $F$ from $\mathcal{C}$ to $\mathcal{D}$ from $\mathcal{C}$ to $\mathcal{D}$ is a correspondence that maps each object $a$ of $\mathcal{C}$ to an object $F(a)$ of $\mathcal{D}$, and each morphism $f$ in $\mathcal{C}(a,b)$ to a morphism $F(f)$ in $\mathcal{D}(F(b),F(a))$ for all $a,b \in \mathcal{C}$, such that:

$$\forall f \in \mathcal{C}(a,b), \forall g \in \mathcal{C}(b,c), F(g \circ f) = F(f) \circ F(g)$$

Equivalently, a contravariant functor is a functor from $\mathcal{C}^{op}$ to $\mathcal{D}$. $\lrcorner$

Being given two categories, we can thus study the class of functors between these two categories. Actually, we can even equip this class with operators, which are called natural transformations:

**Definition 9.15** (Natural transformation). Let $\mathcal{C}$ and $\mathcal{D}$ be two categories, and $F, G : \mathcal{C} \to \mathcal{D}$ two functors. A *natural transformation* $\alpha$ from $F$ to $G$ is a family of morphisms $(\alpha_a)_{a \in \mathcal{C}}$, with $\alpha_a \in \mathcal{D}(F(a), G(a))$ for all $a \in \mathcal{C}$ and such that for all $f \in \mathcal{C}(a,b)$, the following diagram commutes:

$$\begin{tikzcd} F(a) \arrow[r, "F(f)"] \arrow[d, "\alpha_a"'] & F(b) \arrow[d, "\alpha_b"] \\ G(a) \arrow[r, "G(f)"'] & G(b) \end{tikzcd}$$

If in addition, for any object $a \in \mathcal{C}$, the morphism $\alpha_a$ is invertible, we say that $\alpha$ is a natural bijection. A functor $F : \mathcal{C} \to \mathcal{D}$ is then called an *equivalence of categories* when there exists a functor $G : \mathcal{D} \to \mathcal{C}$ and two natural bijections from $F \circ G$ (resp. $G \circ F$) to the identity functor of $C$ (resp. the one of $\mathcal{D}$). This notion generalizes the one of isomorphisms of categories. ⌟

**Definition 9.16** (Adjunction)**.** Let $\mathcal{C}$ and $\mathcal{D}$ be categories, an *adjunction* between $\mathcal{C}$ and $\mathcal{D}$ is a triple $(F, G, \varphi)$ where:

- $F$ is a functor from $\mathcal{D}$ to $\mathcal{C}$;
- $G$ is a functor from $\mathcal{C}$ to $\mathcal{D}$;
- for all $c \in \mathcal{C}, d \in \mathcal{D}$, $\varphi_{c,d}$ is a bijection from $\mathcal{C}(F(d), c)$ to $\mathcal{D}(d, G(c))$, natural in $c$ and $d$.

We denote it by $F \dashv G$, $F$ is said to be the *left adjoint* (of $G$), and vice-versa. ⌟

We introduce a last definition describing a broad class of categories. These categories allow for instance to give a categorical counterpart to the $\lambda$-calculus, see for instance [8] for an introductory presentation.

**Definition 9.17** (Cartesian category)**.** Let $\mathcal{C}$ be a category, $a, b \in \mathcal{C}$. A product of $a$ and $b$ is a triple $(a \times b, \pi_a, \pi_b)$, where $a \times b \in \mathcal{C}$, $\pi^1_{a\times b} \in \mathcal{C}(a \times b, a)$ and $\pi^2_{a\times b} \in \mathcal{C}(a \times b, b)$ are such that for all $f \in \mathcal{C}(c, a), g \in \mathcal{C}(c, b)$, there exists a unique morphism $\langle f, g\rangle \in \mathcal{C}(c, a \times b)$ such that the following diagrams commutes:

$$\begin{tikzcd}
 & c \arrow[dl, "f"'] \arrow[d, "\langle f,g\rangle"] \arrow[dr, "g"] & \\
a & a \times b \arrow[l, "\pi^1_{a\times b}"] \arrow[r, "\pi^2_{a\times b}"'] & b
\end{tikzcd}$$

A category is said *Cartesian* if it contains a terminal object $\top$ and if every pair of objects has a product. A Cartesian category is said to be *closed* if for any object $c \in \mathcal{C}$, the functor $(\cdot) \times c : \mathcal{C} \to \mathcal{C}$ has a right-adjoint, which we write $c \to (\cdot)$. ⌟

### 9.4.5 Hyperdoctrines and triposes

We can now define the structures which allow for a categorical approach of realizability. First, we recall the definition of Heyting algebras:

**Definition• 9.18.** A Heyting algebra $\mathcal{H}$ is a bounded lattice such that for all $a, b \in \mathcal{H}$ there is a greatest element $x$ of $\mathcal{H}$ such that $a \wedge x \leq b$. This element is denoted $a \to b$. ⌟

In any Heyting algebra, one defines the pseudo-complement $\neg a$ of any element $a$ by setting $\neg a \triangleq (a \to \bot)$. By definition, $a \wedge \neg a = \bot$ and $\neg a$ is the largest element having this property. However, it is not true in general that $a \vee \neg a = \top$, thus $\neg$ is only a pseudo-complement, not a real complement, as would be the case in a Boolean algebra. A *complete Heyting algebra* is a Heyting algebra that is complete as a lattice. Observe that Heyting algebras form a category[13] **HA** whose morphisms $F : \mathcal{H} \to \mathcal{H}'$ are the morphisms of the underlying lattice structure preserving Heyting's implication: $F(a \to b) = F(a) \to F(b)$ for all $a, b \in \mathcal{H}$.

