A Model of Countable Nondeterminism in Guarded Type Theory

Aleš Bizjak[1], Lars Birkedal[1], and Marino Miculan[2]

[1]Aarhus University {abizjak,birkedal}@cs.au.dk
[2]University of Udine marino.miculan@uniud.it

July 24, 2015

Contents

1 The topos $\mathbf{Sh}(\omega_1)$

We first describe the topos of sheaves over ω_1, the first uncountable ordinal. More precisely, we consider ω_1 as a topological space equipped with the Alexandrov topology. Thus the topos $\mathbf{Sh}(\omega_1)$ is a full subcategory of the category $\mathbf{PSh}(\omega_1 + 1)$, since the opens of ω_1 are exactly the downwards closed subsets of ω_1 which in turn correspond precisely to $\omega_1 + 1$ (by von Neumann's construction of ordinals, they are exactly $\omega_1 + 1$). We write 0 for the first ordinal. So objects are of the form

$$X(0) \longleftarrow X(1) \longleftarrow \cdots \longleftarrow X(\omega) \longleftarrow X(\omega + 1) \longleftarrow \cdots$$

More precisely the objects of $\mathbf{Sh}(\omega_1)$ are continuous functors from $(\omega_1 + 1)^{\mathrm{op}}$ to $\mathbf{Set}$ and morphisms are natural transformations.

$\mathbf{Sh}(\omega_1)$ is a topos. The inclusion functor $i : \mathbf{Sh}(\omega_1) \to \mathbf{PSh}(\omega_1 + 1)$ has a *left* adjoint $\mathbf{a}$, the associated sheaf functor. Limits and exponentials in $\mathbf{Sh}(\omega_1)$ are computed as in $\mathbf{PSh}(\omega_1 + 1)$, i.e. limits are pointwise and exponential X^Y is given at stage ν as $\mathbf{Hom}_{\mathbf{Sh}(\omega_1)}(\mathbf{Hom}_{\omega_1+1}(\cdot, \nu) \times Y, X)$. Colimits, however, are not constructed as in presheaves, but are computed first as in presheaves followed by an application of the $\mathbf{a}$ functor.

We denote the lattice of subobjects of an object X by $\mathbf{Sub}(X)$ and we denote reindexing along $f : X \to Y$ by $f^* : \mathbf{Sub}(Y) \to \mathbf{Sub}(X)$. Since $\mathbf{Sh}(\omega_1)$ is a topos, each subobject lattice is a complete Heyting algebra. Further, we can show that each subobject lattice is in fact a bi-Heyting algebra, that is, a Heyting algebra with a $\setminus$ operation that is left adjoint to union, i.e. $X \setminus Y \leq Z \leftrightarrow X \leq Y \vee Z$. The existence of $\setminus$ can be shown by using explicit descriptions of operations on the subobject lattice below. This makes $\mathbf{Sh}(\omega_1)$ a bi-Heyting topos [8].

Subobject classifier The subobject classifier Ω is given by closed sieves, which are exactly the maximal sieves. More precisely the subobject classifier at ν is given by sieves S such that $\bigvee S \in S$. These sieves therefore correspond to ordinals smaller or equal to ν. Explicitly

$$\Omega(\nu) = \{\beta \mid \beta \leq \nu\}$$

(note that von Neumann's construction of ordinals gives $\Omega(\nu) = \nu + 1$).

The restriction maps are given by minimum, i.e.

$$r^\beta_\nu : \Omega(\beta) \to \Omega(\nu)$$
$$r^\beta_\nu(\gamma) = \min\{\beta, \nu\}$$

and the map $\mathbf{true} : 1 \to \Omega$ maps $*$ to the maximal sieve

$$\mathbf{true}_\nu = \nu.$$

Note that this is different from the construction of the subobject classifier for presheaves, where $\Omega(\nu)$ is all the sieves on ν, including the empty sieve.

Given a subobject $m : A \leq X$ the characteristic map $\chi^m : X \to \Omega$ is given by

$$\chi^m_\nu(x) = \bigvee \left\{\beta \leq \nu \mid m^{-1}_\beta \left[x|_\beta\right] \neq \emptyset\right\}$$

i.e. (if we assume $A(\nu) \subseteq X(\nu)$, which we are allowed to)

$$\chi^m_\nu(x) = \bigvee \left\{\beta \leq \nu \mid x|_\beta \in A(\beta)\right\}$$

NB: The supremum of an empty set is 0, the first ordinal.

Some of the properties later will not hold for all the sheaves but only for a certain subset.

Definition 1.0.1. *A sheaf X is total if the restriction maps are surjections. This is the same as saying that the* next^X *defined is internally surjective.*

A way to think of totality is by thinking of ordinals as time with smaller ordinals being the future. Then X being total means that elements at any stage ν are only those that have evolved from some previous stages. They did not just suddenly appear.

1.1 Relationship to Set

The global elements geometric morphism $\Delta \vdash \Gamma : \mathbf{Sh}(\omega_1) \to \mathbf{Set}$ is an essential geometric morphism. That is, there is an adjoint triple

$$\Pi_1 \dashv \Delta \dashv \Gamma$$

where $\Pi_1, \Gamma : \mathbf{Sh}(\omega_1) \to \mathbf{Set}$ and $\Delta : \mathbf{Set} \to \mathbf{Sh}(\omega_1)$ are given as follows

$$\Pi_1(X) = X(1) = \operatorname*{colim}_{\nu \leq \omega_1} X(\nu)$$
$$\Gamma(X) = \mathbf{Hom}_{\mathbf{Sh}(\omega_1)}(1, X) = \lim_{\nu \leq \omega_1} X(\nu)$$
$$\Delta(a)(\nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ a & \text{otherwise} \end{cases}$$

Δ is the *constant sheaf* functor (note that it is *not* the *constant presheaf* functor).

The adjunction $\Pi_1 \dashv \Delta$ gives rise to an adjunction between subobject lattices. More precisely for any set a, there is an adjunction

$$\Pi_1^a : \mathbf{Sub}_{\mathbf{Sh}(\omega_1)}(\Delta(a)) \to \mathbf{Sub}_{\mathbf{Set}}(a) : \Delta^a$$

where $\Delta^a(b) = \Delta(b)$ and $\Pi_1^a(A) = A(1)$. These adjunctions are natural in the sense that for any function $\alpha : a' \to a$ we have $\alpha^* \circ \Pi_1^a = \Pi_1^{a'} \circ \Delta(\alpha)^*$, which is easy to check directly.

Thus, $\Delta \dashv \Gamma : \mathbf{Sh}(\omega_1) \to \mathbf{Set}$ is an open geometric morphism [6, Definition IX.6.2] which further means that Δ preserves models of first-order logic in the sense of [6, Theorem X.3.1]. In practice, this means that whatever predicate on a constant set we define in the internal logic using only the first-order fragment and other constant relations and predicates will be constant.

Another way to see that $\Delta \dashv \Gamma$ is an open geometric morphism is by the fact that $\mathbf{Sh}(1)$ (sheaves on a one point space) is isomorphic to the category $\mathbf{Set}$. Since the unique map $\omega_1 \to 1$ is *open*, the induced geometric morphism is open. It can easily be seen that the direct image functor induced by this unique morphism is (isomorphic to) Γ. By uniqueness of adjoints the inverse image functor must then be (isomorphic to) Δ.

Moreover, Π_1 is a logical morphism, meaning it is a cartesian closed functor that preserves Ω. This is easy to see manually, by computing. However, there is a more general argument available. It proceeds as follows. The set $\{0\}$ is an open subset of ω_1. Let $i : \{0\} \to \omega_1$ be the inclusion and let $i_* : \mathbf{Sh}(\{0\}) \to \mathbf{Sh}(\omega_1)$ be the direct image functor. Recall that

$$i_*(F)(U) = F\left(i^{-1}[U]\right)$$

and in our particular case we have

$$i_*(F)(\nu) = \begin{cases} F(\emptyset) & \text{if } \nu = 0 \\ F(\{0\}) & \text{if } \nu \neq 0 \end{cases}$$

Recall that $\mathbf{Sh}(\{0\})$ is isomorphic to $\mathbf{Set}$ with the isomorphism $\xi : \mathbf{Set} \to \mathbf{Sh}(\{0\})$ given by $\xi(a)(\emptyset) = 1$, $\xi(a)(\{0\}) = a$ and the obvious restriction. Thus we see that $\Delta = i_* \circ \xi$. The inverse image functor i^* is left adjoint to i_*. Since we also have $\xi \circ \Pi_1 \dashv \Delta \circ \xi^{-1}$ we have that i^* is naturally isomorphic to $\xi \circ \Pi_1$ or equivalently $\Pi_1 \cong \xi^{-1} \circ i^*$.

Since $\{0\}$ is an open subset of $\mathbf{Sh}(\omega_1)$ this makes $\mathbf{Set}$ (equivalent to) an open subtopos of $\mathbf{Sh}(\omega_1)$ [5, Section A4.5].

Moreover, we have by [6, Theorem 6, Corollary 7] that $\mathbf{Set}$ is equivalent to a category of j-sheaves for some universal closure operator j. We will see in Section 1.2 that this local operator is exactly the $\neg\neg$-closure. Thus there exists a geometric morphism $e : \mathbf{Set} \to \mathbf{Sh}_j(\mathbf{Sh}(\omega_1))$ such that $e^* \circ a \cong \Pi_1$ and $\iota \circ e_* \cong \Delta$ where ι is the inclusion $\mathbf{Sh}_j(\mathbf{Sh}(\omega_1)) \to \mathbf{Sh}(\omega_1)$.

Putting all of it together we have $\mathbf{Sh}_j(\mathbf{Sh}(\omega_1))$ is an open subtopos of $\mathbf{Sh}(\omega_1)$ since it is equivalent to $\mathbf{Set}$ which is equivalent to $\mathbf{Sh}(\{0\})$. By [5, Proposition 4.5.1] this means that $a : \mathbf{Sh}(\omega_1) \to \mathbf{Sh}_j(\mathbf{Sh}(\omega_1))$ is a logical functor and since $\Pi_1 \cong e^* \circ a$, with e^* being part of an equivalence, which means that it is a logical functor, we have that Π_1 is logical.

This means that Π_1 preserves validity of formulas in the internal language.

1.2 Description of the Heyting algebra structure of the subobject lattices

We give here explicit descriptions of operations on each subobject lattice $\mathbf{Sub}(X)$. Let $X \in \mathbf{Sh}(\omega_1)$, $A, B \in \mathbf{Sub}(X)$ and $\beta \leq \omega_1$. We have

$$\begin{aligned}
\top &= X \\
\bot(\beta) &= \begin{cases} 1 & \text{if } \beta = 0 \\ \emptyset & \text{otherwise} \end{cases} \\
(A \wedge B)(\beta) &= A(\beta) \cap B(\beta) \\
\left(\bigwedge_{i \in I} A_i\right)(\beta) &= \bigcap_{i \in I} A_i(\beta) \\
(A \Rightarrow B)(\beta) &= \left\{ x \in X(\beta) \;\middle|\; \forall \gamma \leq \beta, x|_\gamma \in A(\gamma) \to x|_\gamma \in B(\gamma) \right\} \\
\left(\bigvee_{i \in I} A_i\right)(\beta) &= \left\{ x \in X(\beta) \;\middle|\; \bigvee \left\{ \gamma \leq \beta \mid \exists i \in I, x|_\gamma \in A_i(\gamma) \right\} = \beta \right\}
\end{aligned}$$

Let further $Y \in \mathbf{Sh}(\omega_1)$ and $\varphi : X \to Y$. Then $\exists_\varphi, \forall_\varphi : \mathrm{Sub}(X) \to \mathrm{Sub}(Y)$ are given as follows

$$\begin{aligned}
\exists_\varphi(A)(\beta) &= \left\{ y \in Y(\beta) \;\middle|\; \bigvee \left\{ \gamma \leq \beta \mid \exists a \in A(\gamma), \varphi_\gamma(a) = y|_\gamma \right\} = \beta \right\} \\
\forall_\varphi(A)(\beta) &= \left\{ y \in Y(\beta) \;\middle|\; \forall \gamma \leq \beta, \varphi_\gamma^{-1}\left[y|_\gamma\right] \subseteq A(\gamma) \right\}
\end{aligned}$$

and $\varphi^* : \mathrm{Sub}(Y) \to \mathrm{Sub}(X)$ is given by

$$\varphi^*(C)(\beta) = \left\{ x \in X(\beta) \;\middle|\; \varphi_\beta(x) \in C(\beta) \right\}$$

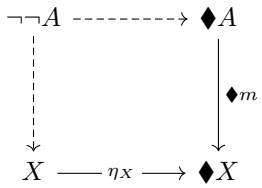
These are standard results from [6, III.8] specialized for a particular space with a particular topology.

Using these descriptions it is easy to see that the $\neg\neg_X : \mathbf{Sub}(X) \to \mathbf{Sub}(X)$ is given as follows

$$(\neg\neg_X A)(\nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ \{x \in X(\nu) \mid x|_1 \in A(1)\} \end{cases}$$

(intuitively, this says that something is not impossible if it will eventually happen (smaller indices are the future)). Observe that in fact $\neg\neg_X P = U_X \to P$ where $U_X \leq X$ is given as $U_X(0) = 1$, $U_X(1) = X(1)$ and $U_X(\nu) = \emptyset$ otherwise. This means that $\neg\neg$ is an *open* local operator [5, Section 4.5].

From these explicit descriptions we can see that $\neg\neg_X$ arises from the functor $\Delta \circ \Pi_1$ as follows. Let $\blacklozenge = \Delta \circ \Pi_1$. Note that $\blacklozenge \dashv \Delta \circ \Gamma$ which means that it preserves all colimits. It is easy to see that $\blacklozenge$ preserves all limits since they are constructed pointwise. We will also see in Section 1.4 that it has a left adjoint which implies that it preserves all limits. In particular it preserves monomorphisms, thus subobjects. Hence given a subobject $m : A \leq X$, $\blacklozenge m : \blacklozenge A \leq \blacklozenge X$. Further, $\blacklozenge$ is a monad, thus there is a unit $\eta_X : X \to \blacklozenge X$. It is then easy to see that



is a pullback diagram. Another way to state this is that $\neg\neg_X : \mathbf{Sub}(X) \to \mathbf{Sub}(X)$ is given as $\neg\neg_X(A) = \eta_X^*(\blacklozenge A)$.

Using this description we can easily see that $\neg\neg_X$ preserves suprema. Indeed

$$\neg\neg_X \left(\bigvee_i A_i \right) = \eta_X^* \left(\blacklozenge \left(\bigvee_i A_i \right) \right)$$

and since suprema in $\mathbf{Sub}(X)$ are constructed using a coproduct in $\mathbf{Sh}(\omega_1)$ followed by images, which are constructed using limits and colimits of $\mathbf{Sh}(\omega_1)$, which are preserved by $\blacklozenge$, we have

$$= \eta_X^* \left(\bigvee_i \blacklozenge A_i \right)$$

and since $\eta_X^* : \mathbf{Sub}(X) \to \mathbf{Sub}(X)$ has a right adjoint, $\forall_{\eta_X}$ functor, it preserves suprema, hence

$$= \bigvee_i \eta_X^* (\blacklozenge A_i) = \bigvee_i \neg\neg_X A_i.$$

Note that as in all toposes, $\neg\neg$ is preserved by reindexing functors, i.e. for any $\varphi : X \to Y$, $\varphi^* \circ \neg\neg_Y = \neg\neg_X \circ \varphi^*$.

We can also easily compute manually using explicit descriptions of operations above, that $\neg\neg_X$ does indeed preserve suprema.

1.3 The $\Box$ modality

Since $\neg\neg_X$ preserves suprema it has a *right* adjoint [2, Corollary 9.32]. We denote this right adjoint to $\neg\neg_X$ by $\Box_X$ and define it as

$$\Box_X(P) = \bigvee \{Q \mid \neg\neg_X Q \leq P\}.$$

$\Box_X$ is an interior operation on the subobject lattice $\mathbf{Sub}(X)$ and $\Box_X P$ it can be characterized as the greatest element smaller than P that is also $\neg\neg_X$ closed. The fact that $\Box_X P$ is $\neg\neg$-closed is proved in Lemma 1.3.1 and the fact that it preserves $\neg\neg$-closed subobjects is proved in Corollary 1.3.3.

Lemma 1.3.1. *For any object X and $P \leq X$, $\neg\neg_X \Box_X(P) = \Box_X(P)$.*

Proof. By definition of $\Box_X$ we have

$$\neg\neg_X (\Box_X(P)) = \neg\neg_X \left(\bigvee \{Q \mid \neg\neg_X Q \leq P\} \right)$$

and since $\neg\neg_X$ preserves suprema

$$= \bigvee \{\neg\neg_X Q \mid \neg\neg_X Q \leq P\} = \bigvee \{Q \mid \neg\neg_X Q \leq P\} = \Box_X(P)$$

The second to last equality holds because for each Q, $\neg\neg_X Q \geq Q$ and $\neg\neg_X \neg\neg_X Q = \neg\neg_X Q$. $\square$

Corollary 1.3.2. *For any object X and $P \leq X$, $\Box_X P \leq P$.*

Proof. Since $\neg\neg_X \dashv \Box_X$ we have $\Box_X P \leq \Box_X P \leftrightarrow \neg\neg_X \Box_X P \leq P$. Since $\leq$ is reflexive Lemma 1.3.1 concludes the proof. $\square$

Corollary 1.3.3. *For any object X and $P \leq X$, $\Box_X(\neg\neg_X P) = \neg\neg_X P$.*

Proof. For any P, $\neg\neg_X \neg\neg_X P = \neg\neg_X P$. Thus $\neg\neg_X \neg\neg_X P \leq \neg\neg_X P$. Since $\Box_X$ is right adjoint to $\neg\neg_X$ we get $\neg\neg_X P \leq \Box_X \neg\neg_X P$. The other direction follows directly from Corollary 1.3.2. $\square$

Corollary 1.3.4. *For any object X and $P \leq X$, $\Box_X(\Box_X P) = \Box_X P$.*

Proof. $\Box_X(\Box_X P) \leq \Box_X P$ follows from Corollary 1.3.2. The other direction follows from the fact that $\Box_X$ is right adjoint to $\neg\neg_X$ and Lemma 1.3.1 since by adjointness we have $\Box_X P \leq \Box_X \Box_X P \leftrightarrow \neg\neg_X \Box_X P \leq \Box_X P$ and the right hand side holds by Lemma 1.3.1. $\square$

We now state and prove how $\Box_X$ commutes with some other operations on $\mathbf{Sub}(X)$. In particular, it commutes with conjunction, top, bottom and universal quantification.

Proposition 1.3.5. *Let X and Y be types, $P, Q \in \mathbf{Sub}(X)$ and $\varphi : X \to Y$ a morphism. The following hold*

1. $\Box_X \top = \top$
2. $\Box_X \bot = \bot$
3. $\Box_X(P \wedge Q) = \Box_X P \wedge \Box_X Q$
4. $\Box_Y (\forall_\varphi P) = \forall_\varphi (\Box_X P)$.

Proof. Since $\Box_X$ is a right adjoint it preserves limits in $\mathbf{Sub}(X)$. $\top$ and $\wedge$ are limits, therefore $\Box_X$ necessarily preserves them.

Corollary 1.3.2 shows that $\Box_X P \leq P$ for any P. In particular, this holds for $\bot$ and since $\bot$ is the least element of $\mathbf{Sub}(X)$, we get $\Box_X \bot = \bot$.

