Toposes for Time Complexity Classes

Jonas Frey and Jakob Grue Simonsen
Department of Computer Science, University of Copenhagen (DIKU)*

Abstract

We outline ongoing work on constructions associating toposes to time complexity classes and present a recent, promising variation that seems to be amenable to 'geometric' reasoning.

1 Introduction

Krivine's classical realizability [7] can be understood as Kleene's (intuitionistic) realizability [6] combined with a negative translation relative to a response type [11] taking the role of falsity. The parameter acting as response type is known as the *pole* in Krivine realizability, and its variability is the reason for the versatility of Krivine's realizability interpretation.

In prior work [3] we have shown how to encode *specifications on program behavior* into the pole, using an extension of the syntax of Krivine realizability by input and output instructions. Recently [4] we have adapted this method to associate toposes to time complexity classes, building on the fact that the Krivine machine with input and output instructions (IOKAM) is a *reasonable* model of computation in the sense of [12], i.e. that it simulates – and is simulated by – Turing machines with polynomial time overhead.

The present abstract describes a promising variation of this approach which we expect to give rise to toposes which are more amenable to analysis by geometric reasoning. To set the stage, we start by recalling the relevant concepts and ideas of Krivine realizability and [3, 4] in Sections 2 and 3 below; the syntax is slightly different to that of [3, 4] as we replace explicit I/O with a special designated variable which allows for easier arguments.

2 Framework

The syntactic classes of *terms, stacks*, and *processes* are given as follows.

Terms:	$t ::= x \mid \lambda x.t \mid t\,t \mid \mathsf{cc} \mid \mathsf{k}_\pi \mid \mathsf{end} \mid \alpha$	
Stacks:	$\pi ::= \varepsilon \mid t{\cdot}\pi$	t closed
Processes:	$p ::= t \star \pi$	t closed

Apart from the standard constructs we use a term end to denote termination, and a special variable α as a placeholder for 'input' (replacing the read instructions of the IOKAM [3, 4]). Formally α is treated differently than the other variables in that it can never be bound by a λ, and accordingly we call a term *closed* if it does not contain any free variables except α.

The sets of closed terms without and possibly with α are denoted by Λ and Λ_α, respectively (thus $\Lambda \subseteq \Lambda_\alpha$), and the sets of stacks without and possibly with α are denoted by Π and Π_α (such that $\Pi \subseteq \Pi_\alpha$). A *proof-like* term is a term not containing end[1] and the set of proof-like terms without and possibly with α are denoted by $\mathsf{PL} \subseteq \Lambda$ and $\mathsf{PL}_\alpha \subseteq \Lambda_\alpha$, respectively. We write $\Lambda_{(\alpha)}, \Pi_{(\alpha)}, \mathsf{PL}_{(\alpha)}$, to refer to both variants at the same time, when explaining arguments which work both with and without α.

*The authors are partially supported by the Danish Council for Independent Research *Sapere Aude* grant "Complexity via Logic and Algebra" (COLA).

[1] Contrary to Krivine [7] we allow continuation terms k_π as subterms of proof-like terms. This difference is no big deal, and is discussed in [3, Sec. 5.2].

On processes $t \star \pi \in \Lambda_{(\alpha)} \times \Pi_{(\alpha)}$ we define the usual reduction relation.

$$\begin{array}{lrcl} \text{(push)} & tu \star \pi & \succ & t \star u{\cdot}\pi \\ \text{(pop)} & (\lambda x\,.\,t[x]) \star u{\cdot}\pi & \succ & t[u] \star \pi \\ \text{(save)} & \mathsf{cc} \star t{\cdot}\pi & \succ & t \star \mathsf{k}_\pi{\cdot}\pi \\ \text{(restore)} & \mathsf{k}_\pi \star t{\cdot}\rho & \succ & t \star \pi \end{array}$$

We use the notation $(t \star \pi)\!\downarrow$ to say that a process $t \star \pi$ terminates, i.e.

$$(t \star \pi)\!\downarrow \quad :\Leftrightarrow \quad \exists \rho\,.\,t \star \pi \succ^* \mathsf{end} \star \rho.$$

