

Compositional Higher-Order Model Checking via ω -Regular Games over Böhm Trees

Takeshi Tsukada

University of Oxford, JSPS Postdoctoral Fellow for Research Abroad

C.-H. Luke Ong

University of Oxford

Abstract—We introduce *type-checking games*, which are ω -regular games over Böhm trees, determined by a type of the Kobayashi-Ong intersection type system. These games are a higher-type extension of parity games over trees, determined by an alternating parity tree automaton. However, in contrast to these games over trees, the “game boards” of our type-checking games are composable, using the composition of Böhm trees. Moreover the winner of a composite game is completely determined by the respective winners of the component games.

To our knowledge, type-checking games give the first compositional analysis of *higher-order model checking*, or the model checking of trees generated by recursion schemes. We study a higher-type analogue of higher-order model checking, namely, the problem to decide the winner of a type-checking game over the Böhm tree generated by a λY -term. We introduce a new type system and use it to prove that the problem is decidable.

On the semantic side, we develop a novel arena game model for type-checking games, which is a cartesian closed category equipped with parametric monad and comonad that themselves form a parametrised adjunction.

Index Terms—higher-order model checking, game semantics, Böhm tree, intersection type system, effect system

I. INTRODUCTION

Higher-order model checking (HOMC), or the model checking problem for trees generated by recursion schemes, is a widely studied problem in connection with the verification of higher-order programs [11, 18]. With respect to monadic second-order (MSO) properties, the model checking of trees generated by recursion schemes was first proved to be decidable in 2006 [15], using game semantics [10]. A variety of semantic and algorithmic techniques, and models of computation have since been employed to study HOMC: notably, intersection types [12], collapsible pushdown automata [9] and Krivine machines [19]. Algorithmic properties that refine and extend the MSO decidability result have also been introduced, such as *logical reflection* [2], *effective selection* [3, 8] and *transfer theorem* [20].

In both theory and practice, model checking is (to date) fundamentally a whole program analysis. Perhaps surprisingly, this is also true of HOMC. There are multiple decision procedures that can model check trees generated by ground-type functional programs, but not by higher-type programs, which generate *trees with binders* i.e. Böhm trees. Indeed it is open as to what is an appropriate higher-type analogue of HOMC *qua* decision problem. It seems natural to consider a Böhm tree as a graph with additional edges from bound variables to their binders, but the MSO theory is undecidable even for

a λY -definable Böhm tree [5]. Symptomatic of our limited understanding of HOMC is a lack of a denotational (i.e. compositional) semantics of higher-order programs suitable for the direct analysis of model checking algorithms. A precise and abstract formulation of such a compositional semantics would be a cartesian closed category of parity games.

Our goal, then, is to build a compositional framework for HOMC, using the λY -calculus as our programming language. The primary task is to identify a well-behaved set \mathcal{C} of properties τ over Böhm trees U , and a mechanism for a satisfaction relation $\models U : \tau$ whose validity is characterised by a system of witnesses s . We list some desiderata.

(1) The satisfaction relation \models should *conservatively extend* the MSO properties of trees (i.e. order-2 Böhm trees).

(2) *Decidability of λY -definable Böhm trees*: It should be decidable, given a λY -term M and $\tau \in \mathcal{C}$, whether $\models \text{BT}(M) : \tau$, writing $\text{BT}(M)$ for the Böhm tree of M .

(3) *Two-Level Compositionality*: Suppose Böhm trees U and V are composable, the set of properties satisfied by $U \circ V$ should be completely determined by those of U and of V . Further, if $\models U : \tau$ and $\models V : \sigma$ imply $\models U \circ V : \delta$, the witnesses s_U^τ of $\models U : \tau$ and s_V^σ of $\models V : \sigma$ should be composable, generating a witness $s_{U \circ V}^\delta$ of $\models U \circ V : \delta$.

(4) *Effective Selection*: If $\models \text{BT}(M) : \tau$, then there should exist, constructively, a λY -definable witness of $\models \text{BT}(M) : \tau$.

To this end, our choice of \mathcal{C} is the set of intersection types of the Kobayashi-Ong type system [12], as formulas of Böhm trees; we introduce a new notion of 2-player game over Böhm trees, called *type-checking games*, as the mechanism of a satisfaction relation whose validity is witnessed by P having a winning strategy for the game in question.

A. Contributions of the paper

We start from an algebraic abstraction of the parity conditions, called *summarisable winning condition*. This general notion—every ω -regular winning condition can be viewed as a finite summarisable winning condition—underpins all subsequent developments: our type-checking games (Sec. III), the concomitant type system (Sec. IV), effect arena and two-level games (Sec. V and VI) are all parametrised on a summarisable winning condition and a finite set of ground types.

We introduce *type-checking games*, which are a kind of higher-type property-checking games played over a Böhm tree, the property in question being an intersection type of

the Kobayashi-Ong type system [12]. On the one hand, type-checking games generalise the property-checking games (such as parity games) which are played over trees, to games which are played over *trees with binders*; on the other, they may be viewed as an extension of Stirling’s *dependency tree automata* [21] to infinite (binding) trees with ω -regular conditions. The table below summarises the various aspects of types as a higher-order analogue of alternating parity tree automata.

Trees and Parity Tree Automata	Böhm trees and Types
Tree constructors	Order-1 free variables
Trees	Order-2 Böhm trees
Recursion schemes	λY -terms with order-1 free vars
Value tree of a grammar	Böhm tree of a term
MSO properties / parity tree automata	Order-2 types
A tree accepted by an automaton	A Böhm tree of a type

A remarkable feature of type-checking games is their *intensional* or *two-level compositionality*. Given two such games of composable simple types, not only are their underlying “game boards” composable as Böhm trees, the winning strategies of the composite game are completely determined by the respective winning strategies of the component games (Theorem 11). A key to this result is Lemma 10: certain (but enough) winning strategies of type-checking games are themselves representable as Böhm trees.

We introduce a new intersection type system that characterises the winner of type-checking games for λY -definable Böhm trees. Since HOMC can be regarded as the 2nd-order fragment of the type-checking games, this type system—which is decidable (Theorem 17)—also characterises HOMC, and may be viewed as a conservative extension and simplification of the Kobayashi-Ong type system. Precisely we prove a *transfer theorem* (Theorem 18): P wins the type-checking game determined by a given intersection type τ over the Böhm tree of a λY -term iff that term is typable by τ . The second major result is a higher-order version of *effective selection* (Theorem 20): given an inhabited intersection type, we can effectively construct a λY -term whose Böhm tree represents a P-winning strategy of the type-checking game of that type.

The semantic foundation, and the ultimate basis of the results mentioned above, is a new category of *effect arena games*. To our knowledge, this is the first example of a cartesian closed category of games and winning strategies of ω -regular conditions. We then construct a *two-level game semantics* (in the sense of [16]) as an accurate model of our type system. Our categorical constructions give HOMC examples of positive and negative actions [14] and of parameterised adjunction [1].

B. Related work

Salvati and Walukiewicz [20] have established a transfer theorem for (infinitary) λY -terms: for a fixed vocabulary of terms, the MSO properties of Böhm trees of terms are effectively MSO properties of the terms themselves. However their theorem is only applicable to *Böhm trees without binders* (i.e. order-2 Böhm trees in our terminology). This restriction seems essential to their development. Their proof of the

transfer theorem relies on the (effective) equivalence between MSOL and non-deterministic parity automata for defining tree languages. However, no such equi-expressivity results for defining Böhm-tree languages are known. Moreover, MSOL is undecidable on the binding structures of λY -definable Böhm trees [5]. In contrast, our framework, and in particular our type-based transfer theorem (Theorem 18), work for Böhm trees of all orders definable by (finitary) λY -terms.

Haddad [8] has given a new proof of *effective selection* [3] for recursion schemes, using a notion of shape-preserving scheme transformation. We (Theorem 20) extend Haddad’s result to all higher orders, based on a derivation-as-term interpretation of a new intersection type system. This approach can be found in the work of van Bakel [22] and of Kobayashi et al. [13], but applied to different problems.

II. ALGEBRAIC STRUCTURE OF PRIORITIES

This section introduces the notion of *summarisable winning conditions*, abstracting the algebraic properties of the parity condition (see Example 3). Types and games of this paper are parametrised by a summarisable winning condition.

First we review the notion of ordered ω -semigroups [17]. We write $\langle e_1, e_2, \dots \rangle$ or $\langle e_i \rangle_{i \in \omega}$ for infinite sequences.