To see $\Box_Y(\forall_\varphi P) = \forall_\varphi(\Box_X P)$ we use the fact that $\forall_\varphi$ is right adjoint to φ^* and that φ^* commutes with $\neg\neg_X$. Thus for any $R \in \mathbf{Sub}(Y)$ we have

$$\begin{aligned} R \leq \Box_Y(\forall_\varphi P) &\leftrightarrow \neg\neg_Y R \leq \forall_\varphi P \\ &\leftrightarrow \varphi^*(\neg\neg_Y R) \leq P \\ &\leftrightarrow \neg\neg_X(\varphi^* R) \leq P \\ &\leftrightarrow \varphi^* R \leq \Box_X P \\ &\leftrightarrow R \leq \forall_\varphi(\Box_X P) \end{aligned}$$

Thus picking R to be $\Box_Y(\forall_\varphi P)$ or $\forall_\varphi(\Box_X P)$ we get both approximations (we could also have just referenced Yoneda's lemma here), thus $\Box_Y(\forall_\varphi P) = \forall_\varphi(\Box_X P)$. □

$\Box$ and substitution In contrast to $\neg\neg_X$ which is preserved by reindexing functors, $\Box_X$ is not, that is, for $\varphi : Y \to X$ it *is not in general the case* that $\Box_Y \circ \varphi^* = \varphi^* \circ \Box_X$. One direction, however, does hold.

Proposition 1.3.6. *For any $\varphi : Y \to X$, $\varphi^* \circ \Box_X \leq \Box_Y \circ \varphi^*$. If φ is an isomorphism then the two sides are also equal.*

Proof. By definition of $\Box_X$ and the fact that φ^* has a right adjoint we have

$$\begin{aligned} \varphi^*(\Box_X(P)) &= \varphi^*\left(\bigvee\{Q \mid \neg\neg_X Q \leq P\}\right) \\ &= \bigvee\{\varphi^* Q \mid \neg\neg_X Q \leq P\} \\ &\leq \bigvee\{\varphi^* Q \mid \varphi^*(\neg\neg_X Q) \leq \varphi^*(P)\} \\ &= \bigvee\{\varphi^* Q \mid \neg\neg_Y(\varphi^* Q) \leq \varphi^*(P)\} \\ &\leq \bigvee\{R \mid \neg\neg_Y R \leq \varphi^*(P)\} \\ &= \Box_Y(\varphi^* P) \end{aligned}$$

If φ were an isomorphism it would preserve and also reflect order and every element of the lattice would be in the image. Thus the chain can be strengthened to show that in this case $\Box_Y(\varphi^* P)$ and $\varphi^*(\Box_X(P))$ are equal. □

The fact that $\Box_X$ is not natural in X implies that there is no morphism $\Box : \Omega \to \Omega$ such that $\Box_X$ would arise from it, as is the case for $\neg\neg_X$. Note that it is not a coincidence that $\Box_X$ is not natural but a fundamental limitation of interior operations. If $\Box_X$ were natural it would have to be the identity. More precisely, any operation on the subobject lattices that preserves $\top$ and is deflationary and natural in X must be the identity (provided the category satisfies some minimal requirements). This is proved in [8, Proposition 4.2] (see also Proposition 4.1 of loc. cit.).

$\Box_X$ does commute with exchange, however, but in general not with contraction and weakening. It does commute with weakening in the special case proved in Corollary 1.3.8 below. For the proof we need the following lemma stating that $\neg\neg$ commutes with $\exists_\pi$ for suitable π.

Lemma 1.3.7. *Let X and Y be types and $\pi : X \times Y \to X$ the projection. Then $\exists_\pi \circ \neg\neg \leq \neg\neg \circ \exists_\pi$ always holds. If Y is total the converse also holds.*

Proof. $\exists_\pi$ is the left adjoint to π^*. Thus

$$\begin{aligned}\exists_\pi(\neg\neg Q) \leq \neg\neg(\exists_\pi Q) &\leftrightarrow \neg\neg Q \leq \pi^*\left(\neg\neg(\exists_\pi Q)\right)\\ &\leftrightarrow \neg\neg Q \leq \neg\neg\left(\pi^*(\exists_\pi Q)\right)\\ &\leftarrow Q \leq \pi^*(\exists_\pi Q)\end{aligned}$$

and the last holds because $\pi^* \circ \exists_\pi$ is a monad, i.e. a closure.

Now suppose Y is total and let $Q \in \mathbf{Sub}(X \times Y)$. First, let ν be a successor ordinal. Let $x \in \neg\neg(\exists_\pi(Q))(\nu)$. By definition $x|_1 \in \exists_\pi(Q)(1)$, which further implies there exists $y \in Y(1)$, such that $(x|_1, y) \in Q(1)$. Since Y is total there exists a $y' \in Y(\nu)$, such that $y'|_1 = y$. Thus $(x|_1, y'|_1) \in Q(1)$ and so $(x, y') \in \neg\neg Q(\nu)$. This means that $x \in \exists_\pi(\neg\neg Q)(\nu)$.

Since this holds for all successor ordinals, it must also hold for limit ordinals (actually, the same manual proof would also suffice, but it is more complicated to write it down). □

The restriction on total objects Y is necessary. Consider any total X and let Y be an object which at stage at stage 1 is some nonempty set and at greater stages is the empty set. Suppose $P = X \times Y$, i.e. the top element. Then $\exists_\pi(\neg\neg P)(2) = \emptyset$, however $\neg\neg(\exists_\pi(P))(2)$ is not empty (since X is total).

Corollary 1.3.8. *Let $\pi : X \times Y \to X$ be the projection. If Y is total (restrictions are surjections) then $\pi^* \circ \Box_X = \Box_{X\times Y} \circ \pi^*$.*

Proof. In light of Proposition 1.3.6 we only need to show. $\pi^* \circ \Box_X \geq \Box_{X\times Y} \circ \pi^*$. We have

$$\begin{aligned}\Box_{X\times Y}(\pi^* Q) \leq \pi^*(\Box_X Q) &\leftrightarrow \exists_\pi \Box_{X\times Y}(\pi^* Q) \leq \Box_X Q\\ &\leftrightarrow \neg\neg\left(\exists_\pi \Box_{X\times Y}(\pi^* Q)\right) \leq Q\\ &\leftrightarrow \exists_\pi \neg\neg\left(\Box_{X\times Y}(\pi^* Q)\right) \leq Q\\ &\leftrightarrow \neg\neg\left(\Box_{X\times Y}(\pi^* Q)\right) \leq \pi^* Q\\ &\leftrightarrow \Box_{X\times Y}(\pi^* Q) \leq \Box_{X\times Y}\left(\pi^* Q\right)\end{aligned}$$

□

Exchange is reindexing by an isomorphism. Therefore by Proposition 1.3.6 it preserves □.

1.3.1 □ on constant types

If $X = \Delta(a)$ for some set a then $\Box_X$ has a much simpler description.

Lemma 1.3.9. *If $X = \Delta(a)$ and $P \leq X$ then*

$$\Box_X(P)(\nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ \bigcap_{\nu=1}^{\omega_1} P(\nu) & \text{otherwise} \end{cases}.$$

Proof. Since adjoints are unique we only need to show that $\Box_X$ is right adjoint to $\neg\neg_X$. On constant objects the definition of $\neg\neg_X$ simplifies to

$$(\neg\neg_X P)(\nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ P(1) & \text{otherwise} \end{cases}.$$

Thus suppose $\neg\neg_X P \leq Q$. In particular, this means that for all $\nu \geq 1$, $P(1) \subseteq Q(\nu)$. Thus $P(1) \leq \bigcap_{\nu=1}^{\omega_1} Q(\nu)$ and since restrictions on X are inclusions, meaning that $P(\nu) \subseteq P(1)$ for any $\nu \geq 1$, we get $P \leq \Box_X Q$.

Conversely, suppose $P \leq \Box_X Q$. Thus $P(\beta) \leq \bigcap_{\nu=1}^{\omega_1} Q(\nu)$ for all $\beta \geq 1$. In particular this means that $P(1) \subseteq \bigcap_{\nu=1}^{\omega_1} Q(\nu)$ and so $P(1) \subseteq Q(\nu)$ for any $\nu \geq 1$, meaning that $\neg\neg_X P \leq Q$. $\Box$

Using this description we can show that $\Box_X$ is preserved by reindexing functors arising from maps between constant sheafs.

Proposition 1.3.10. *Let a, b be sets and $f : a \to b$. Then $\Delta(f)^* \circ \Box_{\Delta(b)} = \Box_{\Delta(a)} \circ \Delta(f)^*$.*

Proof. Let $\nu \geq 1$ (for $\nu = 0$ there is nothing to prove). By a simple calculation we have

$$\left(\Delta(f)^*(\Box_{\Delta(b)}(P))\right)(\nu) = f^{-1}\left[\bigcap_{\nu=1}^{\omega_1} P(\nu)\right]$$

and since preimages preserve intersections we get

$$\begin{aligned} &= \bigcap_{\beta=1}^{\omega_1} f^{-1}\left[P(\beta)\right] \\ &= \bigcap_{\beta=1}^{\omega_1} \left(\Delta(f)^*(P)(\beta)\right) \\ &= \left(\Box_{\Delta(a)}\left(\Delta(f)^*(P)\right)\right)(\nu) \end{aligned}$$

$\Box$

Note that any morphism $\varphi : \Delta(a) \to \Delta(b)$ is of the form $\varphi = \Delta(f)$ for some (unique) $f : a \to b$, i.e. Δ is full and faithful. Thus if we restrict to constant contexts $\Box$ commutes with substitution and so we may work informally with $\Box$ as with $\neg\neg$ or any other logical operation.

1.4 The $\neg\neg$ modality

It is easy to see that $\blacklozenge$ preserves all limits. Indeed, Π_1 also has a left adjoint σ_1 defined as follows

$$\begin{aligned} \sigma_1(a)(0) &= 1 \\ \sigma_1(a)(1) &= a \\ \sigma_1(a)(\nu) &= \emptyset \quad \text{for } \nu \geq 1 \end{aligned}$$

which then means that $\blacklozenge$ has a left adjoint $\sigma_1 \circ \Pi_1$ and thus it must preserve all limits. This then implies, using the fact that $\neg\neg_X(A) = \eta_X^*(\blacklozenge A)$ and that limits in subobject lattices are computed from limits in $\mathbf{Sh}(\omega_1)$, that $\neg\neg_X$ preserves all limits. In particular, it preserves all infima, meaning that it also has a left adjoint, which we call $\diamond_X$. It is given simply as

$$\diamond_X(P) = \bigwedge\{Q \mid P \leq \neg\neg Q\}$$

however it can be described in an elementary way as

Lemma 1.4.1.

$$\diamond_X(P)(\nu) = \begin{cases} 1 & \text{if } \nu = 0 \\ P(1) & \text{if } \nu = 1 \\ \emptyset & \text{otherwise} \end{cases}$$

Proof. By uniqueness of adjoints we only need to show that $\diamond_X$ is left adjoint to $\neg\neg_X$.

First suppose $\diamond_X P \leq Q$. We are to show $P \leq \neg\neg Q$. Let $x \in P(\nu)$. Then $x|_1 \in P(1)$, thus by assumption in $Q(1)$. Hence by the explicit description of $\neg\neg$ we have that $x \in \neg\neg Q(\nu)$.

Now suppose $P \leq \neg\neg Q$ and we are to show $\diamond_X P \leq Q$. The only non-trivial inclusion is at stage 1. So take $x \in P(1)$. Then $x \in Q$ by assumption (since $\neg\neg Q(1) = Q(1)$), concluding the proof. □

Note that $\sigma_1 \circ \Pi_1$ is a comonad, its counit is mono and using the descriptions above we have that $\diamond_X(P) = \sigma_1(\Pi_1(m_P)); \varepsilon_X$ where m_P is the mono belonging to P and ε_X is the counit at X.

In contrast to $\Box$, however, $\diamond$ does commute with reindexing. Thus it defines a map $\diamond : \Omega \to \Omega$ which is simply given as

$$\diamond_\nu(\beta) = \begin{cases} 0 & \text{if } \nu = 0 \\ 1 & \text{otherwise} \end{cases}$$

or equivalently as $\diamond_\nu(\beta) = \min\{1, \beta\}$ which then clearly shows that $\diamond$ is natural, i.e. a morphism $\Omega \to \Omega$.

The fact that $\diamond$ is natural has as a consequence the fact that $\neg\neg$ commutes with universal quantification.

Lemma 1.4.2. *Let $\varphi : X \to Y$ be a morphism and $\forall_\varphi$ the right adjoint to φ^*. Then $\neg\neg \circ \forall_\varphi = \forall_\varphi \circ \neg\neg$.*

Proof. We show two inequalities using properties of adjoints. We have for any R

$$\begin{aligned} R \leq \neg\neg\left(\forall_\varphi Q\right) &\leftrightarrow \diamond R \leq \forall_\varphi Q \\ &\leftrightarrow \varphi^*\left(\diamond R\right) \leq Q \\ &\leftrightarrow \diamond\left(\varphi^* R\right) \leq Q \\ &\leftrightarrow \varphi^* R \leq \neg\neg Q \\ &\leftrightarrow R \leq \forall_\varphi\left(\neg\neg Q\right) \end{aligned}$$

We now apply Yoneda's lemma (or just immediately see that this implies equality) to conclude the proof. □

Note that the lemma states that $\neg\neg$ commutes over all universals, not just the usual ones arising from weakening. This is in contrast to the situation with $\neg\neg$ and existentials. An analogous proof to the above shows that $\diamond$ commutes over existentials. The reason we cannot use the same proof to show that $\neg\neg$ commutes over existentials, even though it has a right adjoint, is that $\Box$ does not commute with reindexing.

Note that the fact that $\diamond$ commutes with reindexing is not contrary to [8, Proposition 4.2] since $\diamond$ does not preserve truth. It is however a comonad, i.e. $\diamond P \leq P$ and $\diamond \diamond P = \diamond P$.

Further, we can show that $\neg\neg$ commutes with implication. Indeed, since $\neg\neg$ preserves limits in subobject lattices we immediately have $\neg\neg(P \to Q) \leq \neg\neg P \to \neg\neg Q$. For the other direction we first recall that $P \to Q = \forall_{m_P}\left(m_P^* Q\right)$ where $m_P : P \to X$ is the inclusion of P into X: To that end we use the fact that $P \wedge Q = m_P^*(Q)$. We have

$$P \wedge R \leq Q \leftrightarrow P \wedge R \leq P \wedge Q \leftrightarrow m_P^*(R) \leq m_P^*(Q) \leftrightarrow R \leq \forall_{m_P}(m_P^*(Q))$$

and by uniqueness of adjoints also $P \to Q = \forall_{m_P}\left(m_P^* Q\right)$. Using this, we can prove the following.

Lemma 1.4.3. $\neg\neg(P \to Q) = P \to \neg\neg Q = \neg\neg P \to \neg\neg Q$

Proof. Since $\neg\neg(P \to Q) \wedge P \leq \neg\neg(P \to Q \wedge P) \leq \neg\neg Q$ we get $\neg\neg(P \to Q) \leq P \to \neg\neg Q$ and also $\neg\neg(P \to Q) \leq \neg\neg P \to \neg\neg Q$.

By the above characterization and the fact that $\neg\neg$ commutes over reindexing and universals we have. $\neg\neg(P \to Q) = \neg\neg\left(\forall_{m_P}\left(m_P^*(Q)\right)\right) = \forall_{m_P}\left(m_P^*(\neg\neg Q)\right) = P \to \neg\neg Q$ It is easy to see that $\neg\neg P \to \neg\neg Q \leq P \to \neg\neg Q$, with the same calculation as above (also, we can always weaken the precondition). We thus have that $\neg\neg P \to \neg\neg Q \leq P \to \neg\neg Q = \neg\neg(P \to Q)$. This concludes the proof. □

1.5 The $\triangleright$ and $\blacktriangleright$ modalities

Recall that the first ordinal, 0, is a limit ordinal. It is the limit of the empty sequence. There is a functor $\blacktriangleright : \mathbf{Sh}\,(\omega_1) \to \mathbf{Sh}\,(\omega_1)$ defined by

$$\begin{aligned} \blacktriangleright(X)(\nu+1) &= X(\nu) \\ \blacktriangleright(X)(\alpha) &= X(\alpha) \qquad \text{for a limit ordinal } \alpha \end{aligned}$$

and the obvious action on morphisms. There is a natural transformation $\mathrm{next}^X : X \to \blacktriangleright(X)$ defined as

$$\begin{aligned} \mathrm{next}^X_{\nu+1} &= r_\nu \\ \mathrm{next}^X_{\alpha} &= \mathrm{id}_{X(\alpha)} \qquad \text{for a limit ordinal } \alpha \end{aligned}$$

where $r_\nu = X(\nu \leq \nu+1) : X(\nu+1) \to X(\nu)$ is the restriction map of X.

Using $\blacktriangleright$ we can define a notion of contractiveness of morphisms.

Definition 1.5.1. *A morphism $\varphi : X \to Y$ is* (externally) contractive *if there exists a morphism $g : \blacktriangleright X \to Y$, such that $f = \mathrm{next}^X; g$.*

A useful property of contractive endomorphisms is that they have unique fixed points.

Proposition 1.5.2. *If $f : X \to X$ is a contractive morphism then there exists a unique $e : 1 \to X$ such that $e; f = e$.*

Proof. Let $f = \mathrm{next}^X; g$. By construction of $\blacktriangleright$ we have that $g_{\nu+1} : X(\nu) \to X(\nu+1)$ and $X(0) = 1$. We thus define a global section by induction as follows[1] (again, α is a limit ordinal)

$$\begin{aligned} e_0(*) &= g_0(*) \\ e_{\nu+1}(*) &= g_{\nu+1}(e_\nu(*)) \\ e_\alpha &= \lim_{\nu<\alpha} e_\nu \end{aligned}$$

where $\lim_{\nu<\alpha} e_\nu$ denotes the unique element of $X(\alpha)$ that restricts to e_ν, $\nu < \alpha$. Now it is obvious that $r_0(e_1(*)) = e_0(*)$. For the successor ordinals we have

$$r_\nu(e_{\nu+1}) = r_\nu(g_{\nu+1}(e_\nu(*))) = f_\nu(e_\nu(*)) = e_\nu(*)$$

and

$$f_{\nu+1}(e_{\nu+1}(*)) = g_{\nu+1}(r_\nu(e_{\nu+1}(*))) = g_{\nu+1}(e_\nu(*)) = e_{\nu+1}(*).$$

[1] More precisely, we define at stage ν a triple of an element $e_\nu \in X(\nu)$ and proofs that it is a fixed point of f_ν and that it restricts to the previous ones.

and for limit ordinals we have that e_α restricts to previous ones by construction. To show that $f_\alpha(e_\alpha(*)) = e_\alpha(*)$ we show that $f_\alpha(e_\alpha(*))$ restricts to the same elements. Let $\nu < \alpha$.

$$f_\alpha(e_\alpha(*))|_\nu = f_\nu\left(e_\alpha|_\nu\right) = f_\nu(e_\nu) = e_\nu$$

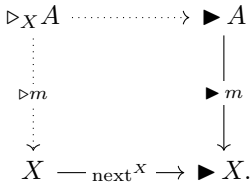
hence $f_\alpha(e_\alpha(*)) = e_\alpha$ by uniqueness of amalgamations.