In the category of Heyting algebras, we have a particular notion of adjunction, which is peculiar to partially ordered sets:

[13] Formally, **HA** is a subcategory of the category **Ord** of pre-orders. This category is sometimes called of Heyting prealgebras since the equality is induced by the preorder relation $a = b \triangleq a \leq b \wedge b \leq a$. In the literature this equality is sometimes written $a \cong b$ and called an isomorphism to distinguish it from the equality of pre-ordered sets.

**Definition 9.19** (Galois connection)**.** A *Galois connection* between two posets $A, B$ is a pair of function $f : A \to B, g : B \to A$ such that:

$$f(x) \leq y \Leftrightarrow x \leq g(y)$$

┘

For instance, the following examples are Galois connections:

- the natural injection and the floor form a Galois connection between $\mathbb{N}$ and $\mathcal{R}$:

$$\forall n \in \mathbb{N}, \forall x \in \mathcal{R}, (n \leq x \Leftrightarrow n \leq \lfloor x \rfloor)$$

- in any Heyting algebra $\mathcal{H}$, given $a \in \mathcal{H}$, we have:

$$\forall x, y \in \mathcal{H}, (a \wedge x \leq y \Leftrightarrow x \leq a \to y)$$

- in any lattice $\mathcal{L}$, (binary) meets and joins are respectively the left and right adjoints of a Galois connection formed with the diagonal morphism $\Delta : \mathcal{L} \to \mathcal{L} \times \mathcal{L}$.

**Proposition 9.20.** *If $(f, g)$ is a Galois connection between two ordered sets $A, B$, then:*

1. *$f$ and $g$ are monotonic functions,*
2. *$g$ is fully determined by $f$ (and thus unique) and vice-versa.*

*Proof.* It is easy to check that indeed, $f$ is uniquely determined by $g$:

$$f(x) = \min \{y \in B : x \leq g(y)\} \qquad \text{(for all } x \in A\text{)}$$

and vice-versa. □

We are now ready to define the key notion of (first-order) hyperdoctrine, due to Lawvere [105]. While there are many definitions of this notion in the literature, they are not always equivalent. Here, we follow Pitt's presentation [136] by adopting a minimal definition. This definition captures exactly the notion of first-order theory with equality.

**Definition 9.21** (Hyperdoctrine)**.** Let $C$ be a Cartesian closed category. A *first-order hyperdoctrine* over $C$ is a contravariant functor $\mathcal{T} : C^{op} \to \mathbf{HA}$ with the following properties:

1. For each diagonal morphism $\delta_X : X \to X \times X$ in $C$, the left adjoint to $\mathcal{T}(\delta_X)$ at the top element $\top \in \mathcal{T}(X)$ exists. In other words, there exists an element $=_X \in \mathcal{T}(X \times X)$ such that for all $\varphi \in \mathcal{T}(X \times X)$:
$$\top \;\leq\; \mathcal{T}(\delta_X)(\varphi) \qquad \Leftrightarrow \qquad =_X \;\leq\; \varphi$$
2. For each projection $\pi^1_{\Gamma,X} : \Gamma \times X \to \Gamma$ in $C$, the monotonic function $\mathcal{T}(\pi^1_{\Gamma,X}) : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma \times X)$ has both a left adjoint $(\exists X)_\Gamma$ and a right adjoint $(\forall X)_\Gamma$:
$$\begin{array}{rcl} \varphi \leq \mathcal{T}(\pi^1_{\Gamma,X})(\psi) & \Leftrightarrow & (\exists X)_\Gamma(\varphi) \leq \psi \\ \mathcal{T}(\pi^1_{\Gamma,X})(\varphi) \leq \psi & \Leftrightarrow & \varphi \leq (\forall X)_\Gamma(\psi) \end{array}$$
3. These adjoints are natural in $\Gamma$, i.e. given $s : \Gamma \to \Gamma'$ in $C$, the following diagrams commute:

$$\begin{tikzcd} \mathcal{T}(\Gamma' \times X) \arrow[r, "\mathcal{T}(s \times \mathrm{id}_X)"] \arrow[d, "(\exists X)_{\Gamma'}"'] & \mathcal{T}(\Gamma \times X) \arrow[d, "(\exists X)_\Gamma"] \\ \mathcal{T}(\Gamma') \arrow[r, "\mathcal{T}(s)"'] & \mathcal{T}(\Gamma) \end{tikzcd} \qquad \begin{tikzcd} \mathcal{T}(\Gamma' \times X) \arrow[r, "\mathcal{T}(s \times \mathrm{id}_X)"] \arrow[d, "(\forall X)_{\Gamma'}"'] & \mathcal{T}(\Gamma \times X) \arrow[d, "(\forall X)_\Gamma"] \\ \mathcal{T}(\Gamma') \arrow[r, "\mathcal{T}(s)"'] & \mathcal{T}(\Gamma) \end{tikzcd}$$

This condition is also called the *Beck-Chevaley conditions*.