If we want to bound the number of steps in a terminating reduction sequence we write $(t \star \pi)\!\downarrow^{\leq n}$ – i.e. for $n \in \mathbb{N}$ we define

$$(t \star \pi)\!\downarrow^{\leq n} \quad :\Leftrightarrow \quad \exists \rho\,.\,t \star \pi \succ^{\leq n} \mathsf{end} \star \rho.$$

A *pole* is a set $\perp\!\!\!\perp \subseteq \Lambda_{(\alpha)} \times \Pi_{(\alpha)}$ of processes that is closed under *inverse reduction*, i.e. it satisfies

$$t \star \pi \succ u \star \rho, \quad u \star \rho \in \perp\!\!\!\perp \quad \Rightarrow \quad t \star \pi \in \perp\!\!\!\perp.$$

For our purposes we assume as additional closure condition on poles that they are also closed under forward application of the (push) rule, i.e.

$$tu \star \pi \in \perp\!\!\!\perp \quad \Rightarrow \quad t \star u{\cdot}\pi \in \perp\!\!\!\perp. \tag{2.1}$$

Instead of $t \star \pi \in \perp\!\!\!\perp$ we shall also write $t \perp_{\perp\!\!\!\perp} \pi$, or more generally $L \perp_{\perp\!\!\!\perp} S$ if $\forall t \in L\ \forall \pi \in S\,.\,t \star \pi \in \perp\!\!\!\perp$, where $L \subseteq \Lambda_{(\alpha)}$ and $S \subseteq \Pi_{(\alpha)}$.

Relative to a fixed pole $\perp\!\!\!\perp$ we can define a realizability model, i.e. a model of higher order logic. Category theoretically, this model is represented by a *tripos*, i.e. an indexed preorder $\mathcal{K}_{\perp\!\!\!\perp} : \mathbf{Set}^{\mathsf{op}} \to \mathbf{Ord}$ (with certain properties making it a model of higher order logic), and an ensuing topos $\mathbf{Set}[\perp\!\!\!\perp]$, obtained via the *tripos-to-topos construction* [5]. In the following we recall the definition of the tripos from [3], making explicit the duality between (sets of) terms and stacks which was left implicit in [3], but which we need in the present deliberations.

3 Truth values, falsity values, triposes, toposes

Truth values are represented either by sets $K, L, M \subseteq \Lambda_{(\alpha)}$ of λ-terms, or by sets $S, T, U \subseteq \Pi_{(\alpha)}$ of stacks. Following Miquel we call the former truth values, and the latter *falsity values*. Given a truth value K, we define the corresponding falsity value by

$$K_{\perp\!\!\!\perp} = \{\pi \in \Pi_{(\alpha)} \mid \forall t \in K\,.\,t \star \pi \in \perp\!\!\!\perp\},$$

and dually the truth value corresponding to a falsity value S is given by

$$S^{\perp\!\!\!\perp} = \{t \in \Lambda_{(\alpha)} \mid \forall \pi \in S\,.\,t \star \pi \in \perp\!\!\!\perp\}.$$

The operations $(-)_{\perp\!\!\!\perp}$ and $(-)^{\perp\!\!\!\perp}$ form a *Galois connection* between the sets $P(\Lambda_{(\alpha)})$ and $P(\Pi_{(\alpha)})$, which restricts to an antitone order-isomorphism between the image of $(-)_{\perp\!\!\!\perp}$ in $P(\Pi_{(\alpha)})$ and the image of $(-)^{\perp\!\!\!\perp}$ in $P(\Lambda_{(\alpha)})$. We refer to the elements of these subsets as *stable* truth and falsity values. It is possible to define the interpretation of logic using only stable truth and falsity values (as Streicher [10] does), but it complicates the constructions (intuitively because it introduces more intermediate double negations in the negative translation than necessary), and we will refrain from doing so and instead closely follow Krivine's account.