Definition 1 (ordered ω -semigroup). A (partially) ordered ω -semigroup is a pair (\mathbb{E}, \mathbb{F}) , where \mathbb{E} and \mathbb{F} are partially ordered sets equipped with the following operations:

- An associative binary operation $\circ : \mathbb{E} \times \mathbb{E} \rightarrow \mathbb{E}$.
- An operation $\otimes : \mathbb{E} \times \mathbb{F} \rightarrow \mathbb{F}$ that defines a semigroup action of \mathbb{E} to \mathbb{F} (i.e. $e_1 \otimes (e_2 \otimes f) = (e_1 \circ e_2) \otimes f$).
- A surjective mapping $\pi : \mathbb{E}^\omega \rightarrow \mathbb{F}$ called *infinite product*.

All the operations must respect the partial orders i.e. (i) $e_1 \preceq e'_1$ and $e_2 \preceq e'_2$ implies $e_1 \circ e_2 \preceq e'_1 \circ e'_2$, (ii) $e \preceq e'$ and $f \preceq f'$ implies $e \otimes f \preceq e' \otimes f'$, and (iii) $e_i \preceq e'_i$ for every i implies $\pi \langle e_i \rangle_{i \in \omega} \preceq \pi \langle e'_i \rangle_{i \in \omega}$. Furthermore the infinite product must satisfy the following laws:

- (i) $e_0 \otimes \pi \langle e_1, e_2, \dots \rangle = \pi \langle e_0, e_1, e_2, \dots \rangle$.
- (ii) If $\langle e_i \rangle_{i \in \omega} \rightsquigarrow \langle e'_j \rangle_{j \in \omega}$, then $\pi \langle e_i \rangle_{i \in \omega} = \pi \langle e'_j \rangle_{j \in \omega}$.

Here \rightsquigarrow is defined by

$$\langle e_1, \dots, e_{i_1}, e_{i_1+1}, \dots, e_{i_2}, e_{i_2+1}, \dots, e_{i_3}, \dots \rangle \rightsquigarrow \langle (e_1 \circ \dots \circ e_{i_1}), (e_{i_1+1} \circ \dots \circ e_{i_2}), (e_{i_2+1} \circ \dots \circ e_{i_3}), \dots \rangle$$

for every infinite chain $0 < i_1 < i_2 < \dots$ of natural numbers.

An *ordered ω -monoid* is an ordered ω -semigroup (\mathbb{E}, \mathbb{F}) such that \mathbb{E} has an identity element. We write ε for the identity element. We have $\varepsilon \otimes f = f$ for every $f \in \mathbb{F}$. An ordered monoid \mathbb{E} has *left-residuals* if there exists a binary operation $e \backslash e'$ s.t. $e \circ d \preceq e'$ iff $d \preceq e \backslash e'$. Then \mathbb{E} as the preorder category is a left-closed (strict) monoidal category.

Definition 2 (Summarisable winning condition). A *summarisable winning condition* (henceforth simply *winning condition*) is a triple $(\mathbb{E}, \mathbb{F}, \Omega)$, where (\mathbb{E}, \mathbb{F}) is an ordered ω -monoid with \mathbb{E} having left-residuals and $\Omega \subseteq \mathbb{F}$ is a lower-closed subset (i.e. $f \in \Omega$ and $f' \preceq f$ implies $f' \in \Omega$).

A winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$ determines the winner of an infinite play associated with an infinite sequence $\langle e_i \rangle_{i \in \omega}$ of elements in \mathbb{E} by: P wins for the play just if $\pi \langle e_i \rangle_{i \in \omega} \in \Omega$.

Given a winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$, we say $\Omega \subseteq \mathbb{F}$ is *stable* if for every $e \in \mathbb{E}$ and $f \in \mathbb{F}$, $f \in \Omega \iff e \otimes f \in \Omega$. A winning condition is *finite* if \mathbb{E} and \mathbb{F} are finite.

Example 3. The parity condition is a finite and stable winning condition. Let $\mathbb{P} = \{0, 1, 2, \dots, 2N\}$ be the set of priorities and $\Omega_{\text{parity}} \subseteq \mathbb{P}^\omega$ be the set of all infinite sequences satisfying the parity condition (i.e. the largest priority that occurs infinitely often is even). The priorities have a monoidal product $e_1 \circ e_2 = \max(e_1, e_2)$ with 0 as the identity. Consider the *sub-priority* order [7]: $2N \prec \dots \prec 4 \prec 2 \prec 0 \prec 1 \prec 3 \prec \dots \prec 2N - 1$. The monoid product respects the sub-priority order. Priorities have left-residuals, defined by:

$$(e \setminus e') := \begin{cases} e' & \text{if } e < e' \text{ or } (e = e' \text{ and } (e \text{ is odd or } 0)) \\ e + 1 & \text{if } e > e' \text{ and } e \text{ is odd} \\ e - 1 & \text{if } e \geq e' \text{ and } e \text{ is even and } e > 0. \end{cases}$$

Consider a winning condition $(\mathbb{P}, \{\mathbf{e}, \mathbf{o}\}, \{\mathbf{e}\})$ with $\mathbf{e} \prec \mathbf{o}$. The infinite product is defined as $\pi \langle e_1, e_2, \dots \rangle = \mathbf{e}$ just if $\langle e_1, e_2, \dots \rangle \in \Omega_{\text{parity}}$ and the mixed product is defined by $e \otimes f = f$. It is finite and $\Omega = \{\mathbf{e}\}$ is stable.

It is well-known that finite ω -semigroups coincide with ω -regular languages [17]. By a similar argument, every ω -regular winning condition can be viewed as a finite summarisable winning condition and vice versa. In particular, for every finite summarisable winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$, the ω -language $\{\langle e_i \rangle_{i \in \omega} \mid \pi \langle e_i \rangle_{i \in \omega} \in \Omega\}$ is ω -regular. See Appendix A-B.

In the sequel, for simplicity, Ω is assumed to be stable. See Appendix F for the general case.

III. TYPE-CHECKING GAMES

This section introduces formulas describing the properties of Böhm trees, which may be viewed as an extension of the Kobayashi-Ong type system [12]. The main result is compositionality (Theorem 11), which will be proved by game semantics in Section VI-B (proofs of other results are in the Appendix).

Henceforth we fix a summarisable winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$ and a finite set Q of ground types.

A. Preliminary: Böhm trees

We briefly review the notion of simply-typed Böhm trees. See Appendix B-A for the formal definitions.

First, we define simple types, which we shall refer to as *kinds* in order to avoid confusion with (intersection) types introduced later. The set of kinds is defined inductively: $\kappa ::= \mathbf{o} \mid \kappa \rightarrow \kappa$, where \mathbf{o} is the unique base kind and $\kappa_1 \rightarrow \kappa_2$ is the kind for functions from κ_1 to κ_2 . A kind environment is a finite sequence of kind bindings of the form $x :: \kappa$. Kind environments are ranged over by Δ . We use $::$ for kind bindings and kind judgements in order to distinguish them from type bindings and type judgements.

We assume the standard notion of Böhm trees, which are possibly infinite trees with binders, defined co-inductively by the grammar: $T ::= \perp \mid x U_1 \dots U_k$ and $U ::= \lambda x_1 \dots \lambda x_k. T$, where \perp (meaning divergence) is a special symbol of kind \mathbf{o} . Böhm trees are required to be well-kinded and η -long. Kind judgements $\Delta \vdash T :: \mathbf{o}$ and $\Delta \vdash U :: \kappa$ have the expected meaning. Böhm trees of appropriate kinds can compose: for Böhm trees U and V of kinds $\kappa \rightarrow \kappa'$ and κ respectively, we write $U @ V$ for the application of U to V , which has kind κ' (see Appendix B-C). It is well-known [6, 10] that Böhm trees of a given kind corresponds bijectively to innocent strategies of the corresponding arena (see Appendix B-B). This bijection bridges the gap between the Böhm-tree representation in this section and the game-semantic analysis in the following sections.

Böhm trees subsume ordinary (node-labelled, ranked) trees. Let $\Sigma = \{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ be a set of tree constructors and $\text{ar}(\mathbf{a}_i)$ be the arity of the constructor \mathbf{a}_i . We write Σ^\perp for $\Sigma \uplus \{\perp\}$, where \perp is of arity 0. A *tree over Σ^\perp* is just a Böhm tree of 2nd-order kind $(\mathbf{o}^{\text{ar}(\mathbf{a}_1)} \rightarrow \mathbf{o}) \rightarrow \dots \rightarrow (\mathbf{o}^{\text{ar}(\mathbf{a}_n)} \rightarrow \mathbf{o}) \rightarrow \mathbf{o}$.

B. Kobayashi-Ong types

We use types of the Kobayashi-Ong type system [12] as formulas specifying properties of Böhm trees. Our syntax simplifies the original system [12] by removing flags from the type environments, and extends it by being parametrised by winning conditions. Furthermore we introduce new operations on types, namely, positive and negative actions of effects.