The elements of $\mathcal{T}(X)$, as $X$ ranges over the objects of $C$, are called the $\mathcal{T}$*-predicates*. ⌟

Let us give some intuitions about this definition, which are related to the informal introduction of hyperdoctrine we did at the beginning of the chapter:

- The base category $C$ is the *domain of discourse*, that is to say that its elements are types or contexts (whence the suggestive notations $X$ and $\Gamma$) on which the predicates range. Its morphisms thus correspond to substitutions, while products $\Gamma \times \Gamma'$ should be understood as the concatenations of contexts.
- The functor $\mathcal{T}$ associates to each context $\Gamma \in C$ the sets of predicates over $\Gamma$. It might be helpful to think of the elements of $\mathcal{T}(\Gamma)$ as formulas $\varphi(x_1,\ldots,x_n)$ of free variables $x_1 : X_1,\ldots,x_n : X_n$ with $\Gamma \equiv X_1,\ldots,X_n$. The structure of Heyting algebra means that predicates can be compound by means of the connectives $\wedge, \vee, \rightarrow$ and that these operations respect the laws of intuitionistic propositional logic.
- The functoriality of $\mathcal{T}$, that is the fact that each morphism $s : \Gamma \rightarrow \Gamma'$ in $C$ induces a morphism $\mathcal{T}(s) : \mathcal{T}(\Gamma') \rightarrow \mathcal{T}(\Gamma)$, is to be understood as the existence of substitutions on formulas. In other words, if $\varphi(x)$ is a predicate ranging over $\Gamma$ and $s$ is as above, then $\mathcal{T}(s)(\varphi)$ is intuitively the predicate $\varphi(s(y))$.
- The ordering on formulas corresponds to the inclusion of predicates in the sense of the associated theory, that is to say:
$$\varphi \leq \psi \quad \equiv \quad \forall(x:\Gamma).(\varphi(x) \Rightarrow \psi(x))$$
The induced equality on formulas is then extensional or, to put it differently, a relation of equi-provability:
$$\varphi = \psi \quad \equiv \quad \forall(x:\Gamma).(\varphi(x) \Leftrightarrow \psi(x))$$
- With these intuitions in mind, the diagonal morphism $\delta_X$ is nothing more than the function which duplicates variables, and the first condition simply means that:
$$\forall(x:X).(\top \Rightarrow \varphi(x,x)) \qquad \Leftrightarrow \qquad \forall(x,y:X).(x=y \Rightarrow \varphi(x,y))$$
- As explained in the introduction, since both quantifiers $\exists x : X.$ and $\forall x : X.$ bind the variable $x$, turning any formula ranging over $\Gamma \times X$ into a formula ranging over $\Gamma$, it is natural to interpret them as morphism from $\mathcal{T}(\Gamma \times X)$ to $\mathcal{T}(\Gamma)$. As for their definitions as left and right adjoints of the projection $\pi^1_{\Gamma\times X}$, *i.e.*:
$$\begin{array}{lcl} \varphi \leq \mathcal{T}(\pi^1_{\Gamma\times X})(\psi) & \Leftrightarrow & (\exists X)_\Gamma(\varphi) \leq \psi \\ \mathcal{T}(\pi^1_{\Gamma\times X})(\varphi) \leq \psi & \Leftrightarrow & \varphi \leq (\forall X)_\Gamma(\psi) \end{array}$$
they correspond to the following logical equivalences which characterize them:
$$\begin{array}{lcl} \forall(y:\Gamma,x:X).(\varphi(y,x) \Rightarrow \psi(y)) & \Leftrightarrow & \forall(y:\Gamma).(\exists(x:X).\varphi(y,x)) \Rightarrow \psi(y) \\ \forall(y:\Gamma,x:X).(\varphi(y) \Rightarrow \psi(y,x)) & \Leftrightarrow & \forall(y:\Gamma).\varphi(y) \Rightarrow \forall(x:X).\psi(y,x) \end{array}$$
- Using the equality predicates and the adjoints for first projections, one can show that in fact for every morphism $f : X \rightarrow Y$, $\mathcal{T}(f) : \mathcal{T}(Y) \rightarrow \mathcal{T}(X)$ has left and right adjoints, which for any $y \in Y$ are intuitively given by:
$$\begin{array}{lcl} \exists(f)(\varphi)(y) & \equiv & \exists(x:X).(f(x) = y \wedge \varphi(x)) \\ \forall(f)(\varphi)(y) & \equiv & \forall(x:X).(f(x) = y \Rightarrow \varphi(x)) \end{array}$$

- Finally, the Beck-Chevaley conditions simply express that the quantifiers are compatible with the substitution. For instance, in the left-hand side diagram for the existential quantifier, given $\Gamma, \Gamma', X \in \mathcal{C}$ and a morphism $s : \Gamma \to \Gamma'$, the commutation of the diagram requires that:

$$\mathcal{T}(s) \circ (\exists X)_{\Gamma'} \quad = \quad (\exists X)_{\Gamma} \circ (\mathcal{T}(s \times \mathtt{id}_X))$$

  In terms of substitutions, the previous equality is nothing more than the requirement that for any $\varphi \in \mathcal{T}(\Gamma' \times X)$ and any $y' \in \Gamma'$:

$$(\exists (x : X).\varphi(y,x))[y := s(y')] \quad = \quad \exists (x : X).(\varphi(s(y'),x))$$

  The commutation of the other diagram gives the same equality for the universal quantifier.