Predicates on a set J are families of falsity values, i.e. functions $\varphi, \psi : J \to P(\Pi_{(\alpha)})$. We define an ordering on predicates by setting

$$\varphi \leq \psi \quad :\Leftrightarrow \quad \exists t \in \mathsf{PL}_{(\alpha)}\ \forall j \in J\,.\,t \star \varphi(j)^{\perp\!\!\!\perp}{\cdot}\psi(j) \subseteq \perp\!\!\!\perp,$$

where $t \star \varphi(j)^{\perp\!\!\!\perp}{\cdot}\psi(j)$ is a shorthand for $\{t \star u{\cdot}\pi \mid u \in \varphi(j)^{\perp\!\!\!\perp}, \pi \in \psi(j)\}$. The assignment of preorders $(P(\Pi_{(\alpha)})^J, \leq)$ to sets J is contravariantly functorial in J and thus gives rise to an indexed preorder

$$\mathcal{K}_{\perp\!\!\!\perp} : \mathbf{Set}^{\mathsf{op}} \to \mathbf{Ord}.$$

In [3] it is shown that $\mathcal{K}_{\perp\!\!\!\perp}$ is a tripos, and thus gives rise to a topos $\mathbf{Set}[\perp\!\!\!\perp]$ for any pole $\perp\!\!\!\perp$. The tripos and the topos are degenerate – i.e. all predicates in the tripos are equivalent, and the topos is equivalent to the terminal category – if and only if the pole contains a process $t \star \pi$ which does not contain the end instruction (i.e., t is proof-like, and π is built up from proof-like terms). Thus, to avoid degeneracy (corresponding to logical inconsistency) one only has to ensure that every process in $\perp\!\!\!\perp$ contains an end instruction.

4 Poles for classes of algorithms

One fundamental example of a pole is the set

$$\mathfrak{T} = \{t \star \pi \mid (t \star \pi)\downarrow\}$$

of all terminating processes. It is clearly closed under inverse reduction, and non-degenerate since every terminating process has to contain the end instruction somewhere. This pole does not seem to be explicitly mentioned anywhere in the literature[2], but it can be viewed as a basic case of the construction of [3], and the first named author learned about some of its interesting properties from Krivine and recorded them in [2].

We call the corresponding topos $\mathbf{Set}[\mathfrak{T}]$ the *termination topos*.

In the following we introduce two variations on this theme which are about termination relative to a class of inputs, using the special variable α.

To this end, given a binary string $\sigma \in \{0,1\}^*$, let $\overline{\sigma}$ represent an appropriate encoding of σ as a λ-term, e.g. the System F encoding. The poles $\mathfrak{R}, \mathfrak{P} \subseteq \Lambda_\alpha \times \Pi_\alpha$ are defined as

$$\mathfrak{R} = \{t \star \pi \in \Lambda_\alpha \times \Pi_\alpha \mid \forall \sigma \in \{0,1\}^* \,.\, (t \star \pi)[\overline{\sigma}/\alpha]\downarrow\}$$
$$\mathfrak{P} = \{t \star \pi \in \Lambda_\alpha \times \Pi_\alpha \mid \exists P \in \mathbb{N}[X]\ \forall \sigma \in \{0,1\}^* \,.\, (t \star \pi)[\overline{\sigma}/\alpha]\downarrow^{\leq P(|\sigma|)}\},$$

where $\mathbb{N}[X]$ is the set of polynomials in the variable X with coefficients in $\mathbb{N}$. Thus, $\mathfrak{R}$ is the set of processes that terminate for all inputs $\sigma \in \{0,1\}^*$, and $\mathfrak{P}$ is the set of processes that do so in a number of steps that is polynomially bounded in the length of σ. Again, $\mathfrak{R}$ and $\mathfrak{P}$ are easily recognized as non-degenerate poles.

Our intuition is that the corresponding toposes $\mathbf{Set}[\mathfrak{R}]$ and $\mathbf{Set}[\mathfrak{P}]$ can be viewed geometrically as generalized spaces 'over' the termination topos $\mathbf{Set}[\mathfrak{T}]$. In this geometric image the termination topos plays the role of the 'point', and $\mathbf{Set}[\mathfrak{R}]$ and $\mathbf{Set}[\mathfrak{P}]$ are to be viewed as generalized topologies on the set $\{0,1\}^*$ of binary stings, whose open sets are respectively the decidable subsets, and the subsets which are decidable in polynomial time.