There are two classes of types, *prime types* and *intersection types*, defined by the following grammar:

$$\begin{aligned} (\text{Prime Types}) \quad \tau, \sigma &::= q \mid \alpha \rightarrow \tau \\ (\text{Intersection Types}) \quad \alpha, \beta &::= \bigwedge_{i \in I} \langle \tau_i; e_i \rangle \end{aligned}$$

where $q \in Q$, $e_i \in \mathbb{E}$ and $I = \emptyset, \omega$ or $\{1, 2, \dots, k\}$ for some $k \in \omega$. Precisely, the set of types is defined by induction on the structure of kinds. See the refinement relation below.

We use the infix notation for intersection, e.g., $\langle \tau_1; e_1 \rangle \wedge \langle \tau_2; e_2 \rangle$ meaning $\bigwedge_{i \in \{1, 2\}} \langle \tau_i; e_i \rangle$. The intersection connective \wedge is *neither* associative, commutative nor idempotent (e.g. $\langle \tau_1; e_1 \rangle \wedge \langle \tau_2; e_2 \rangle \neq \langle \tau_2; e_2 \rangle \wedge \langle \tau_1; e_1 \rangle$ and $\langle \tau; e \rangle \wedge \langle \tau; e \rangle \neq \langle \tau; e \rangle$). However these laws are true modulo the equivalence induced by the subtyping relation (defined below). Note that we allow intersection of countably infinite types $\bigwedge_{i \in \omega} \langle \tau_i; e_i \rangle$. A type is *finite* if all the index sets are finite. We define the projection \mathbf{p}_k by $\mathbf{p}_k(\bigwedge_{i \in I} \langle \tau_i; e_i \rangle) := \langle \tau_k; e_k \rangle$ for each $k \in I$.

We call an element $e \in \mathbb{E}$ an *effect*. The effect e of $\langle \tau; e \rangle$ is the summary of effects from a certain point to the call of τ . For example, e in the judgement $U : \langle \tau; e \rangle$ is the summary of effects from the *beginning of the computation* to the call of U , and e in $\lambda x. T : \langle \tau; e \rangle \rightarrow q$ is the summary of effects from the *call of $\lambda x. T$* to the call of its argument x .

The Kobayashi-Ong type system considers only types that *refine* kinds. The judgement $\tau :: \kappa$ (resp. $\alpha :: \kappa$) means that the prime type τ (resp. the intersection type α) refines the kind

κ . The derivation rules are as follows.

$$\frac{\alpha :: \kappa \quad \tau :: \kappa'}{\bar{q} :: \bar{o} \quad \alpha \rightarrow \tau :: \kappa \rightarrow \kappa'} \quad \frac{\forall i \in I. \tau_i :: \kappa}{\bigwedge_{i \in I} \langle \tau_i; e_i \rangle :: \kappa}$$

The *subtyping relation* is defined by the following mostly standard rules. Precisely it is a kind-indexed family of relations on prime types of the same kind and on intersection types of the same kind. We write $\tau \approx \tau'$ if $\tau \preceq \tau'$ and $\tau' \preceq \tau$.

$$\frac{\alpha' \preceq \alpha \quad \tau \preceq \tau'}{q \preceq q} \quad \frac{\alpha \rightarrow \tau \preceq \alpha' \rightarrow \tau'}{\bigwedge_{i \in I} \langle \tau_i; e_i \rangle \preceq \bigwedge_{j \in J} \langle \sigma_j; d_j \rangle} \quad \frac{\forall j \in J. \exists i \in I. (\tau_i \preceq \sigma_j \text{ and } e_i \succeq d_j)}{\bigwedge_{i \in I} \langle \tau_i; e_i \rangle \preceq \bigwedge_{j \in J} \langle \sigma_j; d_j \rangle}$$

Note that the effect annotation is *contravariant*. Intuitively this is because an effect describes a property of the *caller* (the context) who is contravariant to the callee (the term).

A *type environment* Γ is a finite sequence of intersection type bindings of the form $x : \alpha$ such that the binding variables are pairwise distinct. We write $\Gamma(x) = \alpha$ if $x : \alpha$ is in Γ . The *refinement relation* and the *subtyping relation* for environments are both defined by point-wise extension.

The *positive action* [14] of \mathbb{E} to intersection types and type environments is defined by:

$$e \otimes \left(\bigwedge_{i \in I} \langle \tau_i; d_i \rangle \right) := \bigwedge_{i \in I} \langle \tau_i; e \circ d_i \rangle \quad (e \otimes \Gamma)(x) := e \otimes (\Gamma(x)).$$

Similarly the *negative action* $e \setminus -$ can be defined, e.g.,

$$e \setminus \left(\bigwedge_{i \in I} \langle \tau_i; d_i \rangle \right) := \bigwedge_{i \in I} \langle \tau_i; e \setminus d_i \rangle.$$

They form a Galois connection by the definition of $e \setminus e'$:

$$(e \setminus \Gamma_1) \preceq \Gamma_2 \text{ iff } \Gamma_1 \preceq (e \otimes \Gamma_2).$$

Here $e \setminus -$ is the left-adjoint since the effect annotation is contravariant. The positive action on type environments simplifies and generalises the flag-updating operation in the original type system [12] and the priority-updating operation in the work of Fujima et al. [7]. Its left adjoint (the negative action) is new, to the best of our knowledge.

C. Type-checking games over Böhm trees

It is well-known that the problem of whether a tree is accepted by a given parity tree automaton is equivalent to solving a certain parity game over the tree. This subsection extends such games over trees to games over Böhm trees. Given a Böhm tree U of kind κ and a type $\alpha :: \kappa$, we shall define a *type-checking game*, which is a game between Proponent and Opponent (henceforth, simply P and O) over the Böhm tree U . This is a conservative extension of games over trees, which can be considered as type-checking games restricted to 2nd-order types.

Definition 4 (Type-checking games). A node of the game graph is called a *position*. A position is of the form

$$\Gamma \models T : q \quad \text{or} \quad \Gamma \models (U_1, \dots, U_n) : (\alpha_1, \dots, \alpha_n)$$

where $n \geq 0$. The former is a *P-position* and the latter, written $\Gamma \models (U_i)_{i \leq n} : (\alpha_i)_{i \leq n}$, is an *O-position*. Positions must be

kind-respecting: for the former, $\Gamma :: \Delta$ for some Δ , and $\Delta \vdash T :: o$; and for the latter, $\Gamma :: \Delta$ and $\Delta \vdash U_i :: \kappa_i$ and $\alpha_i :: \kappa_i$ for every i . In our game, *effects* (colours in the standard terminology) are assigned to edges, instead of positions. There are three kinds of edges. The first kind has the form

$$(\Gamma \models x U_1 \dots U_l : q) \xrightarrow{\varepsilon} (\Gamma \models (U_1, \dots, U_l) : (\alpha_1, \dots, \alpha_l))$$

where $p_k(\Gamma(x)) = \langle \alpha_1 \rightarrow \dots \rightarrow \alpha_l \rightarrow q; e \rangle$ with $e \succeq \varepsilon$ for some k . The second kind of edges has the form

$$(\Gamma \models (U_i)_{i \leq n} : (\alpha_i)_{i \leq n}) \xrightarrow{e} ((e \setminus \Gamma), \tilde{x} : \tilde{\beta} \models T : q)$$

where, for some $i \leq n$ and k , $U_i = \lambda x_1 \dots \lambda x_l. T$ and $p_k(\alpha_i) = \langle \beta_1 \rightarrow \dots \rightarrow \beta_l \rightarrow q; e \rangle$ and $\tilde{x} : \tilde{\beta}$ means $x_1 : \beta_1, \dots, x_l : \beta_l$. The third kind of edges is for divergence,

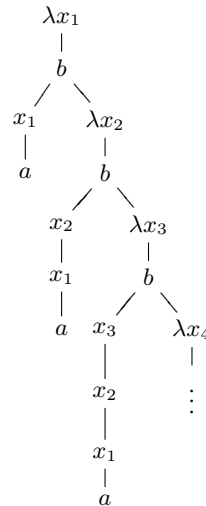
$$(\Gamma \models \perp : q) \xrightarrow{\varepsilon} (\Gamma \models \perp : q).$$

So the unique infinite play starting from $(\Gamma \models \perp : q)$ has effects $\langle \varepsilon, \varepsilon, \dots \rangle$. The winner of an infinite play is determined by the effects of the path: P wins just if $\pi \langle e_1, e_2, \dots \rangle \in \Omega$, where $\langle e_1, e_2, \dots \rangle$ is the sequence of effects along the path. If a finite maximal play ends at a P-position (resp. O-position) then O (resp. P) wins.