Remembering the introduction of this chapter, the definition of Kleene's realizability naturally induces a hyperdoctrine structure where each set $X$ is associated to the Heyting algebra $(\mathcal{P}(\mathbb{N})^X, \vdash_X)$. Actually, any complete Heyting algebra gives rise to a hyperdoctrine whose structure is very similar:

**Example 9.22** (Hyperdoctrine of a complete Heyting algebra)**.** Let $\mathcal{H}$ be a complete Heyting algebra. The functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HA}$ given by:

$$\mathcal{T}(X) = \mathcal{H}^X \qquad \text{and} \qquad \mathcal{T}(f) : \begin{cases} \mathcal{H}^Y \to \mathcal{H}^X \\ g \mapsto (x \mapsto g(f(x))) \end{cases} \text{for any } f \in X \to Y$$

defines a hyperdoctrine. The $\mathcal{T}$-predicates are indexed families of elements of $\mathcal{H}$, ordered componentwise. The equality predicates are given by:

$$=_X (x,x') \triangleq \begin{cases} \top & \text{if } x = x' \\ \bot & \text{if } x \neq x' \end{cases}$$

where $\top$ (resp. $\bot$) is the greatest (resp. least) element of $\mathcal{H}$. The adjoints are defined thanks to the completeness of $\mathcal{H}$:

$$(\exists X)_\Gamma(\varphi)(y) = \bigvee_{x \in X} \varphi(y,x) \qquad\qquad (\forall X)_\Gamma(\varphi)(y) = \bigwedge_{x \in X} \varphi(y,x)$$

The Beck-Chevaley conditions are easily verified. In the case of the existential quantifier, for all $\Gamma, \Gamma', X \in \mathcal{C}$, any $\varphi \in \mathcal{H}^{\Gamma \times X}$ and any $s : \Gamma \to \Gamma'$, we have:

$$\begin{aligned} (\mathcal{T}(s) \circ (\exists X)_{\Gamma'})(\varphi) &= \mathcal{T}(s)(y' \mapsto \textstyle\bigvee_{x \in X} \varphi(y',x)) \\ &= y \mapsto \textstyle\bigvee_{x \in X} \varphi(s(y),x) \\ &= y \mapsto \textstyle\bigvee_{x \in X} \mathcal{T}(s \times \mathtt{id}_X)(\varphi) \\ &= ((\exists X)_\Gamma \circ \mathcal{T}(s \times \mathtt{id}_X))(\varphi) \end{aligned}$$

⌟

Hyperdoctrines are thus tailored to furnish a categorical representation of theories in first-order intuitionistic predicate logic. It was then observed that when a hyperdoctrine has enough structure, the model it gives can be somewhat internalized into a topos[14]. The hyperdoctrines for which this construction is possible were called *triposes* by Hyland, Johnstone and Pitts in [79].

[14]We will not introduce toposes in this thesis. A topos can regarded as a generalization of the category of sets, as such, the set-theoretic foundations of mathematics can expressed in terms of toposes. Toposes are useful structures for the categorical analysis of (high-order) logic. The standard reference for logic interpretation through toposes is Johnstone's book *Sketches of an elephant* [85].

**Definition 9.23** (Tripos)**.** A *tripos* over a Cartesian closed category $\mathcal{C}$ is a first-order hyperdoctrine $\mathcal{T} : \mathcal{C}^{op} \to \mathbf{HA}$ which has a *generic predicate*, *i.e.* there exists an object $\mathtt{Prop} \in \mathcal{C}$ and a predicate $\mathtt{tr} \in \mathcal{T}(\mathtt{Prop})$ such that for any object $\Gamma \in \mathcal{C}$ and any predicate $\varphi \in \mathcal{T}(\Gamma)$, there exists a (not necessarily unique) morphism $\chi_\varphi \in \mathcal{C}(\Gamma, \mathtt{Prop})$ such that:

$$\varphi = \mathcal{T}(\chi_\varphi)(\mathtt{tr})$$

⌟

Before giving some examples, we shall say that:

- the object $\mathtt{Prop} \in \mathcal{C}$, as the notation suggests, is the type of *propositions*;
- the generic predicate $\mathtt{tr} \in \mathcal{T}(\mathtt{Prop})$ is the *truth predicate*;
- for each predicate $\varphi \in \mathcal{T}(\Gamma)$, the arrow $\chi_\varphi \in \mathcal{C}(\Gamma, \mathtt{Prop})$ is then a propositional function representing $\varphi$, since for any $x \in \Gamma$, we intuitively have:

$$\mathtt{tr}(\chi_\varphi(x)) \equiv \varphi(x)$$

**Example 9.24.**

1. The example described in the introduction for Kleene's realizability indeed defines a tripos.
2. Given a complete Heyting algebra, the hyperdoctrine given by the functor $\mathcal{T}(X) = \mathcal{H}^X$ (see Example 9.22) is a tripos, with $\mathtt{Prop}$ being defined as (the underlying set of) $\mathcal{H}$, and the truth predicate being given by $\mathtt{tr} \triangleq \mathtt{id}_{\mathcal{H}} \in \mathcal{T}(\mathcal{H})$.

⌟

## 9.5 Algebraic structures for (classical) realizability

### 9.5.1 OCA: ordered combinatory algebras

Finally, we recall in this section the different algebraic structures arising from realizability. We first present the notion of *ordered combinatory algebras*, abbreviated in OCA, which is a variant[15] of Hofstra and Van Oosten's notion of ordered partial combinatory algebras [76].

**Definition 9.25** (OCA)**.** An *ordered combinatory algebra* is a quintuple $(\mathcal{A}, \leq, \mathsf{app}, \boldsymbol{k}, \boldsymbol{s})$, which we simply write $\mathcal{A}$, where:

- $\leq$ is a partial order over $\mathcal{A}$,
- $\mathsf{app} : (a,b) \mapsto ab$ is a monotonic function[16] from $\mathcal{A} \times \mathcal{A}$ to $\mathcal{A}$,
- $\boldsymbol{k} \in \mathcal{A}$ is such that $\boldsymbol{k}ab \leq a$ for all $a, b \in \mathcal{A}$,
- $\boldsymbol{s} \in \mathcal{A}$ is such that $\boldsymbol{s}abc \leq ac(bc)$ for all $a, b, c \in \mathcal{A}$.

