Revisiting the duality of computation: an algebraic analysis of classical realizability models

Étienne Miquey
Équipe Gallinette, INRIA, Université de Nantes
Email: etienne.miquey@inria.fr

Abstract—In an impressive series of papers, Krivine showed at the edge of the last decade how classical realizability furnishes a surprising technique to build models for classical theories. In particular, he proved that classical realizability subsumes Cohen’s forcing, and even more, gives rise to unexpected models of set theories. Pursuing the algebraic analysis of these models that was first undertaken by Streicher, Miquel recently proposed to lay the algebraic foundation of classical realizability and forcing within new structures which he called implicative algebras. These structures are a generalization of Boolean algebras based on an internal law representing the implication. Notably, implicative algebras allow for the adequate interpretation of both programs (i.e. proofs) and their types (i.e. formulas) in the same structure.

The very definition of implicative algebras takes position on a presentation of logic through universal quantification and the implication and, computationally, relies on the call-by-name λ-calculus. In this paper, we investigate the relevance of this choice, by introducing two similar structures. On the one hand, we define disjunctive algebras, which rely on internal laws for the negation and the disjunction and which we show to be particular cases of implicative algebras. On the other hand, we introduce conjunctive algebras, which rather put the focus on conjunctions and on the call-by-value evaluation strategy. We finally show how disjunctive and conjunctive algebras algebraically reflect the well-known duality of computation between call-by-name and call-by-value.

I. INTRODUCTION

Krivine classical realizability: It is well-known since Griffin’s seminal work [12] that a classical Curry-Howard correspondence can be obtained by adding control operators to the λ-calculus. Several calculi were born from this idea, amongst which Krivine λc-calculus [19], defined as the λ-calculus extended with Scheme’s call/cc operator (for call-with-current-continuation). Elaborating on this calculus, Krivine’s developed in the late 90s the theory of classical realizability [19], which is a complete reformulation of its intuitionistic twin. Originally introduced to analyze the computational content of classical programs, it turned out that classical realizability also provides interesting semantics for classical theories. While it was first tailored to Peano second-order arithmetic (i.e. second-order type systems), classical realizability actually scales to more complex classical theories like ZF [20], and gives rise to surprisingly new models. In particular, its generalizes Cohen’s forcing [20], [26] and allows for the direct definition of a model in which neither the continuum hypothesis nor the axiom of choice holds [22].

Algebraization of classical realizability: During the last decade, the algebraic structure of the models that classical realizability induces has been actively studied. This line of work was first initiated by Streicher, who proposed the concept of abstract Krivine structure [34], followed by Ferrer, Frey, Guillermo, Malherbe and Miquel who introduced other structures peculiar to classical realizability [8], [6], [5], [9], [10]. In addition to the algebraic study of classical realizability models, these works had the interest of building the bridge with the algebraic structures arising from intuitionistic realizability. In particular, Streicher showed in [34] how classical realizability could be analyzed in terms of tripodes [33], the categorical framework emerging from intuitionistic realizability models, while the later work of Ferrer et al. [8] connected it to Hofstra and Van Oosten’s notion of ordered combinatory algebras [15]. More recently, Alexandre Miquel introduced the concept of implicative algebra [28], which appear to encompass the previous approaches and which we present in this paper.

Implicative algebras: In addition to providing an algebraic framework conducive to the analysis of classical realizability, an important feature of implicative structures is that they allow us to identify realizers (i.e. λ-terms) and truth values (i.e. formulas). Concretely, implicative structures are complete lattices equipped with a binary operation $a \to b$ satisfying properties 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 $a \equiv b$ will encompass different intuitions depending on whether we regard $a$ and $b$ as formulas or as terms. Namely, $a \equiv b$ 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 $b$.

In terms of the Curry-Howard correspondence, this means that we not only identify types with formulas and proofs with programs, but we also identify types and programs.

Side effects: Following Griffin’s discovery on control operators and classical logic, several works have renewed the observation that within the proofs-as-programs correspondence, with side effects come new reasoning principles [18], [17], [27], [13], [16]. More generally, it is now clear that computational features of a calculus may have consequences it induces. For instance, computational proofs of the axiom of dependent choice can be obtained by adding a quote instruction [18], using memoisation [14], [31] or with a bar recuror [24]. Yet, such choices may also have an impact on the structures of the corresponding realizability models: the
non-deterministic operator \( \triangledown \) is known to make the model collapse on a forcing situation \([21]\), while the bar recursor requires some continuity properties \([24]\).

If we start to have a deep understanding of the algebraic structure of classical realizability models, the algebraic counterpart of side effects on this structure is still unclear. As a first step towards this problem, it is natural to wonder: does the choice of an evaluation strategy have algebraic consequences on realizability models? This is the main question that this paper addresses.

**Outline of the paper:** We start by recalling the definition of Miquel’s implicative algebras and their main properties in Section [II]. We then introduce the notion of disjunctive algebras in Section [III] which naturally arises from the negative decomposition of the implication \( A \rightarrow B = \neg A \lor B \). We explain how this decomposition induces realizability models based on a call-by-name fragment of Munch-Maccagnoni’s system L, and which we show that disjunctive algebras are in fact particular cases of implicative algebras. In Section [IV], we explore the positive dual decomposition of the implication \( A \rightarrow B \rightarrow (A \lor B)^\bot \), which is known to make the model collapse on a forcing situation \([21]\), while the bar recursor \( \lambda \rightarrow \) does not contain any continuations \([1]\) and such that \( t \in \mathcal{P}(\Pi) \) is the falsity value of \( A \).

Within Krivine realizability models, a formula \( A \) is interpreted as a set of closed terms \( [A] \subseteq \Lambda \), called the *truth value* of \( A \), and whose elements are called the *realizers* of \( A \). Unlike in intuitionistic realizability models, this set is actually defined by orthogonality to a *falsity value* \( [\bot] \) of stacks, which intuitively represents a set of opponents to the formula \( A \). Realizability models are parameterized by a pole \( \downarrow \), a set of processes that is closed under anti-reduction and which somehow plays the role of a referee between terms and stacks. The pole allows us to define the orthogonal set \( X^\bot \) of any falsity value \( X \subseteq \Pi \) by:

\[
X^\bot \triangleq \{ t \in \Lambda : \forall \pi \in X, t \star \pi \in \downarrow \}
\]

Valid formulas are the one admitting a proof-like *realizer*, that is to say a term \( t \in \Lambda \) which does not contain any continuations \([1]\) and such that \( t \in [A]^\bot \) where \( [A] \in \mathcal{P}(\Pi) \) is the falsity value of \( A \).

Before pursuing with the definition of implicative algebras, we would like to draw the reader’s attention on an important observation: through the Curry-Howard interpretation of logic, and especially in realizability, there is an omnipresent lattice structure, which is reminiscent of the concept of subtyping \([3]\).

Indeed, given a pole \( \downarrow \) it is always possible to define a semantic notion of subtyping:

\[
A \triangleleft \downarrow B \triangleq [B] \subseteq [A]
\]

In this case, the relation \( \triangleleft \) 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 \( \lor \) as an intersection. In particular, universal quantifiers \( [\forall x.A] \) is interpreted as unions \( \bigcup_{x \in \mathbb{N}} [A[n/x]] \), *i.e.* a meet, while the logical connectives \( \wedge \) is interpreted as the type of pair, *i.e.* with a computation content. As such, classical realizability corresponds to the following picture:

<table>
<thead>
<tr>
<th>Realizability:</th>
<th>( \forall = \top )</th>
<th>( \wedge = \times )</th>
</tr>
</thead>
</table>

This is to compare with forcing, that can be expressed in terms of Boolean algebras and where both the universal quantifier and the conjunction are interpreted by meets:

\[
\forall = \wedge = \top \quad [1]
\]

**B. Implicative structures**

Implicative structures are tailored to represent both the formulas of second-order logic and realizers arising from Krivine’s \( \lambda \)-calculus. For their logical facet, they are defined as meet-complete lattices (for the universal quantification) with

\[A,B := X \mid A \Rightarrow B \mid \forall X.A\]

The terms of the language are typed with formulas of second-order logic, that is to say of system \( F \):

\[
A,B ::= X \mid A \Rightarrow B \mid \forall X.A
\]

The typing rules are the standard rules of (Curry-style) system \( F \), to which is added a rule to type \( ec \) with Peirce’s law \( \forall A.B.([A \Rightarrow B] \Rightarrow A) \Rightarrow A \).

Otherwise, one can show that as soon as the pole contain a process \( t \star \pi \), any formula can be realized by \( k_{\pi t} \), see \([29]\).
an internal binary operation satisfying the properties of the implication:

**Definition 1** An implicative structure is a complete lattice \((A, \leq)\) equipped with an operation \((a, b) \mapsto (a \to b)\), such that for all \(a, a_0, b, b_0 \in A\) and any subset \(B \subseteq A\):

1. If \(a_0 \leq a\) and \(b \leq b_0\) then \((a \to b) \leq (a_0 \to b_0)\).
2. \(\bigwedge_{b \in B}(a \to b) = a \to \bigwedge_{b \in B} b\)

It is then immediate to embed any closed formula of second-order logic within any implicative structure. Obviously, any complete Heyting algebra or any complete Boolean algebra defines an implicative structure with the canonical arrow. More interestingly, any ordered combinatorial algebras, a structure arising naturally from realizability [15, 35, 34, 7], also induces an implicative structure [30]. Last but not least, any classical realizability model induces as expected an implicative structure on the lattice \((\mathcal{P}(\Omega), \supseteq)\) by considering [28], [30]:

\[ a \to b \equiv a^\perp \cdot b = \{ t \cdot \pi : t \in a^\perp, \pi \in b \} \]

**C. Interpreting the \(\lambda\)-calculus**

We shall now give an overview of the interpretation of \(\lambda\)-terms within implicative structures.

**Definition 2** (\(\lambda\)-calculus). Given two elements \(a, b \in A\) and a function \(f : A \to A\), we define:

\[ \lambda f \equiv \bigwedge_{a \in A} (a \to f(a)) \]

Noteworthily, these definitions fulfill several properties in adequation with the computational behavior of the (call-by-name) \(\lambda\)-calculus.

**Proposition 3** (Properties). For all \(a \in A\) and \(f \in A \to A\), we have:

1. \((\lambda f) a \equiv f(a).\) (\(\beta\)-reduction)
2. \(a \equiv \lambda x (x \to ax).\) (\(\eta\)-expansion)
3. \(ab \equiv c \iff a \equiv (b \to c).\) (Adjunction)

Any closed \(\lambda\)-term \(t\) can thus be interpreted as an element \(t^A\) of the implicative structure. This interpretation is adequate with the interpretation of the second-order formulas.

**Theorem 4** For any closed \(t\), if \(\vdash t : A\), then \(t^A \leq A^A\).

**D. Implicative algebras**

**Definition 5** (Separator). Let \((A, \preceq, \to)\) be an implicative structure. We call a separator over \(A\) any set \(S \subseteq A\) such that for all \(a, b \in A\), the following conditions hold:

1. \(K^A \in S\) and \(S^A \in S\). (Combinators)
2. If \(a \in S\) and \(a \preceq b\), then \(b \in S\). (Upwards closure)
3. If \((a \to b) \in S\) and \(a \in S\), then \(b \in S\). (Closure under modus ponens)

A separator \(S\) is said to be classical if \(cc^A \in S^I\) and consistent if \(\bot \not\in S\). We call implicative algebra any implicative structure \((A, \preceq, \to, S)\) equipped with a separator \(S\) over \(A\).

\(^4\)Where \(cc^A\) is defined through its type by \(\lambda_{a,b}( ((a \to b) \to a) \to a)\).

Intuitively, thinking of elements of an implicative structure as truth values, a separator should be understood as the set which distinguishes the valid formulas. Considering the elements as terms, it should rather be viewed as the set of valid realizers. Indeed, conditions (1) and (3') ensure that all closed \(\lambda\)-terms are in any separator. Reading \(a \preceq b\) as “the formula \(a\) is a subtype of the formula \(b\)”, condition (2) ensures the validity of semantic subtyping. Thinking of the ordering as “\(a\) is a realizer of the formula \(b\)”, condition (2) states that if a formula is realized, then it is in the separator.

**Example 6** (Classical realizability). Any model induces an implicative structure \((A, \preceq, \to)\) where \(A = \mathcal{P}(\Pi)\). \(a \preceq b \iff a \supseteq b\) and \(a \to b = a^\perp \cdot b\). The set of realized formulas, namely \(S = \{a \in A : a^\perp \cap \text{PL} \neq \emptyset\}\), defines a valid separator [28].

**E. Internal logic & implicative tripos**

In order to study the internal logic of implicative algebras, we define an entailment relation: we say that \(a\) entails \(b\) and we write \(a \vdash b\) if \(a \to b \in S\). We say that \(a\) and \(b\) are equivalent and write \(a \equiv_S b\) if \(a \vdash b\) and \(b \vdash a\).

As mentioned earlier, we can define a product \(a \times b\) and a sum \(a + b\) through the usual impredicative encodings in System \(\text{IF}\). This induces a structure of pre-Heyting algebra with respect to the entailment relation (which is a preorder):

\[ a \vdash_S b \iff a \times b \vdash_S c \]

In order to recover a Heyting algebra, it suffices to consider the quotient \(\mathcal{H} = A/\!\!\!\equiv_S\) by the relation \(\equiv_S\), which is equipped with an order relation:

\[ [a] \vdash_S [b] \iff a \vdash_S b \quad \text{for all } a, b \in A \]

where we write \([a]\) for the equivalence class of \(a \in A\). We can then extend the product, the sum and the arrow to equivalences classes in order to obtain a Heyting algebra \((\mathcal{H}, \preceq, \land_{\mathcal{H}}, \lor_{\mathcal{H}}, \to_{\mathcal{H}})\).

The construction of the implicative tripos is quite similar. Recall that a (set-based) tripos is a first-order hyperdoctrine \(\mathcal{T} : \text{Set}^{op} \to \text{HA}\) which admits a generic predicate \(\exists\). To define a tripos, we roughly consider the functor of the form \(I \in \text{Set}^{op} \to \mathcal{A}^I\). Again, to recover a Heyting algebra we quotient the product \(\mathcal{A}^I\) (which defines an implicative structure) by the uniform separator \(S[I]\) defined by:

\[ S[I] \equiv \{ a \in \mathcal{A}^I : \exists s \in S, \forall i, a_i \equiv a \} \]

**Theorem 7** (Implicative tripos). Let \((A, \preceq, \to, S)\) be an implicative algebra. The following functor (where \(f : I \to J\)):

\[ \mathcal{T} : I \to \mathcal{A}^I/S[I] \quad \mathcal{T}(f) : \{ ((a_i)_{i \in I}) \mapsto ((a_{f(j)})_{j \in J}) \}
\]

defines a tripos.

Before presenting the novelties of this paper, we shall make two last observations related to the definition of the implicative tripos.

\(^5\)That is to say that we define \(a \times b \equiv \lambda_{a,b}( ((a \to b) \to c) \to c)\) and \(a + b \equiv \lambda_{a,b}( ((a \to c) \to (b \to c)) \to c)\).

\(^6\)See Definition [57] in the appendices for a complete definition.

\(^7\)See Appendix \(A\) for more details on the tripos construction.
classical realizability (see [19], [29]), the proof that $\mathcal{T}$ is a tripos relies on the definition of the universal quantification of a family of truth values as its meet while the existential quantification is defined through a negative encoding: $\exists e \in \mathcal{E}_i \mathcal{T} \models \bigwedge_{e \in \mathcal{E}} (a_i \rightarrow e) \rightarrow e$. While it could have seemed more natural to define existential quantifiers through joins, we should recall that the arrow does not commute with joins in general.

Second, we could alternatively quotient the product $A^I$ by the separator product $S^I$. Actually, one can check that $A^I / S^I$ is in bijection with $(A/S)^2$, and in the case where $S$ is a classical separator, $A / S$ is actually a Boolean algebra, so that the product $(A/S)^2$ is nothing more than a Boolean-valued model (as in the case of forcing). Since $S[I] \subseteq S^I$, the realizability models that can not be obtained by forcing are exactly those for which $S[I] \neq S^I$.

III. DECOMPOSING THE ARROW: DISJUNCTIVE ALGEBRAS

A. Decomposing the arrow

We shall now introduce the notion of disjunctive algebra, which is a structure primarily based on disjunctions, negations (for the connectives) and meets (for the universal quantifier). Our starting point is more general as the former. The first step in this direction is to decompose the arrow:

\[ \Rightarrow \]

In 2009, Guillaume Munch-Maccagnoni gave a computational account of Girard’s presentation for classical logic [32]. In his calculus, named $L$, each connective corresponds to the type of a particular constructor (or destructor). While $L$ is in essence close to Curien and Herbelin’s $\lambda\mu\nu$-calculus [4] (in particular it is presented with the same paradigm of duality between proofs and contexts), the syntax of terms does not include $\lambda$-abstraction (and neither does the syntax of formulas includes an implication). The two decompositions of the arrow evoked above are precisely reflected in a decomposition of $\lambda$-abstractions (and dually, of stacks) in terms of $L$ constructors. Notably, the choice of a decomposition corresponds to a particular choice of an evaluation strategy for the encoded $\lambda$-calculus. When picking the $\forall$ connective, the corresponding $\lambda$-terms are evaluated according to a call-by-name evaluation strategy, while the decomposition through the $\otimes$ connective induces a $\lambda$-calculus that is reduced in a call-by-value fashion.

We shall begin by considering the call-by-name case, which is closer to the situation of implicative algebras. We start with the presentation of the corresponding fragment of Munch-Maccagnoni’s calculus, which we call $L^\forall$. In particular, we will see how this calculus induces a realizability model whose structure leads us to the definition of disjunctive structures.

We will observe that the encoding of $\lambda$-terms into $L^\forall$ can be directly reflected through an implicative structure induced by each disjunctive structure. Finally, we shall define the notions of (disjunctive) separator and disjunctive algebra and we will see how any disjunctive algebra can be viewed as an implicative algebra.

B. The $L^\forall$ calculus

The $L^\forall$-calculus is the restriction of Munch-Maccagnoni’s system $L$ [32], to the negative fragment corresponding to the connectives $\forall$, $\neg$ (which we simply write $\neg$ since there is no ambiguity here) and $\forall$. To simplify things (and ease the connection with the $\lambda\mu\nu$-calculus [4]), we slightly change the notations of the original paper. As Krivine’s $\lambda\nu$-calculus, this language describes commands of abstract machines that made of a term $t$ taken within its evaluation contexts $e$. The syntax is given by:

\[ e ::= \alpha \mid (e_1, e_2) \mid [t] \mid \mu x.c \]

\[ t ::= x \mid \mu(\alpha_1, \alpha_2).c \mid \mu[x].c \mid \mu\alpha.c \]

\[ c ::= (t \parallel e) \]

We write $E_0$, $T_0$, $C_0$ for the sets of closed contexts, terms and commands. Values are defined by the following fragment of the syntax:\[11\]

\[ V ::= \alpha \mid [V_1, V_2] \mid [t] \]

We denote by $V_0$ the corresponding set of closed values.