Specifying the initial position determines a game. By abuse of notation, we write $\Gamma \models T : q$ and $\Gamma \models U : \alpha$ for the games starting from these positions. Further when the game $\Gamma \models T : q$ has a winning P-strategy, we say $\Gamma \models T : q$ is *valid* and write $\Gamma \models T : q$. We abbreviate $\Gamma \models (U) : (\alpha)$ to $\Gamma \models U : \alpha$ and $\Gamma \models U : \langle \tau; \varepsilon \rangle$ to $\Gamma \models U : \tau$.

In edges from P-positions, P can use only types annotated by ε (or greater). This means that variable x is used immediately without making any effect. The effect ε in the first kind of edges is “meaningless”¹ but ε in the third edges is significant.

Example 5. Let $\Delta = a :: o, b :: o \rightarrow ((o \rightarrow o) \rightarrow o) \rightarrow o$. Consider the order-4 Böhm tree U_0 shown below.



Notice that in U_0 , each bound variable occurs infinitely often, and infinitely many names are required to represent variable binding. Consider the tree property $\varphi = \text{“there are only finitely many occurrences of bound variables in each branch”}$. Let U be a Böhm tree satisfying $\Delta \vdash U :: (o \rightarrow o) \rightarrow o$. Take the parity winning condition consisting of effects $2 \prec 0 \prec 1$, and a single ground type q . Then U satisfies φ iff $\Gamma \models U : (1 \rightarrow 1) \rightarrow 0$ where $\Gamma = a : 1, b : 0 \rightarrow ((1 \rightarrow 1) \rightarrow 0) \rightarrow 1$.² Observe that, because the effect at each covariant position in the typing judgement is 1 and the identity effect $\varepsilon = 0 \preceq 1$, the typing does not

¹Observe that $\pi \langle e_1, e_2, e_3, \dots \rangle = \pi \langle e_1, \varepsilon, e_2, \varepsilon, e_3, \varepsilon, \dots \rangle$.

²Since the ground type is unique, we can abbreviate types α to $\bar{\alpha}$ whereby $\langle \alpha_1 \rightarrow \dots \rightarrow \alpha_n \rightarrow q; e \rangle := \bar{\alpha}_1 \rightarrow \dots \rightarrow \bar{\alpha}_n \rightarrow e$ and $\alpha \wedge \beta := \bar{\alpha} \wedge \bar{\beta}$.

restrict where a bound variable may occur in the input tree U . Intuitively, the effect 1 (at the underlined contravariant position 1) in the type of b accounts for each occurrence of a bound variable.

Consider the type-checking game $\Gamma \models U_0 : (1 \rightarrow 1) \rightarrow 0$. Now $(\Gamma \models (U_0) : ((1 \rightarrow 1) \rightarrow 0)) \xrightarrow{0} (0 \parallel \Gamma, x_1 : 1 \rightarrow 1 \models b(x_1 a) (\lambda x_2. -) : q)$. Since $0 \parallel \Gamma = \Gamma$ and $\varepsilon = 0 \preceq 1$, we have $(\Gamma, x_1 : 1 \rightarrow 1 \models b(x_1 a) (\lambda x_2. -) : q) \xrightarrow{\varepsilon} (\Gamma, x_1 : 1 \rightarrow 1 \models (x_1 a, \lambda x_2. -) : (0, (1 \rightarrow 1) \rightarrow 0))$. Then, with O choosing the right branch and then the left, we have

$$\begin{aligned} & (\Gamma, x_1 : 1 \rightarrow 1 \models (x_1 a, \lambda x_2. -) : (0, (1 \rightarrow 1) \rightarrow 0)) \\ & \xrightarrow{0} (\Gamma, x_1, x_2 : 1 \rightarrow 1 \models b(x_2 (x_1 a)) (\lambda x_3. -) : q) \quad \because \varepsilon \preceq 1 \\ & \xrightarrow{\varepsilon} (\Gamma, x_1, x_2 : 1 \rightarrow 1 \models (x_2 (x_1 a), (\lambda x_3. -)) : (0, (1 \rightarrow 1) \rightarrow 0)) \\ & \xrightarrow{0} (\Gamma, x_1, x_2 : 1 \rightarrow 1 \models x_2 (x_1 a) : q) \\ & \xrightarrow{\varepsilon} (\Gamma, x_1, x_2 : 1 \rightarrow 1 \models (x_1 a) : (1)) \\ & \xrightarrow{1} (\Gamma' \models x_1 a : q) \text{ where } \Gamma' = 1 \parallel (\Gamma, x_1, x_2 : 1 \rightarrow 1) \\ & \xrightarrow{\varepsilon} (\Gamma' \models (a) : (1)) \\ & \xrightarrow{1} (1 \parallel \Gamma' \models a : q) \quad \because (1 \parallel \Gamma')(a) = 1 \parallel 1 = 1 \text{ and } \varepsilon \preceq 1 \\ & \xrightarrow{\varepsilon} (1 \parallel \Gamma' \models () : ()) \end{aligned}$$

and P wins the play. Note that P has a winning strategy.

Remark 6. The game over a tree (i.e. a 2nd-order Böhm tree) determined by a parity tree automaton is a type-checking game over the same tree. The type corresponding to the automaton is constructed as follows. For instance, consider the transition rule $(q_0, a, \{(1, q_1), (2, q_2), (2, q_3)\})$, which means the tree is accepted from q_0 if (i) the root is labelled by a , (ii) the first child of the tree is accepted from q_1 , and (iii) the second child is accepted from both q_2 and q_3 . This rule can be expressed by the type binding $a : \langle \langle q_1; e_1 \rangle \rightarrow \langle \langle q_2; e_2 \rangle \wedge \langle q_3; e_3 \rangle \rangle \rightarrow q_0; 2N - 1 \rangle$, where e_i is the priority of the state q_i and $2N - 1$ is the maximum odd priority (meaning that this rule is available at every position). The type environment corresponding to an APT is just the collection of all such type bindings.

Remark 7. The type-checking games are closely related to (alternating) dependency tree automata [21], automata for (finite) trees with binders like the λ -abstraction of Böhm trees. As recogniser of well-kinded finite Böhm trees (i.e. finite terms in $\beta\eta$ -normal form), dependency tree automata are equi-expressive to type-checking games. From this point of view, type-checking games can be considered as an ω -regular extension of dependency tree automata, which is new, to the best of our knowledge.

Composition of Böhm trees gives us a way to compose the “game boards” of the type-checking games. This is a significant generalisation of the situation of games over trees. The main theorem of the paper states that the winner of a game over $U @ V$ is characterised by the respective winners of the games over its components, U and V .

Theorem 8. $\Gamma \models U_1 @ U_2 : \tau$ if and only if $\Gamma \models U_1 : \alpha \rightarrow \tau$ and $\Gamma \models U_2 : \alpha$ for some α .

This theorem is a consequence of a refined version (Theorem 11) together with Lemma 10 below.

D. Böhm tree representation of a winning strategy

As for a parity game over a tree determined by an alternating parity tree automaton, a winning strategy of the game can be represented by a tree labelled by the transition rules, known as a *run-tree*. Lifting this observation to higher orders, we shall prove that a winning strategy of a type-checking game over a Böhm tree can itself be represented by a Böhm tree.

The basic idea of the Böhm-tree representation of a derivation is similar to our previous work [16], interpreting intersections as products. For example, consider a type environment

$$\Gamma = x : \langle \tau_1; e_1 \rangle \wedge \langle \tau_2; e_2 \rangle \wedge \langle \tau_3; e_3 \rangle, \quad y : \langle \tau_4; e_4 \rangle \wedge \langle \tau_5; e_5 \rangle$$

and assume that $\Gamma \models x U : q$. A strategy-describing Böhm tree is a Böhm tree under the type environment

$$b(\Gamma) = x : \langle \tau_1; e_1 \rangle \times \langle \tau_2; e_2 \rangle \times \langle \tau_3; e_3 \rangle, \quad y : \langle \tau_4; e_4 \rangle \times \langle \tau_5; e_5 \rangle.$$

At the initial position $\Gamma \models x U : q$, P has three options, using $\langle \tau_1; e_1 \rangle$, $\langle \tau_2; e_2 \rangle$ or $\langle \tau_3; e_3 \rangle$. In $b(\Gamma)$, each option corresponds to a component of the product type for x . A strategy choosing $\langle \tau_2; e_2 \rangle$ for the first move is described as $(p_2 x) \hat{V}$, where \hat{V} describes the strategy for succeeding plays. For instance, when $\tau_2 = (\langle \sigma_1; d_1 \rangle \wedge \langle \sigma_2; d_2 \rangle) \rightarrow q$, O can move to $\Gamma \models U : \langle \sigma_1; d_1 \rangle$ and $\Gamma \models U : \langle \sigma_2; d_2 \rangle$. Then \hat{V} is $\hat{V}_1 \sqcap \hat{V}_2$, where \hat{V}_i describes the winning strategy for $\Gamma \models U : \langle \sigma_i; d_i \rangle$ for each $i = 1, 2$.