Thus e defines a natural transformation $1 \to X$ with $e; f = e$. To see that it is unique observe that if $e' : 1 \to X$ is such that $e'; f = e'$ we have

$$e'_{\nu+1}(*) = f_{\nu+1}(e'_{\nu+1}(*)) = g_{\nu+1}(r_\nu(e'_{\nu+1}(*))) = g_{\nu+1}(e'_\nu(*))$$

and the values at limit ordinals α are uniquely determined by the sheaf condition. Thus $e' = e$. □

$\triangleright$ modality $\blacktriangleright$ is a modality on types. It is easy to see that it preserves all limits, since they are constructed pointwise. In particular, it preserves monos, therefore subobjects. Thus we can define an operation on subobject lattices, which we call $\triangleright$. Given $m : A \leq X$ the subobject $\triangleright m : \triangleright A \leq X$ is defined via pullback along next^X as in the following diagram



Since it is defined via pullback it is easy to see that this operation is natural in X, i.e. for any morphism $\varphi : Y \to X$ we have $\varphi^* \circ \triangleright_X = \triangleright_Y \circ \varphi^*$. By the usual Yoneda argument we thus get a morphism $\triangleright : \Omega \to \Omega$ such that given $A \leq X$ with the characteristic map χ_A, the characteristic map of $\triangleright_X A$ is $\chi_A; \triangleright$.

We can compute $\triangleright : \Omega \to \Omega$ more explicitly. It is given by

$$\triangleright_\nu(\beta) = \min(\beta + 1, \nu).$$

1.6 Kripke-Joyal semantics

Let X be a sheaf and φ, ψ formulas in the internal language with a free variable of type X, i.e. $x : X \vdash \varphi : \Omega$ and $x : X \vdash \psi : \Omega$.

Let $\nu \leq \omega_1$ and $\xi \in X(\nu)$. Then

$\nu \Vdash \bot$ iff $\nu = 0$

$\nu \Vdash \top$ iff always

$\nu \Vdash \varphi$ iff $[\![\varphi_\nu(\xi)]\!] = {\downarrow}\nu$

$\nu \Vdash \varphi(\xi) \wedge \psi(\xi)$ iff $\nu \Vdash \varphi(\xi)$ and $\nu \Vdash \psi(\xi)$

$\nu \Vdash \varphi(\xi) \vee \psi(\xi)$ iff $\nu \Vdash \varphi(\xi)$ or $\nu \Vdash \psi(\xi)$ — if ν is a successor ordinal

$\nu \Vdash \varphi(\xi) \vee \psi(\xi)$ iff for all $\beta < \nu, \beta \Vdash \varphi(\xi|_\beta)$ or $\beta \Vdash \psi(\xi|_\beta)$ — if ν is a limit ordinal

$\nu \Vdash \varphi(\xi) \to \psi(\xi)$ iff for all $\beta \leq \nu, \beta \Vdash \varphi(\xi|_\beta)$ implies $\beta \Vdash \psi(\xi|_\beta)$

$\nu \Vdash \neg\varphi(\xi)$ iff for all $\beta \leq \nu, \beta \Vdash \varphi(\xi|_\beta)$ implies $\beta = 0$

Further, if $x : X, y : Y \vdash \varphi : \Omega$, $\nu \leq \omega_1$ and $\xi \in X(\nu)$ then

$\nu \Vdash \exists y, \varphi(\xi, y)$ iff there exists $\xi' \in Y(\nu), \nu \Vdash \varphi(\xi, \xi')$ if ν is a successor ordinal

$\nu \Vdash \exists y, \varphi(\xi, y)$ iff for all $\beta < \nu$ there exists $\xi_\beta \in Y(\beta), \beta \Vdash \varphi(\xi|_\beta, \xi_\beta)$ if ν is a limit ordinal

$\nu \Vdash \forall y, \varphi(\xi, y)$ iff for all $\beta \leq \nu$, for all $\xi_\beta \in Y(\beta), \beta \Vdash \varphi(\xi|_\beta, \xi_\beta)$

These are standard, cf. [6, Theorem VI.7.1]. The semantics of $\triangleright$ is as follows. Let $\varphi : \Omega^X$. Then

$$\nu \Vdash \triangleright\varphi(\alpha) \text{ iff for all } \beta < \nu, \beta \Vdash \varphi(\alpha|_\beta) \tag{1}$$

For successor ordinals $\nu = \nu' + 1$ this reduces to

$$\nu + 1 \Vdash \triangleright\varphi(\alpha) \text{ iff } \nu' \Vdash \varphi(\alpha|_{\nu'})$$

which is easy to check from the definition of $\triangleright$ above. For limit ordinals the characterization in (1) follows from the local character property (see below).

Note that $0 \Vdash \varphi$ for any φ and $1 \Vdash \triangleright\varphi$ for any φ. This is a bit different than the case for presheaves.

Local character By the local character property [6, VI.7 (page 316)] we have for any limit ordinal ξ

$$\xi \Vdash \varphi(\alpha) \text{ iff for all } \beta < \xi, \beta \Vdash \varphi(\alpha|_\beta).$$

Therefore to prove validity of a formula at a limit ordinal, it suffices to do so on all strictly smaller ordinals. This is essentially why one uses sheaves instead of presheaves, to transfer *local* properties to global ones.

Note that from the characterization in (1) it is immediate that $p \to \triangleright p$ for any p.

Proposition 1.6.1. *$\triangleright$ satisfies the Löb induction rule $\forall p : \Omega, (\triangleright p \to p) \to p$ and also satisfies the following properties.*

- *$\triangleright$ preserves $\wedge$, $\vee$, $\top$ and $\to$,*
- $\triangleright(\forall x : X, \varphi(x)) \to \forall x : X, \triangleright\varphi(x)$,
- $\exists x : X, \triangleright\varphi(x) \to \triangleright(\exists x : X, \varphi(x))$.

Proof. The proof of the Löb induction rule is by Kripke-Joyal semantics. More precisely, by induction on ν we prove that for all $\xi \in \Omega(\nu)$, for all $\beta \leq \nu$, $\beta \Vdash (\triangleright p \to p)(\xi|_\beta)$ implies $\xi|_\beta = \beta$.

- If $\nu = 0$ there is nothing to prove.
- If $\nu = \nu' + 1$ we consider two cases.
  - If $\beta \leq \nu'$ the result follows directly from the induction hypothesis.
  - If $\beta = \nu$ assume $\beta \Vdash (\triangleright p \to p)(\xi)$. We need to show $\xi = \beta$. By induction hypothesis and monotonicity $\xi|_{\nu'} = \nu'$, or in other words, $\nu' \Vdash \xi|_{\nu'}$. Since by assumption $\beta \Vdash \triangleright\xi$ implies $\beta \Vdash \xi$ we get $\beta = \xi$.

If ν is a limit ordinal the result follows from the local character property.

$\triangleright$ **preserves** $\wedge$ Given two subobjects $A, B \leq X$ with characteristic maps χ_A, χ_B, the characteristic map of $A \wedge B$ is given by $\langle \chi_A, \chi_B \rangle ; \wedge$, where $\wedge : \Omega \times \Omega \to \Omega$ is given by $\wedge_\nu(\beta, \beta') = \min\{\beta, \beta'\}$. It is thus easy to see that $\wedge$ commutes with $\triangleright$, i.e. that $\triangleright \times \triangleright ; \wedge = \wedge ; \triangleright$.

$\triangleright$ **preserves** $\to$ Similarly to $\wedge$, $\to$ is given on subobjects by composition with $\to$: $\Omega \times \Omega \to \Omega$ which is given as

$$\to_\nu (\beta, \beta') = \begin{cases} \nu & \text{if } \beta' \geq \beta \\ \beta' & \text{otherwise} \end{cases}$$

and since $\triangleright$ preserves order, i.e. $\beta' \geq \beta$ implies $\triangleright_\nu(\beta') \geq \triangleright_\nu(\beta)$, it is easy to see that $\to ; \triangleright = \triangleright \times \triangleright ; \to$.

$\triangleright$ **preserves** $\vee$ $\vee$ is given on subobjects by composition with $\vee : \Omega \times \Omega \to \Omega$ which is given by

$$\vee_\nu(\beta, \beta') = \max\{\beta, \beta'\}.$$

Using this it is easy to see that $\triangleright \times \triangleright ; \vee = \vee ; \triangleright$.

Universal quantifier Take $\nu \leq \omega_1$ and assume $\nu \Vdash \triangleright(\forall x : X, \varphi(x))$. Take $\beta \leq \nu$ and $x_\beta \in X(\beta)$. We are to show $\beta \Vdash \triangleright\varphi(x_\beta)$. Let $\beta' < \beta$. We are to show $\beta' \Vdash \varphi(x_\beta|_{\beta'})$. Since $\beta' < \nu$ we have that $\beta' \Vdash \forall x : X, \varphi(x)$. The conclusion follows.

Existential quantifier Take $\nu \leq \omega_1$ and assume $\nu \Vdash \exists x : X, \triangleright\varphi(x)$. We are to show $\nu \Vdash \triangleright(\exists x : X, \varphi(x))$.

Let $\beta < \nu$. We have to show $\beta \Vdash \exists x : X, \varphi(x)$.

Suppose ν is a successor ordinal. Using the assumption we have that there exists $x_\nu \in X(\nu)$, such that $\nu \Vdash \triangleright\varphi(x_\nu)$ which implies in particular that $\beta \Vdash \varphi(x_\nu|_\beta)$. Choosing $x_\nu|_\beta \in X(\beta)$ we have that $\beta \Vdash \exists x : X, \varphi(x)$.

Suppose ν is a limit ordinal. Let $\beta < \nu$. Then $\beta + 1 < \nu$ and by assumption there exists a $x_{\beta+1} \in X(\beta + 1)$ such that $\beta \Vdash \varphi\left(x_{\beta+1}|_\beta\right)$. Thus $\beta \Vdash \exists x : X, \varphi(x)$.

□

Proposition 1.6.2. *If X is total then $(\forall x : X, \triangleright\varphi(x)) \to \triangleright(\forall x : X, \varphi(x))$ holds.*

Proof. Take $\nu < \omega_1$ and assume $\nu \Vdash \forall x : X, \triangleright\varphi(x)$. Take $\beta < \nu$. We are to show $\beta \Vdash \forall x : X, \varphi(x)$. Take $\xi \leq \beta$ and $x_\xi \in X(\xi)$. Since X is total there is $x_{\xi+1} \in X(\xi + 1)$ that restricts to x_ξ. Since $\xi + 1 \leq \nu$ we have $\xi + 1 \Vdash \triangleright\varphi(x_{\xi+1})$, thus $\xi \Vdash \varphi(x_\xi)$. □

Remark 1.6.3. It is *not* the case that iff X is total and inhabited then $\triangleright(\exists x : X, \varphi(x)) \to \exists x : X, \triangleright\varphi(x)$ holds. In fact, even if X is a constant sheaf this does not necessarily hold.

Proof. Let $X = \delta(\mathbb{N})$. The constant sheaf of natural numbers is

$$1 \longleftarrow \mathbb{N} \xleftarrow{\text{id}} \mathbb{N} \xleftarrow{\text{id}} \mathbb{N} \xleftarrow{\text{id}} \mathbb{N} \xleftarrow{\text{id}} \cdots$$

Let $A \leq X$ be the subobject defined as follows

$$\begin{aligned} A(0) &= 1 \\ A(n) &= \{k \mid n \leq k < \omega\} && \text{for } 0 < n < \omega \\ A(\nu) &= \emptyset && \text{for } \omega \leq \nu < \omega_1 \end{aligned}$$

A is a subsheaf of X. Let φ be the characteristic map of A. If $\triangleright(\exists x : X, \varphi(x)) \to \exists x : X, \triangleright\varphi(x)$ were to hold it would hold at stage $\omega + 1$. But $\omega + 1 \Vdash \triangleright(\exists x : X, \varphi(x))$ since for all $n < \omega$, exists $n \in A(n)$, i.e. we can choose $x_n = n$ and then $n \Vdash \varphi(x_n)$.

But notice that $X(\omega + 1)$ is empty, hence the right hand side of the implication is false. □

Remark 1.6.4. The last remark has implications for fixed points. It seems that the internal Banach fixed point theorem does not hold in the same generality as it does in the topos of trees [4], or rather the proof that works for the topos of trees does not carry over, since it uses the fact that later commutes over existentials for total and inhabited objects.

This state of affairs makes intuitive sense, since [4, Lemma 2.10] makes no intuitive sense for $\mathbf{Sh}(\omega_1)$ since it implies that at any stage we can get a fixed point by a finite iteration, something we would not expect to hold in $\mathbf{Sh}(\omega_1)$.

1.7 Guarded recursive predicates

We now show that suitably *internally contractive* functions have unique fixed-points, thus establishing the existence of recursively defined predicates and relations.

Definition 1.7.1. *Define*

$$\begin{aligned}\mathrm{Inhab}(X) &= \exists x : X.\top \\ \mathrm{Total}(X) &= \forall x : \blacktriangleright X, \exists x' : X, \mathrm{next}^X(x') = x\end{aligned}$$

If $\vdash \mathrm{Inhab}(X)$ then each $X(\nu)$ is non-empty, but this *does not mean* that a global section exists. If $\vdash \mathrm{Total}(X)$ then the restriction maps are surjections (which implies that each stage $X(\nu)$ is inhabited, since X is a sheaf, i.e. $X(1) = 1$).

Proposition 1.7.2 (Internal Banach fixed point theorem)**.** *The following holds in the internal logic of* $\mathbf{Sh}(\omega_1)$.

$$\mathrm{Total}(X) \to \forall f : X^X, \mathrm{Contr} f \to \exists! x : X, f(x) = x$$

Proof. We use the Kripke-Joyal forcing semantics and proceed by induction on ν. If $\nu = 0$ there is nothing to prove. Let $\nu = \nu' + 1$ and assume $\nu \Vdash \mathrm{Total}(X)$. Let $f \in X^X(\nu)$, $\beta \leq \nu$ and $\beta \Vdash \mathrm{Contr} f$. We are to show $\beta \Vdash \exists! x : X, f(x) = x$. We define a sequence of elements $e_\xi \in X(\xi)$ for $\xi \leq \beta$ as follows.

$$\begin{aligned}e_0 &= f_0(*) \\ e_{\xi+1} &= f_{\xi+1}(r_\xi^{-1}(e_\xi)) \\ e_\alpha &= \lim_{\xi<\alpha} e_\xi\end{aligned}$$

This requires some explanations. First, r are the restriction maps of X. Second, α is again a limit ordinal and the limit means the unique element of $X(\alpha)$ that exists by the sheaf condition. Third, since $\xi + 1$ in the definition is smaller or equal to ν we can use the assumption that X is total with the element e_ξ to get some element $r_\xi^{-1}(e_\xi) \in X(\xi + 1)$ that restricts to e_ξ.

First we observe that the choice of $r_\xi^{-1}(e_\xi) \in X(\xi + 1)$ does not matter. This follows from contractivness of f since if u, v both restrict to e_ξ then $f_{\xi+1}(u) = f_{\xi+1}(v)$.

Further, we have that $f_\xi(e_\xi) = e_\xi$ and that $r_\xi(e_{\xi+1}) = e_\xi$. For $\xi = 0$ this is obvious. For successor ordinals we have

$$f_{\xi+1}(e_{\xi+1}) = f_{\xi+1}(f_{\xi+1}(r_\xi^{-1}(e_\xi)))$$

Since f is contractive it suffices to show $r_\xi(e_{\xi+1}) = r_\xi(f_{\xi+1}(r_\xi^{-1}(e_\xi)))$ and this follows by naturality of f and the induction hypothesis.

And

$$r_\xi(e_{\xi+1}) = r_\xi(f_{\xi+1}(r_\xi^{-1}(e_\xi))) = f_\xi(e_\xi) = e_\xi.$$

For limit ordinals the restrictions are automatic. To see that $f_\alpha(e_\alpha) = e_\alpha$ we show that $f_\alpha(e_\alpha)$ restricts to the same elements and this is immediate by naturality of the family f.

Thus there exist a $x_\beta \in X(\beta)$ such that $f_\beta(x_\beta) = x_\beta$. Uniqueness is shown similarly to uniqueness of external fixed points in the proof of Proposition 1.5.2. □

1.8 Predicates defined as least and greatest fixed points

Inductively and coinductively defined predicates are constructed as least and greatest fixed points of suitable maps of type $\mathcal{P}(X) \to \mathcal{P}(X)$ for a suitable X, giving a predicate on X. We will show that these predicates are $\neg\neg$ closed provided the defining functions are sufficiently tame.

Greatest fixed points

Proposition 1.8.1. *Suppose φ is a predicate on $\Gamma, x : X$ and suppose that the sequent*

$$\Gamma \mid \emptyset \vdash \forall x : X, \varphi \leftrightarrow F(\varphi)$$

holds where $F(\varphi)$ is a well-formed formula in context $\Gamma, x : X$ consisting of $\forall, \to, \wedge, \vee, \top, \bot$ and existential quantification over total types and using only $\neg\neg$-closed relational symbols.

Then

$$\Gamma \mid \emptyset \vdash \forall x : X, \neg\neg\varphi \leftrightarrow F(\neg\neg\varphi)$$

also holds.

Proof. From

$$\Gamma \mid \emptyset \vdash \forall x : X, \varphi \leftrightarrow F(\varphi)$$

we immediately get

$$\Gamma \mid \emptyset \vdash \neg\neg\left(\forall x : X, \varphi \leftrightarrow F(\varphi)\right)$$

and using the fact that $\neg\neg$ commutes with implication, universal quantifiers and conjuction we get

$$\Gamma \mid \emptyset \vdash \forall x : X, \neg\neg\varphi \leftrightarrow \neg\neg F(\varphi)$$

and now the restrictions on F guarantee that $\neg\neg F(\varphi) = F(\neg\neg\varphi)$, concluding the proof. □

As a consequence, suppose we define a predicate φ on X coinductively as the greatest fixed point of some $\mathcal{F} : \mathcal{P}(X) \to \mathcal{P}(X)$. Since $\varphi \to \neg\neg\varphi$ and if φ is a fixed point then also $\neg\neg$ is, we have that φ is $\neg\neg$-closed.

Least fixed points

Proposition 1.8.2. *Suppose φ is a predicate on $\Gamma, x : X$ and suppose that the sequent*

$$\Gamma \mid \emptyset \vdash \forall x : X, F(\varphi) \to \varphi$$

holds where $F(\varphi)$ is a well-formed formula in context $\Gamma, x : X$ consisting of $\forall, \to, \wedge, \vee, \top, \bot$ and existential quantification over total types and using only $\neg\neg$-closed relational symbols where in addition φ only occurs positively, i.e. F preserves implication.

Then

$$\Gamma \mid \emptyset \vdash \forall x : X, F(\Box\varphi) \to \Box\varphi$$

also holds.

Proof. Since $\Box$ does not behave as well as $\neg\neg$ we have to prove this using the model. Our assumption gives us that $[\![F(\varphi)]\!] \leq [\![\varphi]\!]$ in the fiber over $\Gamma, x : X$ and we wish to show $[\![F(\Box\varphi)]\!] \leq [\![\Box\varphi]\!] = \Box [\![\varphi]\!]$. Since $\Box$ is the right adjoint to $\neg\neg$ we have

$$[\![F(\Box\varphi)]\!] \leq \Box [\![\varphi]\!] \leftrightarrow \neg\neg [\![F(\Box\varphi)]\!] = [\![\neg\neg F(\Box\varphi)]\!] \leq [\![\varphi]\!]$$

and since the assumptions on F guarantee that $\neg\neg F(\varphi) = F(\neg\neg\varphi)$ we have

$$\leftrightarrow [\![F(\neg\neg\Box\varphi)]\!] \leq [\![\varphi]\!]$$

which by properties of $\neg\neg$ and $\Box$ gives us

$$\leftrightarrow [\![F(\Box\varphi)]\!] \leq [\![\varphi]\!]$$

which holds by the fact that $\Box\varphi \to \varphi$ and the assumption on monotonicity of F. □

It is an easy fact to show that least fixed points of monotone functions are exactly their least prefixed points. Using Proposition 1.8.2 and the facts that $\Box$ is deflationary and for any φ, $\Box\varphi$ is $\neg\neg$-closed we can conclude that inductively defined predicates are constant, provided they can be defined using a suitable subset of the logic.