We shall say a few words about it:

- $(e_1, e_2)$ are pairs of contexts, which we will relate to usual stacks;
- $\mu(\alpha_1, \alpha_2).c$, which binds the co-variables $\alpha_1, \alpha_2$, is the dual destructor for pairs;
- $[t]$ is a constructor for the negation, which allows us to embed a term into a context;
- $\mu[x].c$, which binds the variable $x$, is the dual destructor;
- $\mu\alpha.c$ binds a covariable and allows to capture a context: as such, it implements classical control.

\[8\]When it does, the implicative tripos actually collapses to a forcing tripos, see [28], [29].

\[9\]See [28] for more characterization criteria.

\[10\]To do justice to Girard’s approach, the implication which is considered in linear logic, written $\rightarrow$, is different from the usual one. The difference between both implications is not relevant in our framework.
is naturally defined as a shorthand for the pair \([\langle u \rangle, e]\), which indeed inhabits the type \(\neg A \Jordan B\). Starting from there, the rest of the definitions are straightforward:

\[
\begin{align*}
u \cdot e & \triangleq [\langle u \rangle, e] \\
m([x], \beta).c & \triangleq \mu (\alpha, \beta).\langle \mu [x].c, \alpha \rangle \\
t u & \triangleq \mu \alpha.\langle t \| u \cdot \alpha \rangle
\end{align*}
\]

These definitions are sound with respect to the typing rules expected from the \(\lambda\mu\mu\)-calculus (see Proposition 59). In addition, they induce the usual rules of \(\beta\)-reduction for the call-by-name evaluation strategy in the Krivine abstract machine (notice that in the KAM, all stacks are values):

\[
\begin{align*}
\langle t \| u \| \pi \rangle & \longrightarrow_{\beta} \langle t \| u \cdot \pi \rangle & (\pi \in V) \\
\langle ax.t \| u \cdot \pi \rangle & \longrightarrow_{\beta} \langle t[u/x] \| \pi \rangle & (\pi \in V)
\end{align*}
\]

Realizability models: We briefly go through the definition of the realizability interpretation à la Krivine for \(L^\forall\). As usual, we begin with the definition of a pole:

**Definition 8** (Pole). A pole is defined as any subset \(\perp \subseteq C\) such that \(c, c' \in C\) then \(c \longrightarrow_{\beta} c'\) and \(c' \in \perp\) then \(c \in \perp\).

As it is common in Krivine’s call-by-name realizability, falsity values are defined primitively as sets of contexts. Truth values are then defined by orthogonality to the corresponding falsity values. We say that a term \(t\) is orthogonal (with respect to the pole \(\perp\)) to a context \(e\) when \(\langle t \| e \rangle \in \perp\). A term \(t\) (resp. a context \(e\)) is said to be orthogonal to a set \(S \subseteq \mathcal{E}_0\) (resp. \(S \subseteq \mathcal{T}_0\)) which we write \(t \perp S\), when for all \(e \in S\), \(t\) is orthogonal to \(e\). Due to the call-by-name\(^1\) (which is induced here by the choice of connectives), a formula \(A\) is primitively interpreted by its ground falsity value, which we write \(\| A \|_V\) and which is a set in \(\Pi(V_0)\). Its truth value \(\| A \|\) is then defined by orthogonality to \(\| A \|_V\) (and is a set in \(\Pi(T_0)\)), while its falsity value \(\| \neg A \|_V\) is again obtained by orthogonality to \(\| A \|\). To ease the definitions we assume that for each subset \(S\) of \(\Pi(V_0)\), there is a constant symbol \(\mathcal{S}\) in the syntax. Given a fixed pole \(\perp\), the interpretation is given by:

\[
\begin{align*}
\| \mathcal{S} \| & \triangleq S \\
\| \forall X.A \| & \triangleq \bigcup_{S \in \Pi(V_0)} \| A \{ X := \mathcal{S} \} \|_V \\
\| A \Jordan B \| & \triangleq \{ (V_1, V_2) : \forall V \in \| A \|_V \land V \in \| B \|_V \} \\
\| \neg A \| & \triangleq \{ \langle \pi.t : e/A \rangle \mid \forall \pi.t \in \| A \|, t \perp e \}
\end{align*}
\]

The typing rules are adequate with respect to the realizability interpretation:

**Proposition 9** (Adequacy). We have:

1. If \(V \in \| A \|_V\), then \(V \in \| A \|_V\).
2. If \(t \perp A\), then \(t \in \| A \|\).
3. If \(e \perp A\), then \(e \in \| A \|\).
4. If \(c \perp t\), then \(c \perp t\).

Proof. See Appendix B-A1.\(^3\)

---

\(^1\)The reader might recognize the rule \(\r_\zeta\) of Wadler’s sequent calculus \([36]\).

\(^2\)See \([29]\) Chapter 3] for a more detailed explanation on this point.
C. Disjunctive structures

We are now define the notion of disjunctive structure. We choose negative connectives and in particular a universal quantifier, hence we should define commutations with respect to arbitrary meets. The following properties of the realizability interpretation for $L^\mathbb{Q}$ provides us with a safeguard for the definition to come:

**Proposition 10.** In any $L^\mathbb{Q}$ realizability model, if $X \notin FV(B)$ the following equalities hold:

1. $\|\forall X. (A \land B)\|_V = \|(\forall X. A) \land B\|_V$.
2. $\|\forall X. (B \land A)\|_V = \|(B \land (\forall X. A))\|_V$.
3. $\|\neg (\forall X. A)\|_V = \bigcap_{S \in \mathcal{P}(V_0)} \neg A\{X := S\}\|_V$.

**Proof.** See Section B-B.

Algebraically, the previous proposition advocates for the following definition (recall that the order is defined as the reversed inclusion of primitive falsity values (whence $\cap$ is $\land$) and that the $\forall$ quantifier is interpreted by $\land$):

**Definition 11** (Disjunctive structure). A disjunctive structure is a complete lattice $(A, \preceq)$ equipped with a binary operation $(a, b) \mapsto a \land b$, together with a unary operation $a \mapsto \neg a$, such that for all $a, a', b, b' \in A$ and for any $B \subseteq A$:

1. if $a \preceq a'$ then $\neg a' \preceq \neg a$.
2. if $a \preceq a'$ and $b \preceq b'$ then $a \land b \preceq a' \land b'$.
3. $\bigwedge_{b \in B} (a \land b) = a \land \bigwedge_{b \in B} b$.
4. $\bigwedge_{a \in A} a = \bigvee_{a \preceq a} a$.

Observe that the commutation laws imply the value of the internal laws when applied to the maximal element $\top$: $\top \land a = \top$.

**Example 12** (Dummy disjunctive structure). Given a complete lattice $(\mathcal{L}, \preceq)$, the following definitions give rise to a dummy structure that fulfills the axioms of Definition 11:

$a \land b \triangleq \top$.

**Example 13** (Complete Boolean algebras). Let $B$ be a complete Boolean algebra. It encompasses a disjunctive structure defined by:

$A \triangleq B$.

The different axioms are easy to be satisfied for complete Boolean algebras.

**Example 14** ($L^\mathbb{Q}$ realizability models). If we abstract the structure of the realizability interpretation of $L^\mathbb{Q}$, it is a structure of the form $(\mathcal{T}_0, \mathcal{E}_0, V_0, \cdot, \cdot, [,\cdot], \perp, \equiv)$, where $\cdot, \cdot$ is a binary map from $\mathcal{E}_0^2$ to $\mathcal{E}_0$ (whose restriction to $V_0$ has values in $V_0$), $[,]$ is an operation from $\mathcal{T}_0$ to $V_0$, and $\perp \subseteq \mathcal{T}_0 \times \mathcal{E}_0$ is a relation. From this sextuple, we can define:

- $A \triangleq \mathcal{P}(V_0)$.
- $a \land b \triangleq [(V_1, V_2) : V_1 \in a \land V_2 \in b]$.
- $\neg a \triangleq [a^\perp] = \{[t] : t \in a^\perp\}$.

The resulting quadruple $(A, \preceq, \mathcal{V}, \neg)$ is a disjunctive structure (see Proposition 62 for a proof).

D. Embedding of $L^\mathbb{Q}$

Following the interpretation of the $\lambda$-terms in implicational structures, we can embed $L^\mathbb{Q}$ terms within disjunctive structures. We do not have the necessary step here to fully introduce here (see Appendix B-C), but it is worth mentioning that the orthogonality relation $\perp \perp e$ is interpreted via the ordering $e^A \preceq e^C$.

**Theorem 15** (Adequacy). The typing rules of $L^\mathbb{Q}$ (Figure 1) are adequate with respect to the interpretation of terms, contexts, commands and formulas:

1. If $\cdot \vdash t : A \mid \cdot$, then $t^A \preceq A^\mathbb{Q}$.
2. If $\cdot \mid e : A \vdash \cdot$, then $e^A \preceq A^\mathbb{Q}$.
3. If $e : (\cdot \vdash \cdot)$, then $e^A \in \perp$.

**Proof.** See Appendix B-D.

E. The induced implicational structure

As expected, any disjunctive structures directly induces an implicational structure through the definition $a \rightarrow b \triangleq \neg a \land b$:

**Proposition 16** If $(A, \preceq, \mathcal{V}, \neg)$ is a disjunctive structure, then $(A, \preceq, \rightarrow)$ is an implicational structure.

Therefore, we can again define for all $a, b$ of $A$ the application $ab$ as well as the abstraction $\lambda f$ for any function $f$ from $A$ to $A$; and we get for free the properties of these encodings in implicational structures. Up to this point, we have two ways of interpreting a $\lambda$-term into a disjunctive structures, either through the implicational structure which is induced by the disjunctive one, or by embedding into the $L^\mathbb{Q}$-calculus which is then interpreted within the disjunctive structure. As a sanity check, we verify that both coincide.

**Proposition 17** ($\lambda$-calculus). Let $A^\mathbb{Q} = (A, \preceq, \mathcal{V}, \neg)$ be a disjunctive structure, and $A^\rightarrow = (A, \preceq, \rightarrow)$ the implicational structure it canonically defines, we write $i$ for the corresponding inclusion. Let $t$ be a closed $\lambda$-term (with parameter in $A$), and $[t]$ his embedding in $L^\mathbb{Q}$. Then we have $i(t^A) = [i]A^\mathbb{Q}$.

**Proof.** See Appendix B-E.

F. Disjunctive algebras

**Separation in disjunctive structures:** We shall now introduce the notion of disjunctive separator. To this purpose, we adapt the definition of implicational separators, using Bourbaki’s axioms for the disjunction and the negation instead of Hilbert’s combinators $S$ and $K$. We thus consider the following combinator:

$\sum_1 \triangleq \lambda_{a \in A} [(a \land a) \rightarrow a]$

$\sum_2 \triangleq \lambda_{a, b \in A} [a \rightarrow (a \land b)]$

This is in fact, in Bourbaki’s presentation [22] p.25] the fifth axiom is deducible from the first four, using the so-called “deduction lemma”. This lemma roughly states that if $B$ is a theorem assuming $A$ is an axion, then $A \rightarrow B$ is a theorem. The proof is done by induction on the proof of $B$ in the extended theory, and this induction could be inlined to get a proof of $S5$ from the proof that $(A \land B) \lor C$ is deducible assuming $(A \land (B \lor C))$ as an axiom. Yet, the resulting naive proof for this requires more than 300 steps (see above for such a proof produced by Miquel), thus to ease this presentation we preferred to keep it as an axiom.
that is the unique rule of deduction we have, is actually

Remark 20

The reader may notice that in this section, we do
distinction between classical and intuitionistic separators.
Indeed, \(L^2\) and the corresponding fragment of the sequential
calculus are intrinsically classical. As we shall see thereafter, so
are the disjunctive algebras: the negation is always involutive modulo the equivalence \(\cong\) (Proposition 24).

Remark 20 (Generalized modus ponens). The modus ponens,
that is the unique rule of deduction we have, is actually
compatible with meets. Consider a set \(I\) and two family
\((a_i)_{i \in I}, (b_i)_{i \in I} \in A^I\), we have:

\[
\begin{align*}
al \vdash b & \quad \vdash_1 b \\
\vdash_1 b & \quad \vdash b
\end{align*}
\]

where we write \(a \vdash b\) for \((\bigwedge_{i \in I} a_i \rightarrow b_i) \in \mathcal{S}\) and \(\vdash_1 a\)
for \((\bigwedge_{i \in I} a_i) \in \mathcal{S}\). The proof is straightforward using that the separator is closed upwards by application, and that:

\[
(\bigwedge_{i \in I} a_i \rightarrow b_i)(\bigwedge_{i \in I} a_i) \leq (\bigwedge_{i \in I} b_i)
\]

which is clearly true. As our axioms are themselves expressed
as meets, the results that we will obtain internally (that is by
deduction from the separator’s axioms) can all be generalized
to meets.

Example 21 (Complete Boolean algebras). Once again, if \(B\) is
a complete Boolean algebra, \(B\) induces a disjunctive structure
in which it is easy to verify that the combinators \(s_1^2, s_2^2, s_3^2, s_4^2\)
and \(s_5^2\) are equal to the maximal element \(\top\). Therefore, the
singleton \(\{\top\}\) is a valid separator for the induced disjunctive
structure. In fact, the filters for \(\mathcal{B}\) are exactly its separators.

Example 22 (L^2 realizability model). Recall from Example 14
that any model of classical realizability based on the \(L^2\)-
calculus induces a disjunctive structure. As in the implicatible case, the set of formulas realized by a closed term\(^{15}\)

\[
\mathcal{S}_{\uplus} \triangleq \{a \in \mathcal{P}(V^+_0) : a \uplus \cap \mathcal{T}_0 \neq \emptyset\}
\]

defines a valid separator. The conditions (1) and (3) are clearly
verified (for the same reasons than in the implicatible case).
As for the formulas corresponding to the combinators, it is a
straightforward programming exercise to find \(L^2\)-terms of the
corresponding types, which is enough to conclude using the
adequacy of the realizability model (see Example 81 for more
details).

1) Internal logic: As in the case of implicatible algebras,
we say that \(a\) entails \(b\) and write \(a \vdash_b b\) if \(a \rightarrow b \in \mathcal{S}\).

Proposition 23 (Preorder). For any \(a, b, c \in A\), we have:

\[
\begin{align*}
& a \vdash_S a \quad \text{(Reflexivity)} \\
& a \vdash_S b \quad \text{if } a \vdash_S b \quad \text{and } b \vdash_S c \text{ then } a \vdash_S c \quad \text{(Transitivity)}
\end{align*}
\]

Proof. We first that (2) holds by applying twice the closure
by modus ponens, then we use it with the relation \(a \vdash_S a \forall a\)
and \(a \forall a \vdash_S\) that can be deduced from the combinators \(s_1^2, s_2^2\)
to get 1. \(\Box\)

Negation: We can relate the primitive negation to the
one induced by the underlying implicatible structure, and show
that the principle of double negation elimination is valid with
respect to any separator.

Proposition 24 (Implicative negation). For all \(a \in A\), the
following holds:

\[
\begin{align*}
\text{1. } & a \vdash_S \bot \\
\text{2. } & a \vdash_S \vdash_2 \neg a \\
\text{3. } & a \vdash_S \neg \neg a \\
\text{4. } & a \vdash_S \neg a \\
\end{align*}
\]

Proof. See Appendix B-G \(\Box\)

Sum type: As in implicative structures, we can define the
sum type by:

\[
a + b \triangleq \bigcup_{c \in A} ((a \rightarrow c) \rightarrow (b \rightarrow c) \rightarrow c) \quad (\forall a, b \in A)
\]

We can prove that the disjunction and this sum type are
equivalent from the point of view of the separator:

Proposition 25 (Implicative sum type). For all \(a, b \in A\), the
following holds:

\[
\begin{align*}
\text{1. } & a \forall b \vdash_S a + b \\
\text{2. } & a + b \vdash_S a \forall b
\end{align*}
\]

Proof. See Appendix B-G \(\Box\)

G. Induced implicative algebras

In order to that any disjunctive algebra is a particular case
of implicative algebra, we first verify that Hilbert’s combinators
belong to any disjunctive separator:

Proposition 26 (Combinators). We have:

\[
\begin{align*}
\text{1. } & K^A \in \mathcal{S} \\
\text{2. } & S^A \in \mathcal{S} \\
\text{3. } & CC^A \in \mathcal{S}
\end{align*}
\]

Proof. See Appendix B-G \(\Box\)

As a consequence, we get the expected theorem:

Theorem 27 Any disjunctive algebra is a classical implicative
algebra.
Proof. The conditions of upward closure and closure under modus ponens coincide for implicatve and disjunctive separators, and the previous propositions show that $K, S$ and $CC$ belong to the separator of any disjunctive algebra. □

H. Tripods

Since any disjunctive algebra is a particular case of implicative algebra, it is clear that the construction leading to the implicative tripods can be rephrased in this framework. In particular, the same criterion allows us to determine whether the implicative tripods is isomorphic to a forcing tripods. Notably, a disjunctive algebra with extra-commendations for the disjunction $\lor$ and the negation $\neg$ with arbitrary joins will induce an implicative algebra where the arrow commutes with arbitrary joins. Therefore, the induced tripods would collapse to a forcing situation (see Footnote [8]).

IV. A POSITIVE DECOMPOSITION: CONJUNCTIVE ALGEBRAS

A. Call-by-value realizability models

While there exists now several models build of classical theories constructed via Krivine realizability [21], [23], [24], [27], they all have in common that they rely on a presentation of logic based on negative connectives/quantifiers. If this might not seem shocking from a mathematical perspective, it has the computational counterpart that these models all build on a call-by-name calculus, namely the $\lambda$-calculus [17]. In light of the logical consequences that computational choices have on the induced theory, it is natural to wonder whether the choice of a call-by-name evaluation strategy is anecdotical or fundamental.

As a first step in this direction, we analyze here the algebraic structure of realizability models based on the $L^\forall$ calculus, the positive fragment of Munch-Maccagnoni’s system $L$ based on a negation and a tensor. Through the lenses of the duality of computation between terms and evaluation contexts [21], [32], this fragment is dual to the $L^{\forall}$ calculus and it naturally allows to embed the $\lambda$-terms evaluated in a call-by-value fashion. We shall now reproduce the approach we had for $L^{\forall}$: we first introduce the $L^\forall$ calculus and the realizability models it induces in order to later define conjunctive structures. We then show how these structures can be equipped with a separator and how the resulting conjunctive algebras lead to the construction of a conjunctive tripods. In the next section, we will show how conjunctive and disjunctive algebras are related by an algebraic notion of duality.

B. The $L^\forall$ calculus

The $L^\forall$ calculus corresponds exactly to the restriction of $L$ to the positive fragment induced by the connectives $\otimes, \neg$ and the existential quantifier $\exists$. Its syntax is given by:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Contexts</strong></td>
<td>$e ::= \alpha$ $</td>
</tr>
<tr>
<td><strong>Terms</strong></td>
<td>$t ::= x$ $</td>
</tr>
<tr>
<td><strong>Commands</strong></td>
<td>$c ::= (t \mid e)$</td>
</tr>
</tbody>
</table>

In this framework, values are defined by:

**Values**

$V ::= x \mid (V, V) \mid [e]$

We denote by $V_0, \tau_0, \xi_0, \xi_0$ for the sets of closed values, terms, contexts and commands. The syntax is really close to the one of $L^{\forall}$: The reduction rules correspond again to the intuition one could have from the syntax of the calculus: all destructors and binders reduce in front of the corresponding values, while pairs of terms are expanded if needed. The reduction rules are given by:

$\langle \mu x.c \parallel e \rangle \rightarrow c[e/\alpha]$  
$\langle [e] \parallel \mu[\alpha]. c \rangle \rightarrow c[e/\alpha]$  
$\langle V \parallel \mu x.c \rangle \rightarrow c[V/x]$  
$\langle (V, V') \parallel \mu(x,x').c \rangle \rightarrow c[V/x, V'/x']$  
$\langle (t,u) \parallel e \rangle \rightarrow \langle t \mid \mu x. \langle u \parallel \mu y.((x,y) \parallel e) \rangle \rangle$

where $(t,u) \notin V$ in the last $\beta$-reduction rule.