In order to formalise the above idea, we need to endow Böhm trees and types with indexed products. *Extended Böhm trees* are (co-inductively) defined by the following grammar:

$$\hat{V} ::= \lambda \tilde{x}. \perp \mid \lambda \tilde{x}. (p_k y) \left(\prod_{i \in I_1} \hat{V}_1^i \right) \dots \left(\prod_{i \in I_n} \hat{V}_n^i \right),$$

where \tilde{x} is a possibly empty sequence of variables. In an extended Böhm tree, the subterm at the head position must be a projected variable $p_k y$, and each argument subterm an indexed tuple $\prod_{i \in I} \hat{V}^i$. *Extended types* are closed under indexed products, which are written $\prod_{i \in I} \langle \tau_i; e_i \rangle$, instead of $\bigwedge_{i \in I} \langle \tau_i; e_i \rangle$. We write b for the replacement of intersections by products e.g. $b(\bigwedge_{i \in I} \langle \bigwedge_{j \in J_i} \langle q_{ij}; e_{ij} \rangle \rightarrow q_i; e_i \rangle) = \prod_{i \in I} \langle \prod_{j \in J_i} \langle q_{ij}; e_{ij} \rangle \rightarrow q_i; e_i \rangle$. The notion of type-checking games is naturally extended to extended Böhm trees.

Definition 9. The relation $\hat{V} \in U$ between extended Böhm trees and (standard) Böhm trees is defined by the following rules (see Appendix B-G for details): $\lambda \tilde{x}. \perp \in \lambda \tilde{x}. \perp$, and

$$\lambda \tilde{x}. (p_k y) \left(\prod_{i \in I_1} \hat{V}_1^i \right) \dots \left(\prod_{i \in I_n} \hat{V}_n^i \right) \in \lambda \tilde{x}. y U_1 \dots U_n$$

where $\hat{V}_k^i \in U_k$ for every $k \leq n$ and $i \in I_k$. We write $\hat{V} \blacktriangleright (\Gamma \models U : \tau)$ just if $\hat{V} \in U$ and $b(\Gamma) \models \hat{V} : b(\tau)$. In that case, we say \hat{V} is a *run-Böhm-tree*.

The following lemma justifies the Böhm-tree representation of winning strategies.

Lemma 10. $\Gamma \models U : \tau$ iff $\hat{V} \blacktriangleright (\Gamma \models U : \tau)$ for some \hat{V} .

As the main contribution of the paper, we show how winning strategies can be composed and decomposed. This is the “proof-relevant” version of compositionality (Theorem 8). This theorem will be proved by game semantics in the sequel.

Theorem 11. (i) $\widehat{V}_1 \blacktriangleright (\Gamma \models U_1 : \alpha \rightarrow \tau)$ and $\widehat{V}_2 \blacktriangleright (\Gamma \models U_2 : \alpha)$ implies $(\widehat{V}_1 @ \widehat{V}_2) \blacktriangleright (\Gamma \models U_1 @ U_2 : \tau)$. (ii) If $\widehat{W} \blacktriangleright (\Gamma \models U_1 @ U_2 : \tau)$, then $\widehat{V}_1 \blacktriangleright (\Gamma \models U_1 : \alpha \rightarrow \tau)$ and $\widehat{V}_2 \blacktriangleright (\Gamma \models U_2 : \alpha)$ and $\widehat{W} = \widehat{V}_1 @ \widehat{V}_2$ for some α , \widehat{V}_1 and \widehat{V}_2 .

IV. TYPE SYSTEM FOR MODEL-CHECKING BÖHM TREES

This section studies (decision procedures for) the *Model-Checking Problem for $\lambda\mathbf{Y}$ -definable Böhm Trees* i.e. whether $\models \text{BT}(M) : \tau$ for a given $\lambda\mathbf{Y}$ -term M and type τ , where $\text{BT}(M)$ is the Böhm tree of M . We develop a type system that characterises the winner of type-checking games for $\lambda\mathbf{Y}$ -definable Böhm trees. Since the standard higher-order model-checking can be regarded as the second-order fragment of the type-checking games, the type system also characterises higher-order model checking (for tree-generating $\lambda\mathbf{Y}$ -terms).

In this section, we assume that $(\mathbb{E}, \mathbb{F}, \Omega)$ is finite. Otherwise some algorithmic results would fail.

A. Preliminary: $\lambda\mathbf{Y}$ -calculus

The set of terms is given by: $M ::= x \mid M M \mid \lambda x.M \mid \mathbf{Y}_\kappa$, where \mathbf{Y}_κ is the fixed-point combinator of kind $(\kappa \rightarrow \kappa) \rightarrow \kappa$. The kinding rules are defined as usual. We write $M[N/x]$ for the standard capture-avoiding substitution. The reduction relation is given by the following rules

$$(\lambda x.M) N \longrightarrow M[N/x] \quad \mathbf{Y} M \longrightarrow M(\mathbf{Y} M),$$

with the η -expansion rule and the congruence rule (i.e. if $M \longrightarrow M'$, then $M N \longrightarrow M' N$ and $N M \longrightarrow N M'$).

A $\lambda\mathbf{Y}$ -term M generates the Böhm tree $\text{BT}(M)$ as the result of infinite reductions. An important fact is that $\text{BT}(-)$ and the application commute in the following sense.

Proposition 12. Let $(M N)$ be a well-kinded $\lambda\mathbf{Y}$ -term. Then $\text{BT}(M N) = \text{BT}(M) @ \text{BT}(N)$.

B. Typing rules

Definition 13. Assume a kinded term $\Delta \vdash M :: \kappa$. There are three kinds of judgements for M : *prime type judgement* $\Gamma \vdash M : \tau$, *prime effect judgement* $\Gamma \vdash M : \langle \tau; e \rangle$, and *intersection type judgement* $\Gamma \vdash M : \alpha$, where $\Gamma :: \Delta$, $\tau :: \kappa$ and $\alpha :: \kappa$. The typing rules are as follows:

$$\begin{array}{c} \frac{\langle \tau; e \rangle = p_i(\Gamma(x)) \text{ for some } i \text{ and } e \succeq \varepsilon}{\Gamma \vdash x : \tau} \quad \frac{\Gamma, x : \alpha \vdash M : \tau}{\Gamma \vdash \lambda x.M : \alpha \rightarrow \tau} \\[10pt] \frac{\Gamma \vdash M : \alpha \rightarrow \tau \quad \Gamma \vdash N : \alpha}{\Gamma \vdash M N : \tau} \quad \frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash M : \tau \quad \tau \preceq \tau'}{\Gamma' \vdash M : \tau'} \\[10pt] \frac{\Gamma \vdash M : \tau}{e @ \Gamma \vdash M : \langle \tau; e \rangle} \quad \frac{\forall i \in I. \Gamma \vdash M : \langle \tau_i; e_i \rangle}{\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle} \quad \frac{}{\Gamma \vdash \mathbf{Y} : \tau} \end{array}$$

and the subtyping rule for $\Gamma \vdash M : \langle \tau; e \rangle$. The variable rule requires x to have type τ with effect ε , since x is used immediately without any effects. The positive-action rule

applies e -action to both sides. This rule would be unsound or incomplete if Ω were not stable. The rule for \mathbf{Y} just checks if the judgement is true in the Böhm-tree semantics. Without relying on the type-checking game, we can define an equivalent game for \mathbf{Y} on a finite graph.

Definition 14 (Game for \mathbf{Y}). Given a prime type τ of kind $(\kappa \rightarrow \kappa) \rightarrow \kappa$, we define the game $G(\tau)$ as follows. Here types are considered modulo \approx . A P-position has the form $\alpha \models_P \sigma$, where $\alpha :: \kappa \rightarrow \kappa$ and $\sigma :: \kappa$, and an O-position has the form $\alpha \models_O \beta$, where $\alpha :: \kappa \rightarrow \kappa$ and $\beta :: \kappa$. Recall that σ, τ (resp. α, β) range over prime (resp. intersection) types. Edges are defined by:

$$(\alpha \models_P \tau) \xrightarrow{\varepsilon} (\alpha \models_O \beta),$$

whenever $p_k(\alpha) = \langle \beta \rightarrow \tau'; e \rangle$ with $e \succeq \varepsilon$ and $\tau' \preceq \tau$ for some k , and

$$(\alpha \models_O \beta) \xrightarrow{e} (e \parallel \alpha \models_P \tau)$$

whenever $p_k(\beta) = \langle \tau; e \rangle$ for some k . The initial position of $G(\alpha \rightarrow \tau)$ is $\alpha \models_P \tau$. The winner of an infinite play is determined by the (infinite sequence of) effects along the play.