⌟

Given an ordered combinatory algebra $\mathcal{A}$, we define the set of downward closed subsets of $\mathcal{A}$, which we write $D(\mathcal{A})$:

$$D(\mathcal{A}) \triangleq \{S \subset \mathcal{A} : \forall a \in \mathcal{A}, \forall b \in S, a \leq b \Rightarrow a \in S\}$$

The *standard realizability tripos* on $\mathcal{A}$ is defined by the functor $\mathcal{T}$ which associates to each set $X \in \mathbf{Set}^{op}$ the set of functions $D(\mathcal{A})^X$, which is equipped with the ordering:

$$\varphi \vdash_X \psi \quad \triangleq \quad \exists a \in \mathcal{A}. \forall x \in X. \forall b \in \mathcal{A}. (b \in \varphi(x) \Rightarrow ab \in \psi(x))$$

[15] In partial combinatory algebras, the application is defined as a partial function.

[16] Observe that the application, which is written as a product, is neither commutative nor associative in general.

The type of propositions Prop is defined as $D(\mathcal{A})$ itself and the generic predicate is defined as the identity of $D(\mathcal{A})$. While this definition is standard[17] in the framework of intuitionistic realizability [160]—the reader might in particular recognize the structure underlying the example we gave in the introduction—, its counterpart for classical logic is slightly different.

In his paper [151], Streicher exhibits the notion of *abstract Krivine structure* (which we write AKS), which he shows to be a particular case of OCA. Yet,the so-called *Krivine tripos* he constructs afterwards is defined as a functor mapping any set $X$ to the set of functions $\mathcal{A}^X$ with values in $\mathcal{A}$ (instead of a powerset like $\mathcal{D}(A)$). To this purpose, he considers *filtered ordered combinatory algebras*, which are the given of an OCA with a filter:

**Definition 9.26** (Filter)**.** If $\mathcal{A}$ is an OCA, a *filter* over $\mathcal{A}$ is a subset $\Phi \subseteq \mathcal{A}$ such that:

- $\boldsymbol{k} \in \Phi$ and $\boldsymbol{s} \in \Phi$,
- $\Phi$ is closed under application, *i.e.* if $a, b \in \Phi$ then $ab \in \Phi$.

⌟

**Remark 9.27.** It is a well-known fact that Hilbert's combinators $K$ and $S$ are complete with respect to the $\lambda$-calculus, in the sense that any closed $\lambda$-terms can be encoded as a combination of $K$ and $S$ which is adequate with the $\beta$-reduction. Similarly, in an ordered combinatory algebra, any $\lambda$-terms $t$ can be encoded as a combination $t^*$ of $\boldsymbol{k}$ and $\boldsymbol{s}$ such that the $\beta$-reduction is reflected through the ordering: for any $\lambda$-terms $t(x)$ and $u$, we have[18]:

$$((\lambda x.t)u)^* \leq (t[u/x])^*$$

We shall thus abuse the notation to write closed $\lambda$-terms as if they were elements $\mathcal{A}$. Besides, by definition of the notion of filter, any filter $\Phi$ contains all the closed $\lambda$-terms. ⌟

### 9.5.2 AKS: abstract Krivine structures

Krivine abstract structures are merely an axiomatization of the Krivine abstract machine viewed as an algebraic structure:

**Definition 9.28** (AKS)**.** An *abstract Krivine structure* is a septuple $(\boldsymbol{\Lambda}, \boldsymbol{\Pi}, \mathsf{app}, \mathsf{push}, \mathsf{k}_{\_}, \boldsymbol{k}, \boldsymbol{s}, \boldsymbol{cc}, \mathbf{PL}, \perp\!\!\!\perp)$ where:

1. $\boldsymbol{\Lambda}$ and $\boldsymbol{\Pi}$ are non-empty sets, respectively called the *terms* and *stacks* of the AKS;
2. $\mathsf{app} : t, u \mapsto tu$ if a function (called *application*) from $\boldsymbol{\Lambda} \times \boldsymbol{\Lambda}$ to $\boldsymbol{\Lambda}$;
3. $\mathsf{push} : t, \pi \mapsto t \cdot \pi$ if a function (called *push*) from $\boldsymbol{\Lambda} \times \boldsymbol{\Pi}$ to $\boldsymbol{\Pi}$;
4. $\mathsf{k}_{\_} : \pi \mapsto \mathsf{k}_\pi$ if a function from $\boldsymbol{\Pi}$ to $\boldsymbol{\Lambda}$ ($\mathsf{k}_\pi$ is called a *continuation*);
5. $\boldsymbol{k}$, $\boldsymbol{s}$ and $\boldsymbol{cc}$ are three distinguished terms of $\boldsymbol{\Lambda}$;
6. $\perp\!\!\!\perp \subseteq \boldsymbol{\Lambda} \times \boldsymbol{\Pi}$ (called the *pole*) is a relation between terms and stacks, also written $t \star \pi \in \perp\!\!\!\perp$. This relation fulfills the following axioms for all terms $t, u, v \in \boldsymbol{\Lambda}$ and all stacks $\pi, \pi' \in \boldsymbol{\Lambda}$:

$$\begin{array}{ccc}
tu \star \pi \in \perp\!\!\!\perp & \text{whenever} & t \star u \cdot \pi \in \perp\!\!\!\perp \\
\boldsymbol{k} \star t \cdot u \cdot \pi \in \perp\!\!\!\perp & \text{whenever} & t \star \pi \in \perp\!\!\!\perp \\
\boldsymbol{s} \star t \cdot u \cdot v \cdot \pi \in \perp\!\!\!\perp & \text{whenever} & tv(uv) \star \pi \in \perp\!\!\!\perp \\
\boldsymbol{cc} \star t \cdot \pi \in \perp\!\!\!\perp & \text{whenever} & t \star \mathsf{k}_\pi \cdot \pi \in \perp\!\!\!\perp \\
\mathsf{k}_\pi \star t \cdot \pi' \in \perp\!\!\!\perp & \text{whenever} & t \star \pi \in \perp\!\!\!\perp
\end{array}$$