Finally, we shall present the type system of $L^\forall$. We are intersted in the second-order formulas defined from the positive connectives:

**Formulas**

$A, B ::= X \mid A \otimes B \mid \neg A \mid \exists X.A$

The type system, which uses the same three kinds of sequents for terms, contexts, commands than in $L^{\forall}$ type system, is given in Figure [2].

C. Embedding of the $\lambda$-calculus

Guided by the positive decomposition of the arrow $A \rightarrow B \triangleq \neg(A \otimes \neg B)$, we can follow Munch-Maccagnoni’s paper [32] Appendix E], to embed the $\lambda$-calculus into $L^\forall$.

With such a definition, a stack $u \cdot e$ in $A \rightarrow B$ (that is with $u$ a term of type $A$ and $e$ a context of type $B$) is naturally embedded as a term $(u, [e])$, which is turn into the context $\mu[\alpha].\langle (u, [e]) \parallel \alpha \rangle$ which indeed inhabits the “arrow” type $\neg(A \otimes \neg B)$. The rest of the definitions are then direct:

$\mu(x,[\alpha]).c \triangleq \mu(x,x').(x' \parallel \mu[\alpha].c) \mid t \cdot e \triangleq \mu[\alpha].\langle (t, [e]) \parallel \alpha \rangle$  
$\lambda x.t \triangleq \langle \mu(x,[\alpha]).(t \parallel \alpha) \rangle$  
$tu \triangleq \mu[\alpha].\langle t \parallel u \parallel \alpha \rangle$
One again, these definitions are sound with respect to the usual typing rules of the $\lambda\mu\iota\sigma$-calculus (see Proposition 88). In addition, they entail the expected rules of call-by-value $\beta$-reduction:

\[
\langle t \mid u \mid c \rangle \rightarrow_\beta \langle t \mid u \cdot c \rangle
\]

\[
\langle \lambda x.t \mid u \cdot c \rangle \rightarrow_\beta \langle u \mid c[x \leftarrow \lambda x.t] \rangle
\]

\[
\langle V \mid \mu x.c \rangle \rightarrow_\beta c[V / x]
\]

---

### D. A realizability model based on the $L^5$-calculus

We briefly recall the definitions necessary to the realizability interpretation à la Krivine of $L^5$. Most of the properties being the same as for $L^4$, we spare the reader from a useless copy-paste and go straight to the point. As usual in call-by-value realizability models [28], formulas as primitives interpreted as sets of truth values, which we call ground truth values, while falsity values and truth values are then defined by orthogonality. The interpretation is given by:

\[
\begin{align*}
\hat{S}|_V & \triangleq \mathcal{S} \\
\langle A \otimes B \rangle|_V & \triangleq \{ (t, u) \mid t \in \langle A \rangle|_V \land u \in \langle B \rangle|_V \} \\
\langle \lnot A \rangle|_V & \triangleq \{ e \mid e \in \langle A \rangle \} \\
\langle \exists X.A \rangle|_V & \triangleq \{ e \mid \forall \tilde{V} \in \langle A \rangle|_V \land e \} \\
\langle A \rangle|_V & \triangleq \{ t \mid \forall e \in \langle A \rangle|_V \land e \}
\end{align*}
\]

**Lemma 28 (Adequacy).** We have Let $\Gamma, \Delta$ be typing contexts, $\rho$ be a valuation and $\sigma$ be a substitution which verifies that $\sigma \vdash \Gamma[\rho]$ and $\sigma \vdash \Delta[\rho]$. We have:

1. If $\vdash V : A$, then $V \in \langle A \rangle|_V$.
2. If $\vdash e : A \lnot$, then $e \in \langle A \rangle$.
3. If $\vdash t : A$, then $t \in \langle A \rangle$.
4. If $c : (\lnot \lnot)$, then $c \in \perp$.

**Proof.** By induction over typing derivations, see [32]. □

---

### E. Conjunctive structures

We shall now introduce the notion of conjunctive structure. Following the methodology from the previous section, we begin by observing the existing commutations in the realizability models induced by $L^5$. Since we are in a structure centered on positive connectives, we should pay attention to the commutations with joins:

**Proposition 29.** In any $L^5$ realizability model, if $X \notin FV(B)$ the following equalities hold:

1. $\| \exists X.(A \otimes B) \|_V = (\exists X.A) \otimes B|_V$.
2. $\| \exists X.(B \otimes A) \|_V = B \otimes (\exists X.A)|_V$.
3. $\| \lnot (\exists X.A) \|_V = \bigcap_{e \in \mathcal{P}(\mathcal{V}_0)} \lnot A \{ X := \hat{S} \}|_V$

**Proof.** See Appendix C-C. □

Since we are now interested in primitive truth values, which are logically ordered by inclusion (in particular, the existential quantifier is interpreted by unions, thus joins), the previous proposition advocates for the following definition:

**Definition 30 (Conjunctive structure).** A conjunctive structure is a complete join-semilattice $(A, \ll)$ equipped with a binary operation $(a, b) \mapsto a \otimes b$, and a unary operation $a \mapsto \lnot a$, such that for all $a, a', b, b' \in A$ and for all subset $B \subseteq A$ we have:

1. If $a \ll a'$ then $\lnot a' \ll \lnot a$
2. If $a \ll a'$ and $b \ll b'$ then $a \otimes b \ll a' \otimes b'$
3. $\bigwedge (a \otimes b) = a \otimes (\bigwedge b)$ and $\bigwedge (b \otimes a) = (\bigwedge b) \otimes a$
4. $\lnot \bigwedge_{a \in A} a = \bigwedge_{a \in A} \lnot a$

As in the cases of implicutive and disjunctive structures, the commutations imply that:

1. $\perp \otimes \perp = \perp$
2. $a \otimes \perp = \perp$
3. $\perp = \top$

**Example 31 (Dummy conjunctive structure).** Given a complete lattice $L$, the following definitions give rise to a dummy conjunctive structure:

\[
a \otimes b \triangleq \top \quad \lnot a \triangleq \top \quad \forall a, b \in A
\]

**Example 32 (Complete Boolean algebras).** Let $B$ be a complete Boolean algebra. It embodies a conjunctive structure, that is defined by:

\[
A \triangleq B \quad a \ll b \triangleq a \ll b \quad a \otimes b \triangleq a \land b \quad \lnot a \triangleq \lnot a
\]

**Example 33 (L$^5$ realizability models).** As for the disjunctive case, we can abstract the structure of the realizability interpretation of $L^5$ into a structure of the form $(\mathcal{T}_0, \mathcal{E}_0, \mathcal{V}_0, (\cdot, \cdot), [\cdot], \perp, \top)$, where $(\cdot, \cdot)$ is a map from $\mathcal{T}_0 \times \mathcal{V}_0$ (whose restriction to $\mathcal{V}_0$ has values in $\mathcal{V}_0$), $[\cdot]$ is an operation from $\mathcal{E}_0$ to $\mathcal{V}_0$, and $\perp \subseteq \mathcal{T}_0 \times \mathcal{E}_0$ is a relation. From this sextuple we can define:

\[
\begin{align*}
\mathcal{A} \triangleq \mathcal{P}(\mathcal{V}_0) & \quad \mathcal{A} \otimes \mathcal{B} \triangleq \{(V_1, V_2) \mid V_1 \in a \land V_2 \in b\} \\
\mathcal{A} \otimes b \triangleq a \triangledown b & \quad \lnot a \triangleq \{e \mid e \in a\}
\end{align*}
\]

The quadruple $(\mathcal{A}, \otimes, \triangledown, \lnot)$ is a then conjunctive structure (see Section C for a proof).

It is worth noting that even though we can define an arrow by $a \triangledown b \triangleq \lnot (a \otimes \lnot b)$, it does not induce an implicative structure: indeed, the distributivity law is not true in general\(^\text{[17]}\)\).

In turns, we have another distributivity law which is usually wrong in implicative structure:

\[
\bigwedge_{a \in A} (a \triangledown b) = \bigwedge_{a \in A} (a \triangledown b) \neq \bigwedge_{b \in B} (a \triangledown b)
\]

Actually, implicative structures where both are true corresponds precisely to a degenerated forcing situation.

### F. Interpreting $L^5$ terms

Once again, we can embed $L^5$ commands, contexts and terms into any conjunctive structure. The embedding, given in Appendix C-C, is very similar to the one for $L^4$ in disjunctive structure, and is again adequate:

**Theorem 34 (Adequacy).** The typing rules of $L^5$ (Figure 2) are adequate with respect to the interpretation of terms, contexts, commands and formulas:

\(^{17}\)For instance, it is false in $L^5$ realizability models.
1) If \( \vdash t : A \mid : \), then \( t^A \equiv A^A \); 
2) If \( \vdash e : A \vdash : \), then \( e^A \supset A^A \); 
3) If \( \vdash c : (\vdash -) \), then \( c^A \in \vdash \).

**Proof.** See Appendix C-C

G. Conjunctive algebras

The definition of conjunctive separators turns out to be more subtle than in the disjunctive case. Among others things, conjunctive structures mainly axiomatize joins, while the combinators or usual mathematical axioms that we could wish to have in a separator are more naturally expressed via universal quantifications, hence meets. Yet, an analysis of the sequent calculus underlying \( L^\ominus \) type system shows that we could consider a tensorial calculus where deduction systematically involves a conclusion of the shape \( \neg A \). This justifies to consider the following combinators

\[
\begin{align*}
s_1^\ominus & \triangleq \lambda_{a \in A} \neg \neg (a \otimes a) \otimes a \\
s_2^\ominus & \triangleq \lambda_{a,b \in A} \neg \neg (a \otimes (a \otimes b)) \\
s_3^\ominus & \triangleq \lambda_{a,b \in A} \neg \neg ((a \otimes b) \otimes b) \\
s_4^\ominus & \triangleq \lambda_{a,b,c \in A} \neg \neg ((a \otimes (b \otimes c)) \otimes ((a \otimes b) \otimes c)) \\
s_5^\ominus & \triangleq \lambda_{a,b \in A} \neg \neg ((a \otimes b) \otimes (a \otimes b)) \\
\end{align*}
\]

and to define conjunctive separators as follows:

**Definition 35 (Separator).** We call separator for the conjunctive structure \( A \) any subset \( S \subseteq A \) that fulfills the following conditions for all \( a,b \in A \):

1. If \( a \in S \) and \( a \otimes b \) then \( b \in S \). (upward closure)
2. \( s_1^\ominus, s_2^\ominus, s_3^\ominus, s_4^\ominus \) and \( s_5^\ominus \) are in \( S \). (combinators)
3. If \( \neg (a \otimes b) \in S \) and \( a \in S \) then \( \neg b \in S \). (deduction)
4. If \( a \in S \) and \( b \in S \) then \( a \otimes b \in S \). (pairs)

A separator \( S \) is said to be classical if besides \( \neg a \in S \) implies \( a \in S \).

**Remark 36 (Modus Ponens).** If the separator is classical, it is easy to see that the modus ponens is valid: if \( a \not\rightarrow b \in S \) and \( a \in S \), then \( \neg b \in S \) by (3) and thus \( b \in S \).

**Example 37 (Complete Boolean algebras).** Once again, if \( B \) is a complete Boolean algebra, \( B \) induces a conjunctive structure in which it is easy to verify that the combinators \( s_1^\ominus, s_2^\ominus, s_3^\ominus, s_4^\ominus \) and \( s_5^\ominus \) are equal to the maximal element \( \top \). Therefore, the singleton \( \{ \top \} \) is a valid separator.

**Example 38 (Realizability model).** As expected, the set of realized formulas by a proof-like term:

\[
S_{\dag} \triangleq \{ a \in P(V_0^+) : a^{\dag} \cap T_0 \neq 0 \}
\]
defines a valid separator. The conditions (1) and (3) are clearly verified (for the same reasons as in the implicational case). As for the formulas corresponding to the combinators, it is a straightforward programming exercise to find \( L^\ominus \)-terms of the corresponding types, which is enough to conclude using that the adequacy of the realizability model (see Example 81 for more details).

**Example 39 (Kleene realizability).** We do not want to enter into too much details here, but it is worth mentioning that realizability interpretations à la Kleene of intuitionistic calculi equipped with primitive pairs (e.g. (partial) combinatorial algebras, the \( \lambda \)-calculus) induce conjunctive algebras. Insofar as Kleene realizability takes position against classical reasoning (\( \forall X . X \lor \neg X \) is not realized hence its negation is), these algebras have the interesting properties of not being classical (and are even incompatible with a classical completion).

**Remark 40 (Generalized axioms).** Once again, the axioms (3) and (4) generalize to meet of families \( (a_i)_{i \in I}, (b_i)_{i \in I} \):

\[
\begin{align*}
\vdash_I \neg (a \otimes b) & \quad \vdash_I b \\
\vdash_I a & \quad \vdash_I b
\end{align*}
\]

where we write \( \vdash_I \) for \( (\lambda_{i \in I} a_i) \in S \) and where the negation and conjunction of families are taken pointwise.

H. Internal logic

We consider as usual the entailment relation defined by \( a \vdash_S b \triangleq (\exists b \in S) \). Observe that if the separator is not classical, we do not have that \( a \vdash_S b \) and \( a \in S \) entails \( b \in S \).

**Proposition 41 (Preorder).** For any \( a, b, c \in A \), the following holds:

1. \( a \vdash_S a \) (Reflexivity)
2. If \( a \vdash_S b \) and \( b \vdash_S c \) then \( a \vdash_S c \) (Transitivity)

Negation: Here again, we can relate the negation \( \neg a \) to the one induced by the arrow \( a \not\rightarrow \downarrow \).

**Proposition 42 (Implicative negation).** For all \( a \in A \), the following holds:

1. \( \neg a \vdash_S a \not\rightarrow \downarrow \)
2. \( a \not\rightarrow \downarrow \vdash_S \neg a \)
3. \( a \vdash_S \neg \neg a \)
4. \( \neg \neg a \vdash_S a \)

\( \lambda \)-calculus: We first show that Hilbert’s combinators \( K \) and \( S \) belong to any conjunctive separator.

**Proposition 43 (K and S).** We have:

1. \( \lambda_{a,b \in A} (a \not\rightarrow b \not\rightarrow a) \in S \)
2. \( \lambda_{a,b,c \in A} (a \not\rightarrow b \not\rightarrow c) \not\rightarrow (a \not\rightarrow b) \not\rightarrow a \not\rightarrow c \in S \)

As in implicative structures, we can define the abstraction and application of the \( \lambda \)-calculus:

\[
\lambda f \triangleq \bigcup_{a \in A} (a \not\rightarrow f(a)) \quad ab \triangleq \bigcup_{c \in \neg c} \{ \neg c : a \not\rightarrow b \not\rightarrow c \}
\]

Observe that here we need to add a double negation, since intuitively \( ab \) is a computation of type \( \neg \neg c \) rather than a value of type \( c \). In other words, values are not stable by applications, and extracting a value from a computation requires a form of classical control. Nevertheless, for any separator we have:

\( 20 \) Actually we can consider a different relation \( a \vdash_S b \triangleq (a \otimes b) \) for which \( a \vdash_S b \) and \( a \in S \) entails \( b \). This one turns out to be useful to ease proofs, but from a logical perspective, the significant entailment is the one given by \( a \vdash_S b \).
Proposition 44. If \( a \in S \) and \( b \in S \) then \( ab \in S \).

Similarly, the beta reduction rule now involves a double-negation on the reduced term:

Proposition 45. \( (\lambda f) a \leq \neg f(a) \)

Proof. We first show that \( t \leq a \Rightarrow b \) implies \( ta \leq \neg b \), and we use that \( \lambda f \leq a \Rightarrow f(a) \) to conclude.

In the case where the separator is classical, we can prove that it contains the interpretation of all closed \( \lambda \)-terms:

Theorem 46. \( \lambda \)-calculus. If \( S \) is classical and \( t \) is a closed \( \lambda \)-term, then \( tA \in S \).

Proof. By combinatorial completeness, we have the existence of a combinatorial term \( t_0 \) (i.e., a composition of \( K \) and \( S \)) such that \( t_0 \rightarrow^* t \). Since \( K \in S \), \( S \in S \) and \( S \) is closed under application, \( t_0 \in S \). For each step \( t_n \rightarrow t_{n+1} \), we have \( t_n \leq t_{n+1} \), and thus \( t_n \in S \) implies \( t_{n+1} \in S \). We conclude by induction on the length of the reduction \( t_0 \rightarrow^* t \).

Induced Heyting algebra: Once more, the entailment relation induces a structure of (pre-)Heyting algebra, whose conjunction and disjunction are naturally given by \( a \times b \equiv a\otimes b \) and \( a + b \equiv \neg(\neg a \otimes \neg b) \).

Proposition 47. (Heyting Algebra). For any \( a, b, c \in A \), for any \( a, b, c \in A \), we have:

1. \( a \times b \vdash_S a \)
2. \( a \times b \vdash_S b \)
3. \( a \vdash_S a + b \)
4. \( b \vdash_S a + b \)
5. \( a \vdash_S b \Rightarrow c \) if and only if \( a \times b \vdash_S c \)

We can thus quotient the algebra by the equivalence relation \( \equiv_S \) and extend the previous operation to equivalence classes in order to obtain a Heyting algebra \( A/\equiv_S \).

I. Conjunctive tripos

We are now ready to reproduce the construction of the implicational tripos in our setting.

Theorem 48. (Conjunctive tripos). Let \( (A, \leq, \to, S) \) be a classical conjunctive algebra. The following functor (where \( f : I \to J \)):

\[ T : I \to A^I/S[I] \quad T(f) : \begin{cases} A^I/S[I] & \to A^J/S[J] \\ \{(a_i)_{i \in I} \} & \mapsto \{(a_{f(i)})_{j \in J} \} \end{cases} \]

defines a tripos.

Proof. The proof mimicks the proof in the case of implicational algebras, see Appendix C-E.

\[ \lambda \]

21 Actually, since we always have that if \( \neg\neg\neg\neg a \in S \) the \( \neg\neg \neg \neg a \in S \), the same proof shows that in the intuitionistic case we have at \( \neg\neg\neg\neg tA \in S \).

22 For technical reasons, we only give the proof in case where the separator is classical (recall that it allows to directly use \( \lambda \)-terms), but as explained, by adding double negation everywhere the same reasoning should work for the general case as well. Yet, this is enough to express our main result in the next section which only deals with the classical case.

V. THE DUALITY OF COMPUTATION, ALGEBRARICALLY

In [4], Curien and Herbelin introduce the \( \lambda\mu \)\( \ddot{\mu} \) in order to emphasize the so-called duality of computation between terms and evaluation contexts. They define a simple translation inverting the role of terms and stacks within the calculus, which has the notable consequence of translating a call-by-value calculus into a call-by-name calculus and vice-versa. The very same translation can be expressed within L, in particular it corresponds to the trivial translation from mapping every constructor on terms (resp. destructors) in \( L^\ddot{\mu} \) to the corresponding constructor on stacks (resp. destructors) in \( L^\ddot{\mu} \). We shall now see how this fundamental duality of computation can be retrieved algebraically between disjunctive and conjunctive algebras.

We first show that we can simply pass from one structure to another by reversing the order relation. We know that reversing the order in a complete lattice yields a complete lattice in which meets and joins are exchanged. Therefore, it only remains to verify that the axioms of disjunctive and conjunctive structures can be deduced through this duality one from each other, which is the case.

Proposition 49. Let \( (\ll, \gg, -) \) be a disjunctive structure. Let us define:

\[ A^\ll A^\gg \quad \ll A^\gg A^\ll \quad a \ll b \equiv a \gg b \]

then \( (A^\ll, \ll, a, \gg, ~) \) is a conjunctive structure.