Consequence for the external interpretation Since Π_1 is a logical functor it preserves structure of an elementary topos. More precisely, $\Pi_1 : \mathbf{Sh}(\omega_1) \to \mathbf{Set}$ gives rise to the morphism of higher-order fibrations

$$\begin{array}{ccc} \mathbf{Sub}_{\mathbf{Sh}(\omega_1)} & \longrightarrow & \mathbf{Sub}_{\mathbf{Set}} \\ \Big\downarrow{\scriptstyle \mathbf{cod}} & & \Big\downarrow{\scriptstyle \mathbf{cod}} \\ \mathbf{Sh}(\omega_1) & \xrightarrow{\Pi_1} & \mathbf{Set} \end{array}$$

that preserves the structure of a higher-order fibration (i.e. constructs to model higher-order logic). In particular this means it preserves validity of formulas in the internal language that use only the usual connectives of higher-order logic (i.e. everything but $\triangleright$ and $\Box$).

Thus given a formula φ of higher-order logic (i.e. no $\Box$ and no $\triangleright$) in some context Γ and choosing some interpretations of relational symbols, the denotation $\Pi_1([\![\varphi]\!])$ as a subobject of $\Pi_1([\![\Gamma]\!])$ is the same as the denotation of φ in **Set**, replacing the interpretations of relational symbols and types by their values at stage 1.

For instance if

$$x : \Delta(a), y : \Delta(b) \mid \emptyset \vdash \forall z : \mathcal{P}(\Delta(a)), \exists w : \mathcal{P}(\Delta(b)), P(x, w) \to Q(z, y)$$

holds in the logic of $\mathbf{Sh}(\omega_1)$ with P and Q being interpreted as some $\Delta(p)$ and $\Delta(q)$, respectively, then

$$x : a, y : b \mid \emptyset \vdash \forall z : \mathcal{P}(a), \exists w : \mathcal{P}(b), P(x, w) \to Q(z, y)$$

holds with P being interpreted as p and Q as q (we used the that $\Pi_1 \circ \Delta = \mathrm{id}_{\mathbf{Set}}$).

Combining this with the construction of fixed points we get that least and greatest fixed points constructed internally are mapped correspond to external least and greatest fixed points.

Proposition 1.8.3. *Let φ be a predicate symbol on $\Delta(a)$ and let $F(\varphi)$ be a formula in context $x : \Delta(a)$. Suppose $F(\varphi)$ is monotone in φ and suppose it involves only quantifiers over constant sets and constant predicate symbols. If*

$$\emptyset \mid \emptyset \vdash (\forall x : \Delta(a), \varphi(x) \leftrightarrow F(\varphi)) \tag{2}$$

$$\wedge \left(\forall \psi : \mathcal{P}\left(\Delta(a)\right), (\forall x : \Delta(a), \psi(x) \leftrightarrow F(\psi)) \rightarrow (\forall x : \Delta(a), \psi(x) \rightarrow \varphi(x))\right) \tag{3}$$

holds in the logic of $\mathbf{Sh}\left(\omega_1\right)$ *then*

$$\emptyset \mid \emptyset \vdash (\forall x : a, \varphi(x) \leftrightarrow \Pi_1(F)(\varphi)) \tag{4}$$

$$\wedge \left(\forall \psi : \mathcal{P}\left(a\right), (\forall x : a, \psi(x) \leftrightarrow \Pi_1(F)(\psi)) \rightarrow (\forall x : a, \psi(x) \rightarrow \varphi(x))\right) \tag{5}$$

holds in the logic of $\mathbf{Set}$ *with interpretations of relational symbols being Π_1-images of their chosen interpretations in* $\mathbf{Sh}\left(\omega_1\right)$. *Here $\Pi_1(F)$ means replacing occurrences of $\Delta(b)$ in quantifiers with b.*

Remark 1.8.4. Note that if φ is a predicate on $\Delta(a)$ (in the empty context) then $\varphi(x)$ is a formula in context $x : \Delta(a)$. These formulas characterize φ as the greatest predicate (or subset) satisfying F. We can state and prove an analogous property for least fixed points.

From Proposition 1.8.1 we have that if formulas (2) and (3) hold then $\varphi = \neg\neg \circ \varphi$. Thus $x : \Delta(a) \mid \emptyset \vdash \varphi(x)$ corresponds to a constant subobject of $\Delta(a)$, say $\Delta\left(\underline{\varphi}\right)$. Formulas (4) and (5) then state that $\underline{\varphi}$ is the greatest fixed point of "the same" formula.

1.9 Transitive closure

We prove that given a $\neg\neg$ closed relation R on a set with decidable equality its reflexive and transitive closure R^* is also $\neg\neg$-closed.

Lemma 1.9.1. *Let $R \subseteq X \times X$ be a relation on a decidable total type X. If R is $\neg\neg$-closed (meaning for all $x, x' : X$, $(x, x') \in R \leftrightarrow \neg\neg((x, x') \in R)$) then the reflexive transitive closure R^* is also $\neg\neg$-closed.*

Proof. By construction $R^* = \bigvee_{n \in \mathbb{N}} R^n$ where the sequence $\{R^n\}_{n \in \mathbb{N}}$ is defined by induction as

$$R^0 = \{(x, x') \mid x = x'\}$$
$$R^{n+1} = \{(x, x'') \mid \exists x' : X, (x, x') \in R \wedge (x', x'') \in R^n\}$$

Thus it suffices to show that all R^n are $\neg\neg$-closed and we do this by induction. R^0 is $\neg\neg$-closed since X is assumed to have decidable equality. Assuming R^n is $\neg\neg$-closed we have

$$\neg\neg((x, x') \in R^{n+1}) \leftrightarrow \neg\neg\exists x' : X, (x, x') \in R \wedge (x', x'') \in R^n$$

and since X is assumed to be total we have by Lemma 1.3.7 and the fact that $\neg\neg$ commutes with conjunction that

$$\leftrightarrow \exists x' : X, \neg\neg((x, x') \in R) \wedge \neg\neg((x', x'') \in R^n)$$

which is by assumption on R and the induction hypothesis equivalent to

$$\leftrightarrow \exists x' : X, (x, x') \in R \wedge (x', x'') \in R^n$$

which is by definition of R^{n+1} equivalent to

$$\leftrightarrow (x, x') \in R^{n+1}.$$

□

In particular, all constant sets are total and decidable and so the lemma applies.

2 The model of a language with countable choice

In this section we introduce $\mathbf{F}^{\mu,?}$, a call-by-value functional language akin to System F, i.e., with impredicative polymorphism, existential and general recursive types, extended with a countable choice expression `?`. We work informally in the internal logic of $\mathbf{Sh}(\omega_1)$ outlined above except where explicitly stated.

2.1 Syntax and operational semantics

The syntax of types, terms and values is defined in Figure 1. These should be understood as initial algebras of suitable polynomial functors.

$$\begin{aligned}
\tau &::= \alpha \mid \mathbf{1} \mid \tau_1 \times \tau_2 \mid \tau_1 + \tau_2 \mid \tau_1 \to \tau_2 \mid \mu\alpha.\tau \mid \forall\alpha.\tau \mid \exists\alpha.\tau \\
v &::= x \mid \langle\rangle \mid \langle v_1, v_2\rangle \mid \lambda x.e \mid \texttt{inl}\ v \mid \texttt{inr}\ v \mid \Lambda.e \mid \texttt{pack}\, v \\
e &::= x \mid \langle\rangle \mid \langle e_1, e_2\rangle \mid \lambda x.e \mid \texttt{inl}\ e \mid \texttt{inr}\ e \mid \Lambda.e \mid \texttt{pack}\, e \\
&\quad \mid \texttt{?} \mid \texttt{proj}_i\, e \mid e_1\, e_2 \mid \texttt{case}\,(e, x_1.e_1, x_2.e_2) \mid e[] \mid \texttt{unpack}\ e_1\ \texttt{as}\ x\ \texttt{in}\ e_2 \\
&\quad \mid \texttt{unfold}\, e \mid \texttt{fold}\, e \\
E &::= - \mid \langle E, e\rangle \mid \langle v, E\rangle \mid \texttt{inl}\ E \mid \texttt{inr}\ E \mid \texttt{pack}\, E \\
&\quad \mid \texttt{proj}_i\, E \mid E\, e \mid v\, E \mid \texttt{case}\,(E, x_1.e_1, x_2.e_2) \mid E[] \mid \texttt{unpack}\ E\ \texttt{as}\ x\ \texttt{in}\ e \\
&\quad \mid \texttt{unfold}\, E \mid \texttt{fold}\, E \\
C &::= - \mid \langle C, e\rangle \mid \langle e, C\rangle \mid \lambda x.C \mid \texttt{inl}\ C \mid \texttt{inr}\ C \mid \Lambda.C \mid \texttt{pack}\, C \\
&\quad \mid \texttt{proj}_i\, C \mid C\, e_2 \mid e\, C \mid \texttt{case}\,(C, x_1.e_1, x_2.e_2) \mid \texttt{case}\,(e, x_1.C, x_2.e_2) \\
&\quad \mid \texttt{case}\,(e, x_1.e_1, x_2.C) \mid C[] \mid \texttt{unpack}\ C\ \texttt{as}\ x\ \texttt{in}\ e \mid \texttt{unpack}\ e\ \texttt{as}\ x\ \texttt{in}\ C \\
&\quad \mid \texttt{unfold}\, C \mid \texttt{fold}\, C
\end{aligned}$$

Figure 1: Types, terms and evaluation contexts

The types include polymorphic, existential and recursive types. The terms are standard, apart from the countable choice expression `?`. We assume disjoint, countably infinite sets of *type variables*, ranged over by α, and *term variables*, ranged over by x. The free type variables of types and terms, $ftv(\tau)$ and $ftv(e)$, and free term variables $fv(e)$, are defined in the usual way. The notation $(\cdot)[\vec{\tau}/\vec{\alpha}]$ denotes the simultaneous capture-avoiding substitution of types $\vec{\tau}$ for the free type variables $\vec{\alpha}$ in types and terms; similarly, $e[\vec{v}/\vec{x}]$ denotes simultaneous capture-avoiding substitution of values $\vec{v}$ for the free term variables $\vec{x}$ in e.

Composition of evaluation contexts Evaluation contexts can be composed. Given two evaluation contexts E, E' we define $E \circ E'$ by induction on E as follows

$$
\begin{aligned}
[] \circ E' &= E' \\
\langle E, e\rangle \circ E' &= \langle E \circ E', e\rangle \\
\langle v, E\rangle \circ E' &= \langle v, E \circ E'\rangle \\
(\mathtt{proj}_i E) \circ E' &= \mathtt{proj}_i E \circ E' \\
(\mathtt{inl}\, E) \circ E' &= \mathtt{inl}\, (E \circ E') \\
(\mathtt{inr}\, E) \circ E' &= \mathtt{inr}\, (E \circ E') \\
(\mathtt{fold}\, E) \circ E' &= \mathtt{fold}\, (E \circ E') \\
(\mathtt{unfold}\, E) \circ E' &= \mathtt{unfold}\, (E \circ E') \\
(\mathtt{pack}\, E) \circ E' &= \mathtt{pack}\, (E \circ E') \\
(\mathtt{unpack}\ E\ \mathtt{as}\ x\ \mathtt{in}\ e) \circ E' &= \mathtt{unpack}\ (E \circ E')\ \mathtt{as}\ x\ \mathtt{in}\ e \\
(\mathtt{case}\, (E, x_1.e_1, x_2.e_2)) \circ E' &= \mathtt{case}\, (E \circ E', x_1.e_1, x_2.e_2) \\
(\mathtt{case}\, (E, x_1.e_1, x_2.e_2)) \circ E' &= \mathtt{case}\, (E \circ E', x_1.e_1, x_2.e_2) \\
(E\, e) \circ E' &= (E \circ E')\, e \\
(v\, E) \circ E' &= v\, (E \circ E')
\end{aligned}
$$

Lemma 2.1.1. *For any pair of evaluation contexts E, E' and an expression e,*

$$E[E'[e]] = (E \circ E')[e]$$

Types We define the type of natural numbers $\mathtt{nat}$ as $\mathtt{nat} = \mu\alpha.1 + \alpha$ and the corresponding numerals as $\underline{0} = \mathtt{fold}\, (\mathtt{inl}\ \langle\rangle)$ and $\underline{n+1} = \mathtt{fold}\, (\mathtt{inr}\ \underline{n})$ by induction on n.

The type system is defined in Figure 3. The judgment $\Delta \vdash \tau$ is defined in Figure 2. The judgments are mostly standard, apart from the typing of $\mathtt{?}$.

$$
\frac{\alpha \in \Delta}{\Delta \vdash \alpha} \qquad \Delta \vdash \mathbf{1}
$$

$$
\frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 \times \tau_2} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 + \tau_2} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 \to \tau_2}
$$

$$
\frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \exists\alpha.\tau} \qquad \frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \forall\alpha.\tau} \qquad \frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \mu\alpha.\tau}
$$

Figure 2: Well-formed types. The judgment $\Delta \vdash \tau$ means $ftv(\tau) \subseteq \Delta$.

We write $\mathbf{Type}(\Delta)$ for the set of types well-formed in context Δ and $\mathbf{Type}$ for the set of closed types τ, i.e., where $ftv(\tau) = \emptyset$. We write $\mathbf{Val}\,(\tau)$ and $\mathbf{Tm}\,(\tau)$ for the sets of *closed* values and terms of type τ, respectively. We write $\mathbf{Val}$ and $\mathbf{Tm}$ for the set of all closed values and closed terms, respectively. $\mathbf{Stk}\,(\tau)$ denotes the set of τ-accepting evaluation contexts, i.e. evaluation contexts E, such that given any closed term e of type τ, $E[e]$ is a typeable term. $\mathbf{Stk}$ denotes the set of all evaluation contexts.

$$\frac{x{:}\tau \in \Gamma \qquad \Delta \vdash \Gamma}{\Delta;\Gamma \vdash x : \tau} \qquad \frac{\Delta \vdash \Gamma}{\Delta;\Gamma \vdash \langle\rangle : \mathbf{1}} \qquad \frac{\Delta;\Gamma \vdash e_1 : \tau_1 \qquad \Delta;\Gamma \vdash e_2 : \tau_2}{\Delta;\Gamma \vdash \langle e_1, e_2\rangle : \tau_1 \times \tau_2}$$

$$\frac{\Delta;\Gamma, x{:}\tau_1 \vdash e : \tau_2}{\Delta;\Gamma \vdash \lambda x.e : \tau_1 \to \tau_2} \qquad \frac{\Delta;\Gamma \vdash e : \tau_1 \qquad \Delta \vdash \tau_2}{\Delta;\Gamma \vdash \mathtt{inl}\ e : \tau_1 + \tau_2} \qquad \frac{\Delta;\Gamma \vdash e : \tau_2 \qquad \Delta \vdash \tau_1}{\Delta;\Gamma \vdash \mathtt{inr}\ e : \tau_1 + \tau_2}$$

$$\frac{\Delta;\Gamma, x_1{:}\tau_1 \vdash e_1 : \tau \qquad \Delta;\Gamma, x_2{:}\tau_2 \vdash e_2 : \tau \qquad \Delta;\Gamma \vdash e : \tau_1 + \tau_2}{\Delta\ \Gamma \vdash \mathtt{case}\,(e, x_1.e_1, x_2.e_2) : \tau}$$

$$\frac{\Delta, \alpha;\Gamma \vdash e : \tau}{\Delta;\Gamma \vdash \Lambda.e : \forall\alpha.\tau} \qquad \frac{\Delta;\Gamma \vdash e : \tau_1 \times \tau_2}{\Delta;\Gamma \vdash \mathtt{proj}_i\, e : \tau_i} \qquad \frac{\Delta;\Gamma \vdash e : \tau' \to \tau \qquad \Delta;\Gamma \vdash e' : \tau'}{\Delta;\Gamma \vdash e\, e' : \tau}$$

$$\frac{\Delta \vdash \tau_1 \qquad \Delta;\Gamma \vdash e : \tau[\tau_1/\alpha]}{\Delta;\Gamma \vdash \mathtt{pack}\, e : \exists\alpha.\tau}$$

$$\frac{\Delta;\Gamma \vdash e : \exists\alpha.\tau_1 \qquad \Delta \vdash \tau \qquad \Delta, \alpha;\Gamma, x : \tau_1 \vdash e' : \tau}{\Delta;\Gamma \vdash \mathtt{unpack}\ e\ \mathtt{as}\ x\ \mathtt{in}\ e' : \tau}$$

$$\frac{\Delta;\Gamma \vdash e : \mu\alpha.\tau}{\Delta;\Gamma \vdash \mathtt{unfold}\, e : \tau[\mu\alpha.\tau/\alpha]} \qquad \frac{\Delta;\Gamma \vdash e : \tau[\mu\alpha.\tau/\alpha]}{\Delta;\Gamma \vdash \mathtt{fold}\, e : \mu\alpha.\tau}$$

$$\frac{\Delta;\Gamma \vdash e : \forall\alpha.\tau \qquad \Delta \vdash \tau'}{\Delta;\Gamma \vdash e[] : \tau[\tau'/\alpha]} \qquad \frac{\Delta \vdash \Gamma}{\Delta;\Gamma \vdash \mathtt{?} : \mathtt{nat}}$$

Figure 3: Typing of terms, where $\Gamma ::= \emptyset \mid \Gamma, x{:}\tau$ and $\Delta ::= \emptyset \mid \Delta, \alpha$. The notation $\Delta \vdash \tau$ means that $\mathit{ftv}(\tau) \subseteq \Delta$.

For a typing context $\Gamma = x_1{:}\tau_1, \ldots, x_n{:}\tau_n$ with $\tau_1, \ldots, \tau_n \in \mathbf{Type}$, let

$$\mathbf{Subst}(\Gamma) = \{\gamma \in \mathbf{Val}^{\vec{x}} \mid \forall 1 \le i \le n.\ \gamma(x_i) \in \mathbf{Val}\,(\tau_i)\}$$

denote the set of type-respecting value substitutions. In particular, if $\Delta;\Gamma \vdash e : \tau$ then $\varnothing;\varnothing \vdash e\gamma : \tau\delta$ for any $\delta \in \mathbf{Type}^{\Delta}$ and $\gamma \in \mathbf{Subst}(\Gamma\delta)$, and the type system satisfies standard properties of progress and preservation and a canonical forms lemma.

The operational semantics of the language is given in Figure 4 by a reduction relation $e \rightsquigarrow e'$. In particular, the choice operator ? evaluates nondeterministically to any numeral $\underline{n}$ ($n \in \mathbb{N}$). We use evaluation contexts E, and write $E[e]$ for the term obtained by plugging e into E.

To define the logical relation we need further reduction relations. These will allow us to ignore most reductions in the definition of logical relation.