Lemma 15. P wins for $\models \text{BT}(\mathbf{Y}) : \tau$ iff P wins for $G(\tau)$.

Remark 16. Though our game $G(\tau)$ has the same flavour as the game used in the Kobayashi-Ong system [12], it requires careful effect handling. That is because \mathbf{Y} can be applied to an open term, whereas the recursion of recursion schemes always takes closed terms. For a closed term M , $\vdash M : \tau$ is equivalent to $\vdash M : \langle \tau; e \rangle$ for every e , so the effect of this position is not significant. In contrast, as for open terms, the effect e of $\Gamma \vdash M : \langle \tau; e \rangle$ affects Γ . This is handled by the left-residual operation $e \parallel \alpha$ in edges from O-positions.

Assuming a finite winning condition, the set of types modulo \approx is finite for each kind. It follows that $G(\tau)$ is an ω -regular game on a finite graph; hence $G(\tau)$ is decidable. Further, for every term $\Delta \vdash M :: \kappa$, there are only finitely many type judgements $\Gamma \vdash M : \tau$ modulo \approx . Therefore, by an induction on the structure of the term, the type-checking problem is decidable.

Theorem 17 (Decidability). $\Gamma \vdash M : \tau$ is decidable.

The type system is sound and complete with respect to the type-checking games. The proof is by induction on the structure of the term, using Proposition 12 and Theorem 8.

An *order- n transfer theorem* for a set \mathcal{C} of properties says that the \mathcal{C} -definable properties of order- n $\lambda\mathbf{Y}$ -definable Böhm trees are effectively the \mathcal{C} -definable properties of the term-generators themselves. See Appendix G for further details.

Theorem 18 (Transfer). $\Gamma \vdash M : \tau$ iff $\Gamma \models \text{BT}(M) : \tau$.

Proof: (Sketch) If $M = \mathbf{Y}$, both directions are trivial by the typing rule. Assume $M = M_1 M_2$. If $\Gamma \vdash M_1 M_2 : \tau$, then $\Gamma \vdash M_1 : \alpha \rightarrow \tau$ and $\Gamma \vdash M_2 : \alpha$ for some α by the typing rules. By the induction hypothesis, we have $\Gamma \models \text{BT}(M_1) : \alpha \rightarrow \tau$ and $\Gamma \models \text{BT}(M_2) : \alpha$. By Theorem 8, we

have $\Gamma \models \text{BT}(M_1) @ \text{BT}(M_2) : \tau$. By Proposition 12, we have $\Gamma \models \text{BT}(M_1 M_2) : \tau$ as required. The converse is similar. See Appendix C-D for other cases. ■

Example 19. Recall the Böhm tree U_0 in Example 5. Set $M = \mathbf{Y}(\lambda f^{\circ \rightarrow (\circ \rightarrow \circ) \rightarrow \circ} . \lambda y^{\circ} . \lambda x^{\circ \rightarrow \circ} . b(x y)(f(x y))) a$ with $\Delta \vdash M :: (\circ \rightarrow \circ) \rightarrow \circ$. Then $U_0 = \text{BT}(M)$. By $\vdash \mathbf{Y} : \langle \langle \tau; 0 \rangle \rightarrow \tau; 0 \rangle \rightarrow \tau$, where $\tau = \langle q; 1 \rangle \rightarrow (\langle \langle q; 1 \rangle \rightarrow q; 1 \rangle) \rightarrow q$, we have $\Gamma \vdash M : \langle \langle \langle q; 1 \rangle \rightarrow q; 1 \rangle \rightarrow q; 0 \rangle$ (which is abbreviated to $\Gamma \vdash M : (1 \rightarrow 1) \rightarrow 0$ in Example 5). The tree $\text{BT}(M)$, due to Clairambault and Murawski [5], is interesting. When viewed as a graph, consisting of the tree edges augmented by the binding relation (so that each bound variable points to its binder), $\text{BT}(M)$ has an undecidable MSO theory [5]. However, thanks to Theorems 17 and 18, problems such as whether $\text{BT}(M)$ satisfies a given type-describable property (e.g. φ in Example 5) are decidable.

C. Effective selection

As we have seen, if P wins for $\Gamma \models \text{BT}(M) : \tau$, there exists a Böhm tree representing a winning strategy. It is natural to ask: can we effectively construct a $\lambda\mathbf{Y}$ -term D such that $\text{BT}(D)$ represents a winning strategy of the game i.e. $\text{BT}(D) \blacktriangleright (\Gamma \models \text{BT}(M) : \tau)$? This subsection proves that D can easily be constructed from a derivation of $\Gamma \vdash M : \tau$ by induction on the derivation. This result extends the effective selection result [3, 8] to higher-order judgements.

A derivation is *finite* if all types in it are finite. If $\Gamma \vdash M : \tau$ is derivable and Γ and τ are finite, then it has a finite derivation. A *representation* of a finite derivation $\Gamma \vdash M : \tau$ is a $\lambda\mathbf{Y}$ -term $\flat(\Gamma) \vdash D : \flat(\tau)$ (augmented with finite products), defined by induction on the derivation. We write $D \triangleright (\Gamma \vdash M : \tau)$ if D is a representation of a derivation concluding $\Gamma \vdash M : \tau$. See Appendix C-E for the complete list of rules.

$$\begin{array}{c} \frac{\text{BT}(D) \blacktriangleright (\models \text{BT}(\mathbf{Y}) : \tau)}{D \triangleright (\Gamma \vdash \mathbf{Y} : \tau)} \quad \frac{\mathbf{p}_i(\Gamma(x)) = \langle \tau; e \rangle \text{ and } e \succeq \varepsilon}{(\mathbf{p}_i x) \triangleright (\Gamma \vdash x : \tau)} \\ \frac{D_i \triangleright (\Gamma \vdash M : \langle \tau_i; e_i \rangle) \quad (i = 1, 2, \dots, k)}{(\prod_{i \in \{1, \dots, k\}} D_i) \triangleright (\Gamma \vdash M : \bigwedge_{i \in \{1, \dots, k\}} \langle \tau_i; e_i \rangle)} \\ \frac{D_0 \triangleright (\Gamma \vdash M : \alpha \rightarrow \tau) \quad D_1 \triangleright (\Gamma \vdash N : \alpha) \quad \text{some } \alpha}{(D_0 D_1) \triangleright (\Gamma \vdash M N : \tau)} \end{array}$$

The second main theorem states that a representation D actually generates a winning strategy. The proof is the same as that of the soundness direction of Theorem 18, but uses the “proof-relevant” version (Theorem 11) instead of Theorem 8.

Theorem 20. $D \triangleright (\Gamma \vdash M : \tau)$ implies $\text{BT}(D) \blacktriangleright (\Gamma \models \text{BT}(M) : \tau)$.

Now what is left is the effective construction of D for the base case of \mathbf{Y} . Since $G(\tau)$ is an ω -regular game, it has a finite-memory winning strategy. The expected $\lambda\mathbf{Y}$ -term can be extracted from this winning strategy, using mutual recursion. For more details, see Appendix C-E.

Lemma 21. If P wins for $\models \mathbf{Y} : \tau$, then one can effectively construct a $\lambda\mathbf{Y}$ -term D s.t. $\text{BT}(D) \blacktriangleright (\models \mathbf{Y} : \tau)$.

If $\Gamma \models \text{BT}(M) : \tau$, one can effectively construct a finite derivation of $\Gamma \vdash M : \tau$ and compute its term representation by using Lemma 21, resulting in a $\lambda\mathbf{Y}$ -term D such that $D \triangleright (\Gamma \vdash M : \tau)$. Recall that, if M is of 2nd-order, $\text{BT}(M)$ is the value tree and a winning strategy is an accepting run-tree. So this result gives an effective construction of an accepting run-tree, which is known as *effective selection* [3, 8]. In fact, our result leads to a generalisation of the previous result to arbitrary higher-order judgement.

V. EFFECT ARENA GAMES

Given a summarisable winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$ and a finite set Q of ground types, we construct a category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ of effect arena games. The category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is cartesian closed and, when Ω is stable as we have assumed, it is equipped with an adjunction $(-\| -) \dashv (- \otimes -)$ parametrised by \mathbb{E} in the sense of Atkey [1] where \mathbb{E} is considered a preorder category. Furthermore $(-\otimes -) : \mathbb{E}^{\text{op}} \times \mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)} \rightarrow \mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is itself a positive action (equivalently, a parametric comonad) [14].

An important result of this section is the well-definedness of the category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$; the main technical lemmas are concerned with the preservation of the winning condition by strategy composition.