Proposition 50. Let \( (\ll, \gg, -) \) be a conjunctive structure. Let us define:

\[ A^\gg A^\ll \quad \gg A^\ll A^\gg \quad a \gg b \equiv a \ll b \]

then \( (A^\gg, \gg, a, \ll, ~) \) is a conjunctive structure.

Intuitively, by considering stacks a realizers, we somehow reverse the algebraic structure, and we consider as valid formulas the one that whose orthogonal were valid. In terms of separator, it means that when reversing a structure we should consider the separator defined as the preimage through the negation of the original separator.

Theorem 51. Let \( (A^\ll, S^\ll) \) be a conjunctive algebra, the set:

\[ S^\ll \equiv \neg^{-1}(S^\gg) = \{a \in A : \neg a \in S^\gg\} \]

is a valid separator for the dual disjunctive structure \( A^\gg \).

Theorem 52. Let \( (A^\gg, S^\gg) \) be a disjunctive algebra. The set:

\[ S^\gg \equiv \neg^{-1}(S^\ll) = \{a \in A : \neg a \in S^\ll\} \]

is a classical separator for the dual conjunctive structure \( A^\ll \).

Proof. See Appendix D.

\[ \lambda \]

It is worth noting that reversing in both cases, the dual separator is classical. This is to connect with the fact that classical reasoning principles are true on negated formulas. Moreover, starting from a non-classical conjunctive algebra,
one can reverse it twice to get a classical algebra. This corresponds to a classical completion of the original separator \( S \): by definition, \( \neg\neg(S) = \{ a : \neg\neg a \in S \} \), and it is easy to see that \( a \in S \) implies \( \neg\neg a \in S \), hence \( S \subseteq \neg\neg(S) \).

Actually, the duality between disjunctive and (classical) conjunctive algebras is even stronger, in the sense that through the translation, the induced triposes are isomorphic. Recall that an isomorphism \( \varphi \) between two (Set-based) triposes \( T, T' \) is defined as a natural isomorphism \( \varphi \) in \( \text{Set} \)-based triposes \( T \Rightarrow T' \) in the category \( \text{HA} \), that is as a family of isomorphisms \( \varphi_I \) \( : T(I) \cong T'(I) \) (indexed by all \( I \in \text{Set} \)) that is natural in \( I \).

**Theorem 53** (Main result). Let \( (A, S) \) be a disjunctive algebra and \( (\bar{A}, \bar{S}) \) its dual conjunctive algebra. The family of maps:

\[
\varphi_I : \begin{cases} 
A/S[I] & \leftrightarrow A/S[I] \\
[a_i] & \leftrightarrow [\neg a_i] 
\end{cases}
\]

defines a tripos isomorphism.

**Proof.** The naturality of \( \varphi \) is clear by construction. Recall that to prove that \( \varphi \) is an isomorphism of Heyting algebras, it is enough to show that it is an isomorphism of posets. The whole proof rely once again on the key fact that for any \( a, b \):

\[ a \vdash_S b \iff \neg a \vdash_S \neg b \]

which directly implies that \( \varphi \) is order-preserving. Let \( I \) be a fixed set. Let us show that \( \varphi_I \) is injective. Let \((a_i)_{i \in I}, (b_i)_{i \in I}\) be two family of elements of \( A \) such that \( \varphi_I([a_i]) = \varphi_I([b_i]) \), i.e. \( \neg a_i \equiv_S [a_i] \neg b_i \). Using the lemma above, if for any \( i \in I \), \( a_i \equiv_S [a_i] \neg b_i \) then for any \( i \in I \), \( a_i \equiv_S b_i \), i.e. \([a_i] = [b_i]\). To show that \( \varphi_I \) is surjective, it suffices to see that for any family \((a_i)_{i \in I}\), we have \([a_i] = [\neg a_i] = \varphi_I([\neg a_i]) \) (in \( A/S[I] \)) \( O \).

VI. CONCLUSION

A. An algebraic view on the duality of computation

To sum up, in this paper we saw how the two decompositions of the arrow \( a \rightarrow b \) as \( \neg a \wedge b \) and \( (a \vee \neg b) \), which respectively induce decompositions of a call-by-name and call-by-value \( \lambda \) calculi within Munch-Maccagnoni’s system \( L \) \[32\], yield two different algebraic structures reflecting the corresponding realizability models. Namely, call-by-name models give rise to disjunctive algebras, which are particular cases of Miquel’s implicational algebras \[28\], while conjunctive algebras correspond to call-by-value realizability models.

The well-known duality of computation between terms and contexts is reflected here by simple translations from conjunctive to disjunctive algebras and vice-versa, where the underlying lattices are simply reversed. Besides, we showed that (classical) conjunctive algebras induce triposes that are isomorphic to disjunctive triposes. The situation is summarized in Figure 3, where \( \otimes \text{-algebras} \) denotes classical conjunctive algebras.

B. From Kleene to Krivine via negative translation

We could now re-read within our algebraic landscape the result of Miquel stating that Krivine realizability models for PA2 can be obtained as a composition of Kleene realizability for HA2 and Friedman’s negative translation \[26\]. Interestingly, in his setting the fragment of formulas that is interpreted in HA2 correspond exactly to the positive formulas of \( L^\circ \), so that it gives rise to an (intuitionistic) conjunctive algebra. Friedman’s translation is then used to encode the type of stacks within this fragment via a negation. In the end, realized formulas are precisely the one that are realized through Friedman’s translation: the whole construction exactly matches the passage from an intuitionistic conjunctive structure defined by Kleene realizability to a classical implicational algebras through the arrow from \( \rightarrow \text{-algebras} \) to \( \rightarrow \text{-algebras} \) via \( \neg \text{-algebras} \).

C. Future work

While Theorem 53 implies that call-by-value and call-by-name models based on the \( L^\circ \) and \( L^\neg \) calculi are equivalents, it does not provide us with a definitive answer to our original question. Indeed, just as (by-name) implicational algebras are more general than disjunctive algebras, it could be the case that there exists a notion of (by-value) implicational algebras that is strictly more general than conjunctive algebras and which is not isomorphic to a by-name situation.

Also, if we managed to obtain various results about conjunctive algebras, there is still a lot to understand about them. Notably, the interpretation we have of the \( \lambda \)-calculus is a bit disappointing in that it does not provide us with an adequacy result as nice as in implicational algebras. In particular, the fact that each application implicitly gives rise to a double negation breaks the compositionality. This is of course to connect with the definition of truth values in by-value models which requires three layers and a double orthogonal. Yet, we feel that many things remain to understand about the underlying structure of by-value realizability models.

Finally, on a long-term perspective, the next step is obviously to understand the algebraic impact of more sophisticated evaluation strategy (e.g., call-by-need) or side-effects (e.g., a monotonic memory). While both have been used in concrete cases to give a computational content to certain axioms (e.g., the axiom of dependent choice \[14\]) or model constructions (e.g., forcing \[20\]), for the time being we have no idea on how to interpret in the realm of implicational algebras.
The author would like to thank Alexandre Miquel to which several ideas in this paper, especially the definition of conjunctive separators, should be credited; as well as the Uruguayan national research agency ANII for financing a visit in Montevideo which somehow led to the present paper.

ACKNOWLEDGMENT

REFERENCES

\[(x : A) \in \Gamma \quad \Gamma \vdash x : A \]
\[(\Gamma, x : A \vdash t : B) \quad \Gamma \vdash \lambda x : A \rightarrow B \quad \Gamma \vdash t : A \rightarrow B \quad \Gamma \vdash t : A\]
\[\Gamma \vdash t : A \quad (X \notin FV(\Gamma)) \quad \Gamma \vdash t : \forall X.A \quad \Gamma \vdash t : \forall X.A \]
\[\Gamma \vdash t : A \{X := B\} \quad \Gamma \vdash \text{cc} : ((A \rightarrow B) \rightarrow A) \rightarrow A\]

Fig. 4. Second-order type system for the $\lambda$-calculus

Appendix A

Implicative algebras

We shall now sketch the formalization of the former result. First, we extend the usual formulas of System F by defining second-order formulas with parameters as:

\[A, B ::= a \mid X \mid A \Rightarrow B \mid \forall X.A\]  \hspace{1cm} (a \in A)

We can then embed closed formulas with parameters into the implicative structure $A$. The embedding is trivially defined by:

\[a^A \triangleq a \quad (A \Rightarrow B)^A \triangleq A^A \rightarrow B^A \quad (\forall X.A)^A \triangleq \bigwedge_{a \in A} (A \{X := a\})^A\]

where $a \in A$. We define a type system for the $\lambda$-calculus with parameters (that is, $\lambda$-terms with parameter plus an instruction $\text{cc}$). Typing contexts are defined as usual by finite lists of hypotheses of the shape $(x : A)$ where $x$ is a variable and $A$ a formula with parameters. The inference rules, given in Figure 4, are the same as in System F (with the extended syntaxes of terms and formulas with parameters), plus the additional rules for $\text{cc}$.

In order to prove the adequacy of the type system with respect to the embedding, we define substitutions, which we write $\sigma$, as functions mapping variables (of terms and types) to element of $A$:

\[\sigma ::= \varepsilon \mid \sigma[x \mapsto a] \mid \sigma[X \mapsto a]\]  \hspace{1cm} (a \in A, \ x, X \text{ variables})

In the spirit of the proof of adequacy in classical realizability, we say that a substitution $\sigma$ realizes a typing context $\Gamma$, which we write $\sigma \vdash \Gamma$, if for all bindings $(x : A) \in \Gamma$ we have $\sigma(x) \approx (A[\sigma])^A$.

**Theorem 54** (Adequacy). The typing rules of Figure 4 are adequate with respect to the interpretation of terms and formulas: if $t$ is a $\lambda$-term with parameters, $A$ a formula with parameters and $\Gamma$ a typing context such that $\Gamma \vdash t : A$ then for all substitutions $\sigma \vdash \Gamma$, we have $(t[\sigma])^A \approx (A[\sigma])^A$.

**Proof.** The proof resembles the usual proof of adequacy in classical realizability (see [19], [29]), namely by induction on typing derivations. \qed

**Corollary 55** For all $\lambda$-terms $t$, if $\vdash t : A$, then $t^A \approx A^A$.

A. Implicative tripos

**Definition 56** (Hyperdoctrine). Let $C$ be a Cartesian closed category. A first-order hyperdoctrine over $C$ is a contravariant functor $T : C^{op} \rightarrow HA$ with the following properties:

1) For each diagonal morphism $\delta_X : X \rightarrow X \times X$ in $C$, the left adjoint to $T(\delta_X)$ at the top element $\top \in T(X)$ exists. In other words, there exists an element $=X \in T(X \times X)$ such that for all $\varphi \in T(X \times X)$:

\[\top \approx T(\delta_X)(\varphi) \iff =X \approx \varphi\]

2) For each projection $\pi^1_{\Gamma, X} : \Gamma \times X \rightarrow \Gamma$ in $C$, the monotonic function $T(\pi^1_{\Gamma, X}) : T(\Gamma) \rightarrow T(\Gamma \times X)$ has both a left adjoint $(\exists X)_\Gamma$ and a right adjoint $(\forall X)_\Gamma$:

\[\varphi \leq T(\pi^1_{\Gamma, X})(\psi) \iff (\exists X)_\Gamma(\varphi) \leq \psi\]
\[T(\pi^1_{\Gamma, X})(\varphi) \leq \psi \iff \varphi \leq (\forall X)_\Gamma(\psi)\]

3) These adjoints are natural in $\Gamma$, i.e. given $s : \Gamma \rightarrow \Gamma'$ in $C$, the following diagrams commute:

\[
\begin{array}{ccc}
T(\Gamma' \times X) & \xrightarrow{T(s \times id_X)} & T(\Gamma \times X) \\
(\exists X)_{\Gamma'} & \downarrow & (\exists X)_{\Gamma} \\
T(\Gamma') & \xrightarrow{T(s)} & T(\Gamma) \\
\end{array}
\]

\[
\begin{array}{ccc}
T(\Gamma' \times X) & \xrightarrow{T(s \times id_X)} & T(\Gamma \times X) \\
(\forall X)_{\Gamma'} & \downarrow & (\forall X)_{\Gamma} \\
T(\Gamma') & \xrightarrow{T(s)} & T(\Gamma) \\
\end{array}
\]
This condition is also called the Beck-Chevalley conditions.
The elements of $\mathcal{T}(X)$, as $X$ ranges over the objects of $C$, are called the $\mathcal{T}$-predicates.

**Definition 57 (Tripos).** A tripos over a Cartesian closed category $C$ is a first-order hyperdoctrine $\mathcal{T} : C^{op} \to \text{HA}$ which has a generic predicate, i.e. there exists an object $\text{Prop} \in C$ and a predicate $\text{tr} \in \mathcal{T}(\text{Prop})$ such that for any object $\Gamma \in C$ and any predicate $\varphi \in \mathcal{T}(\Gamma)$, there exists a (not necessarily unique) morphism $\chi_{\varphi} \in C(\Gamma, \text{Prop})$ such that:

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

**Induced tripos:** In order to recover a Heyting algebra, it suffices to consider the quotient $\mathcal{A}/_{\equiv_S}$ by the relation $\equiv_S$, which is equipped with an order relation:

$$[a] \preceq \mathcal{H} [b] \iff a \vdash_S b$$

(for all $a, b \in \mathcal{A}$) where we write $[a]$ for the equivalence class of $a \in \mathcal{A}$. We define:

$$\begin{align*}
[a] \to_H [b] & \equiv [a \to b] \\
[a] \land_H [b] & \equiv [a \times b] \\
[a] \lor_H [b] & \equiv [a + b] \\
\perp_H & \equiv [\perp] = \{a \in \mathcal{A} : \neg a \in S\}
\end{align*}$$

The quintuple $(\mathcal{H}, \preceq_H, \land_H, \lor_H, \to_H)$ is a Heyting algebra.

We define:

$$\begin{align*}
[a] \to_J [b] & \equiv [a \to b] \\
[a] \land_J [b] & \equiv [a \times b] \\
[a] \lor_J [b] & \equiv [a + b] \\
\perp_J & \equiv [\perp] = \{a \in \mathcal{A} : \neg a \in S\}
\end{align*}$$

The quintuple $(\mathcal{H}, \preceq_H, \land_H, \lor_H, \to_H)$ is a Heyting algebra.

**Theorem 58 (Implicative tripos).** Let $(\mathcal{A}, \preceq, \to, S)$ be an implicative algebra. The following functor:

$$\mathcal{T} : I \mapsto \mathcal{A}^I/S[I]$$

$$\mathcal{T}(f) : \begin{cases}
\mathcal{A}^I/S[I] & \to \mathcal{A}^J/S[J] \\
[(a_i)_{i \in I}] & \mapsto [(a_{f(j)})_{j \in J}]
\end{cases}$$

(\forall f \in J \to I)

defines a tripos.

**Proof.** We verify that $\mathcal{T}$ satisfies all the necessary conditions to be a tripos.

- The functoriality of $\mathcal{T}$ is clear.
- For each $I \in \text{Set}$, the image of the corresponding diagonal morphism $\mathcal{T}(\delta_I)$ associates to any element $[(a_{ij})_{(i,j) \in I \times I}] \in \mathcal{T}(I \times I)$ the element $[(a_i)_{i \in I}] \in \mathcal{T}(I)$. We define:

$$\lambda a \in \mathcal{A} \quad (i,j) \mapsto \begin{cases}
\lambda a \in \mathcal{A} (a \to a) & \text{if } i = j \\
\perp \to \top & \text{if } i \neq j
\end{cases}$$

and we need to prove that for all $[a] \in \mathcal{T}(I \times I)$:

$$[\top]_I \preceq_S [I] \mathcal{T}(\delta_I)(a) \iff [\top]_I \preceq_S [I \times I] [a]$$

Let then $[(a_{ij})_{i,j \in I}]$ be an element of $\mathcal{T}(I \times I)$. From left to right, assume that $[\top]_I \preceq_S [I] \mathcal{T}(\delta_I)(a)$, that is to say that there exists $s \in S$ such that for any $i \in I$, $s \preceq \top \to a_{ii}$. Then it is easy to check that for all $i, j \in I$, $\lambda z.s(\lambda x.x) \preceq i = j \to a_{ij}$. Indeed, using the adjunction and the $\beta$-reduction it suffices to show that for all $i, j \in I$, $(i = j) \preceq (s(\lambda x.x)) \to a_{ij}$.

If $i = j$, this follows from the fact that $(s(\lambda x.x)) \preceq a_{ii}$. If $i \neq j$, this is clear by subtyping.

From right to left, if there exists $s \in S$ such that for any $i, j \in I$, $s \preceq i = j \to a_{ij}$, then in particular for all $i \in I$ we have $s \preceq (\lambda x.x) \to a_{ii}$, and then $\lambda x.s(\lambda x.x) \preceq \top \to a_{ii}$ which concludes the case.

- For each projection $\pi^1_{J \times I} : I \times J \to I$ in $C$, the monotone function $\mathcal{T}(\pi^1_{J \times I}) : \mathcal{T}(I) \to \mathcal{T}(I \times J)$ has both a left adjoint $\exists J$ and a right adjoint $\forall J$ which are defined by:

$$\forall J_I \{ [(a_{ij})_{i \in I \times J}] \} \equiv \{ \forall_{j \in J} a_{ij} \}_{i \in I}$$

$$\exists J_I \{ [(a_{ij})_{i \in I \times J}] \} \equiv \{ \exists_{j \in J} a_{ij} \}_{i \in I}$$

Note that the definition of the functor on functions $f : J \to I$ assumes implicitly the possibility of picking a representative in any equivalent class $[a] \in \mathcal{A}/S[I]$, i.e. the full axiom of choice.

The reader familiar with classical realizability might recognize the usual interpretation of Leibniz’s equality.
The proofs of the adjointness of this definition are again easy manipulation of \( \lambda \)-calculus. We only give the case of \( \exists \), the case for \( \forall \) is easier. We need to show that for any \([a_{ij}]_{i,j} \in T(I \times J)\) and for any \([b_i]_{i \in I}\), we have:

\[
[(a_{ij})_{i,j} \in I \times J] \preceq_{S[I \times J]} [(b_i)_{i \in I}] \iff [(\exists_j a_{ij})_{i \in I}] \preceq_{S[I]} [(b_i)_{i \in I}]
\]

Let us fix some \([a]\) and \([b]\) as above. From left to right, assume that there exists \(s \in S\) such that for all \(i \in I, j \in J\), \(s \preceq a_{ij} \rightarrow b_i\), and thus \(s a_{ij} \preceq b_i\). Using the semantic elimination rule of the existential quantifier, we deduce that for all \(i \in I\), if \(t \preceq \exists_j a_{ij}\), then \(t(\lambda x.sx) \preceq b_i\). Therefore, for all \(i \in I\) we have \(\lambda y.y(\lambda x.sx) \preceq \exists_j a_{ij} \rightarrow b_i\).

From right to left, assume that there exists \(s \in S\) such that for all \(i \in I, j \in J\), \(s \preceq \exists_j a_{ij} \rightarrow b_i\). For any \(j \in J\), using the semantic introduction rule of the existential quantifier, we deduce that for all \(i \in I\), \(\lambda x.xa_{ij} \preceq \exists_j a_{ij}\). Therefore, for all \(i \in I\) we have \(\lambda x.s(\lambda z.zx) \preceq a_{ij} \rightarrow b_i\).