Let $\rightsquigarrow^*$ be the reflexive transitive closure of $\rightsquigarrow$. We call reductions of the form $\mathtt{unfold}\,(\mathtt{fold}\, v) \rightsquigarrow v$ *unfold-fold* reductions and reductions of the form $\mathtt{?} \rightsquigarrow \underline{n} \quad (n \in \mathbb{N})$ *choice* reductions. We define

- $e \overset{p}{\rightsquigarrow} e'$ if $e \rightsquigarrow^* e'$ and *none* of the reductions is a choice reduction
- $e \overset{0}{\rightsquigarrow} e'$ if $e \rightsquigarrow^* e'$ and *none* of the reductions is an unfold-fold reduction
- $e \overset{1}{\rightsquigarrow} e'$ if $e \rightsquigarrow^* e'$ and *exactly one* of the reductions is an unfold-fold reduction

Basic reductions $\longmapsto$

$$\begin{array}{rlrl}
\mathtt{proj}_i\,\langle v_1, v_2\rangle &\longmapsto v_i & \mathtt{unfold}\,(\mathtt{fold}\,v) &\longmapsto v \\
(\lambda x.e)\,v &\longmapsto e[v/x] & \mathtt{unpack}\ (\mathtt{pack}\,v)\ \mathtt{as}\ x\ \mathtt{in}\ e &\longmapsto e[v/x] \\
(\Lambda.e)[] &\longmapsto e & \mathtt{case}\,(\mathtt{inl}\,v, x_1.e_1, x_2.e_2) &\longmapsto e_1[v/x_1] \\
\mathtt{?} &\longmapsto \underline{n} \quad (n \in \mathbb{N}) & \mathtt{case}\,(\mathtt{inr}\,v, x_1.e_1, x_2.e_2) &\longmapsto e_2[v/x_2]
\end{array}$$

One step reduction relation $\rightsquigarrow$

$$E[e] \rightsquigarrow E[e'] \qquad \text{if } e \longmapsto e'$$

Figure 4: Operational semantics.

- $\overset{p,1}{\rightsquigarrow} = \overset{p}{\rightsquigarrow} \cap \overset{1}{\rightsquigarrow}$
- $\overset{p,0}{\rightsquigarrow} = \overset{p}{\rightsquigarrow} \cap \overset{0}{\rightsquigarrow}$

The $\overset{1}{\rightsquigarrow}$ reduction relation will be used in the stratified definition of divergence, see Section 2.2 and the other reduction relations will be used to state additional properties of the logical relation.

2.2 Termination relations

In order to define the $\top\top$ closure we need to define our observations. Recall that previously in [3] this was achieved by defining the termination relations $\Downarrow$ and $\downarrow$ and stratified versions of these, $\Downarrow_\beta$ and $\downarrow_n$ for $\beta < \omega_1$ and $n < \omega$. Focusing on $\Downarrow_\beta$, it is defined by induction on β as $e \Downarrow_\beta \leftrightarrow \forall e', e \overset{1}{\rightsquigarrow} e' \to \exists \nu < \beta, e' \Downarrow_\nu$. It is easy to see that $\Downarrow_\beta \subseteq \Downarrow_{\beta+1}$ so $\{\Downarrow_\beta\}_{\beta<\omega_1}$ does not define a subobject of the constant sheaf $\Delta(\mathbf{Tm})$.

But observe that defining $\uparrow_\beta = \mathbf{Tm} \setminus \Downarrow_\beta, \uparrow_\beta$ may be defined by induction on β as

$$e\!\uparrow_\beta \leftrightarrow \exists e', e \overset{1}{\rightsquigarrow} e' \wedge \forall \nu < \beta, e'\!\uparrow_\nu.$$

Using this, we define internally $\uparrow : \mathcal{P}(\mathbf{Tm})$ as the unique fixed point of $\Psi : \mathcal{P}(\mathbf{Tm}) \to \mathcal{P}(\mathbf{Tm})$ given by

$$\Psi(m) = \left\{ e : \mathbf{Tm} \;\middle|\; \exists e', e \overset{1}{\rightsquigarrow} e' \wedge \triangleright(m(e')) \right\}.$$

This is the stratified definition of *may-divergence* (there is a diverging path). We can also define (internally) non-stratified divergence predicate $\uparrow$ as the greatest fixed-point of $\Phi : \mathcal{P}(\mathbf{Tm}) \to \mathcal{P}(\mathbf{Tm})$ given as

$$\Phi(m) = \left\{ e : \mathbf{Tm} \;\middle|\; \exists e', e \rightsquigarrow e' \wedge m(e') \right\}.$$

Since Φ is monotone the greatest fixed point exists by Knaster-Tarski fixed-point theorem (which holds in any topos). Observe that Ψ is almost the same as $\Phi \circ \triangleright$, apart from using a different reduction relation.

Lemma 2.2.1. *Let $e, e' \in \mathbf{Tm}$. The following hold in the internal language.*

1. *if $e \overset{p}{\rightsquigarrow} e'$ then $e\!\uparrow \leftrightarrow e'\!\uparrow$*
2. *if $e \overset{p,0}{\rightsquigarrow} e'$ then $e\!\uparrow \leftrightarrow e'\!\uparrow$*

3. *if* $e \overset{0}{\rightsquigarrow} e'$ *then* $(e'\!\uparrow) \to e\!\uparrow$

4. *if* $e \overset{1}{\rightsquigarrow} e'$ *then* $\triangleright(e'\!\uparrow) \to e\!\uparrow$

Proof. 1. Suppose $e \overset{p}{\rightsquigarrow} e'$. If $e = e'$ there is nothing to do. Otherwise, the crucial property we have is that there exists a unique e'', such that $e \rightsquigarrow e''$ and $e'' \overset{p}{\rightsquigarrow} e'$. Using this property this case follows by induction on the number of steps in $\overset{p}{\rightsquigarrow}$.

2. This property follows from the fact that if $e \overset{p,0}{\rightsquigarrow} e'$ then $e \overset{1}{\rightsquigarrow} e''$ if and only if $e' \overset{1}{\rightsquigarrow} e''$ which is easy to see directly from the definition of these relations.

3. Suppose $e \overset{0}{\rightsquigarrow} e'$ and $e'\!\uparrow$. Then by definition there exists a e'', such that $e' \overset{1}{\rightsquigarrow} e''$ and $\triangleright(e''\!\uparrow)$. By definition of $\overset{1}{\rightsquigarrow}$ we also have $e \overset{1}{\rightsquigarrow} e''$ and so $e\!\uparrow$.

4. This follows directly from the definition of the $\uparrow$ relation.

□

2.3 Must-contextual and must-CIU preorders

Contexts can be typed as second-order terms, by means of a typing judgment of the form $C : (\Delta \mid \Gamma \Rrightarrow \tau) \to (\Delta' \mid \Gamma' \Rrightarrow \sigma)$, stating that whenever $\Delta \mid \Gamma \vdash e : \tau$ holds, $\Delta' \mid \Gamma' \vdash C[e] : \sigma$ also holds. The typing of contexts can be defined as an inductive relation defined by suitable typing rules, which we omit here since they are standard; see [1]. We write $C : (\Delta \mid \Gamma \Rrightarrow \tau)$ to mean there exists a type σ, such that $C : (\Delta \mid \Gamma \Rrightarrow \tau) \to (\varnothing \mid \varnothing \Rrightarrow \sigma)$ holds.

We define *contextual must-approximation* using the *may-divergence* predicate. This is in contrast with the definition in [3] which uses the must-convergence predicate. However externally, in the model, this definition is precisely the one used in [3].

Definition 2.3.1 (Must-contextual approximation)**.** *We define the must-contextual approximation relation as*

$$\Delta \mid \Gamma \vdash e_1 \lesssim^{ctx}_{\Downarrow} e_2 : \tau \overset{\triangle}{\leftrightarrow} \Delta \mid \Gamma \vdash e_1 : \tau \wedge \Delta \mid \Gamma \vdash e_2 : \tau \wedge \forall C, C : (\Delta \mid \Gamma \Rrightarrow \tau) \wedge C[e_2]\!\uparrow \to C[e_1]\!\uparrow$$

Note the order of terms in the implication: if $C[e_{\mathbf{2}}]$ may-diverges then $C[e_{\mathbf{1}}]$ may-diverges. This is the contrapositive of the usual definition which states that if $C[e_{\mathbf{1}}]$ must-converges then $C[e_{\mathbf{2}}]$ must-converges. Must-contextual approximation defined explicitly using contexts can be shown to be the largest compatible adequate and transitive relation, so it coincides with contextual approximation defined in [3].

In practice it is difficult to work with contextual approximation directly. An alternative characterization of the contextual approximation and equivalence relations can be given in terms of CIU preorders [7], which we define below. We will use the logical relation to prove that CIU and contextual approximations coincide and that both of them coincide with the logical approximation relation.

The definition of the CIU approximation is the same as in [3], only with changed termination relations. We state it here for completeness and reference.

Definition 2.3.2 (CIU approximation)**.** Must-CIU approximation, $\lesssim^{CIU}$*, is the type-indexed relation defined as follows: for all* e, e' *with* $\Delta; \Gamma \vdash e : \tau$ *and* $\Delta; \Gamma \vdash e' : \tau$*,*

$$\Delta \mid \Gamma \vdash e \lesssim^{CIU}_{\Downarrow} e' : \tau \;\leftrightarrow\; \forall \delta \in \mathbf{Type}(\Delta),\, \gamma \in \mathbf{Subst}(\Gamma\delta),\, E \in \mathbf{Stk}\,(\tau\delta)\,,\; E[\mathbf{e}'\gamma]\!\uparrow \to E[\mathbf{e}\gamma]\!\uparrow$$

Note again the order of terms e and e' in the implication.

It is simple to see that must-contextual approximation implies must-CIU approximation. However the converse is not so simple to see. We will prove it by constructing the logical relation and proving that all three relations coincide.

2.4 Logical approximation relation

We now define the logical relation, internalizing the definition in [3]. The major difference is that instead of using must-termination and stratified must-termination predicates we use may-divergence and stratified may-divergence predicates.

Relational interpretation of types Given two closed types $\tau, tau' \in \mathbf{Type}$ let $\mathbf{VRel}(\tau,\tau') = \mathcal{P}(\mathbf{Val}(\tau) \times \mathbf{Val}(\tau'))$, $\mathbf{TRel}(\tau,\tau') = \mathcal{P}(\mathbf{Tm}(\tau) \times \mathbf{Tm}(\tau'))$ and $\mathbf{SRel}(\tau,\tau') = \mathcal{P}(\mathbf{Stk}(\tau) \times \mathbf{Stk}(\tau'))$. We implicitly use the inclusion $\mathbf{VRel}(\tau,\tau') \subseteq \mathbf{TRel}(\tau,\tau')$. For a type variable context Δ we define $\mathbf{VRel}(\Delta)$ as the extension of $\mathbf{VRel}(\cdot,\cdot)$

$$\mathbf{VRel}(\Delta) = \left\{(\varphi_1,\varphi_2,\varphi_r) \;\middle|\; \varphi_1,\varphi_2 : \Delta \to \mathbf{Type}, \forall \alpha \in \Delta, \varphi_r(\alpha) \subseteq \mathbf{VRel}(\varphi_1(\alpha),\varphi_2(\alpha))\right\}$$

where the first two components give syntactic types for the left and right hand sides of the relation and the third component is a relation between those types.

The interpretation of types, $[\![\cdot \vdash \cdot]\!]$ is by induction on the judgement $\Delta \vdash \tau$. Given a judgment $\Delta \vdash \tau$ and $\varphi \in \mathbf{VRel}(\Delta)$ $[\![\Delta \vdash \tau]\!](\varphi) \subseteq \mathbf{VRel}(\varphi_1(\tau),\varphi_2(\tau))$ where the φ_1 and φ_2 are the first two components of φ and $\varphi_1(\tau)$ denotes substitution of types in φ_1 for free type variables in τ. It is defined below. For the sake of readability we omit the typing judgments in each case, but they should be understood to be there.

$$\begin{aligned}
[\![\Delta \vdash \alpha]\!](\varphi) &= \varphi_r(\alpha)\\
[\![\Delta \vdash \mathbf{1}]\!](\varphi) &= \mathbf{Id_1}\\
[\![\Delta \vdash \tau_1 \times \tau_2]\!](\varphi) &= \left\{(\langle v,u\rangle,\langle v',u'\rangle) \;\middle|\; (v,v') \in [\![\Delta \vdash \tau_1]\!](\varphi), (u,u') \in [\![\Delta \vdash \tau_2]\!](\varphi)\right\}\\
[\![\Delta \vdash \tau_1 + \tau_2]\!](\varphi) &= \left\{(\mathtt{inl}\, v, \mathtt{inl}\, v') \;\middle|\; (v,v') \in [\![\Delta \vdash \tau_1]\!](\varphi)\right\} \cup\\
&\qquad \left\{(\mathtt{inr}\, u, \mathtt{inr}\, u') \;\middle|\; (u,u') \in [\![\Delta \vdash \tau_2]\!](\varphi)\right\}\\
[\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi) &= \left\{(\lambda x.e, \lambda y.e') \;\middle|\; \forall (v,v') \in [\![\Delta \vdash \tau_1]\!](\varphi), (e[v/x], e'[v'/y]) \in [\![\Delta \vdash \tau_2]\!](\varphi)^{\top\top}\right\}\\
[\![\Delta \vdash \forall \alpha.\tau]\!](\varphi) &= \left\{(\Lambda.e, \Lambda.e') \;\middle|\; \forall \sigma,\sigma' \in \mathbf{Type}, \forall s \in \mathbf{VRel}(\sigma,\sigma'), (e,e') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto (\sigma,\sigma',s)])^{\top\top}\right\}\\
[\![\Delta \vdash \exists \alpha.\tau]\!]_{\varphi,\psi}(\varphi) &= \left\{(\mathtt{pack}\, v, \mathtt{pack}\, v') \;\middle|\; \exists \sigma,\sigma' \in \mathbf{Type}, \exists s \in \mathbf{VRel}(\sigma,\sigma'), (v,v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto (\sigma,\sigma',s)])\right\}\\
[\![\Delta \vdash \mu\alpha.\tau]\!](\varphi) &= \mathtt{fix}\left(\lambda s.\left\{(\mathtt{fold}\, v, \mathtt{fold}\, v') \;\middle|\; \triangleright((v,v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto s]))\right\}\right)
\end{aligned}$$

where the $\cdot^{\top\top} : \mathbf{VRel}(\tau,\tau') \to \mathbf{TRel}(\tau,\tau')$ is defined with the help of $\cdot^{\top} : \mathbf{VRel}(\tau,\tau') \to \mathbf{SRel}(\tau,\tau')$ as follows

$$\begin{aligned}
r^{\top} &= \left\{(E,E') \;\middle|\; \forall (v,v') \in r, E'[v']\!\uparrow \to E[v]\!\Updownarrow\right\}\\
r^{\top\top} &= \left\{(e,e') \;\middle|\; \forall (E,E') \in r^{\top}, E'[e']\!\uparrow \to E[e]\!\Updownarrow\right\}
\end{aligned}$$

The internal Banach's fixed point theorem implies that the interpretation of recursive types is well defined.

The following lemmata are simple to prove.

Lemma 2.4.1. $\cdot^{\top\top}$ *is monotone and inflationary, i.e.*

- *For all* r, $r \subseteq r^{\top\top}$
- *For all* r, s, $r \subseteq s \to r^{\top\top} \subseteq s^{\top\top}$

Lemma 2.4.2. *Let* $r \in \mathbf{VRel}(\tau, \tau')$.

- *If* $e \overset{p,0}{\rightsquigarrow} e_1$ *and* $e' \overset{p}{\rightsquigarrow} e_1'$ *then* $(e, e') \in r^{\top\top} \leftrightarrow (e_1, e_1') \in r^{\top\top}$.
- *If* $e \overset{1}{\rightsquigarrow} e_1$ *then for all* $e' \in \mathbf{Tm}(\tau')$, *if* $\triangleright((e_1, e') \in r^{\top\top})$ *then* $(e, e') \in r^{\top\top}$.

A crucial property of the interpretation of types is that it respects substitution and weakening.

Lemma 2.4.3 (Substitution)**.** *Let* $\Delta \vdash \tau$ *and* $\Delta, \alpha \vdash \sigma$. *Then*

$$[\![\Delta \vdash \sigma[\tau/\alpha]]\!](\varphi) = [\![\Delta, \alpha \vdash \sigma]\!](\varphi[\alpha \mapsto [\![\Delta \vdash \tau]\!](\varphi)]).$$

Lemma 2.4.4 (Weakening)**.** *Suppose* $\Delta \vdash \tau$. *Then for all* $\varphi \in \mathbf{VRel}(\Delta)$, $s \in \mathbf{VRel}(\sigma, \sigma')$, *for all* $\alpha \notin \Delta$,

$$[\![\Delta \vdash \tau]\!](\varphi) = [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto (\sigma, \sigma', s)]).$$

The actual "logical relation" is defined on open terms by reducing it to the above relations on closed terms by substitution. We first extend the interpretation of types to the interpretation of contexts as

$$[\![\Delta \vdash \Gamma]\!](\varphi) = \left\{(\gamma, \gamma') \;\middle|\; \gamma, \gamma' : \mathbf{Val}^{\mathrm{dom}(\Gamma)}, \forall x \in \mathrm{dom}(\Gamma), (\gamma(x), \gamma'(x)) \in [\![\Delta \vdash \Gamma(x)]\!](\varphi)\right\}$$

Definition 2.4.5 (Logical relation)**.**

$$\Delta; \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau = \forall \varphi \in \mathbf{VRel}(\Delta), \forall (\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!](\varphi), (e\gamma, e'\gamma') \in [\![\Delta \vdash \tau]\!](\varphi)^{\top\top}$$

2.4.1 Properties of relations

Definition 2.4.6 (Type-indexed relation)**.** *A* type-indexed relation $\mathcal{R}$ *is a set of tuples* $(\Delta, \Gamma, e, e', \tau)$ *such that* $\Delta \vdash \Gamma$ *and* $\Delta \vdash \tau$ *and* $\Delta \mid \Gamma \vdash e : \tau$ *and* $\Delta \mid \Gamma \vdash e' : \tau$. *We write* $\Delta; \Gamma \vdash e \mathcal{R} e' : \tau$ *for* $(\Delta, \Gamma, e, e', \tau) \in \mathcal{R}$.