Since open sets correspond to truth values in sheaf toposes, we will start by associating truth values to sets of strings. In the following we focus on $\mathfrak{P}$, similar arguments can be made for $\mathfrak{R}$ as well. For $U \subseteq \{0,1\}^*$, define the falsity value P_U by

$$P_U = \{t{\cdot}\varepsilon \in \Pi_\alpha \mid \exists P \in \mathbb{N}[X]\ \forall \sigma \in U \,.\, (t \star \varepsilon)[\overline{\sigma}/\alpha]\downarrow^{\leq P(|\sigma|)}\},$$

i.e. P_U is the set of all stacks containing a single term which terminates in polynomial time without arguments and for inputs in U. The following lemma gives a first idea of how concepts from computational complexity are reflected in the internal logic of $\mathbf{Set}[\mathfrak{P}]$.

Lemma 4.1 *For $U \subseteq \{0,1\}^*$, we have*

1. $P_{U^\complement} \leq \neg P_U$
2. $\neg P_U \leq P_{U^\complement}$ *iff U is decidable in polynomial time.*

Proof. First observe that a realizer of P_U (i.e. an element of $(P_U)^{\mathfrak{P}}$) is a term t such that $t[\overline{\sigma}/\alpha]$ either terminates or calls its first argument, both after a polynomially bounded number of steps. For $\sigma \notin U$, $t[\overline{\sigma}]$ necessarily has to terminate, since the first argument can not be expected to do so.

For the first claim we have to find a proof-like term t such that $t \star u{\cdot}v{\cdot}\pi \in \mathfrak{P}$ for all $u \in (P_{U^\complement})^{\mathfrak{P}}$, $v \in (P_U)^{\mathfrak{P}}$, and $\pi \in \Pi_\alpha$. The identity term $\lambda x \,.\, x$ fulfills this requirement, since $(\lambda x \,.\, x) \star u{\cdot}v{\cdot}\pi \succ u \star v{\cdot}\pi$, and u as a realizer of $P_{U^\complement}$ terminates on U, and if it doesn't terminate it calls its argument v which terminates on $U^\complement$ as a realizer of P_U.

[2] The referees point to analogies with work of Brunel [1] and Riba [8].

For the first half of the second claim assume that $\neg P_U \leq P_{U^{\complement}}$. Then there exists a $t \in \mathsf{PL}_\alpha$ such that $t \perp_{\mathfrak{P}} (\neg P_U)^{\mathfrak{P}} \cdot P_{U^{\complement}}$. Let $v, w \in \Lambda_\alpha$ be arbitrary terms which terminate in polynomial time on input in U, and in $U^{\complement}$, such that $u \cdot \varepsilon \in P_U$, and $v \cdot \varepsilon \in P_{U^{\complement}}$. Then $\mathsf{k}_{u\cdot\varepsilon} \in (\neg P_U)^{\mathfrak{P}}$ by [3, Lemma 5.2], and thus $t \star \mathsf{k}_{u\cdot\varepsilon} \cdot v \cdot \varepsilon \in \mathfrak{P}$, i.e. it terminates in polynomial time for arbitrary input. Since we have no guarantee on the behavior of u outside of U, and of v outside of $U^{\complement}$, t has to decide whether to put u of v in head position by inspecting the argument substituted in α. Since it has to do so in polynomially many steps in $|\sigma|$ – and thus in polynomial time, since the KAM is a reasonable machine – we see that U has to be decidable in polynomial time.

For the second half of the second claim assume that U is decidable in polynomial time. Then – again since the KAM is reasonable – there exists a term $t \in \Lambda_\alpha$ such that for all $\sigma \in \{0,1\}^*$ we have

$$t[\overline{\sigma}/\alpha] \star u \cdot v \cdot \pi \succ^* \begin{cases} u \star \pi & \text{if } \sigma \in U \\ v \star \pi & \text{else,} \end{cases}$$

in polynomially many steps in $|\sigma|$.

We claim that then $w = \lambda xy\,.\,t(x(\lambda x.x))y$ is a realizer of $(\neg P_U)^{\mathfrak{P}} \cdot P_{U^{\complement}}$. Indeed, let $u \in (\neg P_U)^{\mathfrak{P}}$ and $v \cdot \varepsilon \in P_{U^{\complement}}$.