- These adjoints clearly satisfy the Beck-Chevalley condition. For instance, for the existential quantifier, we have for all \(I, I', J, \) for any \([a_{i'j'}]_{i',j'} \in T(I' \times J)\) and any \(s : I \rightarrow I'\),

\[
(T(s) \circ (\exists J)_{I'})([(a_{i'j'})_{i',j'}]_{I' \times J}) = T(s)([(\exists_j a_{i'j})_{i \in I}])
= [(\exists_j a_{i'j})_{i \in I}]
= ((\exists J)_{I'}([(a_{i'j'})_{i',j'}]_{I' \times J})
= ((\exists J)_{I'} \circ T(s \times \text{id}_J)([(a_{i'j'})_{i',j'}]_{I' \times J}))
\]

- Finally, we define \(\text{Prop} \triangleq A\) and verify that \(\text{tr} \triangleq [\text{id}_A] \in T(\text{Prop})\) is a generic predicate. Let then \(I\) be a set, and \(a = [(a_i)_{i \in I}] \in T(I)\). We let \(\chi_a : i \mapsto a_i\) be the characteristic function of \(a\) (it is in \(I \rightarrow \text{Prop}\)), which obviously satisfies that for all \(i \in I\):

\[
T(\chi_a)(\text{tr}) = [(\chi_a(i))_{i \in I}] = [(a_i)_{i \in I}]
\]

\(\square\)
A. The $L^\forall$ calculus

Following Munch-Maccagnoni’s paper [32, Appendix E], we can embed the $\lambda$-calculus into the $L^\forall$-calculus. To this end, we are guided by the expected definition of the arrow $A \to B \triangleq \neg A \forall B$. It is easy to see that with this definition, a stack $u \cdot e$ in $A \to B$ (that is with $u$ a term of type $A$ and $e$ a context of type $B$) is naturally defined as a shorthand for the pair $([u], e)$, which indeed inhabits the type $\neg A \forall B$. Starting from there, the rest of the definitions are straightforward:

\[
\begin{align*}
\lambda x.t & \triangleq [\mu([x], \beta) \cdot (t \parallel \beta)] \\
u \cdot e & \triangleq ([u], e)
\end{align*}
\]

**Proposition 59.** The following typing rules are admissible:

\[
\begin{array}{c}
\Gamma, x : A \vdash t : B \\
\Gamma \vdash \lambda x.t : A \to B
\end{array}
\quad
\begin{array}{c}
\Gamma \vdash u : A \mid \Delta \\
\Gamma \vdash e : B \mid \Delta
\end{array}
\quad
\begin{array}{c}
\Gamma \vdash t : A \to B \mid \Delta \\
\Gamma \vdash u : A \mid \Delta
\end{array}
\quad
\begin{array}{c}
\Gamma \vdash u \cdot e : A \to B \mid \Delta
\end{array}
\]

**Proof.** Each case is directly derivable from $L^\forall$ type system. We abuse the notation to denote by $(\text{def})$ a rule which simply consists in unfolding the shorthands defining the $\lambda$-terms.

- **Case $\mu([x], \alpha)$.c:**

  \[
  \begin{array}{c}
  \Gamma \mid \alpha : \neg A \mid \Delta, \alpha : \neg A, \beta : B \\
  \Gamma \vdash [\mu([x], \beta) \cdot (\mu([x], \alpha) \cdot \alpha : \neg A \mid \Delta, \beta : B)] (\text{cut})
  \end{array}
  \]

  \[
  \begin{array}{c}
  \Gamma \mid \mu([x], \beta) \cdot (\mu([x], \alpha) \cdot \alpha : \neg A \mid \Delta, \beta : B) \\
  \Gamma \vdash \mu([x], \beta) \cdot (\mu([x], \alpha) \cdot \alpha : \neg A \mid \Delta, \beta : B) (\text{(r-\mu)})
  \end{array}
  \]

- **Case $\lambda x.t$:**

  \[
  \begin{array}{c}
  \Gamma \mid \alpha : \neg A \mid \Delta, \alpha : \neg A, \beta : B \\
  \Gamma \vdash \lambda x.t : A \to B \mid \Delta
  \end{array}
  \]

  \[
  \begin{array}{c}
  \Gamma \mid \beta : B \mid \Delta, \beta : B \\
  \Gamma \vdash \lambda x.t : A \to B \mid \Delta (\text{cut})
  \end{array}
  \]

- **Case $u \cdot e$:**

  \[
  \begin{array}{c}
  \Gamma \mid u : A \mid \Delta \\
  \Gamma \mid [u] : A \mid \Delta, e : B \mid \Delta \\
  \Gamma \mid \mu([u], \alpha) : \neg A \forall B \mid \Delta (\text{cut})
  \end{array}
  \]

- **Case $tu$:**

  \[
  \begin{array}{c}
  \Gamma \mid \alpha : B \mid \Delta, \alpha : B \\
  \Gamma \mid u : A \mid \Delta, \alpha : B \\
  \Gamma \mid [u] \cdot \alpha : A \to B \mid \Delta (\text{cut})
  \end{array}
  \]

  \[
  \begin{array}{c}
  \Gamma \mid \mu([u], \alpha) : \neg A \forall B \mid \Delta (\text{(r-\mu)})
  \end{array}
  \]

  \[
  \begin{array}{c}
  \Gamma \mid t u : B \mid \Delta
  \end{array}
  \]

\[
\square
\]

1) **Realizability interpretation:** Given a fixed pole $\bot$, the interpretation is given by:

\[
\begin{align*}
\|S\|_V & \triangleq S \\
\|\forall X.A\|_V & \triangleq S \in \mathcal{P}(V_0) \cup \{A[X := \hat{S}]\}_V \\
\|A \forall B\|_V & \triangleq \{(V_1, V_2) : V_1 \in \|A\|_V \land V_2 \in \|B\|_V\} \\
\|\neg A\|_V & \triangleq \{[t] : t \in \|A\|\} \\
\|A\|_V & \triangleq \{t : \forall V \in \|A\|_V, t \parallel V\} \\
\|\|A\|_V & \triangleq \{e : \forall t \in \|A\|_V, t \perp e\}
\end{align*}
\]

We shall now verify that the type system of $L^\forall$ is indeed adequate with this interpretation. We first prove the following simple lemma:
Lemma 60 (Substitution). Let $A$ be a formula whose only free variable is $X$. For any closed formula $B$, if $S = \|B\|_V$, then $\|A[B/X]\|_V = \|A[S/X]\|_V$.

Proof. Easy induction on the structure of formulas, with the observation that the statement for primitive falsity values implies the same statement for truth values ($\|A[B/X]\| = \|A[S/X]\|$) and falsity values ($\|A[B/X]\| = \|A[S/X]\|$). The key case is for the atomic formula $A \equiv X$, where we easily check that

$$\|X[B/X]\|_V = \|B\|_V = S = \|S\|_V = \|X[S/X]\|_V$$

We define $\Gamma \cup \Delta$ as the union of both contexts where we annotate the type of hypothesis $(\kappa : A) \in \Delta$ with $\kappa : A^\perp$:

$$\Gamma \cup (\Delta, \kappa : A) \triangleq (\Gamma \cup \Delta), \kappa : A^\perp$$

The last step before proving adequacy consists in defining substitutions and valuations. We say that a valuation, which we write $\rho$, is a function mapping each second-order variable to a primitive falsity value $\rho(X) \in \mathcal{P}(\mathcal{V}_0)$. A substitution, which we write $\sigma$, is a function mapping each variable $x$ to a closed term $c$ and each variable $\alpha$ to a closed value $V \in \mathcal{V}_0$:

$$\sigma ::= \varepsilon | \sigma, x \mapsto t | \sigma, \alpha \mapsto V^+$$

We say that a substitution $\sigma$ realizes a context $\Gamma$ and note $\sigma \vdash \Gamma$ when for each binding $(x : A) \in \Gamma$, $\sigma(x) \in |A|$. Similarly, we say that $\sigma$ realizes a context $\Delta$ if for each binding $(\alpha : A) \in \Delta$, $\sigma(\alpha) \in ||A||_V$.

We can now state the property of adequacy of the realizability interpretation:

**Proposition** (Adequacy). Let $\Gamma, \Delta$ be typing contexts, $\rho$ be a valuation and $\sigma$ be a substitution such that $\sigma \vdash \Gamma[\rho]$ and $\sigma \vdash \Delta[\rho]$. We have:

1. If $V^+$ is a positive value such that $\Gamma \vdash V^+ : A \vdash \Delta$, then $V^+[\sigma] \in ||A[\rho]||_V$.
2. If $t$ is a term such that $\Gamma \vdash t : A \vdash \Delta$, then $t[\sigma] \in ||A[\rho]||$.
3. If $e$ is a context such that $\Gamma \vdash e : A \vdash \Delta$, then $e[\sigma] \in ||A||_V$.
4. If $c$ is a command such that $c : (\Gamma \vdash \Delta)$, then $c[\sigma] \in \perp$.

Proof. The proof is almost the same as for the proof of adequacy for the call-by-name $\lambda \mu \iota$-calculus. We only give some key cases which are peculiar to this setting. We proceed by induction over the typing derivations. Let $\sigma$ be a substitution realizing $\Gamma[\rho]$ and $\Delta[\rho]$.

**Case** ($\vdash \neg$). Assume that we have:

$$\frac{c : \Gamma, x : A \vdash \Delta}{\Gamma \vdash \neg \mu[x].c}$$

and let $[t]$ be a term in $||A[\rho]||_V$, that is to say that $t \in ||A[\rho]||$. We know by induction hypothesis that for any valuation $\sigma' \vdash (\Gamma, x : A)[\rho], c[\sigma'] \in \perp$ and we want to show that $\mu[x].c[\sigma] \parallel [t]$. We have that:

$$\mu[x].c[\sigma][t/x] \rightarrow_\beta e[\sigma][t/x] = c[\sigma, x \mapsto t]$$

hence it is enough by saturation to show that $c[\sigma][u/x] \in \perp$. Since $t \in ||A[\rho]||$, $\sigma(x \mapsto t) \vdash (\Gamma, x : A)[\rho]$ and we can conclude by induction hypothesis. The cases for ($\mu \vdash$), ($\iota \mu$) and ($\neg \neg$) proceed similarly.

**Cases** ($\neg \vdash$). Trivial by induction hypotheses.

**Case** ($\forall \vdash$). Assume that we have:

$$\frac{\Gamma \vdash e_1 : A \vdash \Delta \quad \Gamma \vdash u : B \vdash \Delta}{\Gamma \vdash \forall x : B \vdash \Delta}$$

Let then $t$ be a term in $||A \forall B||[\rho]$, to show that $\langle t \parallel (e_1, e_2) \rangle \in \perp$, we proceed by anti-reduction:

$$\langle t \parallel (e, e') \rangle \rightarrow_\beta \langle \mu \alpha. \langle \mu \alpha'. (t \parallel (\alpha, \alpha')) \parallel e' \parallel e \rangle \parallel e \rangle$$

It now easy to show, using the induction hypotheses for $e$ and $e'$ that this command is in the pole: it suffices to show that the term $\mu \alpha. \langle \mu \alpha'. (t \parallel (\alpha, \alpha')) \parallel e' \parallel e \rangle \in |A|$, which amounts to showing that for any value $V_1 \in ||A||_V$:

$$\langle \mu \alpha. \langle \mu \alpha'. (t \parallel (\alpha, \alpha')) \parallel V \rangle \parallel e' \parallel e \rangle \rightarrow_\beta \langle \mu \alpha'. (t \parallel (V, \alpha')) \parallel e' \parallel e \rangle \in \perp$$

Again this holds by showing that for any $V' \in |B|$, 

$$\langle \mu \alpha'. (t \parallel (V, \alpha')) \parallel V' \parallel e' \parallel e \parallel e \rangle \rightarrow_\beta \langle t \parallel (V, V') \parallel e \parallel e \rangle \in \perp$$
By induction hypothesis, we obtain that \( e[\sigma] \in \{ A[B/X][\rho] \} \); so that if we denote \( \{ B[\rho] \} V \in P(V_0) \) by \( S \), we have:
\[
e[\sigma] \in \{ A[\hat{S}/X] \} \subseteq S \in P(V_0) \cup \{ A[\hat{S}/X][\rho] \} V \wedge V_2 \in \{ B[\rho] \} V
\]
where we make implicit use of Lemma 60.

### B. Disjunctive structures

We should now define the notion of disjunctive structure. Regarding the expected commutations, as we choose negative connectives and in particular a universal quantifier, we should define commutations with respect to arbitrary meets. The following properties of the realizability interpretation for \( L^\ominus \) provides us with a safeguard for the definition to come:

**Proposition (Commutations).** In any \( L^\ominus \) realizability model (that is to say for any pole \( \bot \)), the following equalities hold:

1. If \( X \notin FV(B) \), then \( \{ \forall X.(A \ominus B) \} V = \{ \forall X.A \ominus B \} V \).
2. If \( X \notin FV(A) \), then \( \{ \forall X.(A \ominus B) \} V = \{ A \ominus (\forall X.B) \} V \).
3. \( \{ \neg(\forall X.A) \} V = \{ \forall A\{ X := \hat{S}\} \} V \)

**Proof.**

1. Assume the \( X \notin FV(B) \), then we have:
\[
\{ \forall X.(A \ominus B) \} V = S \in P(V_0) \cup \{ A\{ X := \hat{S}\} \ominus B \} V
\]
\[
= S \in P(V_0) \cup \{ \forall A\{ X := \hat{S}\} \} V \wedge V_2 \in \{ B[\rho] \} V
\]
\[
= \{ (V_1, V_2) : V_1 \in P(V_0) \cup \{ A\{ X := \hat{S}\} \} V \wedge V_2 \in \{ B[\rho] \} V \}
\]
\[
= \{ (V_1, V_2) : V_1 \in \{ \forall X.A \} V \wedge V_2 \in \{ B \} V \}
\]
\[
= \{ \forall (\forall X.A) \ominus B \} V
\]

2. Identical.

3. The proof is again a simple unfolding of the definitions:
\[
\{ \neg(\forall X.A) \} V = \{ [t] : t \in \forall X.A \} = \{ [t] : t \in \bigcap S \in P(V_0) \{ A\{ X := \hat{S}\} \} \}
\]
\[
= \bigcap S \in P(V_0) \{ [t] : t \in \forall A\{ X := \hat{S}\} \} = \bigcap S \in P(V_0) \neg \forall A\{ X := \hat{S}\} V
\]

**Proposition 61.** If \( (\forall, \ominus, \ominus, \neg) \) is a disjunctive structure, then the following hold for all \( a \in A \):

1. \( \top \ominus a = \top \)
2. \( a \ominus \top = \top \)
3. \( \neg \top = \bot \)

**Proof.** Using the axioms of disjunctive structures, we prove:

1. for all \( a \in A \), \( \top \ominus a = (\emptyset) \ominus a = \bigcup x \in A \{ x \ominus a : x \in \emptyset \} = \bigcup \emptyset = \top \)
2. for all \( a \in A \), \( a \ominus \top = a \ominus (\emptyset) = \bigcup x \in A \{ a \ominus x : x \in \emptyset \} = \bigcup \emptyset = \top \)
3. \( \neg \top = \neg (\emptyset) = \bigcup x \in A \{ \neg x : x \in \emptyset \} = \bigcup \emptyset = \bot \)

If we abstract the structure of the realizability interpretation of \( L^\ominus \), it is a structure of the form \( (\mathcal{T}_0, \mathcal{E}_0, V_0, (\cdot, \cdot), [\cdot], \bot) \), where \((\cdot, \cdot)\) is a binary map from \( \mathcal{E}_0^2 \) to \( \mathcal{E}_0 \) (whose restriction to \( V_0 \) has values in \( V_0 \)), \([\cdot]\) is an operation from \( \mathcal{T}_0 \) to \( V_0 \), and \( \bot \in \mathcal{T}_0 \times \mathcal{E}_0 \) is a relation. From this sextuple, we can define:

- \( A \triangleq P(V_0) \)
- \( a \ominus b \triangleq \{ (V_1, V_2) : V_1 \in a \wedge V_2 \in b \} \)
- \( a \ominus b \triangleq \{ [t] : t \in a \ominus b \} \)

**Proposition 62.** The quadruple \( (\forall, \ominus, \ominus, \neg) \) is a disjunctive structure.

**Proof.** We show that the axioms of Definition \([\text{III}]\) are satisfied.

1. (Contravariance) Let \( a, a' \in A \), such that \( a \preceq a' \) ie \( a' \subseteq a \). Then \( a^\bot \subseteq a'^\bot \) and thus
\[
\neg a = \{ [t] : t \in a^\bot \} \subseteq \{ [t] : t \in a'^\bot \} = \neg a'
\]
Proposition 69 (Properties of $\Jan$). For all functions $c, c' : \mathcal{A} \to \mathcal{C}_A$, the following hold:

\[
\Jan c \subseteq \Jan c' \quad \text{and} \quad \Jan c = \Jan c' \iff \forall a \in \mathcal{A}, c(a) = c'(a)
\]
1. If for all \( a \in \mathcal{A} \), \( c(a) \leq c'(a) \), then \( \mu^+.c \leq \mu^+.c' \) \hspace{1cm} \text{(Variance)}

2. For all \( t \in \mathcal{A} \), then \( \langle t \parallel \mu^+.c \rangle \leq c(t) \) \hspace{1cm} \text{(\( \beta \)-reduction)}

3. For all \( e \in \mathcal{A} \), then \( t = \mu^+.\langle a \mapsto \langle a \parallel e \rangle \rangle \) \hspace{1cm} \text{(\( \eta \)-expansion)}

**Proof.** 1) Direct consequence of Proposition 65

2. Trivial by definition of \( \mu^+ \).

\[ \square \]

**Remark 70** (Subject reduction). The \( \beta \)-reduction \( c \rightarrow_\beta c' \) is reflected by the ordering relation \( c \leq c' \), which reads “if \( c \) is well-formed, then so is \( c' \)”. In other words, this corresponds to the usual property of subject reduction. In the sequel, we will see that \( \beta \)-reduction rules of \( \mathcal{L}_\beta^\gamma \) will always been reflected in this way through the embedding in disjunctive structures.

3) Terms: Dually to the definitions of (positive) contexts \( \mu^+ \) as a join, we define the embedding of (negative) terms, which are all binders, by arbitrary meets:

**Definition 71** (\( \mu^- \)). For all \( c : \mathcal{A} \rightarrow \mathcal{C}_A \), we define:

\[ \mu^- . c := \bigwedge_{a \in \mathcal{A}} \{ a : c(a) \in \bot \} \]

**Definition 72** (\( \mu^0 \)). For all \( c : \mathcal{A}^2 \rightarrow \mathcal{C}_A \), we define:

\[ \mu^0 . c := \bigwedge_{a,b \in \mathcal{A}} \{ a \land b : c(a,b) \in \bot \} \]

**Definition 73** (\( \mu^\| \)). For all \( c : \mathcal{A} \rightarrow \mathcal{C}_A \), we define:

\[ \mu^\| . c := \bigwedge_{a \in \mathcal{A}} \{ a : c(a) \in \bot \} \]

These definitions also satisfy some variance properties with respect to the preorder \( \leq \) and the order relation \( \leq_\| \), namely, negative binders for variable ranging over positive contexts are covariant, while negative binders intended to catch negative terms are contravariant.