Definition 2.4.7 (Precongruence)**.** *A type-indexed relation* $\mathcal{R}$ *is* reflexive *if* $\Delta; \Gamma \vdash e : \tau$ *implies* $\Delta; \Gamma \vdash e \mathcal{R} e : \tau$. *It is* transitive *if* $\Delta; \Gamma \vdash e \mathcal{R} e' : \tau$ *and* $\Delta; \Gamma \vdash e' \mathcal{R} e'' : \tau$ *implies* $\Delta; \Gamma \vdash e \mathcal{R} e'' : \tau$. *It is compatible if it is closed under the rules in Figure 5.*

A precongruence *is a reflexive, transitive and compatible type-indexed relation.*

$$\frac{}{\Delta;\Gamma \vdash x \mathrel{\mathcal{R}} x : \tau}\ x{:}\tau \in \Gamma \qquad \frac{}{\Delta;\Gamma \vdash \langle\rangle \mathrel{\mathcal{R}} \langle\rangle : \mathbf{1}}$$

$$\frac{\Delta;\Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau_1 \qquad \Delta;\Gamma \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau_2}{\Delta;\Gamma \vdash \langle e_1, e_2\rangle \mathrel{\mathcal{R}} \langle e_1', e_2'\rangle : \tau_1 \times \tau_2} \qquad \frac{\Delta;\Gamma, x{:}\tau_1 \vdash e \mathrel{\mathcal{R}} e' : \tau_2}{\Delta;\Gamma \vdash \lambda x.e \mathrel{\mathcal{R}} \lambda x.e' : \tau_1 \to \tau_2}$$

$$\frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_1}{\Delta;\Gamma \vdash \mathtt{inl}\, e \mathrel{\mathcal{R}} \mathtt{inl}\, e' : \tau_1 + \tau_2} \qquad \frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_2}{\Delta;\Gamma \vdash \mathtt{inr}\, e \mathrel{\mathcal{R}} \mathtt{inr}\, e' : \tau_1 + \tau_2}$$

$$\frac{\Delta;\Gamma, x_1{:}\tau_1 \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau \qquad \Delta;\Gamma, x_2{:}\tau_2 \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau \qquad \Delta;\Gamma \vdash e\ e' {:}\ \tau_1 + \tau_2}{\Delta\ \Gamma \vdash \mathtt{case}\,(e, x_1.e_1, x_2.e_2) \mathrel{\mathcal{R}} \mathtt{case}\,(e', x_1.e_1', x_2.e_2') : \tau}$$

$$\frac{\Delta, \alpha;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau}{\Delta;\Gamma \vdash \Lambda.e \mathrel{\mathcal{R}} \Lambda.e' : \forall\alpha.\tau} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau[\tau_1/\alpha]}{\Delta;\Gamma \vdash (\mathtt{pack}\, e) \mathrel{\mathcal{R}} (\mathtt{pack}\, e') : \exists\alpha.\tau}$$

$$\frac{\Delta;\Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \exists\alpha.\tau_1 \qquad \Delta \vdash \tau \qquad \Delta, \alpha;\Gamma, x : \tau_1 \vdash e \mathrel{\mathcal{R}} e' : \tau}{\Delta;\Gamma \vdash (\mathtt{unpack}\ e_1\ \mathtt{as}\ x\ \mathtt{in}\ e) \mathrel{\mathcal{R}} (\mathtt{unpack}\ e_1'\ \mathtt{as}\ x\ \mathtt{in}\ e') : \tau}$$

$$\frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_1 \times \tau_2}{\Delta;\Gamma \vdash \mathtt{proj}_i\, e \mathrel{\mathcal{R}} \mathtt{proj}_i\, e' : \tau_i} \qquad \frac{\Delta;\Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau' \to \tau \qquad \Delta;\Gamma \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau'}{\Delta;\Gamma \vdash e_1\, e_2 \mathrel{\mathcal{R}} e_1'\, e_2' : \tau}$$

$$\frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \mu\alpha.\tau}{\Delta;\Gamma \vdash \mathtt{unfold}\, e \mathrel{\mathcal{R}} \mathtt{unfold}\, e' : \tau[\mu\alpha.\tau/\alpha]} \qquad \frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau[\mu\alpha.\tau/\alpha]}{\Delta;\Gamma \vdash \mathtt{fold}\, e \mathrel{\mathcal{R}} \mathtt{fold}\, e' : \mu\alpha.\tau}$$

$$\frac{\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \forall\alpha.\tau}{\Delta;\Gamma \vdash e[] \mathrel{\mathcal{R}} e'[] : \tau[\tau'/\alpha]}\ ftv(\tau') \subseteq \Delta \qquad \frac{}{\Delta;\Gamma \vdash \mathtt{?} \mathrel{\mathcal{R}} \mathtt{?} : \mathtt{nat}}$$

Figure 5: Compatibility properties of type-indexed relations

2.4.2 The fundamental property

To prove the fundamental property (reflexivity) we start with some simple properties relating evaluation contexts and relations. The proof of the compatibility properties in most of the cases will be a simple consequence of these lemmata.

The following is a direct consequence of the fact that $p \to \triangleright p$ for any $p : \Omega$, we only state it here for reference.

Lemma 2.4.8. *The interpretations of types satisfy the following monotonicity properties.*

- *If* $(v, v') \in [\![\Delta \vdash \tau]\!]\,(\varphi)$ *then* $\triangleright\,((v, v') \in [\![\Delta \vdash \tau]\!]\,(\varphi))$.
- *If* $(e, e') \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top\top}$ *then* $\triangleright\left((e, e') \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top\top}\right)$.
- *If* $(E, E') \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top}$ *then* $\triangleright\left((E, E') \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top}\right)$.

Lemma 2.4.9. *If* $(v, v') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!]\,(\varphi)$ *and* $(E, E') \in [\![\Delta \vdash \tau_2]\!]\,(\varphi)^{\top}$ *then* $(E \circ (v\,[]), E' \circ (v'\,[])) \in [\![\Delta \vdash \tau_1]\!]\,(\varphi)^{\top}$.

This follows directly from the definition of the interpretation of types, Lemma 2.2.1 and Lemma 2.1.1.

Corollary 2.4.10. *If* $(e, e') \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top\top}$ *and* $(E, E') \in [\![\Delta \vdash \tau_2]\!](\varphi)^{\top}$ *then*

$$(E \circ ([]\, e), E' \circ ([]\, e')) \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)^{\top}.$$

Proof. Take $(v, v') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)$. By Lemma 2.4.9 $(E \circ (v\, []), E' \circ (v'\, [])) \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top}$ so using Lemma 2.1.1 we have

$$E'[v'\, e']\uparrow \to E[v\, e]\Uparrow$$

concluding the proof. □

Corollary 2.4.11. *If* $(e, e') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)^{\top\top}$ *and* $(E, E') \in [\![\Delta \vdash \tau_2]\!](\varphi)^{\top}$ *then* $(E \circ ((\lambda x.e\, x)\, []), E' \circ ((\lambda x.e'\, x)$ $[\![\Delta \vdash \tau_1]\!](\varphi)^{\top}$.

Proof. If $(e, e') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)^{\top\top}$ then $(\lambda x.e, \lambda x.e') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)$. Then use Lemma 2.4.9. □

Lemma 2.4.12. *If* $(E, E') \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\varphi)^{\top}$ *then*

$$(E \circ (\mathtt{unfold}\, []), E' \circ (\mathtt{unfold}\, [])) \in [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)^{\top}.$$

Proof. Take $(\mathtt{fold}\, v, \mathtt{fold}\, v') \in [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)$ with $\triangleright((v, v') \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\varphi))$. Using Lemma 2.1.1, Lemma 2.4.1 and Lemma 2.4.2 we have

$$E'[\mathtt{unfold}\, (\mathtt{fold}\, v')]\uparrow \leftrightarrow E'[v']\uparrow \to \triangleright(E'[v']) \to \triangleright(E[v]\Uparrow) \to E[\mathtt{unfold}\, (\mathtt{fold}\, v)]\Uparrow.$$

concluding the proof. □

Lemma 2.4.13. *If* $(E, E') \in [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)^{\top}$ *then*

$$(E \circ (\mathtt{fold}\, []), E' \circ (\mathtt{fold}\, [])) \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\varphi)^{\top}.$$

Proof. Easily follows from the fact that if (v, v') are related at the unfolded type then $(\mathtt{fold}\, v, \mathtt{fold}\, v')$ are related at the folded type. □

Lemma 2.4.14. *If* $(E, E') \in [\![\Delta \vdash \exists\alpha.\tau]\!](\varphi)^{\top}$ *then for all* $\Delta \vdash \tau_1$

$$(E \circ (\mathtt{pack}\, []), E' \circ (\mathtt{pack}\, [])) \in [\![\Delta \vdash \tau[\tau_1/\alpha]]\!](\varphi)^{\top}.$$

Proof. Take $(v, v') \in [\![\Delta \vdash \tau[\tau_1/\alpha]]\!](\varphi)$. Lemma 2.4.3 implies $(v, v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi\,[\alpha \mapsto [\![\Delta \vdash \tau]\!](\varphi)])$ which further means that $(\mathtt{pack}\, v, \mathtt{pack}\, v') \in [\![\Delta \vdash \exists\alpha.\tau]\!](\varphi)$ which is easily seen to imply the conclusion. □

Lemma 2.4.15. *Let* $\Delta \vdash \tau$, $(E, E') \in [\![\Delta \vdash \tau]\!](\varphi)^{\top}$. *If for all* $\sigma, \sigma' \in \mathbf{Type}$ *and* $s \in \mathbf{VRel}\,(\sigma, \sigma')$ *and for all* $(v, v') \in [\![\Delta, \alpha \vdash \tau_1]\!](\varphi\,[\alpha \mapsto s])$,

$$(e[v/x], e'[v'/x]) \in [\![\Delta \vdash \tau]\!](\varphi)^{\top\top}$$

then

$$(E \circ (\mathtt{unpack}\ []\ \mathtt{as}\ x\ \mathtt{in}\ e), E' \circ (\mathtt{unpack}\ []\ \mathtt{as}\ x\ \mathtt{in}\ e')) \in [\![\Delta \vdash \exists\alpha.\tau_1]\!](\varphi)^{\top}.$$

Proof. Take $(\mathtt{pack}\, v, \mathtt{pack}\, v') \in [\![\Delta \vdash \exists\alpha.\tau_1]\!]\,(\varphi)$. This implies there exist $\sigma, \sigma' \in \mathbf{Type}$ and an $s \in \mathbf{VRel}\,(\sigma, \sigma)$, such that $(v, v') \in [\![\Delta, \alpha \vdash \tau_1]\!]\,(\varphi\,[\alpha \mapsto s])$. An assumption of this lemma further implies $(e[v/x], e'[v'/x]) \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top\top}$. It is now easy to conclude the proof using Lemma 2.4.2. □

The other lemmata concerning context composition are proved in an analogous way. The next lemma will be used to prove compatibility of `?`.

Lemma 2.4.16. *For all $n \in \mathbb{N}$, $(\underline{n}, \underline{n}) \in [\![\vdash \mathtt{nat}]\!]$.*

Proof. By induction on n.

$n = 0$ Then $\underline{n} = \mathtt{fold}\,\mathtt{inl}\,\langle\rangle$. It is easy to see that $(\mathtt{inl}\,\langle\rangle, \mathtt{inl}\,\langle\rangle) \in [\![\vdash 1 + \mathtt{nat}]\!]$ and so the result follows by the definition of interpretation of recursive types and Lemma 2.4.8.

$n = m + 1$ Then $\underline{n} = \mathtt{fold}\,\mathtt{inr}\,\underline{m}$. By assumption $(\underline{m}, \underline{m}) \in [\![\vdash \mathtt{nat}]\!]$ and so $(\mathtt{inr}\,\underline{m}, \mathtt{inr}\,\underline{m}) \in [\![\vdash 1 + \mathtt{nat}]\!]$ and the result easily follows follows by the definition of interpretation of recursive types and Lemma 2.4.8. □

We are now ready to prove that the logical approximation relation is compatible.

Proposition 2.4.17. *The relation $\lesssim^{log}_{\Downarrow}$ is closed under all the rules in Figure 5, i.e. it is compatible.*

Proof. We only show some cases. The general rule is that compatibility rules are proved either by directly showing two values are related at the value relation or relying on the above lemmata and extending the evaluation contexts.

- Introduction of recursive types.

$$\frac{\Delta; \Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \tau[\mu\alpha.\tau/\alpha]}{\Delta; \Gamma \vdash \mathtt{fold}\, e \lesssim^{log}_{\Downarrow} \mathtt{fold}\, e' : \mu\alpha.\tau}$$

  Take $\varphi \in \mathbf{VRel}\,(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!]\,(\varphi)$. Let $f = e\gamma$ and $f' = e'\gamma'$. We have to show $(\mathtt{fold}\, f, \mathtt{fold}\, f') \in [\![\Delta \vdash \mu\alpha.\tau]\!]\,\varphi^{\top\top}$. So take $(E, E') \in [\![\Delta \vdash \mu\alpha.\tau]\!]\,\varphi^{\top}$. By assumption $(f, f') \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!]\,\varphi^{\top\top}$ so it suffices to show $(E \circ (\mathtt{fold}\,[]), E' \circ \mathtt{fold}\,[]) \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!]\,\varphi^{\top}$, but this is exactly the content of Lemma 2.4.13.

- Elimination of recursive types.

$$\frac{\Delta; \Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \mu\alpha.\tau}{\Delta; \Gamma \vdash \mathtt{unfold}\, e \lesssim^{log}_{\Downarrow} \mathtt{unfold}\, e' : \tau[\mu\alpha.\tau/\alpha]}$$

  Exactly the same reasoning as in the previous case, only this time we use Lemma 2.4.12 in place of Lemma 2.4.13.

- The `?` expression.

$$\frac{}{\Delta; \Gamma \vdash ? \lesssim^{log}_{\Downarrow} ? : \mathtt{nat}}$$

  By Lemma 2.4.16 we have $\forall n, (\underline{n}, \underline{n}) \in [\![\vdash \mathtt{nat}]\!]$.

  Take $(E, E') \in [\![\vdash \mathtt{nat}]\!]^{\top}$ and assume $E'[?]\!\uparrow$. By definition of the $\uparrow$ relation there exists an e', $? \rightsquigarrow e'$ and $E[e']\!\uparrow$. Inspecting the operational semantics we see that $e' = \underline{n}$ for some $n \in \mathbb{N}$. This implies $E[\underline{n}]\!\uparrow$ which further implies by Lemma 2.2.1 that $E[?]\!\uparrow$.

- Introduction of existential type.

$$\frac{\Delta \vdash \tau_1 \qquad \Delta; \Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \tau[\tau_1/\alpha]}{\Delta; \Gamma \vdash \mathtt{pack}\, e \lesssim^{log}_{\Downarrow} \mathtt{pack}\, e' : \exists \alpha.\tau}$$

  Take $\varphi \in \mathbf{VRel}\,(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!]\,(\varphi)$. Let $f = e\gamma$ and $f' = e'\gamma'$. We need to show

$$(\mathtt{pack}\, f, \mathtt{pack}\, f') \in [\![\Delta \vdash \exists \alpha.\tau]\!]\,(\varphi)^{\top\top}$$

  Again by assumption $(f, f') \in [\![\Delta \vdash \tau[\tau_1/\alpha]]\!]\,(\varphi)^{\top\top}$. We use Lemma 2.4.14 to finish the proof, as we did above for the case of introduction of recursive types.

- Elimination of existential types.

$$\frac{\Delta; \Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \exists \alpha.\tau_1 \qquad \Delta \vdash \tau \qquad \Delta, \alpha; \Gamma, x : \tau_1 \vdash e_1 \lesssim^{log}_{\Downarrow} e'_1 : \tau}{\Delta; \Gamma \vdash (\mathtt{unpack}\ e\ \mathtt{as}\ x\ \mathtt{in}\ e_1) \lesssim^{log}_{\Downarrow} (\mathtt{unpack}\ e'\ \mathtt{as}\ x\ \mathtt{in}\ e'_1) : \tau}$$

  Take $\varphi \in \mathbf{VRel}\,(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!]\,(\varphi)$. Let $f = e\gamma$ and $f' = e'\gamma'$, $f_1 = e_1\gamma$, $f'_1 = e'_1\gamma$. We need to show

$$(\mathtt{unpack}\ f\ \mathtt{as}\ x\ \mathtt{in}\ f_1, \mathtt{unpack}\ f'\ \mathtt{as}\ x\ \mathtt{in}\ f'_1) \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top\top}.$$

  The premise of this case shows that for all $\sigma, \sigma' \in \mathbf{Type}$ and $s \in \mathbf{VRel}\,(\sigma, \sigma')$, for all $(v, v') \in [\![\Delta, \alpha \vdash \tau_1]\!]\,(\varphi\,[\alpha \mapsto s])$, we have

$$(f_1[v/x], f'_1[v'/x]) \in [\![\Delta, \alpha \vdash \tau]\!]\,(\varphi\,[\alpha \mapsto s])^{\top\top}$$

  and by Lemma 2.4.4 this is the same as for all $\sigma, \sigma' \in \mathbf{Type}$, for all $s \in \mathbf{VRel}\,(\sigma, \sigma')$, for all $(v, v') \in [\![\Delta, \alpha \vdash \tau_1]\!]\,(\varphi\,[\alpha \mapsto s])$,

$$(f_1[v/x], f'_1[v'/x]) \in [\![\Delta \vdash \tau]\!]\,(\varphi)^{\top\top}.$$

  We now use Lemma 2.4.15 to conclude the proof.

□

Corollary 2.4.18 (Fundamental theorem of logical relations)**.** *If* $\Delta; \Gamma \vdash e : \tau$ *then* $\Delta; \Gamma \vdash e \lesssim^{log}_{\Downarrow} e : \tau$

Proof. Every compatible relation is reflexive. This can be shown by an easy induction on the typing derivation. Proposition 2.4.17 shows that the logical relation is compatible, hence it is reflexive. □

We need the next corollary to relate the logical approximation relation to must-contextual approximation.

Corollary 2.4.19. *For any expressions* e, e' *and context* C*, if* $\Delta \mid \Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \tau$ *and* $C : (\Delta \mid \Gamma \Rrightarrow \tau) \to (\Delta' \mid \Gamma' \Rrightarrow \sigma)$ *then* $\Delta' \mid \Gamma' \vdash C[e] \lesssim^{log}_{\Downarrow} C[e'] : \tau'$*.*

Proof. By induction on the judgment $C : (\Delta \mid \Gamma \Rrightarrow \tau) \to (\Delta' \mid \Gamma' \Rrightarrow \sigma)$, using Proposition 2.4.17. □

2.4.3 All three approximation relations coincide

We have already mentioned that it is easy to see that must-contextual approximation implies must-CIU approximation. We now show that must-CIU approximation implies logical approximation.

We start with a lemma showing that the logical approximation relation is closed under post-composition with the must-CIU approximation relation.

Lemma 2.4.20. *For any terms e, e' and e'' of type τ in context $\Delta \mid \Gamma$. If $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' :$ and $\Delta \mid \Gamma \vdash e' \lesssim_{\Downarrow}^{CIU} e'' : \tau$ then $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e'' :$.*

Proof. Take $\varphi \in \mathbf{VRel}(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!](\varphi)$. Take $(E, E') \in [\![\tau]\!](\varphi)^{\top}$ and assume $E'[e''\gamma']\!\uparrow$. Since $\Delta \mid \Gamma \vdash e' \lesssim_{\Downarrow}^{CIU} e'' : \tau$ we have $E'[e'\gamma']\!\uparrow$ and since $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' :$ we further have $E[e\gamma]\!\Uparrow$, concluding the proof. □

Corollary 2.4.21. *For any terms e and e' of type τ in context $\Delta \mid \Gamma$. If $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{CIU} e' : \tau$ then $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' :$.*

Proof. By Corollary 2.4.18 we have that $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e :$. The previous lemma concludes the proof. □

The only missing link in the chain of inclusions is the implication from the logical relation to contextual approximation. This, however, requires some more work.

Adequacy

2.5 Adequacy

We wish to show soundness of the logical relation with respect to must-contextual approximation. However, the implication

$$\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau \to \Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{ctx} e' : \tau$$

does not hold internally, due to the different divergence relations used in the definition of the logical relation. To see precisely where the proof fails let us attempt it. Let $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau$ and take a well-typed closing context C with result type σ. Then by Corollary 2.4.19, $\varnothing \mid \varnothing \vdash C[e] \lesssim_{\Downarrow}^{log} C[e'] : \sigma$. Unfolding the definition of the logical relation we get $(C[e], C[e']) \in [\![\varnothing \vdash \sigma]\!]^{\top\top}$. It is easy to see that $(-,-) \in [\![\varnothing \vdash \sigma]\!]^{\top}$ and so we get by definition of $\top\top$ that $C[e']\!\uparrow \to C[e]\!\Uparrow$. However the definition of contextual equivalence requires the implication $C[e']\!\uparrow \to C[e]\!\uparrow$, which is not a consequence of the previous one.