Then we have

$$(w \star u \cdot v \cdot \varepsilon)[\overline{\sigma}/\alpha] \succ^* \begin{cases} u[\overline{\sigma}/\alpha](\lambda x.x) \star \pi & \text{if } \sigma \in U \\ v[\overline{\sigma}/\alpha] \star \pi & \text{else.} \end{cases}$$

In the second case we have termination by the definition of $P_{U^{\complement}}$. In the first case we have termination because $u[\overline{\sigma}/\alpha]$ expects an argument that behaves like a realizer of P_U, and $\lambda x.x$ does so on U. ∎

5 Set[$\mathfrak{R}$] and Set[$\mathfrak{P}$] as spaces over Set[$\mathfrak{T}$]

To view **Set**[$\mathfrak{R}$] and **Set**[$\mathfrak{P}$] as spaces over **Set**[$\mathfrak{T}$], we want to construct (regular) functors of types

$$\mathbf{Set}[\mathfrak{T}] \to \mathbf{Set}[\mathfrak{R}] \qquad \text{and} \qquad \mathbf{Set}[\mathfrak{T}] \to \mathbf{Set}[\mathfrak{P}],$$

which should be viewed as inverse image parts of geometric morphisms.

The construction of these functors relies on a conjecture on the orthogonality relations $\perp_{\mathfrak{T}}$ and $\perp_{\mathfrak{P}}$ that we shall explain now. We spell out the ideas using $\perp_{\mathfrak{P}}$, but the arguments apply to $\perp_{\mathfrak{T}}$ in exactly the same way.

Let $K, L \subseteq \Lambda_\alpha$ and $S \subseteq \Pi_\alpha$. A simple orthogonality argument shows that we always have

$$K \perp_{\mathfrak{P}} S \quad \Leftrightarrow \quad K \perp_{\mathfrak{P}} (S^{\mathfrak{P}})_{\mathfrak{P}},$$

and since $\mathfrak{P}$ satisfies condition (2.1) we can also show

$$K \perp_{\mathfrak{P}} L \cdot S \quad \Leftrightarrow \quad K \perp_{\mathfrak{P}} L \cdot (S^{\mathfrak{P}})_{\mathfrak{P}},$$

thus falsity values can be replaced with their corresponding double-orthogonal closures in many arguments. Our conjecture is that the same also applies to *truth values*. More specifically:

Conjecture 5.1 *For all $K, L \subseteq \Lambda_\alpha$ and $S \subseteq \Pi_\alpha$ we have $K \perp_{\mathfrak{P}} L \cdot S \;\Leftrightarrow\; K \perp_{\mathfrak{P}} (L_{\mathfrak{P}})^{\mathfrak{P}} \cdot S$.*

This is a conjecture about applicative contexts $(-) \star \pi$ versus contexts of the form $s \star (-) \cdot \pi$, and can be rephrased as saying that the latter are not more discerning than the former – for every context $s \star (-) \cdot \pi$ there exists a set $S \subseteq \Pi_\alpha$ of stacks such that for all terms t we have

$$s \star t \cdot \pi \in \mathfrak{P} \quad \Leftrightarrow \quad t \perp_{\mathfrak{P}} S.$$

In this way it is related to *Milner's Context Lemma* [9, Section 8], but is different in that we are working in an untyped framework, and moreover it is not clear whether we can deduce the conjecture from arguments about observational *orderings* (which can be defined for every pole, but the process loses information).

Using the conjecture, we can define an indexed monotone map from $\mathcal{K}_{\mathfrak{T}}$ to $\mathcal{K}_{\mathfrak{P}}$. The idea is that we embed truth values w.r.t. $\mathfrak{T}$ (without α) into truth values w.r.t. $\mathfrak{P}$ (with α) using the inclusion $\Lambda \subseteq \Lambda_\alpha$. Since we view *falsity values* – not *truth values* – as basic, we have to pre- and post-compose with the orthogonality maps $(-)^{\mathfrak{T}}$ and $(-)_{\mathfrak{P}}$, which (leaving the inclusion $\Lambda \subseteq \Lambda_\alpha$ implicit) gives the mapping

$$\Delta : P(\Pi) \to P(\Pi_\alpha), \qquad S \mapsto (S^{\mathfrak{T}})_{\mathfrak{P}}.$$

To show that postcomposition with Δ gives rise to an indexed monotone map of type $\mathcal{K}_{\mathfrak{T}} \to \mathcal{K}_{\mathfrak{P}}$ it is sufficient to show the following.