**Proposition 74** (Variance). For any functions \( c, c' \) with the corresponding arities, the following hold:

1. If \( c(a) \leq c'(a) \) for all \( a \in \mathcal{A} \), then \( \mu^- . c \leq \mu^- . c' \)
2. If \( c(a,b) \leq c'(a,b) \) for all \( a,b \in \mathcal{A} \), then \( \mu^0 . c \leq \mu^0 . c' \)
3. If \( c(a) \leq c'(a) \) for all \( a \in \mathcal{A} \), then \( \mu^\| . c' \leq \mu^\| . c \)

**Proof.** Direct consequences of Proposition 65

\[ \square \]

The \( \eta \)-expansion is also reflected as usual by the ordering relation \( \leq_\| \):

**Proposition 75** (\( \eta \)-expansion). For all \( t \in \mathcal{A} \), the following holds:

1. \( t = \mu^- \langle a \mapsto \langle t \parallel a \rangle \rangle \)
2. \( t \leq \mu^0 \langle a \mapsto \langle t \parallel \langle a,b \rangle \rangle \rangle \)
3. \( t \leq \mu^\| \langle a \mapsto \langle t \parallel \langle a \rangle \rangle \rangle \rangle \)

**Proof.** Trivial from the definitions.

\[ \square \]

The \( \beta \)-reduction is reflected by the preorder \( \leq \):

**Proposition 76** (\( \beta \)-reduction). For all \( e, e_1, e_2, t \in \mathcal{A} \), the following holds:

1. \( \langle \mu^- . c \parallel e \rangle \leq c(e) \)
2. \( \langle \mu^0 . c \parallel \langle e_1, e_2 \rangle \rangle \leq c(e_1, e_2) \)
3. \( \langle \mu^\| . c \parallel [t] \rangle \leq c(t) \)

**Proof.** Trivial from the definitions.

\[ \square \]

Finally, we call a \( \mathcal{L}_\beta^\gamma \) term with parameters in \( \mathcal{A} \) (resp. context, command) any \( \mathcal{L}_\beta^\gamma \) term (possibly) enriched with constants taken in the set \( \mathcal{A} \). Commands with parameters are equipped with the same rules of reduction as in \( \mathcal{L}_\beta^\gamma \), considering parameters

\[ \square \]
as inert constants. To every closed \( L^\mathcal{Y} \) term \( t \) (resp. context \( e \), command \( c \)) we associate an element \( t^A \) (resp. \( e^A \), \( c^A \)) of \( \mathcal{A} \), defined by induction on the structure of \( t \) as follows:

<table>
<thead>
<tr>
<th>Contexts :</th>
<th>Terms :</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a^A )</td>
<td>( a^A )</td>
</tr>
<tr>
<td>( (e_1, e_2)^A )</td>
<td>( (\mu \alpha.c)^A )</td>
</tr>
<tr>
<td>( (\mu x.c)^A )</td>
<td>( \mu^{-}(a \mapsto (c[\alpha := a])^A) )</td>
</tr>
</tbody>
</table>

We shall now prove that the interpretation of \( L^\mathcal{Y} \) is adequate with respect to the interpretation of terms, contexts, commands and formulas. Indeed, for all contexts \( \Gamma \), \( \Delta \), for all formulas with parameters \( A \) then for all substitutions \( \sigma \) such that \( \sigma \vdash \Gamma \) and \( \sigma \vdash \Delta \), we have:

1. for any term \( t \), if \( \Gamma \vdash t : A \mid \Delta \), then \( (t[\sigma])^A \preceq A[\sigma]^A \);
2. for any context \( e \), if \( \Gamma \vdash e : A \mid \Delta \), then \( (e[\sigma])^A \succeq A[\sigma]^A \);
3. for any command \( c \), if \( c : (\Gamma \vdash \Delta) \), then \( c[\sigma]^A \in \perp \).

**Proof.** By induction over the typing derivations.

**Case** (\( \text{Cut} \)).: Assume that we have:

\[
\frac{\Gamma \vdash t : A \mid \Delta \quad \Gamma \mid e : A \vdash \Delta}{\Gamma \mid (t \parallel e) : \Gamma \vdash \Delta} \tag{\text{Cut}}
\]

By induction hypotheses, we have \( (t[\sigma])^A \preceq A[\sigma]^A \) and \( (e[\sigma])^A \succeq A[\sigma]^A \). By transitivity of the relation \( \preceq \), we deduce that \( (t \parallel e)[\sigma]^A \in \perp \).

**Case** (\( \text{ax} \)).: Straightforward, since if \( (x : A) \in \Gamma \), then \( (x[\sigma])^A \preceq (A[\sigma])^A \). The case (\( ax \)) is identical.
By induction hypothesis, we have that \((\alpha \mapsto (A[\sigma])^A)^A \in \bot\). Then, by definition we have:

\(((\mu \alpha.c)[\sigma])^A = (\mu \alpha.(c[\sigma]))^A = \bigcup_{b \in A} \{ b : (c[\sigma, \alpha \mapsto b])^A \in \bot \} \preceq (A[\sigma])^A\)

**Case (\mu \vdash):** Similarly, assume that we have:

\[
\frac{\Gamma, x : A \vdash \Delta}{\Gamma \vdash \mu \alpha.x : A \mid \Delta} \quad (\mu \vdash)
\]

By induction hypothesis, we have that \((c[\sigma, x \mapsto (A[\sigma])^A])^A \in \bot\). Therefore, we have:

\(((\mu x.c)[\sigma])^A = (\mu x.(c[\sigma]))^A = \bigcup_{b \in A} \{ b : (c[\sigma, x \mapsto b])^A \in \bot \} \succ (A[\sigma])^A\).

**Case (\forall \vdash):** Assume that we have:

\[
\frac{\Gamma \vdash e_1 : A_1 \mid \Delta \quad \Gamma \vdash e_2 : A_2 \mid \Delta}{\Gamma \mid (e_1, e_2) : A_1 \forall \alpha x \forall \alpha x \vdash A_2 \mid \Delta} \quad (\forall \vdash)
\]

By induction hypotheses, we have that \((e_1[\sigma])^A \succ (A_1[\sigma])^A\) and \((e_2[\sigma])^A \succ (A_2[\sigma])^A\). Therefore, by monotonicity of the \(\forall\) operator, we have:

\(((e_1, e_2)[\sigma])^A = (e_1[\sigma], e_2[\sigma])^A = (e_1[\sigma])^A \forall (e_2[\sigma])^A \succ (A_1[\sigma])^A \forall (A_2[\sigma])^A\).

**Case (\neg \vdash):** Assume that we have:

\[
\frac{\Gamma \vdash t : A \mid \Delta}{\Gamma \vdash \neg t : \neg A \mid \Delta} \quad (\neg \vdash)
\]

By induction hypothesis, we have that \((t[\sigma])^A \preceq (A[\sigma])^A\). Then by definition of \([\cdot]^A\) and covariance of the \(\neg\) operator, we have:

\(((\neg t[\sigma])^A = -(t[\sigma])^A \preceq -(A[\sigma])^A\).

**Case (\neg \vdash):** Assume that we have:

\[
\frac{\Gamma \vdash e : A \mid \Delta}{\Gamma \vdash \mu [x].c : \neg A \mid \Delta} \quad (\neg \vdash)
\]

By induction hypothesis, we have that \((c[\sigma, x \mapsto (A[\sigma])^A])^A \in \bot\). Therefore, we have:

\(((\mu [x].c)[\sigma])^A = (\mu [x].(c[\sigma]))^A = \bigcup_{b \in A} \{ b : (c[\sigma, x \mapsto b])^A \in \bot \} \preceq -(A[\sigma])^A\).

**Case (\forall \vdash):** Assume that we have:

\[
\frac{\Gamma \vdash e : A[X := B] \mid \Delta}{\Gamma \vdash e : \forall X.A \mid \Delta} \quad (\forall \vdash)
\]

By induction hypothesis, we have that \((e[\sigma])^A \preceq (A[X := B])^A\) and \((A[\sigma, X \mapsto (B[\sigma])^A])^A\). Therefore, we have that \((e[\sigma])^A \preceq (A[\sigma, X \mapsto (B[\sigma])^A])^A \preceq \bigcup_{b \in A} (A[X := b])^A\).
Case \((\vdash \forall)\) : Similarly, assume that we have:

\[
\Gamma \vdash t : A \mid \Delta \quad X \notin FV(\Gamma, \Delta) \quad \vdash t : \forall X. A
\]

By induction hypothesis, we have that \((t[\sigma])^A \preceq (A[\sigma, X \mapsto b])^A\) for any \(b \in A\). Therefore, we have that \((t[\sigma])^A \preceq \bigwedge_{b \in A} (A[X := b][\sigma]^A)\).

E. The induced implicative structure

As expected, any disjunctive structures directly induces an implicative structure:

**Proposition 78** If \((A, \preceq, \forall, \neg)\) is a disjunctive structure, then \((A, \preceq, \rightarrow)\) is an implicative structure.

**Proof.** We need to show that the definition of the arrow fulfills the expected axioms:

1) (Variance) Let \(a, b, a', b' \in A\) be such that \(a \preceq a\) and \(b \preceq b'\), then we have:

\[
a \rightarrow b = \neg a \forall b \preceq \neg a' \forall b' = a' \rightarrow b'
\]

since \(\neg a \preceq \neg a'\) by contra-variance of the negation and \(b \preceq b'\).

2) (Distributivity) Let \(a \in A\) and \(B \subseteq A\), then we have:

\[
\bigwedge_{b \in B} (a \rightarrow b) = \bigwedge_{b \in B} (\neg a \forall b) = \neg a \forall (\bigwedge_{b \in B} b) = a \rightarrow (\bigwedge_{b \in B} b)
\]

by distributivity of the infimum over the disjunction.

**Lemma 79** The shorthand \(\mu([x], \alpha).c\) is interpreted in \(A\) by:

\[
(\mu([x], \alpha).c)^A = \bigwedge_{a, b \in A} \{(\neg a) \forall b : c[x := a, \alpha := b] \in \preceq\}
\]

**Proof.**

\[
\mu([x], \alpha).c)^A = (\mu(x_0, \alpha).\langle \mu[x].c \parallel x_0 \rangle)^A
\]

\[
= \bigwedge_{a', b \in A} \{(a') \forall b : (\mu[x].c[a := b] \parallel a')^A \in \preceq\}
\]

\[
= \bigwedge_{a', b \in A} \{(a') \forall b : (\bigwedge_{a \in A} \{\neg a : c^A[x := a, \alpha := b] \in \preceq\} \preceq a'\}
\]

\[
= \bigwedge_{a, b \in A} \{(\neg a) \forall b : c^A[x := a, \alpha := b] \in \preceq\}
\]

**Proposition** (\(\lambda\)-calculus). Let \(A^\forall = (A, \preceq, \forall, \neg)\) be a disjunctive structure, and \(A^\rightarrow = (A, \preceq, \rightarrow)\) the implicative structure it canonically defines, we write \(\iota\) for the corresponding inclusion. Let \(t\) be a closed \(\lambda\)-term (with parameter in \(A\)), and \([t]\) his embedding in \(L^\forall\). Then we have

\[
\iota(t^{A^\rightarrow}) = [t]^A^\forall
\]

where \(t^{A^\rightarrow}\) (resp. \(t^{A^\forall}\)) is the interpretation of \(t\) within \(A^\rightarrow\) (resp. \(A^\forall\)).

In other words, this proposition expresses the fact that the following diagram commutes:

\[
\begin{array}{ccc}
\lambda\text{-calculus} & \Rightarrow & L^\forall \\
\downarrow \iota^{A^\rightarrow} & & \downarrow \iota^{A^\forall} \\
(A^\rightarrow, \preceq, \rightarrow) & \Rightarrow & (A^\forall, \preceq, \forall, \neg)
\end{array}
\]

**Proof.** By induction over the structure of terms.

**Case** \(a\) for some \(a \in A^\forall\): This case is trivial as both terms are equal to \(a\).
Case \(\lambda x.u:\) We have \([\lambda x.u] = \mu([x], \alpha).([\alpha] \uparrow \alpha)\) and
\[
(\mu([x], \alpha).([\alpha] \uparrow \alpha))^{A^R} = \bigwedge_{a, b \in A} \{ \neg a \ \forall b : ([t[x := a]]^{A^R}, b) \in \bot \}
\]
\[
= \bigwedge_{a, b \in A} \{ \neg a \ \forall b : [[t[x := a]]^{A^R} \nleq b \}
\]
\[
= \bigwedge_{a \in A} (\neg a \ [[t[x := a]]^{A^R})
\]
On the other hand,
\[
\iota(\lambda x.t)^{A^R} = \iota (\bigwedge_{a \in A} (a \rightarrow ([t[x := a]])^{A^R})) = \bigwedge_{a \in A} (\neg a \ \forall \iota ([t[x := a]])^{A^R})
\]
Both terms are equal since \([[t[x := a]]^{A^R} = (\iota ([t[x := a]])^{A^R}) by induction hypothesis.

Case \(u.v:\)
On the one hand, we have \(\[uv] = \mu(\alpha).([u] \uparrow ([v], \alpha))\) and
\[
(\mu(\alpha).([u] \uparrow ([v], \alpha)))^{A^R} = \bigwedge_{a \in A} \{ a : ([u]^{A^R}, ([v]^{A^R} \ \forall a)) \in \bot \}
\]
\[
= \bigwedge_{a \in A} \{ a : [u]^{A^R} \nleq (\neg[v]^{A^R} \ \forall a) \}
\]
On the other hand,
\[
\iota([u]^{A^R}) = \iota (\bigwedge_{a \in A} \{ a : (\neg a^{A^R} \nleq (v^{A^R} \rightarrow a)) = \bigwedge_{a \in A} \{ a : \iota(\neg a^{A^R} \nleq (\iota(v^{A^R}) \ \forall a) \})
\]
Both terms are equal since \([u]^{A^R} = \iota(u^{A^R})\) and \([v]^{A^R} = \iota(v^{A^R})\) by induction hypotheses.

\(F.\) Disjunctive algebras

Separation in disjunctive structures:

**Definition 80** (Separator). We call separator for the disjunctive structure \(A\) any subset \(S \subseteq A\) that fulfills the following conditions for all \(a, b \in A:\)

1. (upward closure) If \(a \in S\) and \(a \nleq b\) then \(b \in S.\)
2. (combinators) \(S_1, S_2, S_3, S_4\) and \(S_5\) are in \(S.\)
3. (modus ponens) If \(a \rightarrow b \in S\) and \(a \in S\) then \(b \in S.\)

A separator \(S\) is said to be consistent if \(\bot \notin S.\)

**Example 81** (Realizability model). Recall from Example\[14\] that any model of classical realizability based on the \(L^R\)-calculus induces a disjunctive structure. As in the implicative case, the set of formulas realized by a closed term\[^{25}\]

\[S_{\bot} \triangleq \{ a \in P(V_0^+) : a_{\bot} \cap T_0 \neq \emptyset \}\]
defines a valid separator. The conditions (1) and (3) are clearly verified (for the same reasons as in the implicative case), but we should verify that the formulas corresponding to the combinators are indeed realized.

Let us then consider the following closed terms:

\[
\begin{align*}
PS_1 & \triangleq \mu([x], \alpha).([x] \uparrow ([x], \alpha)) \\
PS_2 & \triangleq \mu([x], \alpha).([\alpha_1, \alpha_2].([x \uparrow \alpha_1] \uparrow \alpha)) \\
PS_3 & \triangleq \mu([x], \alpha).([\alpha_1, \alpha_2].([x \uparrow ([x], \alpha)] \uparrow \alpha) \\
PS_4 & \triangleq \mu([x], \alpha).([\beta_1, \beta_2].([\beta_1 \uparrow \gamma \uparrow \alpha]) \uparrow \alpha) \\
PS_5 & \triangleq \mu([x], \alpha).([\beta_1, \beta_2].([\beta_1 \uparrow \gamma \uparrow \alpha, \alpha_1, \alpha_2].([x \uparrow ([\alpha_1, \alpha_2, \alpha_3])] \uparrow \beta] \uparrow \alpha) \\
\end{align*}
\]

**Proposition 82.** The previous terms have the following types in \(L^R:\)

1. \(\vdash PS_1 : \forall A. (A \ \forall A \rightarrow A) \rightarrow A\)
2. \(\vdash PS_2 : \forall AB.A \rightarrow A \ \forall B\)
3. \(\vdash PS_3 : \forall AB. A \ \forall B \rightarrow B \ \forall A\)
4. \(\vdash PS_4 : \forall ABC. (A \rightarrow B) \rightarrow (C \ \forall A \rightarrow C \ \forall B)\)

\[^{25}\]Proof-like terms in \(L^R\) simply correspond to closed terms.
5) \( \vdash PS_S : \forall ABC.(A \triangledown (B \triangledown C)) \rightarrow ((A \triangledown B) \triangledown C) \)

**Proof.** Straightforward typing derivations in \( L^\triangledown \).

We deduce that \( S_\bot \) is a valid separator:

**Proposition 83.** The quintuple \( (P(\forall_0), \preceq, \triangledown, \neg, S_\bot) \) as defined above is a disjunctive algebra.

**Proof.** Conditions (1) and (3) are trivial. Condition (2) follows from the previous proposition and the adequacy lemma for the realizability interpretation of \( L^\triangledown \) (Theorem [15]).

**G. Internal logic**

**Proposition (Preorder).** For any \( a, b, c \in A \), we have:

1] \( a \vdash_S a \) \hspace{1cm} (Reflexivity)

2] if \( a \vdash_S b \) and \( b \vdash_S c \) then \( a \vdash_S c \) \hspace{1cm} (Transitivity)

**Proof.** We first that (2) holds by applying twice the closure by modus ponens, then we use it with the relation \( a \vdash_S a \triangledown a \) and \( a \triangledown a \vdash_S S \) that can be deduced from the combinators \( S_1, S_2 \) to get 1.

**Negation:** We can relate the primitive negation to the one induced by the underlying implicative structure:

**Proposition 84 (Implicative negation).** For all \( a \in A \), the following holds:

1] \( \neg a \vdash_S a \rightarrow \bot \)

2] \( a \rightarrow \bot \vdash_S \neg a \)

**Proof.** The first item follows directly from \( S_2 \) belongs to the separator, since \( a \rightarrow \bot = \neg a \triangledown \bot \) and that \( \neg a \vdash_S \neg a \triangledown \bot \).

For the second item, we use the transitivity with the following hypotheses:

\( (a \rightarrow \bot) \vdash_S a \rightarrow \neg a \) \hspace{1cm} \( (a \rightarrow \neg a) \vdash_S a \rightarrow \neg a \)

The statement on the left hand-side is proved by subtyping from the identity \( \bigvee_{a \in A} (a \rightarrow a) \), which is in \( S \) as the generalized version of \( a \vdash_S a \) above). On the right hand-side, we use twice the modus ponens to prove that \( (a \rightarrow a) \vdash_S (\neg a \rightarrow \neg a) \rightarrow (a \rightarrow \neg a) \rightarrow \neg a \)

The two extra hypotheses are trivially subtypes of the identity again. This statement follows from this more general property (recall that \( a \rightarrow a = \neg a \triangledown a \)):

\( \bigvee_{a,b \in A} ((a \triangledown b) \rightarrow a + b) \in S \)

that we shall prove thereafter (see Proposition [25]).