A. Effect arenas

The category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is based on \mathcal{Inn} , the category of arenas (consisting of only question moves) and innocent strategies [10]. An object A of $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is an arena $|A|$, called the *underlying arena*, equipped with an assignment of an effect for each move. The hom-set $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}(A, B)$ can naturally be viewed as a subset of $\mathcal{Inn}(|A|, |B|)$.

Definition 22 (Effect arena). An *effect arena* (or *arena* for short) A is a tuple $(\mathcal{M}_A, \vdash_A, \lambda_A, \vartheta_A, E_A)$ where

- 1) \mathcal{M}_A is a set of *moves*,
- 2) $\vdash_A \subseteq (\mathcal{M}_A \times \mathcal{M}_A)$ is an *enabling relation*,
- 3) $\lambda_A : \mathcal{M}_A \rightarrow \{P, O\}$ is the *ownership function*,
- 4) $\vartheta_A : \mathcal{M}_A \rightarrow Q$ is a *ground-type assignment*, and
- 5) $E_A : \mathcal{M}_A \rightarrow \mathbb{E}$ is an *effect assignment*.

We write $\vdash_A m$ if there is no m' such that $m' \vdash_A m$. The enabling relation must satisfy the conditions: (i) For each $m \in \mathcal{M}_A$, either $\vdash_A m$ or $m' \vdash_A m$ for a unique $m' \in \mathcal{M}_A$. (ii) If $\vdash_A m$, then $\lambda_A(m) = O$. If $m \vdash_A m'$, then $\lambda_A(m) \neq \lambda_A(m')$.

An effect arena is just a standard arena equipped with effect and ground-type assignments, which are used to determine the set of “good” innocent strategies, called *consistent* strategies.

For an arena A , the set $\mathcal{M}_A^{\text{init}} \subseteq \mathcal{M}_A$ of *initial moves* of A is $\{m \in \mathcal{M}_A \mid \vdash_A m\}$. A move $m \in \mathcal{M}_A$ is called an *O-move* if $\lambda_A(m) = O$ and a *P-move* if $\lambda_A(m) = P$. We often write m^P (resp. m^O) to make the owner explicit.

A *justified sequence* of an arena A is a sequence of moves such that each element except the first is equipped with a *justification pointer* to some previous move. A *play* of an arena A is a justified sequence s that satisfies: (i) Well-openness, (ii) Alternation, (iii) Justification, and (iv) Visibility. (The definition of these standard notions can be found in [10].)

A *P-strategy* (or *strategy*) \mathfrak{s} of an arena A is a prefix-closed subset of plays of A that satisfies Determinacy and Contingent Completeness. An infinite justified sequence is said to be an *infinite play* if all its finite prefixes are plays. A strategy contains an infinite play just if it contains all its finite prefixes. An *infinite P-view* is an infinite play all of whose finite prefixes are P-views. We use bold symbols like \mathfrak{s} and \mathfrak{p} for infinite plays and P-views.

Strategies are not required to be total. If a play $s \in \mathfrak{s}$ ending with an O-move is not answered by \mathfrak{s} , we write $s \cdot \perp \in \mathfrak{s}$. A strategy \mathfrak{s} is *innocent* just if for every pair of plays $s \cdot m$, $s' \cdot m' \in \sigma$ ending with P-moves m and m' , $\lceil s \rceil = \lceil s' \rceil$ implies $\lceil s \cdot m \rceil = \lceil s' \cdot m' \rceil$, writing $\lceil s \rceil$ to mean the P-view of s .

Consistency of innocent strategies has three criteria: Ground-type Reflection, Correctness of Summary and Winning Condition. Ground-type Reflection was introduced in our previous work [16] (called *Colour Reflection* therein): An innocent strategy is *ground-type reflecting* if $s \cdot m_1^O \cdot m_2^P \in \mathfrak{s}$ implies $\vartheta(m_1^O) = \vartheta(m_2^P)$. This captures well-typedness in the simple type system with multiple ground types.

Definition 23 (Correctness of Summary). Let A be an effect arena and s be a play of the underlying arena $|A|$. The effect of a P-move m^P in s is a *correct summary* if $E(m^P)$ is a summary of the respective effects of the O-moves between m and its justifier. Formally, assume $s = s_0 \cdot m^P \cdot s_1$ and let

$$\lceil s_0 \rceil \cdot m^P = p_0 \cdot n_1^O \cdot n_2^P \cdot n_3^O \cdot \dots \cdot n_{2k-1}^O \cdot m^P.$$

Then the effect $E(m^P)$ is a correct summary just if

$$\varepsilon \circ E(n_3) \circ E(n_5) \circ \dots \circ E(n_{2k-1}) \preceq E(m).$$

A play s is *summarising* if for every P-move m that occurs in s , the effect of m is a correct summary. A strategy is *summarising* just if every play in the strategy is summarising.

Notice that we only consider P-views in the definition of correctness of summary (because there may be an irrelevant O-move outside of the P-view between the moves in question).

We introduce some convenient notations. Given a finite sequence of moves $s = m_1 \cdot m_2 \cdot \dots \cdot m_k$ (not necessarily a play), we write $s|_O$ for the subsequence consisting of O-moves. When applied to plays, the resulting subsequence is no longer a play. The effect map is naturally extended to sequence of moves as $E^*(m_1 \cdot m_2 \cdot \dots \cdot m_k) := E(m_1) \circ E(m_2) \circ \dots \circ E(m_k)$ (and ε if $k = 0$). Using these notations, the correctness of summary can be expressed as $p \cdot n \cdot s \cdot m \in \mathfrak{s}$ implies $E^*(s|_O) \preceq E(m)$. For an infinite sequence of moves, we define $E^\omega(m_1 \cdot m_2 \cdot \dots) := \pi \langle E(m_1), E(m_2), \dots \rangle$.

Definition 24 (Winning Condition). An innocent strategy \mathfrak{s} of an effect arena A satisfies the *winning condition* (or simply Ω) if the following conditions hold:

- 1) **Infinite P-views:** For every *infinite P-view* $\mathfrak{p} = m_1^O \cdot m_2^P \cdot m_3^O \cdot \dots \in \mathfrak{s}$, the sequence of effects of O-moves satisfies Ω , i.e. $\pi \langle E(m_1^O), E(m_3^O), E(m_5^O), \dots \rangle \in \Omega$.

- 2) **Infinite Chattering:** For every *unanswered finite P-view* $m_1^O \cdot \dots \cdot m_{2k-1}^O \cdot \perp \in \mathfrak{s}$, considering \perp as the infinite ε s, one requires $\pi \langle E(m_1^O), \dots, E(m_{2k-1}^O), \varepsilon, \varepsilon, \varepsilon, \dots \rangle \in \Omega$.

The condition on P-views is rephrased as $E^\omega(\mathfrak{p}|_O) \in \Omega$.

Remark 25. If $\pi \langle \varepsilon, \varepsilon, \dots \rangle \in \Omega$, then every unanswered finite P-views satisfies Ω . If not, consistent strategies must be total.

Definition 26 (Consistency). An innocent strategy \mathfrak{s} of an effect arena A is *consistent* if \mathfrak{s} is ground-type reflecting, summarising and satisfies the winning condition.

B. Products and exponentials

Given two effect arenas A and B , we construct the product $A \times B$ and the exponential $A \Rightarrow B$. The underlying arenas of the product and of the exponential are the product and the exponential of underlying arenas. I.e. $|A \times B| = |A| \times |B|$ and $|A \Rightarrow B| = |A| \Rightarrow |B|$. Recall that $\mathcal{M}_{A \times B} = \mathcal{M}_A + \mathcal{M}_B$ and $\mathcal{M}_{A \Rightarrow B} = \mathcal{M}_B^{\text{init}} \times \mathcal{M}_A + \mathcal{M}_B$. The effect assignments for $A \times B$ and $A \Rightarrow B$ are defined by:

$$E_{A \times B}(m) := \begin{cases} E_A(m) & (\text{if } m \in \mathcal{M}_A) \\ E_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$$

and

$$\begin{aligned} E_{A \Rightarrow B}((m, n)) &:= E_B(m) \setminus E_A(n) & (\text{if } n \in \mathcal{M}_A^{\text{init}}) \\ E_{A \Rightarrow B}((m, n)) &:= E_A(n) & (\text{if } n \notin \mathcal{M}_A^{\text{init}}). \\ E_{A \Rightarrow B}(m_B) &:= E_B(m_B) & (\text{if } m_B \in \mathcal{M}_B) \end{aligned}$$

The use of residuals corresponds to the negative action to environment in edges from O-positions in type-checking games.

C. Category of consistent strategies

We say an arena A *satisfies the winning condition* (or simply Ω) just if $m_1 \vdash m_2 \vdash \dots$ implies $\pi \langle E(m_1), E(m_2), \dots \rangle \in \Omega$.