Lemma 5.2 *Given $S, T \subseteq \Pi$, if $t \perp_{\mathfrak{T}} S^{\mathfrak{T}} \cdot T$ then $t \perp_{\mathfrak{P}} (\Delta S)^{\mathfrak{P}} \cdot \Delta T$.*

Proof. Using the previously introduced notations and arguments, the proof is given by the following derivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{t \perp_{\mathfrak{T}} S^{\mathfrak{T}} \cdot T}{tS^{\mathfrak{T}} \perp_{\mathfrak{T}} T}}{tS^{\mathfrak{T}} \subseteq T^{\mathfrak{T}}}}{tS^{\mathfrak{T}} \perp_{\mathfrak{P}} (T^{\mathfrak{T}})_{\mathfrak{P}}}}{t \perp_{\mathfrak{P}} S^{\mathfrak{T}} \cdot (T^{\mathfrak{T}})_{\mathfrak{P}}}}{t \perp_{\mathfrak{P}} ((S^{\mathfrak{T}})_{\mathfrak{P}})^{\mathfrak{P}} \cdot (T^{\mathfrak{T}})_{\mathfrak{P}}} \;=\; (\Delta S)^{\mathfrak{P}} \cdot \Delta T$$

The conjecture is used in the last step, and the preceding step uses condition (2.1). ∎

It is straightforward to translate this argument into a proof that postcomposition with Δ gives rise to an indexed monotone map $\mathcal{K}_{\mathfrak{T}} \to \mathcal{K}_{\mathfrak{P}}$, and it is relatively easy to show that this map preserves finite meets, and thus gives rise to a functor $\mathbf{Set}[\mathfrak{T}] \to \mathbf{Set}[\mathfrak{P}]$ (see [5]).

We expect this functor to be *regular* – which means that it preserves epimorphisms, and allows to construct $\mathbf{Set}[\mathfrak{P}]$ from a tripos on $\mathbf{Set}[\mathfrak{T}]$ (substantiating our claim that $\mathbf{Set}[\mathfrak{P}]$ is a topos 'over' $\mathbf{Set}[\mathfrak{T}]$) – but the proof of this is expected to require more substantial work.

References

[1] A. Brunel. Quantitative classical realizability. *Inf. Comput.*, 241:62–95, 2015.

[2] J. Frey. Computability and Krivine realizability. note of a conversation with J.L. Krivine, available at https://sites.google.com/site/jonasfreysite/krivine-comp.pdf.

[3] J. Frey. Realizability toposes from specifications. In *13th International Conference on Typed Lambda Calculi and Applications, TLCA 2015, July 1-3, 2015, Warsaw, Poland*, pages 196–210.

[4] J. Frey and J.G. Simonsen. New topos constructions for time complexity classes. *submitted to LICS 2016*, 2016.

[5] J.M.E. Hyland, P.T. Johnstone, and A.M. Pitts. Tripos theory. *Math. Proc. Cambridge Philos. Soc.*, 88(2):205–231, 1980.

[6] S.C. Kleene. On the interpretation of intuitionistic number theory. *J. Symb. Log.*, 10(4):109–124, 1945.

[7] J.L. Krivine. Realizability in classical logic. *Panoramas et synthèses*, 27:197–229, 2009.

[8] C. Riba. Strong normalization as safe interaction. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 13–22, 2007.

[9] T. Streicher. *Domain-theoretic foundations of functional programming.* World Scientific, 2006.

[10] T. Streicher. Krivine's classical realisability from a categorical perspective. *Mathematical Structures in Computer Science*, 23(06):1234–1256, 2013.

[11] T. Streicher and B. Reus. Classical logic, continuation semantics and abstract machines. *Journal of functional programming*, 8(06):543–572, 1998.

[12] P. van Emde Boas. Machine models and simulations. In *Handbook of theoretical computer science, Vol. A*, pages 1–66. Elsevier, Amsterdam, 1990.