Intuitively, the gist of the problem is that $\uparrow$ defines a time-independent predicate, whereas $\Uparrow$ depends on the time, as explained in the introduction. However, in the model in we can show the following rule is admissible.

Lemma 2.5.1. *$e : \mathbf{Tm} \mid \varnothing \vdash \Box(e\!\Uparrow) \to e\!\uparrow$ holds in the logic.*

Thus we additionally assume this principle in our logic. Using this, we are led to the following corrected statement of adequacy using the $\Box$ modality.

Theorem 2.5.2 (Adequacy)**.** *If e and e' are of type τ in context $\Delta \mid \Gamma$ then $\Box(\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau)$ implies $\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{ctx} e' : \tau$.*

To prove this theorem we first observe that all the lemmata used in the proof of Corollary 2.4.19 are proved in constant contexts using only other constant facts and so Corollary 2.4.19 can be strengthened. More precisely, we can prove the following restatement.

Proposition 2.5.3. $\Box[\forall \Delta, \Delta', \Gamma, \Gamma', \tau, \sigma, C, e, e', C : (\Delta \mid \Gamma \Rrightarrow \tau) \to (\Delta' \mid \Gamma' \Rrightarrow \sigma)$
$\to \Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau \to \Delta' \mid \Gamma' \vdash C[e] \lesssim_{\Downarrow}^{log} C[e'] : \tau']$.

Note that all the explicit universal quantification in the proposition is over constant types. One additional ingredient we need to complete the proof is the fact that $\uparrow$ is $\neg\neg$-closed, i.e. $e\uparrow \leftrightarrow \neg\neg(e\uparrow)$. We can show this in the logic using the fact that $\uparrow$ is the greatest post-fixed point by showing that $\neg\neg\uparrow$ is another one. This fact further means that $\Box(e\uparrow) \leftrightarrow (e\uparrow)$. We are now ready to proceed with the proof of Theorem 2.5.2.

Theorem 2.5.2. Continuing the proof we started above we get, using Proposition 1.3.5, that $\Box(C[e']\uparrow \to C[e]\Updownarrow)$ and thus also $\Box(C[e']\uparrow) \to \Box(C[e]\Updownarrow)$. $\Box(C[e']\uparrow) \leftrightarrow C[e']\uparrow$ and by Lemma 2.5.1 $\Box(C[e]\Updownarrow) \leftrightarrow C[e]\uparrow$. We thus conclude $C[e']\uparrow \to C[e]\uparrow$, as required. □

Using Proposition 1.8.3 and Proposition 1.8.2 we can see that for each Δ, Γ, e, e' and τ,

$$\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{ctx} e' : \tau \leftrightarrow \neg\neg(\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{ctx} e' : \tau)$$

and similarly

$$\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{CIU} e' : \tau \leftrightarrow \neg\neg(\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{CIU} e' : \tau).$$

Combining this observation with the above we have

Theorem 2.5.4. *For any Δ, Γ, e, e' and τ,*

$$\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{CIU} e' : \tau \leftrightarrow \Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{ctx} e' : \tau \leftrightarrow \Box(\Delta \mid \Gamma \vdash e \lesssim_{\Downarrow}^{log} e' : \tau)$$

Proof. The only missing link is the implication from CIU approximation to "boxed" logical approximation. However using the previous observation that CIU approximation is $\neg\neg$-closed with Corollary 2.4.21 and the fact that $\neg\neg$ is the left adjoint to $\Box$, we get the desired implication. □

2.6 Examples of the use of logical relation

We show the syntactic minimal invariance example. We start with two simple lemmata.

Lemma 2.6.1. *Let $\tau, \sigma \in \mathbf{Type}$, let $e \in \mathbf{Tm}\,(\tau \to \sigma)$, $v \in \mathbf{Val}\,(\tau \to \sigma)$. Then*

$$(e, v) \in [\![\tau \to \sigma]\!]^{\top\top} \to (\lambda x.e\,x, v) \in [\![\tau \to \sigma]\!]\,.$$

Proof. By assumption $v = \lambda x.e'$ for some x and e'. Take $(u, u') \in [\![\tau]\!]$ and we're supposed to show $(e\,u, e'[u'/x]) \in [\![\sigma]\!]^{\top\top}$. By Lemma 2.4.2 it suffices to show $(e\,u, v\,u') \in [\![\sigma]\!]^{\top\top}$ and this is a simple consequence of Corollary 2.4.10. □

Lemma 2.6.2. *Let $\tau, \sigma \in \mathbf{Type}$, let $e \in \mathbf{Tm}\,(\tau \to \sigma)$, $v \in \mathbf{Val}\,(\tau \to \sigma)$. Then*

$$(v, e) \in [\![\tau \to \sigma]\!]^{\top\top} \to (v, \lambda x.e\,x) \in [\![\tau \to \sigma]\!]\,.$$

2.6.1 Syntactic minimal invariance

Let $\mathtt{fix} : \forall\alpha,\beta.((\alpha\to\beta)\to(\alpha\to\beta)) \to (\alpha\to\beta)$ be the term $\Lambda.\Lambda.\lambda f.\delta_f(\mathtt{fold}\,\delta_f)$ where δ_f is the term $\lambda y.\mathtt{let}\ y' = \mathtt{unfold}\,y\ \mathtt{in}\ f\,(\lambda x.y'\,y\,x)$.

Consider the type $\tau = \mu\alpha.\mathtt{nat} + \alpha \to \alpha$. Let $\mathrm{id} = \lambda x.x$ and consider the term

$$f \;\equiv\; \lambda h, x.\mathtt{case}\,(\mathtt{unfold}\,x, y.\mathtt{fold}\,(\mathtt{inl}\,y), g.\mathtt{fold}\,(\mathtt{inr}\,\lambda y.h(g(h\,y))))\,.$$

We show that $\mathtt{fix}[][]\,f \cong_{\Downarrow} \mathrm{id} : \tau\to\tau$. First we show that they either logically approximates the other and then use the fact that we have proved this in the context of only constant facts to conclude that the statement always holds. Thus we use Theorem 2.5.4 to conclude that the terms are contextually equivalent.

$\Rightarrow$ We first show by Löb induction that $(\mathtt{fix}[][]\,f, \mathrm{id}) \in [\![\tau\to\tau]\!]^{\top\top}$. It is easy to see that

$$\mathtt{fix}[][]\,f \overset{p,1}{\rightsquigarrow} \lambda x.\mathtt{case}\,(\mathtt{unfold}\,x, y.\mathtt{fold}\,(\mathtt{inl}\,y), g.\mathtt{fold}\,(\mathtt{inr}\,\lambda y.h(g(h\,y))))\,.$$

where $h = \lambda x.\delta_f(\mathtt{fold}\,\delta_f)\,x$. Let

$$\varphi = \lambda x.\mathtt{case}\,(\mathtt{unfold}\,x, y.\mathtt{fold}\,(\mathtt{inl}\,y), g.\mathtt{fold}\,(\mathtt{inr}\,\lambda y.h(g(h\,y))))\,.$$

We now show directly that $(\varphi, \mathrm{id}) \in [\![\tau\to\tau]\!]$ which suffices by Lemma 2.4.8 and Lemma 2.4.2.

So take $(u, u') \in [\![\tau]\!]$. By the definition of the interpretation of recursive types there are two cases

- $u = \mathtt{fold}\,(\mathtt{inl}\,\underline{n})$ and $u' = \mathtt{fold}\,(\mathtt{inl}\,\underline{n})$ for some $n \in \mathbb{N}$. This case is immediate.
- $u = \mathtt{fold}\,(\mathtt{inr}\,g)$, $u' = \mathtt{fold}\,(\mathtt{inr}\,g')$ and $\triangleright((g, g') \in [\![\tau\to\tau]\!])$. We then have that $\varphi\,u \overset{p,1}{\rightsquigarrow} \mathtt{fold}\,(\mathtt{inr}\,\lambda y.h(g(h\,y)))$ and $\mathrm{id}\,u' \overset{p}{\rightsquigarrow} u'$ and so it suffices to show

  $$\triangleright\,(\lambda y.\,(h(g(h\,y)), g') \in [\![\tau\to\tau]\!])\,.$$

  We again show that these are related as values so take $\triangleright((v, v') \in [\![\tau]\!])$ and we need to show $\triangleright\left((h(g(h\,v)), g'\,v') \in [\![\tau]\!]^{\top\top}\right)$. Take $\triangleright((E, E') \in [\![\tau]\!]^{\top})$. Löb induction hypothesis gives us that $\triangleright((h', \mathrm{id}) \in [\![\tau\to\tau]\!]^{\top\top})$, where h' is the body of h, i.e $h = \lambda x.h'\,x$. By Lemma 2.6.1 $\triangleright((h, \mathrm{id}) \in [\![\tau\to\tau]\!]^{\top\top})$ and so by using Lemma 2.4.9 three times we get $\triangleright\left((E[h\,(g\,(h\,[]))], E'[g'\,[]]) \in [\![\tau]\!]^{\top}\right)$.

  So assuming $\triangleright(E'[g'\,v']\!\uparrow)$ we get $\triangleright(E[h\,(g\,(h\,v))]\!\uparrow)$, concluding the proof.

$\Leftarrow$ This direction is essentially the same, only using Lemma 2.6.2 in place of Lemma 2.6.1.

2.6.2 Least prefixed point

We now prove the following recursion induction principle for the fixed-point combinator. The rule is the same as in [3] and the proof is morally the same, except that we replace induction on ordinals by Löb induction, thus removing a lot of unnecessary bookkeeping. More precisely, we prove

$$\frac{\Delta \mid \Gamma \vdash v \lesssim^{ctx}_{\Downarrow} v' : \tau_1\to\tau_2}{\Delta \mid \Gamma \vdash \mathtt{fix}[][]\,v \lesssim^{ctx}_{\Downarrow} v' : \tau_1\to\tau_2}$$

We do this in a few stages. For simplicity we only consider the case where Δ and Γ are empty. The general case is proved in the same way.

It is easy to see that $\mathtt{fix}[][]\, v \overset{1}{\rightsquigarrow} v\, h$ where $h = \lambda x.\delta_f\,(\mathtt{fold}\,\delta_f)\,x$. We first show $(h, v') \in [\![\tau_1 \to \tau_2]\!]$ by Löb induction. So assume $\triangleright((h, v') \in [\![\tau_1 \to \tau_2]\!])$. Since v' is a value of the function type $v' = \lambda y.e$ for some y and typeable e. Take $(u, u') \in [\![\tau_1]\!]$ and $(E, E') \in [\![\tau_2]\!]^\top$. Then using Corollary 2.4.10, then Corollary 2.4.18 applied to v and then Corollary 2.4.11 for v we have $(E[((\lambda x.v\,x)\ -)\,u], E[((\lambda x.v\,x)\ -)\,u]) \in [\![\tau_1 \to \tau_2]\!]^\top$. Using the assumption that $v\,v'$ approximates v' and Theorem 2.5.4 we get

$$E'[v'\,u']\!\uparrow \to E'[(v\,v')\,u']\!\uparrow$$

and thus

$$\triangleright\,(E'[(v\,v')\,u']\!\uparrow)$$

and now using the induction hypothesis $\triangleright((h, v') \in [\![\tau_1 \to \tau_2]\!])$ and the fact that $(E[((\lambda x.v\,x)\ -)\,u], E[((\lambda x.v\,x)\ -)\,u]) \in [\![\tau_1 \to \tau_2]\!]^\top$ we get

$$\triangleright\,(E[(v\,h)\,u]\!\Uparrow)$$

and since $h\,u \overset{1}{\rightsquigarrow} (v\,h)\,u$ we can use Lemma 2.2.1 to get

$$E[h\,u]\!\Uparrow$$

concluding the proof.

Since the proof relies only on constant facts we have, using Theorem 2.5.4, that h contextually approximates v'. Thus

$$\mathtt{fix}[][]\, v \lesssim^{ctx}_{\Downarrow} v\,h \lesssim^{ctx}_{\Downarrow} v\,v' \lesssim^{ctx}_{\Downarrow} v'$$

2.6.3 Parametricity

We now characterize the values of type $\forall\alpha.\alpha \times \alpha \to \alpha$. We start by proving some expected properties of values of the polymorphic and function types.

Lemma 2.6.3. *Let $\alpha \vdash \tau$ and $v \in \mathbf{Val}\,(\forall\alpha.\tau)$. Then for all types σ, σ' and $s \in \mathbf{VRel}\,(\sigma, \sigma')$, $(v[], v[]) \in [\![\alpha \vdash \tau]\!]\,(\varphi)^{\top\top}$ where φ maps α to (σ, σ', s).*

Let $\mathbf{\Omega} = \forall\alpha.\mathtt{fix}[](\lambda f.f)\langle\rangle$ be the term of type $\forall\alpha.\alpha$ that deterministically diverges when instantiated (applied).

Lemma 2.6.4. *Let $v \in \mathbf{Val}\,(\forall\alpha.\alpha \times \alpha \to \alpha)$. If $v[]$ may-diverges then $\emptyset \mid \emptyset \vdash v =^{ctx}_{\Downarrow} \forall\alpha.\mathbf{\Omega}[] : \forall\alpha.\alpha \times \alpha \to \alpha$.*

Proof. This is a simple consequence of Lemma 2.6.10. □

Lemma 2.6.5. *Let $v \in \mathbf{Val}\,(\forall\alpha.\alpha \times \alpha \to \alpha)$. If $v[]$ does not may-diverge and there exist a type τ and a value $u \in \mathbf{Val}\,(\tau \times \tau)$ such that $v[]\,u$ may-diverges then for all types σ and for all values $w \in \mathbf{Val}\,(\sigma \times \sigma)$, $v[]\,w$ may-diverges and $\emptyset \mid \emptyset \vdash v[]\,w =^{ctx}_{\Downarrow} \mathbf{\Omega}[] : \sigma$.*

Proof. It is obvious that $\mathbf{\Omega}[]$ approximates $v[]\,w$ for any w. For the other approximation we use Theorem 2.5.4.

By the canonical forms lemma $u = \langle u_1, u_2\rangle$ for some $u_1, u_2 \in \mathbf{Val}\,(\tau)$. Let $w \in \mathbf{Val}\,(\sigma \times \sigma)$. Again by the canonical forms lemma $w = \langle w_1, w_2\rangle$ for some $w_1, w_2 \in \mathbf{Val}\,(\sigma)$. Now let $s = \{(w_1, u_1), (w_2, u_2)\} \in \mathbf{VRel}\,(\sigma, \tau)$. It is obvious that $(w, u) \in s$. Using Lemma 2.6.3 and the compatibility lemma for the application we have $(v[]\,w, v[]\,u) \in [\![\alpha \vdash \alpha]\!]\,(s)^{\top\top}$.[2] Since the empty context is always related to itself and $v[]\,u\!\uparrow$ we have that $v[]\,w\!\uparrow$. Since Lemma 2.6.3 is easily strengthened to a boxed one we thus have $v[]\,w\!\uparrow$ using Lemma 2.5.1.

We have thus established that $v[]\,w\!\uparrow$ for arbitrary w which implies that it CIU-approximates any other term and thus using Theorem 2.5.4 we conclude the proof. □

For the rest of the cases we need some properties of the $\uparrow$ relation which we now prove.

Lemma 2.6.6. *Let $e \in \mathbf{Tm}\,(\tau)$. If $\neg(e\!\uparrow)$ then there exists a $v\mathbf{Val}\,(\tau)$, such that $e \rightsquigarrow^* v$.*

Proof. We first prove by coinduction that the set

$$\mathcal{N} = \left\{e' \in \mathbf{Val}\,(\tau) \;\middle|\; \forall v \in \mathbf{Val}\,(\tau), \neg(e' \rightsquigarrow^* v)\right\}$$

is included in $\uparrow$ using the universal property of $\uparrow$, i.e. that it is the greatest post-fixed point.

Given $e' \in \mathcal{N}$ we have to show there exists e'' such that $e' \rightsquigarrow e''$ and $e'' \in \mathcal{N}$. By the progress lemma e' is either a value or there exists an expression e'' that e' reduces to. Since $e' \in \mathcal{N}$ it cannot be a value. Thus there exists an expression e'' such that $e' \rightsquigarrow e''$. We have to show $e'' \in \mathcal{N}$. Suppose $v \in \mathbf{Val}\,(\tau)$ and $e'' \rightsquigarrow^* v$. Then $e' \rightsquigarrow^* v$. A contradiction.

Thus $\mathcal{N} \subseteq \uparrow$ and thus $\neg\!\uparrow \subseteq \neg\mathcal{N}$. But $e \in \mathcal{N} \leftrightarrow \forall v \in \mathbf{Val}\,(\tau), \neg(e' \rightsquigarrow^* v) \leftrightarrow \neg(\exists v \in \mathbf{Val}\,(\tau), e \rightsquigarrow^* v)$ and since we can show using properties of $\neg\neg$ that $\exists v \in \mathbf{Val}\,(\tau), e \rightsquigarrow^* v$ is $\neg\neg$-closed we have proved the lemma. □

In the rest of this section we define $\mathbf{2} = \mathbf{1} + \mathbf{1}$ to be the type of booleans. We then write $\mathbf{true} = \mathtt{inl}\,\langle\rangle$ and $\mathbf{false} = \mathtt{inr}\,\langle\rangle$. By the canonical forms lemma these are the only two closed values of this type.

Lemma 2.6.7. *Let $E \in \mathbf{Stk}\,(\mathbf{2})$ and $e \in \mathbf{Tm}\,(\mathbf{2})$. Suppose $E[\mathbf{false}]\!\uparrow$ but $\neg(E[\mathbf{true}]\!\uparrow)$. If $\neg(e \rightsquigarrow^* \mathbf{false})$ and $E[e]\!\uparrow$ then $e\!\uparrow$.*

Proof. We prove this by coinduction. Let

$$\mathcal{N} = \left\{e \in \mathbf{Tm}\,(\mathbf{2}) \;\middle|\; \neg(e \rightsquigarrow^* \mathbf{false}) \wedge E[e]\!\uparrow\right\}$$

and we wish to show that $\mathcal{N} \subseteq \uparrow$. Suppose $e \in \mathcal{N}$. We need to exhibit an e', such that $e \rightsquigarrow e'$ and $e' \in \mathcal{N}$. By the progress lemma we e is either a value or steps to some e'. First we observe that e cannot be a value since the only two values are **true** and **false**. If $e = \mathbf{false}$ then $e \rightsquigarrow^* \mathbf{false}$ and if $e = \mathbf{true}$ then $E[e]\!\uparrow$ does not hold by assumption on E.

So e is not a value. By assumption $E[e]\!\uparrow$ and so there exists e'', such that $E[e] \rightsquigarrow e''$ and $e''\!\uparrow$. Since e is not a value we have $e'' = E[e']$ for some e' such that $e \rightsquigarrow e'$. For the first condition, suppose $e' \rightsquigarrow^* \mathbf{false}$. Then clearly $e \rightsquigarrow^* \mathbf{false}$, a contradiction.

We have thus exhibited an e', such that $e \rightsquigarrow e'$ and $e' \in \mathcal{N}$, thus concluding the proof. □

Naturally we can exchange the roles of **true** and **false** in the last lemma. We record the next lemma for reference. The proof is by simple coinduction.

Lemma 2.6.8. *If $e'\!\uparrow$ and $e \rightsquigarrow^* e'$ then $e\!\uparrow$.*

[2] We abused the notation by writing s instead of a function that maps φ to (σ, τ, s), but the meaning is clear.