Additionally, we can show that the principle of double negation elimination is valid with respect to any separator:

**Proposition (Double negation).** For all \( a \in A \), the following holds:

1] \( a \vdash_S \neg \neg a \)

2] \( \neg \neg a \vdash_S a \)

**Proof.** The first item is easy since for all \( a \in A \), we have \( a \rightarrow \neg \neg a = (\neg a) \triangledown a \equiv_S \neg a \triangledown \neg a \triangledown a = a \rightarrow \neg a \). As for the second item, we use Lemma [23] and Proposition [84] to it reduce to the statement:

\( \bigvee_{a \in A} ((\neg a) \rightarrow \bot) \rightarrow a \in S \)

We use again Lemma [23] to prove it, by showing that:

\( \bigvee_{a \in A} ((\neg a) \rightarrow \bot) \rightarrow (\neg a) \rightarrow a \in S \) \hspace{1cm} \( \bigvee_{a \in A} ((\neg a) \rightarrow a) \rightarrow (\neg a) \rightarrow a \in S \)

where the statement on the left hand-side from by subtyping from the identity. For the one on the right hand-side, we use the same trick as in the last proof in order to reduce it to:

\( \bigvee_{a \in A} (a \rightarrow \neg a) \rightarrow (a \rightarrow a) \rightarrow (\neg a \rightarrow a) \rightarrow a \in S \)
Sum type: As in implicative structures, we can define the sum type by:

\[ a + b \triangleq \bigcup_{c \in A} ((a \rightarrow c) \rightarrow (b \rightarrow c) \rightarrow c) \quad (\forall a, b \in A) \]

We can prove that the disjunction and this sum type are equivalent from the point of view of the separator:

**Proposition (Implicative sum type).** For all \( a, b \in A \), the following holds:

1. \( a \nabla b \vdash_S a + b \)
2. \( a + b \vdash_S a \nabla b \)

**Proof.** We prove in both cases a slightly more general statement, namely that the meet over all \( a, b \) or the corresponding implication belongs to the separator. For the first item, we have:

\[ \bigcup_{a, b \in A} (a \nabla b) \rightarrow a + b = \bigcup_{a, b \in A} (a \nabla b) \rightarrow (a \rightarrow c) \rightarrow (b \rightarrow c) \rightarrow c \]

Swapping the order of the arguments, we prove that \( \bigcup_{a, b, c \in A} (b \rightarrow c) \rightarrow (a \nabla b) \rightarrow (a \rightarrow c) \rightarrow c \in S \). For this, we use Lemma 23 and the fact that:

\[ \bigcup_{a, c \in A} (a \rightarrow c) \rightarrow (a \nabla c) \rightarrow c \in S \]

The left hand-side statement is proved using \( S^3_1 \), while on the right hand-side we prove it from the fact that:

\[ \bigcup_{a, c \in A} (a \rightarrow c) \rightarrow (a \nabla c) \rightarrow c \nabla c \in S \]

which is a subtype of \( S^3_1 \), by using Lemma 23 again with \( S^3_1 \) and by manipulation on the order of the argument.

The second item is easier to prove, using Lemma 23 again and the fact that:

\[ \bigcup_{a, b \in A} (a + b) \rightarrow (a \rightarrow (a \nabla b)) \rightarrow (b \rightarrow (a \nabla b)) \rightarrow (a \nabla b) \in S \]

which is a subtype of \( I^A \) (which belongs to \( S \)). The other part, which is to prove that:

\[ \bigcup_{a, b \in A} ((a \rightarrow (a \nabla b)) \rightarrow (b \rightarrow (a \nabla b)) \rightarrow (a \nabla b) \in S \]

follows from Lemma 20 and the fact that \( \bigcup_{a, b \in A} (a \rightarrow (a \nabla b)) \) and \( \bigcup_{a, b \in A} (b \rightarrow (a \nabla b)) \) are both in the separator.

**H. Induced implicative algebras**

**Proposition 85 (Combinator \( K^A \)).** We have \( K^A \in S \).

**Proof.** This directly follows by upwards closure from the fact that \( \bigcup_{a, b \in A} (a \rightarrow (b \nabla a)) \in S \).

**Proposition 86 (Combinator \( S^A \)).** For any disjunctive algebra \((A, \leq, \nabla, \neg, S)\), we have \( S^A \in S \).

**Proof.** See Section [B] We make several applications of Lemmas 23 and 20 consecutively. We prove that:

\[ \bigcup_{a, b, c \in A} ((a \rightarrow (a \rightarrow b) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow b \rightarrow a \rightarrow c) \in S \]

is implied by Lemma 23 and:

\[ \bigcup_{a, b, c \in A} ((a \rightarrow (a \rightarrow b) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow b \rightarrow a \rightarrow c) \in S \]

The statement on the left hand-side is an ad-hoc lemma, while the other is proved by generalized transitivity (Lemma 20), using a subtype of \( S^3_1 \) as hypothesis, from:

\[ \bigcup_{a, b, c \in A} ((a \rightarrow b) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow c) \in S \]
The latter is proved, using again generalized transitivity with a subtype of $s^3_4$ as premise, from:

$$\bigwedge_{a,b,c \in A} (a \to a \to c) \to (a \to c) \in S$$

This is proved using again Lemmas 23 and 20 with $s^2_5$ and a variant of $s^3_4$. □

**Proposition 87** (Combinator CC$^A$). We have CC$^A \in S$.

**Proof.** We make several applications of Lemmas 23 and 20 consecutively. We prove that:

$$\bigwedge_{a,b \in A} ((a \to b) \to a) \to a \in S$$

is implied by generalized modus ponens (Lemma 23) and:

$$\bigwedge_{a,b \in A} ((a \to b) \to a) \to (\neg a \to a \to b) \to \neg a \to a \in S$$

and

$$\bigwedge_{a,b \in A} ((\neg a \to a \to b) \to \neg a \to a) \to a \in S$$

The statement above is a subtype of $s^3_4$, while the other is proved, by Lemma 23, from:

$$\bigwedge_{a,b \in A} ((\neg a \to a \to b) \to \neg a \to a) \to \neg a \to a \in S$$

and

$$\bigwedge_{a \in A} (\neg a) \to a \in S$$

The statement below is proved as in Proposition 24, while the statement above is proved by a variant of the modus ponens and:

$$\bigwedge_{a,b \in A} (\neg a \to a \to b) \in S$$

We conclude by proving this statement using the connections between $\neg a$ and $a \to \bot$, reducing the latter to:

$$\bigwedge_{a,b \in A} (a \to \bot) \to a \to b \in S$$

which is a subtype of the identity. □
A. Embedding of the $\lambda$-calculus

Guided by the expected definition of the arrow $A \to B \triangleq \neg(A \otimes \neg B)$, we can follow Munch-Maccagnoni's paper [32, Appendix E], to embed the $\lambda$-calculus into $L^\otimes$.

With such a definition, a stack $u \cdot e$ in $A \to B$ (that is with $u$ a term of type $A$ and $e$ a context of type $B$) is naturally embedded as a term $(u, [e])$, which is turned into the context $\mu[\alpha]$. $(u, [e]) \parallel \alpha$ which indeed inhabits the “arrow” type $\neg(A \otimes \neg B)$. The rest of the definitions are then direct:

$$\mu(x, [\alpha], c \triangleq \mu(x, x'), x' \parallel \mu[\alpha], c)$$

$$\lambda x.t \triangleq [\mu(x, [\alpha]), (t \parallel \alpha)]$$

These shorthands allow for the expected typing rules:

**Proposition 88.** The following typing rules are admissible:

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B} \quad \frac{\Gamma \vdash u : A \mid \Delta \quad \Gamma \vdash e : B \mid \Delta}{\Gamma \vdash \mu(x, [\alpha], c \parallel e) \parallel \Delta} \quad \frac{\Gamma \vdash t : A \mid \Delta \quad \Gamma \vdash u : A \mid \Delta}{\Gamma \vdash tu : B \mid \Delta}$$

**Proof.** Each case is directly derivable from $L^\otimes$ type system. We abuse the notations to denote by (def) a rule which simply consists in unfolding the shorthands defining the $\lambda$-terms.

a) **Case $\mu(x, [\alpha], c)$:**

$$\frac{c : (\Gamma, x : A \to \Delta, \alpha : B)}{\Gamma \vdash \mu[x].c \parallel \neg A \mid \Delta, \beta : B} \quad \frac{\Gamma, x : A, x' : \neg B \vdash x' : \neg B \mid \Delta}{\Gamma \vdash \mu(x, x').x' \parallel \mu[\alpha], c) \parallel A \otimes \neg B \mid \Delta}$$

b) **Case $\lambda x.t$:**

$$\frac{\Gamma \vdash \mu(x, [\beta]).(t \parallel \beta) \parallel A \otimes \neg B \mid \Delta}{\Gamma \vdash \lambda x.t : A \to B \mid \Delta}$$

c) **Case $u \cdot e$:**

$$\frac{\Gamma \vdash u : A \mid \Delta}{\Gamma \vdash \langle u, [e] \rangle : A \otimes \neg B \mid \Delta} \quad \frac{\Gamma \vdash e : B \mid \Delta}{\Gamma \vdash \mu[\alpha], \langle u, [e] \rangle \parallel \alpha \parallel \neg (A \otimes \neg B) \mid \Delta}$$

d) **Case $tu$:**

$$\frac{\Gamma \vdash t : A \to B \mid \Delta}{\Gamma \vdash \mu \alpha, (t \parallel \alpha) : B \mid \Delta}$$

Besides, the usual rules of $\beta$-reduction for the call-by-value evaluation strategy are simulated through the reduction of $L^\otimes$:
Proposition 89 (β-reduction). We have the following reduction rules:

\[
\begin{align*}
(t \parallel e) & \rightarrow_\beta (t \parallel u \cdot e) \\
(\lambda x.t \parallel u \cdot e) & \rightarrow_\beta (u \parallel \mu x.(t \parallel e)) \\
(V \parallel \mu x.e) & \rightarrow_\beta \epsilon[V/x]
\end{align*}
\]

Proof. The third rule is included in \(L^\circ\) reduction system, the first follows from:

\[
\langle tu \parallel e \rangle = \langle \mu a.(t \parallel u \cdot a) \parallel e \rangle \rightarrow_\beta \langle t \parallel u \cdot e \rangle
\]

For the second rule, we first check that we have:

\[
\langle (V, [e]) \parallel \mu(x, [\alpha]).c \rangle = \langle (V, [e]) \parallel \mu(x, x').(x' \parallel \mu[\alpha].c) \rangle \rightarrow_\beta \langle [e] \parallel \mu[\alpha].c[V/X] \rangle \rightarrow_\beta c[V/X][e/\alpha]
\]

from which we deduce:

\[
\langle \lambda x.t \parallel u \cdot e \rangle = \langle [\mu(x, [\alpha]).(t \parallel \alpha)] \parallel \mu[\alpha].((u, [e]) \parallel \alpha) \rangle \\
\rightarrow_\beta \langle (u, [e]) \parallel \mu(x, [\alpha]).(t \parallel \alpha) \rangle \\
\rightarrow_\beta \langle u \parallel \mu y.(y, [e]) \parallel \mu(x, [\alpha]).(t \parallel \alpha) \rangle \\
\rightarrow_\beta \langle u \parallel \mu x.(t \parallel e) \rangle
\]

\[\Box\]

B. Conjunctive structures

We shall now introduce the notion of conjunctive structure. Following the methodology from the previous chapter, we begin by observing the existing commutations in the realizability models induced by \(L^\circ\). Since we are in a structure centered on positive connectives, we should pay attention to the commutations with joins:

Proposition (Commutations). In any \(L^\circ\) realizability model, if \(X \notin FV(B)\) the following equalities hold:

1) \(\exists X.(A \otimes B)[V] = (\exists X.A) \otimes B[V]\).
2) \(\exists X.(B \otimes A)[V] = B \otimes (\exists X.A)[V]\).
3) \(\neg(\exists X.A)[V] = \bigcap_{S \in \mathcal{P}(\mathcal{V})} \neg A\{X := \hat{S}\}[V]\)

Proof. 1) Assume the \(X \notin FV(B)\), then we have:

\[
\exists X.(A \otimes B)[V] = S \in \mathcal{P}(\mathcal{V}) \cup A\{X := \hat{S}\} \otimes B[V]
\]

\[
S \in \mathcal{P}(\mathcal{V}) \cup \{V_1, V_2 : V_1 \in A\{X := \hat{S}\}[V] \land V_2 \in B[V]\}
\]

\[
= \{\epsilon_1, \epsilon_2 : \epsilon_1 \in S \in \mathcal{P}(\mathcal{V}) \cup A\{X := \hat{S}\}[V] \land \epsilon_2 \in B[V]\}
\]

\[
= \{\epsilon_1, \epsilon_2 : \epsilon_1 \in \exists X.A[V] \land \epsilon_2 \in B[V]\}
\]

2) Identical.

3) The proof is again a simple unfolding of the definitions:

\[
\neg(\exists X.A)[V] = \{[t] : t \in \exists X.A[V]\} = \{[t] : t \in \bigcap_{S \in \mathcal{P}(\mathcal{V})} A\{X := \hat{S}\}[V]\}
\]

\[
= \bigcap_{S \in \mathcal{P}(\mathcal{V})} \{[t] : t \in A\{X := \hat{S}\}[V]\} = \bigcap_{S \in \mathcal{P}(\mathcal{V})} \neg A\{X := \hat{S}\}[V]
\]

\[\Box\]

Example 90 (Realizability models). As for the disjunctive case, we can abstract the structure of the realizability interpretation of \(L^\circ\) into a structure of the form \((\mathcal{T}_0, \mathcal{E}_0, \mathcal{V}_0, (\cdot, \cdot), [], \sqsubseteq)\), where \((\cdot, \cdot)\) is a map from \(\mathcal{T}_0^2\) to \(\mathcal{T}_0\) (whose restriction to \(\mathcal{V}_0\) has values in \(\mathcal{V}_0\)), \([]\) is an operation from \(\mathcal{E}_0\) to \(\mathcal{V}_0\), and \(\sqsubseteq \subseteq \mathcal{T}_0 \times \mathcal{E}_0\) is a relation. From this sextuple we can define:

- \(\mathcal{A} \triangleq \mathcal{P}(\mathcal{V}_0)\)
- \(a \otimes b \triangleq (a, b) = \{(V_1, V_2) : V_1 \in a \land V_2 \in b\}\)
- \(\lnot a \triangleq [a] = \{[e] : e \in a^\bot\}\)

\[\forall a, b \in \mathcal{A}\]

Proposition 91. The quadruple \((\mathcal{A}, \sqsubseteq, \otimes, \lnot)\) is a conjunctive structure.

Proof. We show that the axioms of Definition 30 are satisfied.

1) Anti-monotonicity. Let \(a, a' \in \mathcal{A}\), such that \(a \sqsubseteq a'\) ie \(a \subseteq a'\). Then \(a^\bot \subseteq a'^\bot\) and thus

\[
\lnot a' = \{[t] : t \in a'^\bot\} \subseteq \{[t] : t \in a^\bot\} = \lnot a
\]
2) Covariance of the conjunction. Let \( a, a', b, b' \in \mathcal{A} \) such that \( a' \subseteq a \) and \( b' \subseteq b \). Then we have
\[
a \otimes b = \{(t, u) : t \in a \land u \in b\} \subseteq \{(t, u) : t \in a' \land u \in b'\} = a' \otimes b'
\]
i.e. \( a \otimes b \preccurlyeq a' \otimes b' \)

3) Distributivity. Let \( a \in \mathcal{A} \) and \( B \subseteq \mathcal{A} \), we have:
\[
\bigwedge_{b \in B} (a \otimes b) = \bigwedge_{b \in B} \{(v, u) : t \in a \land u \in b\} = \{(t, u) : t \in a \land u \in \bigwedge_{b \in B} b\} = a \otimes (\bigwedge_{b \in B} b)
\]

4) Commutation. Let \( B \subseteq \mathcal{A} \), we have (recall that \( \bigwedge_{b \in B} b = \bigcap_{b \in B} b \)):
\[
\bigwedge_{b \in B} \{\neg b\} = \bigwedge_{b \in B} \{[t] : t \in b\} = \{[t] : t \in \bigwedge_{b \in B} b\} = \{[t] : t \in (\bigwedge_{b \in B} b)\} = \neg (\bigwedge_{b \in B} b)
\]

C. Interpreting \( L^\otimes \) terms

We shall now see how to embed \( L^\otimes \) commands, contexts and terms into any conjunctive structure. For the rest of the section, we assume given a conjunctive structure \( (\mathcal{A}, \preccurlyeq, \otimes) \).

1) Commands: Following the same intuition as for the embedding of \( L^\oplus \) into disjunctive structures, we define the commands \( \langle a \mid b \rangle \) of the conjunctive structure \( \mathcal{A} \) as the pairs \( (a, b) \), and we define the pole \( \bot \) as the ordering relation \( \preccurlyeq \). We write \( \mathcal{C}_A = \mathcal{A} \times \mathcal{A} \) for the set of commands in \( \mathcal{A} \) and \( (a, b) \in \bot \) for \( a \preccurlyeq b \).

We consider the same relation \( \preccurlyeq \) over \( \mathcal{C}_A \), which was defined by:
\[
c \preccurlyeq c' \iff \text{ if } c \in \bot \text{ then } c' \in \bot \quad (\forall c, c' \in \mathcal{C}_A)
\]

Since the definition of commands only relies on the underlying lattice of \( \mathcal{A} \), the relation \( \preccurlyeq \) has the same properties as in disjunctive structures and in particular it defines a preorder (see Section B-C1).

2) Terms: The definitions of terms are very similar to the corresponding definitions for the dual contexts in disjunctive structures.

Definition 92. pairing[Pairing] For all \( a, b \in \mathcal{A} \), we let \( (a, b) \triangleq a \otimes b \).

Definition 93. box[Boxing] For all \( a \in \mathcal{A} \), we let \( [a] \triangleq \neg a \).

Definition 94. map[\( \mu^+ \)]
\[
\mu^+.c \triangleq \bigcup_{a \in \mathcal{A}} \{a : c(a) \in \bot\}
\]

We have the following properties for \( \mu^+. \), whose proofs are trivial:

Proposition 95 (Properties of \( \mu^+ \)). For any functions \( c, c' : \mathcal{A} \to \mathcal{C}_A \), the following hold:

1. If for all \( a \in \mathcal{A}, c(a) \preccurlyeq c'(a) \), then \( \mu^+.c \preccurlyeq \mu^+.c' \) (Variance)
2. For all \( t \in \mathcal{A} \), then \( t = \mu^+. (a \mapsto (t \upharpoonright a)) \) (\( \eta \)-expansion)
3. For all \( e \in \mathcal{A} \), then \( \mu^+.c \upharpoonright e \preccurlyeq c(e) \) (\( \beta \)-reduction)

Proof. 1) Direct consequence of Proposition 65
2,3. Trivial by definition of \( \mu^+ \).

3) Contexts: Dually to the definitions of the (positive) contexts \( \mu^+ \) as a meet, we define the embedding of (negative) terms, which are all binders, by arbitrary joins:

Definition 96 (\( \mu^- \)). For all \( c : \mathcal{A} \to \mathcal{C}_A \), we define:
\[
\mu^-.c \triangleq \bigcap_{a \in \mathcal{A}} \{a : c(a) \in \bot\}
\]

Definition 97 (\( \mu^0 \)). For all \( c : \mathcal{A}^2 \to \mathcal{C}_A \), we define:
\[
\mu^0.c \triangleq \bigcap_{a,b \in \mathcal{A}} \{a \otimes b : c(a, b) \in \bot\}
\]
Definition 98 ($\mu^\|_\| A$). For all $c : A \rightarrow C_A$, we define:

$$\mu^\|_\| c \triangleq \bigwedge_{a \in A} \{ \neg a : c(a) \in \bot \}$$

Again, these definitions satisfy variance properties with respect to the preorder $\preceq$ and the order relation $\preceq$. Observe that the $\mu^\|$ and $\mu^\|$-binders, which are negative binders catching positive terms, are contravariant with respect to these relations while the $\mu^\|$-binder, which catches a negative context, is covariant.