[17] To be exact, the very central notion is the one of *partial combinatory algebras* [160], which is not ordered and where app is defined as a partial function. In this case, the tripos associates to each sets the set of functions $\mathcal{P}(\mathcal{A})^X$ with values in the powerset of $\mathcal{A}$ rather than in $D(A)$.

[18] See [45] for instance for a proof.

7. $\mathbf{PL} \subseteq \mathbf{\Lambda}$ is a subset of $\mathbf{\Lambda}$ (whose elements are called the *proof-like terms*), which contains $\boldsymbol{k}, \boldsymbol{s}, \boldsymbol{cc}$ and is closed under application.

┘

It is obvious that any realizability model (in the sense given in Chapter 3) induces an abstract Krivine structure. In fact, almost all the definitions that we used in the previous chapters when defining realizability interpretations can be restated in terms of abstract Krivine structures. Given any subset of stacks $X \subseteq \mathbf{\Pi}$ (which we call a *falsity value*), we write $X^{\perp\!\!\!\perp}$ for its orthogonal set with respect to the pole:

$$X^{\perp\!\!\!\perp} \triangleq \{t \in \mathbf{\Lambda} : \forall \pi \in X, t \star \pi \in \perp\!\!\!\perp\}$$

Orthogonality for subsets $X \subseteq \mathbf{\Lambda}$ (*i.e.* a *truth value*) is defined identically. As usual we write $t \perp\!\!\!\perp \pi$ for $t \star \pi \in \perp\!\!\!\perp$ and $t \perp\!\!\!\perp X$ (resp. $X \perp\!\!\!\perp \pi$) for $t \in X^{\perp\!\!\!\perp}$ (resp. $\pi \in X^{\perp\!\!\!\perp}$). The set of falsity values closed under bi-orthogonality is then defined by:

$$\mathcal{P}_{\perp\!\!\!\perp}(\mathbf{\Pi}) \triangleq \{X \in \mathcal{P}(\mathbf{\Pi}) : X = X^{\perp\!\!\!\perp\perp\!\!\!\perp}\}$$

With these definitions, from any abstract Krivine structure can be constructed a filtered ordered combinatory algebra:

**Proposition 9.29** (From AKS to OCA). *If* $(\mathbf{\Lambda}, \mathbf{\Pi}, \mathsf{app}, \mathsf{push}, \mathsf{k}_{\_}, \boldsymbol{k}, \boldsymbol{s}, \boldsymbol{cc}, \mathbf{PL}, \perp\!\!\!\perp)$ *is an abstract Krivine structure, then the quintuple* $(\mathcal{P}_{\perp\!\!\!\perp}(\mathbf{\Pi}), \leq, \mathsf{app}', \{\boldsymbol{k}\}^{\perp\!\!\!\perp}, \{\boldsymbol{s}\}^{\perp\!\!\!\perp})$ *is an OCA, with:*

- $X \leq Y \triangleq X \supseteq Y$
- $\mathsf{app}'(X, Y) \triangleq \{\pi \in \Pi : \forall t \in Y^{\perp\!\!\!\perp}. t \cdot \pi \in X\}^{\perp\!\!\!\perp\perp\!\!\!\perp}$

*Besides,* $\Phi \triangleq \{X \in \mathcal{P}_{\perp\!\!\!\perp}(\Pi) : \exists t \in \mathbf{PL}. t \perp\!\!\!\perp X\}$ *defines a filter for this OCA.*

*Proof.* See [151] or [45]. □

Given a filtered ordered combinatory algebra $(\mathcal{A}, \Phi)$, one can define the functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathcal{A}$:

$$\mathcal{T}(X) = \mathcal{A}^X \qquad \text{and} \qquad \mathcal{T}(f) : \left\{ \begin{array}{l} \mathcal{A}^Y \to \mathcal{A}^X \\ g \mapsto (x \mapsto g(f(x))) \end{array} \right. \quad \text{for any } f \in X \to Y$$

endowed with the following *entailment* relation:

$$\varphi \vdash_X \psi \triangleq \exists a \in \Phi. \forall x \in X. a\varphi(x) \leq \psi(x) \qquad \text{(for all } X \in \mathbf{Set})$$

In such a case, we shall refer to $a$ as a *realizer*. It is easy to show that the entailment relation $\vdash_X$ actually defines an order relation on $\mathcal{T}(X)$. Therefore, this functor always defines what is called an indexed preorder. In the particular case where the filtered OCA arises from an AKS, it can even be shown that the functor $\mathcal{T}$ actually defines a tripos, which Streicher calls a *Krivine tripos* [151, Theorem 5.10].