We now prove the converse of Lemma 2.5.1 for well-typed expressions.

Lemma 2.6.9. *Let* $\tau \in \mathbf{Type}$ *and* $e \in \mathbf{Tm}(\tau)$. *Then* $e{\uparrow} \to \Box(e{\updownarrow})$.

Proof. Using the fundamental theorem and the fact that $\Box((-,-) \in [\![\emptyset \vdash \tau]\!]^{\top})$ holds we have that $\Box(e{\uparrow} \to e{\updownarrow})$ which implies the lemma since $\Box$ distributes over implication in the correct direction. $\Box$

Note that we cannot directly use Löb induction to prove that $e{\uparrow} \to e{\updownarrow}$ since we use two different step relations in the definitions of $\uparrow$ and $\updownarrow$.

We record the functional extensionality property for values of the function type. The proof is the same as in [3] so we omit it.

Lemma 2.6.10. *Let* $\tau, \sigma \in \mathbf{Type}, f, g \in \mathbf{Val}(() \tau \to \sigma)$ *and assume* $\forall u \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash f\, u =^{ctx}_{\Downarrow} g\, u : \sigma$. *Then* $\emptyset \mid \emptyset \vdash f =^{ctx}_{\Downarrow} g : \tau \to \sigma$.

We now have all the ingredients to prove the last case in the characterization of the values of type $\forall\alpha.\alpha \times \alpha \to \alpha$.

Lemma 2.6.11. *Let* $v \in \mathbf{Val}(\forall\alpha.\alpha \times \alpha \to \alpha)$. *Suppose that for all* τ *and for all* $u \in \mathbf{Val}(\tau \times \tau)$, $\neg(v[]\, u{\uparrow})$. *Then one of the following three cases holds.*

1. $\forall\tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} y : \tau$
2. $\forall\tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} y : \tau$
3. $\forall\tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} x\ \texttt{or}\ y : \tau$

where $x\ \texttt{or}\ y$ *is the binary choice expression* $\texttt{case}(\texttt{unfold}\ ?, \_.x, \_.y)$.

Proof. By Lemma 2.6.3 and the compatibility for application we have that for any $\tau \in \mathbf{Type}$,

$$\forall r \in \mathbf{VRel}(\mathbf{2}, \tau), \forall (b, w) \in r \times r, (v[]\, b, v[]\, w) \in r^{\top\top} \tag{6}$$

and

$$\forall s \in \mathbf{VRel}(\tau, \mathbf{2}), \forall (w, b) \in s \times s, (v[]\, w, v[]\, b) \in s^{\top\top} \tag{7}$$

where we write $r \times r$ and $s \times s$ for the construction on value relations used to interpret product types.

We consider 4 cases. In all cases let $x, y \in \mathbf{Val}(\tau)$ and let $s = \{(x, \mathbf{true}), (y, \mathbf{false})\} \in \mathbf{VRel}(\tau, \mathbf{2})$ and $r = \{(\mathbf{true}, x), (\mathbf{false}, y)\} \in \mathbf{VRel}(\mathbf{2}, \tau)$.

- Suppose $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{true}$ and $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{false}$. We will show that in this case $\forall\tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} x\ \texttt{or}\ y : \tau$ and we do this by establishing CIU-equivalence. We prove two approximations.
  - Take a well-typed evaluation context E and assume $E[x\ \texttt{or}\ y]{\uparrow}$. We need to show that $E[v[]\langle x, y\rangle]{\uparrow}$. $E[x\ \texttt{or}\ y]{\uparrow}$ implies that at least one of $E[x]{\uparrow}$, $E[y]{\uparrow}$ holds. Without loss of generality suppose that $E[x]{\uparrow}$. Lemma 2.6.9 implies that $(E, (\lambda x.\texttt{if}\ z\ \texttt{then}\ \mathbf{\Omega}[]\ \texttt{else}\ z)\ -) \in [\![s]\!]^{\top}$. Using Lemma 2.6.8 and the assumption that $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{true}$ we have that $(\lambda x.\texttt{if}\ z\ \texttt{then}\ \mathbf{\Omega}[]\ \texttt{else}\ z)\,(v[]\langle \mathbf{true}, \mathbf{false}\rangle){\uparrow}$. Hence we have from (7) that $E[v[]\langle x, y\rangle]{\updownarrow}$ which we can improve to $E[v[]\langle x, y\rangle]{\uparrow}$ using Lemma 2.5.1 and the fact that we only used constant properties to prove it (in particular, s and r are $\neg\neg$-closed relations).

– Take a well-typed evaluation context E and assume that $E[v[]\langle x, y\rangle]\uparrow$. We need to show $E[x \texttt{ or } y]\uparrow$ and using Lemma 2.6.8 it suffices to show that either $E[x]\uparrow$ or $E[y]\uparrow$. Assume for the sake of contraiction that the negation holds. Since in intuitionistic logic $\neg(P \vee Q) \leftrightarrow \neg P \wedge \neg Q$ holds we have that neither of $E[x]$ and $E[y]$ may-diverges. This together with the assumption that $E[v[]\langle x, y\rangle]\uparrow$ means that $(-, E) \in r^{\top}$. However this implies, using (6) and Lemma 2.5.1 that $v[]\langle \mathbf{true}, \mathbf{false}\rangle\uparrow$, contradicting the assumption of the lemma. Thus we have proved $\neg\neg E[x \texttt{ or } y]\uparrow$ and since may-divergence is $\neg\neg$-closed also $E[x \texttt{ or } y]\uparrow$.

- Suppose $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{true}$ and not $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{false}$. We will show that in this case $\forall \tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} x : \tau$ and we again do this by establishing CIU-equivalence using two approximations.

  – Take a well-typed evaluation context E and assume that $E[x]\uparrow$. We are to show $E[v[]\langle x, y\rangle]\uparrow$. $E[x]\uparrow$ implies, using Lemma 2.6.9, that $(E, (\lambda z.\texttt{if } z \texttt{ then } \mathbf{\Omega}[] \texttt{ else } z)\ -) \in s^{\top}$. Using (7) and Lemma 2.6.8 we thus have that $E[v[]\langle x, y\rangle]\uparrow$. Note that in this direction we have not used the assumption that $v[]\langle \mathbf{true}, \mathbf{false}\rangle$ does not evaluate to **false**. We shall need it in the other direction, however.

  – Take a well-typed evaluation context E and assume that $E[v[]\langle x, y\rangle]\uparrow$. We are to show $E[x]\uparrow$. Assume the converse for the sake of contradiction. This then means that $((\lambda z.\texttt{if } z \texttt{ then } z \texttt{ else } \mathbf{\Omega}[])\ -, E) \in r^{\top}$. Using this and (6) we have that $(\lambda z.\texttt{if } z \texttt{ then } z \texttt{ else } \mathbf{\Omega}[])\, v[]\langle \mathbf{true}, \mathbf{false}\rangle\uparrow$. We now use Lemma 2.6.7 to conclude that $v[]\langle \mathbf{true}, \mathbf{false}\rangle\uparrow$ (note that here is the place where we used the assumption that $v[]\langle \mathbf{true}, \mathbf{false}\rangle$ does not reduce to **false**). However this contradicts the assumption that $v[]\langle \mathbf{true}, \mathbf{false}\rangle$ does not may-diverge. We have thus established $\neg\neg(E[x]\uparrow)$ and so $E[x]\uparrow$.

- Suppose $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{false}$ and not $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{true}$. In this case $\forall \tau \in \mathbf{Type}, \forall x, y \in \mathbf{Val}(\tau), \emptyset \mid \emptyset \vdash v[]\langle x, y\rangle =^{ctx}_{\Downarrow} y : \tau$. The proof is completely analogous to the previous case so we omit the details.

- Suppose $v[]\langle \mathbf{true}, \mathbf{false}\rangle$ evaluates to neither **true** nor **false**. We claim that this case is impossible. Indeed, by assumption $v[]\langle \mathbf{true}, \mathbf{false}\rangle$ does not may-diverge. By Lemma 2.6.6 there is a value $z \in \mathbf{Val}(\mathbf{2})$ such that $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* z$. However by the canonical forms lemma we z must be either **true** or **false**. A contradiction.

We claim that the four cases we have considered cover everything. As a consequence of Lemma 1.9.1 we have for any two expressions e and e', that either $e \rightsquigarrow^* e'$ or $\neg(e \rightsquigarrow^* e')$. In particular we have $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{false}$ or not and $v[]\langle \mathbf{true}, \mathbf{false}\rangle \rightsquigarrow^* \mathbf{true}$ or not, which give exactly the four cases we have considered. □

3 View from the outside

We now sketch the interpretation of the types and relations defined in the internal language of $\mathbf{Sh}(\omega_1)$ in the category **Set**.

3.1 Interpretation of the model

Types The set of terms, types and evaluation contexts can be constructed as initial algebras of polynomial functors, hence are preserved by Δ, so $\mathbf{Val} = \Delta(\underline{\mathbf{Val}})$, $\mathbf{Tm} = \Delta(\underline{\mathbf{Tm}})$ and

Δ ($\underline{\mathbf{Stk}}$) (here $\underline{\mathbf{Tm}}$, $\underline{\mathbf{Stk}}$ and $\underline{\mathbf{Val}}$ are sets of expressions, evaluation contexts and values defined in **Set**). Moreover, since the initial algebra structure is preserved, in particular catamorphisms are preserved. Thus, functions from constant sets to constant sets "defined by induction" on, say **Stk** are the ones coming from **Set**.

Predicates The basic evaluation relation $\longmapsto$ can be defined by a simple case analysis on the set of closed expressions. More precisely it can be defined in the geometric fragment of first-order logic as a predicate on constant sets, since the type of closed terms is constant and Δ preserves products. Thus if $\underline{\longmapsto}$ is the basic one-step relation defined in **Set**, then $\longmapsto = \Delta\left(\underline{\longmapsto}\right)$. The one-step reduction relation $\rightsquigarrow$ can be defined using $\longmapsto$ using a function from evaluation contexts and closed expressions to closed expressions, which "plugs the hole". This function can be defined by induction on the structure of the evaluation context, technically as a function from **Stk** to $\mathbf{Tm}^{\mathbf{Tm}}$ and since Δ also preserves exponentials, $\mathbf{Tm}^{\mathbf{Tm}}$ is constant. Thus this function arises from the analogous function in **Set**.

Thus $\rightsquigarrow$ is a constant predicate. The transitive closure $\rightsquigarrow$, the relation $\rightsquigarrow^*$ can also be seen to be constant. Similarly, the relations $\overset{1}{\rightsquigarrow}$, $\overset{p}{\rightsquigarrow}$ ..., can be defined *positively* by starting with a smaller $\longmapsto$ relation and by relational composition. It is easy to see that composing two constant relations gives a constant relation. Thus, all the step relations are constant and equivalent to the inclusion by Δ of analogous relations defined in sets.

Interpretation of $\uparrow$ $\uparrow$ is defined internally as the greatest fixed point of Φ given as $\Phi(m) = \left\{e : \mathbf{Tm} \;\middle|\; \exists e', e \rightsquigarrow e' \wedge m(e')\right\}$. Thus it satisfies $\forall e : \mathbf{Tm}, e\uparrow \leftrightarrow \exists e', e \rightsquigarrow e' \wedge e'\uparrow$ and is the largest predicate that satisfies this formula. We will now show that $\uparrow = \Delta\underline{\uparrow}$, where $\underline{\uparrow}$ is the may-divergence relation defined in **Set**. We use Kripke-Joyal forcing semantics.

Suppose ν is a successor ordinal. Let $e \in \mathbf{Tm}(\nu)$. Thus $e \in \underline{\mathbf{Tm}}$ and by Kripke-Joyal we have

$$\nu \Vdash e\uparrow \text{ iff } \exists e' \in \underline{\mathbf{Tm}}, \nu \Vdash e \rightsquigarrow e' \text{ and } \nu \Vdash e'\uparrow$$

As we described above, $\nu \Vdash e \rightsquigarrow e'$ if and only if $e\underline{\rightsquigarrow}e'$ this implies that at each successor ordinal[3] ν, $\uparrow(\nu)$ is a fixed point of

$$\Phi'(S) = \left\{e : \underline{\mathbf{Tm}} \;\middle|\; \exists e', e \rightsquigarrow e' \wedge e' \in S\right\}$$

defined in **Set**. Since $\underline{\uparrow}$ is defined as the greatest fixed point of Φ' we have that for all successor ordinals ν, $\uparrow(\nu) \subseteq \underline{\uparrow}$, thus $\uparrow \leq \Delta\left(\underline{\uparrow}\right)$. It is easy to see (same sequence of steps we did just now) that $\Delta\left(\underline{\uparrow}\right)$ is a fixed point of Φ. Hence, $\uparrow \geq \Delta\left(\underline{\uparrow}\right)$ and so $\uparrow = \Delta\left(\underline{\uparrow}\right)$.

It is easy to see that $\underline{\uparrow}$ is exactly the complement of the must-termination predicate $\Downarrow$ defined in [3].

Interpretation of the stratified may-divergence predicate The predicate $\uparrow_!$ is defined internally as the unique fixed point of Ψ given as $\Psi(m) = \left\{e : \mathbf{Tm} \;\middle|\; \exists e', e \overset{1}{\rightsquigarrow} e' \wedge m(e')\right\}$.

For a successor ordinal ν we thus have that

$$\nu \Vdash e\uparrow_! \text{ iff } \exists e' \in \underline{\mathbf{Tm}}, \nu \Vdash e \overset{1}{\rightsquigarrow} e' \text{ and } \text{ for all } \beta < \nu, \beta \Vdash e'\uparrow_!$$

[3]To see the need for assuming that ν is a successor ordinal see the Kripke-Joyal semantics of existentials for limit ordinals.

Since for $\nu \geq 1$ $\nu \Vdash e \overset{1}{\rightsquigarrow} e'$ means that externally $e \underline{\overset{1}{\rightsquigarrow}} e'$ this is exactly the same as the definition of $\underline{\uparrow}$ given in Section 2.2.

Thus,

$$\underline{\uparrow}(\nu) = \left\{ e \in \underline{\mathbf{Tm}} \;\middle|\; \exists e' \in \underline{\mathbf{Tm}}, e \underline{\overset{1}{\rightsquigarrow}} e' \wedge \forall \beta < \nu, e' \in \underline{\uparrow}(\beta) \right\}$$

which is exactly the pointwise negation of the stratified must-termination predicate $\{\Downarrow_\beta\}$ defined in [3], i.e. $\Downarrow_\beta^c = \underline{\uparrow}(\beta)$.

Proposition 3.1.1.

$$\bigcap_{\nu=1}^{\omega_1} \underline{\uparrow}(\beta) \subseteq \underline{\uparrow}$$

Proof. Since $\Downarrow_\beta^c = \underline{\uparrow}(\beta)$ and $\underline{\uparrow} = \Downarrow^c$ we have

$$\bigcap_{\nu=1}^{\omega_1} \underline{\uparrow}(\beta) \subseteq \underline{\uparrow} \;\leftrightarrow\; \bigcap_{\nu=1}^{\omega_1} \Downarrow_\beta^c \subseteq \Downarrow^c \;\leftrightarrow\; \left(\bigcup_{\nu=1}^{\omega_1} \Downarrow_\beta \right)^c \subseteq \Downarrow^c \;\leftrightarrow\; \Downarrow \subseteq \left(\bigcup_{\nu=1}^{\omega_1} \Downarrow_\beta \right)$$

and the last inclusion holds by [3, Lemma 5.2] (to be completely precise we cannot immediately apply the same lemma, since we have a slightly different language, but the proof is exactly the same). □

Note that this last proposition would not hold, were we to interpret the construction in the topos of trees $\mathcal{S}$, since we would only take the intersection of the first ω approximations. Thus, we need to work in the topos $\mathbf{Sh}(\omega_1)$.

As a consequence, we can add the following principle to our logic

$$e : \mathbf{Tm} \mid \emptyset \vdash \Box(e\underline{\uparrow}) \to e\uparrow$$

which enables us to prove adequacy of the logical relation with respect to contextual must-approximation.

$\uparrow$ is constant We can actually show in the logic that $\uparrow$ is $\neg\neg$-closed. From this it follows that $\Box\uparrow = \uparrow$. Indeed, $\uparrow$ is defined as the greatest fixed point of a monotone operation on a complete lattice and we now show that $\neg\neg\uparrow$ is another fixed point. Hence $\neg\neg\uparrow \leq \uparrow$ but since $\neg\neg$ is a closure operation, we get $\neg\neg\uparrow = \uparrow$.

To show that $\neg\neg\uparrow$ is a fixed point of Φ we have to show $\neg\neg(e\uparrow) \leftrightarrow \exists e', e \rightsquigarrow e' \wedge \neg\neg e'\uparrow$. Since $\uparrow$ is a fixed point of φ we have

$$\neg\neg(e\uparrow) \leftrightarrow \neg\neg\left(\exists e', e \rightsquigarrow e' \wedge e'\uparrow\right)$$

and since we quantify over a constant, therefore total, type we can use Lemma 1.3.7 to get

$$\leftrightarrow \exists e', \neg\neg e \rightsquigarrow e' \wedge \neg\neg(e'\uparrow)$$

(we also used the fact that $\neg\neg$ preserves conjunction) and since $\rightsquigarrow$ is constant, therefore $\neg\neg$-closed (this can be also shown in the logic if we write it out precisely) we get

$$\leftrightarrow \exists e', e \rightsquigarrow e' \wedge \neg\neg(e'\uparrow)$$

showing that $\neg\neg\uparrow$ is a fixed point of Φ and concluding the proof.

Note that another way to prove that $\uparrow = \Delta\left(\underline{\uparrow}\right)$ is to use the fact that $\neg\neg\uparrow = \uparrow$. One uses the fact (which we have not stated or proved, but is easy to see) that $\neg\neg$-closed predicates on a constant set are constant, i.e. they are $\Delta(\varphi')$ for some predicate φ' on $\mathbf{Tm}$. We then use the fact that Π_1 is a logical morphism to get that $\Pi_1(\uparrow)$ is the greatest fixed point characterized by the same formula in $\mathbf{Set}$, thus, $\Pi_1(\uparrow) = \underline{\uparrow}$ and hence (since $\uparrow$ is constant), $\uparrow = \Delta\left(\underline{\uparrow}\right)$.

References

[1] Amal Ahmed. Step-indexed syntactic logical relations for recursive and quantified types. Technical report, Harvard University, 2006.

[2] S. Awodey. *Category Theory*. Oxford Logic Guides. Ebsco Publishing, 2006.

[3] Lars Birkedal, Ales Bizjak, and Jan Schwinghammer. Step-indexed relational reasoning for countable nondeterminism. *Logical Methods in Computer Science*, 9(4), 2013.

[4] Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Logical Methods in Computer Science*, 8(4), 2012.

[5] Peter T. Johnstone. *Sketches of an elephant: a topos theory compendium. Vol. 1*, volume 43 of *Oxford Logic Guides*. The Clarendon Press Oxford University Press, New York, 2002.

[6] S. MacLane and I. Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Mathematical Sciences Research Institute Publications. Springer New York, 1992.

[7] Ian A. Mason and Carolyn L. Talcott. Equivalence in functional languages with effects. *Journal of Functional Programming*, 1(3):287–327, 1991.

[8] GonzaloE. Reyes and Houman Zolfaghari. Bi-heyting algebras, toposes and modalities. *Journal of Philosophical Logic*, 25(1):25–43, 1996.