Proposition 99 (Variance). For any functions $c, c'$ with the corresponding arities, the following hold:

1. If $c(a) \preceq c'(a)$ for all $a \in A$, then $\mu^\|_\| c' \ll \mu^\|_\| c$.
2. If $c(a, b) \preceq c'(a, b)$ for all $a, b \in A$, then $\mu^\|_\| c' \ll \mu^\|_\| c$.
3. If $c(a) \preceq c'(a)$ for all $a \in A$, then $\mu^\|_\| c \ll \mu^\|_\| c'$.

Proof. Direct consequences of Proposition 65.

The $\eta$-expansion is also reflected by the ordering relation $\preceq$:

Proposition 100 ($\eta$-expansion). For all $t \in A$, the following holds:

1. $\mu^\|_\| (a \mapsto \langle t \| a \rangle) = t$.
2. $\mu^\|_\| (a, b \mapsto \langle t \| (a, b) \rangle) \ll t$.
3. $\mu^\|_\| (a \mapsto \langle t \| [a] \rangle) \ll t$.

Proof. Trivial from the definitions.

The $\beta$-reduction is again reflected by the preorder $\preceq$ as the property of subject reduction:

Proposition 101 ($\beta$-reduction). For all $e, e_1, e_2, t \in A$, the following holds:

1. $\langle \mu_\| c \| e \rangle \preceq c(e)$.
2. $\langle \mu^\|_\| c \| (e_1, e_2) \rangle \preceq c(e_1, e_2)$.
3. $\langle \mu^\|_\| c \| [t] \rangle \preceq c(t)$.

Proof. Trivial from the definitions.

D. Adequacy

We shall now prove that the interpretation of $L^\$ is adequate with respect to its type system. Again, we extend the syntax of formulas to define second-order formulas with parameters by:

$$A, B ::= a \mid X \mid A \otimes B \mid \exists X.A$$

This allows us to define an embedding of closed formulas with parameters into the conjunctive structure $A$;

$$a^A \triangleq a \quad (\neg A)^A \triangleq \neg A^A \quad (A \otimes B)^A \triangleq A^A \otimes B^A \quad (\exists X.A)^A \triangleq \bigwedge_{a \in A} (A[X := a])^A$$

As in the previous chapter, we define substitutions, which we write $\sigma$, as functions mapping variables (of terms, contexts, commands) and formulas: for all contexts $\Gamma$, for all formulas $\sigma$ such that $\sigma \vdash \Gamma$, we have $\sigma(x) \preceq (A[\sigma])^A$. Dually, we say that $\sigma$ realizes $\Delta$ if for all bindings $\langle \alpha : A \rangle \in \Delta$ we have $\sigma(\alpha) \preceq (A[\sigma])^A$.

Theorem 102 (Adequacy). The typing rules of $L^\$ (Figure 2) are adequate with respect to the interpretation of terms (contexts, commands) and formulas: for all contexts $\Gamma, \Delta$, for all formulas with parameters $A$ and for all substitutions $\sigma$ such that $\sigma \vdash \Gamma$ and $\sigma \vdash \Delta$, we have:

1. For any term $t$, if $\Gamma \vdash t : A \mid \Delta$, then $(t[\sigma])^A \ll A[\sigma]^A$;
2. For any context $e$, if $\Gamma \vdash e : A \mid \Delta$, then $(e[\sigma])^A \gg A[\sigma]^A$;
3. For any command $c$, if $c : (\Gamma \mid \Delta)$, then $(c[\sigma])^A \in \bot$.

Proof. By induction on the typing derivations. Since most of the cases are similar to the corresponding cases for the adequacy of the embedding of $L^\$ into disjunctive structures, we only give some key cases.
Case ($\vdash \otimes$): Assume that we have:

$$
\frac{\Gamma \vdash t_1 : A_1 | \Delta \quad \Gamma \vdash t_2 : A_2 | \Delta}{\Gamma \vdash (t_1, t_2) : A_1 \otimes A_2 | \Delta \quad (\vdash \otimes)}
$$

By induction hypotheses, we have that $(t_1[\sigma])^A \preceq (A_1[\sigma])^A$ and $(t_2[\sigma])^A \preceq (A_2[\sigma])^A$. Therefore, by monotonicity of the $\otimes$ operator, we have:

$$
((t_1, t_2)[\sigma])^A = (t_1[\sigma], t_2[\sigma])^A = (t_1[\sigma])^A \otimes (t_2[\sigma])^A \preceq (A_1[\sigma])^A \otimes (A_2[\sigma])^A.
$$

Case ($\otimes \vdash$): Assume that we have:

$$
\frac{c : \Gamma, x_1 : A_1, x_2 : A_2 \vdash \Delta}{\Gamma \mid \mu(x_1, x_2), c : A_1 \otimes A_2 \vdash \Delta \quad (\otimes \vdash)}
$$

By induction hypothesis, we get that $(c[\sigma, x_1 \mapsto (A_1[\sigma])^A, x_2 \mapsto (A_2[\sigma])^A])^A \in \bot$. Then by definition we have

$$
((\mu(x_1, x_2).c)[\sigma])^A = \bigwedge_{a, b \in A} \{ a \not\forall b : (c[\sigma, x_1 \mapsto a, x_2 \mapsto b])^A \in \bot \} \preceq (A_1[\sigma])^A \otimes (A_2[\sigma])^A.
$$

Case ($\exists \vdash$): Assume that we have:

$$
\frac{\Gamma \mid e : A \vdash \Delta \quad X \notin FV(\Gamma, \Delta)}{\Gamma \mid e : \exists X.A \vdash \Delta \quad (\exists \vdash)}
$$

By induction hypothesis, we have that for all $a \in A$, $(e[\sigma])^A \preceq ((A)[\sigma, x \mapsto a])^A$. Therefore, we have that $(e[\sigma])^A \preceq \bigwedge_{a \in A} (A( X := a)[\sigma])^A$.

Case ($\vdash \exists$): Similarly, assume that we have:

$$
\frac{\Gamma \mid t : A \{ X := B \} | \Delta}{\Gamma \mid t : \exists X.A | \Delta \quad (\vdash \exists)}
$$

By induction hypothesis, we have that $(t[\sigma])^A \preceq (A[\sigma, X \mapsto (B[\sigma])^A])^A$. Therefore, we have that $(t[\sigma])^A \preceq \bigwedge_{b \in A} (A \{X := b\}[\sigma])^A$.

\textbf{E. Conjunctive algebras}

If we analyse the tensorial calculus underlying $L^\otimes$ type system and try to inline all the typing rules involving commands and contexts within the one for terms, we are left with the following four rules:

$$
\frac{\Gamma, A \vdash A}{\Gamma \vdash A \otimes B} \quad \frac{\Gamma \vdash B}{\Gamma \vdash A \otimes B} \quad \frac{\Gamma, A, B \vdash C}{\Gamma \vdash (A \otimes B) \rightarrow C} \quad \frac{\Gamma, A \vdash C}{\Gamma, A \vdash \neg C} \quad \frac{\Gamma, A \vdash \neg a}{\Gamma \vdash \neg a}
$$

\textbf{Induced Heyting algebra:} As in the implicational case, the entailment relation induces a structure of (pre)Heyting algebra, whose conjunction and disjunction are naturally given $a + b \triangleq \neg (\neg a \otimes \neg b)$ and $a \times b \triangleq a \otimes b$.

\textbf{Proposition 103} (Heyting Algebra). For any $a, b, c \in A$ For any $a, b, c \in A$, we have:

\begin{align*}
1. & \quad a \times b \vdash_S a \\
2. & \quad a \times b \vdash_S b \\
3. & \quad a \vdash_S a + b \\
4. & \quad b \vdash_S a + b \\
5. & \quad a \vdash_S b \Rightarrow c \quad \text{if and only if} \quad a \times b \vdash_S c
\end{align*}

\textbf{Conjunctive tripos: We will need the following lemma:}

\textbf{Lemma 104} (Adjunction). If $a \trianglelefteq b \Rightarrow c$ then $a \trianglelefteq \neg c$.

In order to obtain a conjunctive tripos, we define:

$$
\exists \frac{a_i}{i \in I} \trianglelefteq \bigwedge_{i \in I} a_i \quad \forall \frac{a_i}{i \in I} \trianglelefteq \neg (\bigvee_{i \in I} a_i)
$$

\textbf{Theorem} (Conjunctive tripos). Let $(A, \trianglelefteq, \rightarrow, S)$ be a classical conjunctive algebra. The following functor (where $f : J \rightarrow I$):

$$
\mathcal{T} : I \mapsto A^I / S[I] \quad \mathcal{T}(f) : \left\{ \begin{array}{c} A^I / S[I] \rightarrow A^J / S[J] \\
([a_i]_{i \in I}) \mapsto ([a_{f(i)}]_{j \in J}) \end{array} \right.
$$

defines a tripos.

\textbf{Proof.} The proof mimicks the proof in the case of implicative algebras, see Section \textit{C-E}.
We verify that $\mathcal{T}$ satisfies all the necessary conditions to be a tripos.

- The functoriality of $\mathcal{T}$ is clear.
- For each $I \in \text{Set}$, the image of the corresponding diagonal morphism $\mathcal{T}(\delta_I)$ associates to any element $[(a_{ij})_{i,j} \in I \times I] \in \mathcal{T}(I \times I)$ the element $[(a_i)_{i} \in I] \in \mathcal{T}(I)$. We define

$$
(=_{I}) : i,j \mapsto \begin{cases} 
\bigwedge_{a \in A} (\lambda a : a \to a) & \text{if } i = j \\
\bot \to \top & \text{if } i \neq j
\end{cases}
$$

and we need to prove that for all $[a] \in \mathcal{T}(I \times I)$:

$$
\lceil [a] \rceil_I \sqsubseteq [S[I]] \mathcal{T}(\delta_I)(a) \iff [a] \sqsubseteq [S[I \times I]] [a]
$$

Let then $[(a_{ij})_{i,j} \in I]$ be an element of $\mathcal{T}(I \times I)$.

From left to right, assume that $\lceil [a] \rceil_I \sqsubseteq [S[I]] \mathcal{T}(\delta_I)(a)$. That is to say that there exists $s \in S$ such that for any $i \in I$, $s \sqsubseteq \top \to a_{ii}$. We would like to reproduce the proof in the implicic case, which uses $\lambda z.(\lambda x.x)$. Here, due to the double-negation induced by the application (see Section [IV-H], we can only show that:

$$\lambda z.z(s(\lambda x.x)) \not\equiv i = j \to \neg
$$

Indeed, if $i \neq j$, we have that:

$$
\lambda z.z(s(\lambda x.x)) \not\equiv (\top \to \top) \to \neg \neg \neg \neg a_{ij}
$$

the last one being true by assumption. We conclude using the fact that any $\lambda$-terms with parameters in $S$ belongs to $S$ using a slight variant of Theorem 46.

From right to left, if there exists $s \in S$ such that for any $i,j \in I$, $\bar{s} \equiv i = j \to a_{ij}$, then in particular for all $i$ $\lambda a : a \to a \vdash a_{ii}$. We use the transitivity of $\vdash$ to show that $\top \vdash a_{ij}$ follows from $\lambda a : a \to a \vdash a_{ii}$ and $\bot \vdash \lambda a : a \to a$.

Writing $Id$ for $\lambda a : a \to a$, the latter is obtained by using the deduction rule:

$$
\vdash \neg(Id \otimes (\top \otimes Id)) \vdash Id
$$

Using $s^0_\top$ to get $\neg(Id \otimes (\top \otimes Id))$ from $\neg(\neg(Id \otimes \top) \otimes Id)$ which follows from $s^0_\top$.

- For each projection $\pi_{1 \times J}^{1} : I \times J \to I$ in $C$, the monotone function $\mathcal{T}(\pi_{1 \times J}^{1}) : \mathcal{T}(I) \to \mathcal{T}(I \times J)$ has both a left adjoint $(\exists J)_I$ and a right adjoint $(\forall J)_I$ which are defined by:

$$
(\forall J)_I \{ ([a_{ij})_{i,j} \in I \times J] \} \triangleq \{ \left( \bigwedge_{j \in J} a_{ij} \right)_{i \in I} \} \\
(\exists J)_I \{ ([a_{ij})_{i,j} \in I \times J] \} \triangleq \{ \left( \bigvee_{j \in J} a_{ij} \right)_{i \in I} \}
$$

We only give the case of $\forall$, the case for $\exists$ is easier (it corresponds to [this] and [this] Coq lemmas). We need to show that for any $[b_{ij})_{i,j} \in I \times J$, $\mathcal{T}(\pi_{1 \times J}^{1}) \{ ([a_{ij})_{i,j} \in I \times J] \} \iff \mathcal{T}(\pi_{1 \times J}^{1}) \{ ([b_{ij})_{i,j} \in I \times J] \} \iff \mathcal{T}(\pi_{1 \times J}^{1}) \{ ([a_{ij})_{i \in I} \iff \mathcal{T}(\pi_{1 \times J}^{1}) \{ ([b_{ij})_{i \in I}] \}

Let us fix some $[a]$ and $[b]$ as above.

From left to right, assume that for all $i \in I$, $j \in J$, $a_{ij} \vdash b_{ij}$, we want to prove that $\forall i \in I$, we have $a_{i} \vdash \neg \forall j \in J \neg b_{ij}$. We first show that for any $a, b, c$, the following rule is valid (it mainly amount to $s^0_\top$):

$$
\frac{a \vdash b}{\neg (c \otimes b) \in S} \\
\frac{\neg (c \otimes a) \in S}{a \vdash b}$$
and prove instead that \(\neg \forall j \in J \neg b_{ij} \vdash \forall j \in J \neg b_{ij} \) and \(\neg (a_i \otimes \forall j \in J \neg b_{ij}) \in S\). The former amount to \(\neg \neg \neg a \vdash a\) while we can use commutation rule on the latter to rewrite it as: \(\bigvee_{j \in J} \neg (a_i \otimes \neg b_{ij}) \in S\) which follows from the assumption. From right to left the processus is almost the same and relies on the fact we have for all \(i \in I, j \in J\), \((a_i \stackrel{\equiv}{\Rightarrow} b_{ij}) \in S\) if and only if for all \(i \in I\), \(\bigvee_{j \in J} (a_i \Rightarrow b_{ij}) \in S\) if and only if for all \(i \in I\), \((a_i \otimes \neg b_{ij}) \in S\). We then use the same lemma with the reverse law \(a \vdash \neg \neg a\).

- These adjoints clearly satisfy the Beck-Chevalley condition as in the implicative cases.
- Finally, we define \(\text{Prop} \triangleq \mathcal{A}\) and verify that \(\text{tr} \triangleq [\text{id}_\mathcal{A}] \in T(\text{Prop})\) is a generic predicate, as in the implicative case.

\(\square\)
APPENDIX D
THE DUALITY OF COMPUTATION

Proposition. Let \((\mathcal{A}, \preceq, \triangledown, \neg)\) be a disjunctive structure. Let us define:

\[
\begin{align*}
\mathcal{A}^\odot & \triangleq \mathcal{A}^\triangledown \\
\mathcal{A} & \triangleq \lambda^\triangledown \\
\mathcal{A} & \triangleq \lambda^\triangledown \\
\mathcal{A} & \triangleq \lambda^\triangledown \\
\mathcal{A}^\odot & \triangleq \mathcal{A}^\triangledown
\end{align*}
\]

then \((\mathcal{A}^\odot, \preceq, \triangledown, \neg)\) is a conjunctive structure.

**Proof.** We check that for all \(a, a', b, b' \in \mathcal{A}\) and for all subsets \(A \subseteq \mathcal{A}\), we have:

1. If \(a \preceq a'\) then \(\neg a' \preceq \neg a\)  
   (Variance)
2. If \(a \preceq a'\) and \(b \preceq b'\) then \(a \odot b \preceq a' \odot b'\).  
   (Variance)
3. \((\bigwedge_{a \in A} a) \odot b = \bigwedge_{a \in A} (a \odot b)\) and \(b \odot (\bigwedge_{a \in A} a) = \bigwedge_{a \in A} (b \odot a)\)  
   (Distributivity)
4. \(\neg (\bigwedge_{a \in A} a) = \bigwedge_{a \in A} (\neg a)\)  
   (Commutation)

All the proofs are trivial from the corresponding properties of disjunctive structures.

□

Proposition. Let \((\mathcal{A}, \preceq, \triangledown, \neg)\) be a conjunctive structure. Let us define:

\[
\begin{align*}
\mathcal{A}^\triangledown & \triangleq \mathcal{A}^\odot \\
\mathcal{A}^\triangledown & \triangleq \lambda^\odot \\
\mathcal{A}^\triangledown & \triangleq \lambda^\odot \\
\mathcal{A}^\triangledown & \triangleq \lambda^\odot \\
\mathcal{A}^\triangledown & \triangleq \lambda^\odot
\end{align*}
\]

then \((\mathcal{A}^\triangledown, \preceq, \triangledown, \neg)\) is a disjunctive structure.

**Proof.** We check that for all \(a, a', b, b' \in \mathcal{A}\) and for all subsets \(A \subseteq \mathcal{A}\), we have:

1. If \(a \preceq a'\) then \(\neg a' \preceq \neg a\).  
   (Variance)
2. If \(a \preceq a'\) and \(b \preceq b'\) then \(a \triangledown b \preceq a' \triangledown b'\).  
   (Variance)
3. \((\bigwedge_{a \in A} a) \triangledown b = \bigwedge_{a \in A} (a \triangledown b)\) and \(a \triangledown (\bigwedge_{a \in A} b) = \bigwedge_{a \in A} (a \triangledown b)\)  
   (Distributivity)
4. \(\neg (\bigwedge_{a \in A} a) = \bigwedge_{a \in A} (\neg a)\)  
   (Commutation)

All the proofs are trivial from the corresponding properties of conjunctive structures.

□

Theorem. Let \((\mathcal{A}^\odot, \mathcal{S}^\odot)\) be a conjunctive algebra, the set:

\[
\mathcal{S}^\triangledown \triangleq \neg^{-1}(\mathcal{S}^\odot) = \{a \in \mathcal{A} : \neg a \in \mathcal{S}^\odot\}
\]

is a valid separator for the dual disjunctive structure \(\mathcal{A}^\triangledown\).

Theorem. Let \((\mathcal{A}^\triangledown, \mathcal{S}^\triangledown)\) be a disjunctive algebra. The set:

\[
\mathcal{S}^\odot \triangleq \neg^{-1}(\mathcal{S}^\triangledown) = \{a \in \mathcal{A} : \neg a \in \mathcal{S}^\triangledown\}
\]

is a classical separator for the dual conjunctive structure \(\mathcal{A}^\odot\).

**Proof.** Both proofs rely on the fact that:

\[
a \vdash_{\mathcal{S}^\odot} b \iff \neg a \vdash_{\mathcal{S}^\triangledown} \neg b \quad \text{and} \quad a \vdash_{\mathcal{S}^\triangledown} b \iff \neg a \vdash_{\mathcal{S}^\odot} \neg b
\]

In particular, to prove that the modus ponens is valid when passing from \(\mathcal{A}^\odot\) to \(\mathcal{A}^\triangledown\), we need to show that if \(a, a \rightarrow b \in \neg^{-1}(\mathcal{S}^\odot)\), then \(b \in \neg^{-1}(\mathcal{S}^\triangledown)\) i.e. \(\neg b \in \mathcal{S}^\triangledown\). By hypothesis, we thus have that \(\neg a \rightarrow \neg b \in \mathcal{S}^\triangledown\) from which we deduce that \(\neg(b \odot \neg a) \in \mathcal{S}^\triangledown\) (by internal contraposition). Using the deduction axiom (since \(\neg a \in \mathcal{S}^\triangledown\)), we finally get \(\neg b \in \mathcal{S}^\triangledown\). □