### 9.5.3 $^{\mathcal{I}}$OCA: implicative ordered combinatory algebras

In the continuity of Streicher's work, Ferrer *et al.* defined a subclass of ordered combinatory algebras which possess precisely the additional structure necessary to make of the previous functor a tripos [45]. These algebras, which they call *Krivine ordered combinatory algebras* ($^{\mathcal{K}}$OCA), thus provide us with an algebraic interpretation of Krivine classical realizability. It turns out that they are naturally definable as a particular case of a slightly more general class of algebras, called *implicative ordered combinatory algebras* ($^{\mathcal{I}}$OCA). As we shall see, a $^{\mathcal{K}}$OCA, which is the classical counterpart of an $^{\mathcal{I}}$OCA, is obtained by adding to the latter a combinator corresponding to the usual `call/cc` operator.

**Definition 9.30** ($^{\mathcal{I}}$OCA)**.** An *implicative ordered combinatory algebra* consists of an octuple of the shape $(\mathcal{A}, \leq, \mathtt{app}, \mathtt{imp}, \boldsymbol{k}, \boldsymbol{s}, \boldsymbol{e}, \Phi)$, which we simply write $\mathcal{A}$ or $(\mathcal{A}, \Phi)$, where:

- $\leq$ is a partial order over $\mathcal{A}$, and $\mathcal{A}$ is meet-complete as a poset;
- $\mathtt{app} : (a,b) \mapsto ab$ is a monotonic function from $\mathcal{A} \times A$ to $\mathcal{A}$,
- $\mathtt{imp} : a,b \mapsto a \to b$ is a monotonic function from $\mathcal{A}^{op} \times \mathcal{A} \to \mathcal{A}$ (*i.e.* $\mathtt{imp}$ is monotonic in its second component, antitonic in the first);
- $\Phi \subseteq \mathcal{A}$ is a *filter*, closed by application and such that $\boldsymbol{k}, \boldsymbol{s}, \boldsymbol{e} \in \Phi$;
- the following holds for all $a,b,c \in \mathcal{A}$:
  - $\boldsymbol{k}ab \leq a$
  - $\boldsymbol{s}abc \leq ac(bc)$
  - if $a \leq b \to c$ then $ab \leq c$
  - if $ab \leq c$ then $\boldsymbol{e}a \leq b \to c$

┘

Observe that in particular, any $^{\mathcal{I}}$OCA is a filtered OCA. The extra requirement of an arrow, as the reader might have guessed, equips the sets $(\mathcal{A}^X, \vdash_X)$ with a structure of Heyting algebra. In other words, when $\mathcal{A}$ is an $^{\mathcal{I}}$OCA, the functor $\mathcal{T} : X \mapsto \mathcal{A}^X$ is a tripos. Indeed, thanks to combinatorial completeness of $\boldsymbol{k}$ and $\boldsymbol{s}$, we can define a meet through the usual encoding of pairs in $\lambda$-calculus. We define:

$$\mathsf{t} \triangleq \lambda xy.x \qquad \mathsf{f} \triangleq \lambda xy.y \qquad \mathsf{p} \triangleq \lambda xyz.zxy \qquad \mathsf{p}_0 \triangleq \lambda x.(x\mathsf{t}) \qquad \mathsf{p}_1 \triangleq \lambda x.(x\mathsf{f})$$

which ensures that $\mathsf{p}_0(\mathsf{p}ab) \leq a$ and $\mathsf{p}_1(\mathsf{p}ab) \leq b$. This allows us to define a map $\wedge : \mathcal{A} \times \mathcal{A} \to \mathcal{A}$ by $a \wedge b \triangleq \mathsf{p}ab$. As for the arrow, the $\mathtt{imp}$ operations naturally induces an arrow on formulas such that for any $X \in \mathbf{Set}$, and any $\varphi, \psi, \theta \in \mathcal{A}^X$, we have:

$$\varphi \vdash_X \psi \to \theta \qquad \text{if and only if} \qquad \varphi \wedge \psi \vdash_X \theta$$

Since we believe it might help the reader to see the connection with realizability, we sketch the proof of this statement. From left to right, the implication is trivial since if there exists $u \in \Phi$ such that for all $a \in \varphi(x), b \in \psi(x)$ and $c \in \theta(x)$, $ua \leq b \to c$, then by definition of the arrow $(ua)b \leq c$. Therefore, we can define the realizer $r \triangleq \lambda x.(xu)$ which belongs to $\Phi$ and verifies that $r(\mathsf{p}ab) \leq c$.

From right to left, the proof is very similar: if there exists $u \in \Phi$ such that for all $a \in \varphi(x), b \in \psi(x)$ and $c \in \theta(x)$, $u(\mathsf{p}ab) \leq c$, in particular we have $(\lambda y.u(\mathsf{p}ay))b \leq c$. Therefore, by definition of the arrow, we have that $\boldsymbol{e}(\lambda y.u(\mathsf{p}ay)) \leq b \to c$ and thus $\lambda x.\boldsymbol{e}(\lambda y.u(\mathsf{p}xy))$ is the expected realizer.

The complete proof that the functor $\mathcal{T}$ is a tripos can be found in [45].

### 9.5.4 $^{\mathcal{K}}$OCA: Krivine ordered combinatory algebras

This notion of $^{\mathcal{I}}$OCA can be slightly enforced to obtain the notion of *Krivine ordered combinatory algebras*, that should be simply understood as the usual addition of `call/cc` to go from an intuitionistic setting to the classical one:

**Definition 9.31** ($^{\mathcal{K}}$OCA)**.** A *Krivine ordered combinatory algebra* is an implicative combinatory algebra equipped with a distinguished element $\boldsymbol{c} \in \Phi$ such that for all $a,b \in \mathcal{A}$:

$$\boldsymbol{c} \leq ((a \to b) \to a) \to a$$

┘

**Example 9.32.** Any complete Boolean algebra $\mathcal{B}$ induces a $^{\mathcal{K}}$OCA by defining:

$$ab \triangleq a \wedge b \qquad a \to b \triangleq \neg a \vee b \qquad \Phi \triangleq \{\top\} \qquad \boldsymbol{s} \triangleq \boldsymbol{k} \triangleq \boldsymbol{e} \triangleq \boldsymbol{c} \triangleq \top$$

Broadly, Boolean algebras are trivial $^{\mathcal{K}}$OCA where all the realized elements are collapsed to $\top$. ┘

Interestingly, any abstract Krivine structure gives rise to a Krivine ordered combinatory algebra, and vice-versa. In both cases, the induced triposes (by the AKS and the ${}^{\mathcal{K}}$OCA) are equivalent. This justifies the claim that the latter indeed captures the necessary additional structure that allows an OCA induced from an AKS to be a tripos. These results are a refinement of Proposition 9.29:

**Proposition 9.33** (From AKS to ${}^{\mathcal{K}}$OCA)**.** *If* $(\mathbf{\Lambda},\mathbf{\Pi},\mathsf{app},\mathsf{push},\mathsf{k}_{\_},\boldsymbol{k},\boldsymbol{s},\boldsymbol{cc},\mathbf{PL},\perp\!\!\!\perp)$ *is an abstract Krivine structure, then the nonuple* $(\mathcal{P}_{\perp\!\!\!\perp}(\mathbf{\Pi}),\leq,\mathsf{app}',\mathtt{imp}',\{\boldsymbol{k}\}^{\perp\!\!\!\perp},\{\boldsymbol{s}\}^{\perp\!\!\!\perp},\{\boldsymbol{cc}\}^{\perp\!\!\!\perp},\{\boldsymbol{e}\}^{\perp\!\!\!\perp},\Phi)$ *is a* ${}^{\mathcal{K}}$*OCA, with:*

- $X \leq Y \;\triangleq\; X \supseteq Y$;
- $\mathsf{app}'(X,Y) \triangleq \{\pi \in \Pi : \forall t \in Y^{\perp\!\!\!\perp}. t\cdot\pi \in X\}^{\perp\!\!\!\perp\perp\!\!\!\perp}$;
- $\mathtt{imp}'(X,Y) \triangleq \{t\cdot\pi \in \Pi : t \in X^{\perp\!\!\!\perp} \wedge \pi \in Y\}^{\perp\!\!\!\perp\perp\!\!\!\perp}$;
- $\boldsymbol{e} \triangleq \boldsymbol{s}(\boldsymbol{k}(\boldsymbol{skk}))$;

*Besides,* $\Phi \triangleq \{X \in \mathcal{P}_{\perp\!\!\!\perp}(\Pi) : \exists t \in \mathbf{PL}. t \perp\!\!\!\perp X\}$ *defines a filter for this OCA.*

**Proposition 9.34** (From ${}^{\mathcal{K}}$OCA to AKS )**.** *If* $(\mathcal{A},\leq,\mathsf{app}_{\mathcal{A}},\mathtt{imp}_{\mathcal{A}},\boldsymbol{k},\boldsymbol{s},\boldsymbol{c},\boldsymbol{e},\Phi)$ *is a* ${}^{\mathcal{K}}$*OCA, then the septuple defined by* $(\mathcal{A},\mathcal{A},\mathsf{app},\mathsf{push},\mathsf{k}_{\_},\kappa,\varsigma,\textsc{c},\mathbf{PL},\perp\!\!\!\perp)$ *is an abstract Krivine structure, where:*

- $\perp\!\!\!\perp \triangleq \leq$ i.e. $t \perp\!\!\!\perp \pi \triangleq t \leq \pi$;
- $\mathsf{app}(t,u) \triangleq \mathsf{app}_t(t,u) = tu$;
- $\mathsf{push}(t,\pi) \triangleq \mathtt{imp}(t,\pi) = t\cdot\pi$;
- $\mathsf{k}_\pi \triangleq \pi \to \perp$;
- $\mathbf{PL} \triangleq \Phi$;
- $\kappa \triangleq \boldsymbol{e}(\boldsymbol{bek})$, $\varsigma \triangleq \boldsymbol{e}(\boldsymbol{b}(\boldsymbol{be}(\boldsymbol{be}))\boldsymbol{s})$, $\textsc{c} \triangleq \boldsymbol{e}\,\boldsymbol{c}$,

*where* $\boldsymbol{b}$ *is an abbreviation for* $\boldsymbol{s}(\boldsymbol{ks})\boldsymbol{k}$.

*Proof.* See [45, Theorem 5.11] for the first proposition, [45, Theorem 5.13] for the second. □

Without considering in details the proofs of the correspondences between AKS and ${}^{\mathcal{K}}$OCA or their associated triposes, it is worth noting that when going from a ${}^{\mathcal{K}}$OCA $\mathcal{A}$ to a AKS, both sets $\mathbf{\Lambda}$ and $\mathbf{\Pi}$ are defined as $\mathcal{A}$. This means in particular that realizers and their opponents live in the same world, and the orthogonality relation is simply reflected by the order. That is $t \perp\!\!\!\perp \pi$ if $t \leq \pi$, and more generally if $X \subseteq \mathcal{P}(\mathbf{\Pi})$, $t \perp\!\!\!\perp X$ if for any $x \in X$, $t \leq x$. If, as advocated in Section 9.2, we identify a closed formula $A$ with its falsity values $\|A\|$, we recover the intuition that $t \Vdash A$ is reflected by the ordering $t \leq \|A\|$. With these ideas in mind, we are now ready to see the more general notion of implicative algebra.