Definition 27. The category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ of effect arenas and consistent strategies is defined by the following data.

- **Objects:** An effect arena that satisfies Ω .
- **Maps:** $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}(A, B)$ is the set of all innocent and consistent strategies of $A \Rightarrow B$.

The identity maps and composition are inherited from the underlying arena game model.

To prove that $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is well-defined, it suffices to show (i) consistency of the identity maps, and (ii) consistency of the composite of consistent strategies. The consistency of the identity is rather straightforward.

Lemma 28. For every effect arena A that satisfies Ω , the identity id_A of the underlying arena is consistent.

We prove that for every consistent strategies $\mathfrak{s} : A \Rightarrow B$ and $\mathfrak{t} : B \Rightarrow C$, the composite $(\mathfrak{s}; \mathfrak{t}) : A \Rightarrow C$ is consistent.

First we consider correctness of summary. An interaction sequence $s \in \text{Int}(A, B, C)$ is said to *generate a P-view* just if $s \upharpoonright_{A \Rightarrow C}$ is a P-view. Given an interaction sequence s , we write $s \upharpoonright_{A \Rightarrow C}^O$ for the subsequence of $s \upharpoonright_{A \Rightarrow C}$ consisting of O-moves. The first key lemma is about effects on an interaction sequence of summarising strategies.

Lemma 29. Assume that $\mathfrak{s} : A \Rightarrow B$ and $\mathfrak{t} : B \Rightarrow C$ are summarising strategies. Let $s \in \text{Int}(\mathfrak{s}, \mathfrak{t})$ be an interaction sequence generating a P-view. Take any move m in s and let

$$s = s_1 \cdot \overbrace{n \cdot s_2 \cdot m}^{\text{arc}} \cdot s_3.$$

Then $E_{(A \Rightarrow B) \Rightarrow C}^*((s_2 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \preceq E_{(A \Rightarrow B) \Rightarrow C}(m)$.

Corollary 30. If \mathfrak{s} and \mathfrak{t} are summarising, so is $\mathfrak{s}; \mathfrak{t}$.

Now we study an infinite interaction sequence generating an infinite P-view, in order to prove the preservation of the winning condition by composition. Let $\mathfrak{s} = m_1 \cdot m_2 \cdots \in \text{Int}(|A|, |B|, |C|)$ be an infinite interaction sequence. Its *sub-view* is an infinite subsequence $\mathfrak{p} = n_1 \cdot n_2 \cdots$ of \mathfrak{s} s.t.

$$\mathfrak{s} = s_1 \cdot n_1 \cdot \overbrace{n_2 \cdot s_3 \cdot n_3}^{\text{arc}} \cdot \overbrace{n_4 \cdot s_5 \cdot n_5}^{\text{arc}} \cdot \overbrace{n_6 \cdot s_7 \cdot n_7}^{\text{arc}} \cdots,$$

where n_1 is an initial C -move or an initial B -move. A sub-view starting from an initial C -move is a P-view of $B \Rightarrow C$, and a sub-view starting from an initial B -move can be viewed as a P-view of $A \Rightarrow B$. The second key lemma says that every infinite interaction sequence has an infinite sub-view. The proof relies on a result of Clairambault and Harmer [4].

Lemma 31. Let $\mathfrak{s} : |A| \Rightarrow |B|$ and $\mathfrak{t} : |B| \Rightarrow |C|$ be innocent strategies and $\mathfrak{s} \in \text{Int}(\mathfrak{s}, \mathfrak{t})$ be an infinite interaction sequence generating a P-view. Then \mathfrak{s} has an infinite P-view of \mathfrak{s} or \mathfrak{t} as its sub-view.

Lemma 32. If \mathfrak{s} and \mathfrak{t} are summarising and satisfies Ω then $\mathfrak{s}; \mathfrak{t}$ satisfies Ω .

Proof: (Sketch) Here we check that the winning condition for infinite P-views. See Appendix D-F for more details. Let $\mathfrak{p} \in (\mathfrak{s}; \mathfrak{t})$ be an infinite P-view in the composite, and $\mathfrak{s} \in \text{Int}(\mathfrak{s}, \mathfrak{t})$ be the interaction sequence such that $\mathfrak{p} = \mathfrak{s} \upharpoonright_{A \Rightarrow C}$. By Lemma 31, \mathfrak{s} has an infinite P-view of \mathfrak{s} or \mathfrak{t} as its sub-view. Consider the case that it is an infinite P-view \mathfrak{q} of \mathfrak{s} . The other case can be proved similarly. Assume that $\mathfrak{s} = m_1 \cdot m_2 \cdots$ and $\mathfrak{q} = n_1 \cdot n_2 \cdots$. Since \mathfrak{q} is a sub-view of \mathfrak{s} , we have

$$\mathfrak{s} = m_1 \cdot s_1 \cdot \overbrace{n_1 \cdot n_2 \cdot s_3 \cdot n_3}^{\text{arc}} \cdot \overbrace{n_4 \cdot s_5 \cdot n_5}^{\text{arc}} \cdot n_6 \cdots.$$

Note that n_{2k} cannot be an O-move of $A \Rightarrow C$ and hence

$$\mathfrak{p} \upharpoonright^O = \mathfrak{s} \upharpoonright_{A \Rightarrow C}^O = m_1 \cdot ((s_1 \cdot n_1) \upharpoonright_{A \Rightarrow C}^O) \cdot ((s_3 \cdot n_3) \upharpoonright_{A \Rightarrow C}^O) \cdots.$$

Let us define $t_{2k+1} = (s_{2k+1} \cdot n_{2k+1}) \upharpoonright_{A \Rightarrow C}^O$. Then $\mathfrak{p} \upharpoonright^O = m_1 \cdot t_1 \cdot t_3 \cdots$. By Lemma 29, for every k , we have

$$E_{(A \Rightarrow B) \Rightarrow C}^*(t_{2k+1}) \preceq E_{(A \Rightarrow B) \Rightarrow C}(n_{2k+1}).$$

Evaluating $E_{A \Rightarrow C}^\omega(\mathfrak{p} \upharpoonright^O)$ carefully, we have

$$\begin{aligned} E_{A \Rightarrow C}^\omega(\mathfrak{p} \upharpoonright^O) &= \pi \langle E_{A \Rightarrow C}(m_1) \circ E_{A \Rightarrow C}^*(t_1), E_{A \Rightarrow C}^*(t_3), \dots \rangle \\ &\preceq \pi \langle E_{A \Rightarrow B}(n_1), E_{A \Rightarrow B}(n_3), \dots \rangle. \end{aligned}$$

The last expression is in Ω because \mathfrak{s} satisfies Ω . Hence $E_{A \Rightarrow C}^\omega(\mathfrak{p} \upharpoonright^O) \in \Omega$ by lower-closedness of Ω . ■

The next theorem follows from Corollary 30 and Lemma 32.

Theorem 33. The composite of consistent strategies is consistent. Therefore $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ is well-defined.

D. Categorical structure of $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$

For simplicity, we write \mathcal{G} for $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$. Since an effect arena is an ordinary arena with additional information, it is natural to consider the forgetful functor $|-| : \mathcal{G} \rightarrow \text{Inn}$. Because $\mathcal{G}(A, B)$ maps forgetfully to $\text{Inn}(|A|, |B|)$ and the identities and strategy composition are inherited from Inn , $|-|$ surely defines a functor.

Proposition 34. $A \times B$ is a cartesian product and $A \Rightarrow B$ is an exponential in \mathcal{G} . Furthermore $|-|$ is a CCC functor.

For an arena A and effect e , we define $e \otimes A$ to be an arena that has the same underlying arena and the same ground-type assignment as A , equipped with the effect assignment

$$E_{e \otimes A}(m) := \begin{cases} e \circ E_A(m) & (\text{if } m \in \mathcal{M}_A^{\text{init}}) \\ E_A(m) & (\text{otherwise}). \end{cases}$$

By defining $e \otimes \mathfrak{s} = \mathfrak{s}$ for strategies, $e \otimes -$ is a functor for every e . (This follows from the fact that Ω is stable.) Further it is a (strict) positive action of \mathbb{E}^{op} [14].

Lemma 35. $- \otimes -$ is a bifunctor $\mathbb{E}^{\text{op}} \times \mathcal{G} \rightarrow \mathcal{G}$. Furthermore $(e_1 \circ e_2) \otimes (-) = e_1 \otimes (e_2 \otimes (-))$ and $\varepsilon \otimes (-) = \text{id}_{\mathcal{G}}$.

For each $e \in \mathbb{E}$, $e \otimes -$ has a left-adjoint $e \backslash - : \mathcal{G} \rightarrow \mathcal{G}$. The arena $e \backslash A$ has the same underlying arena as A . The effect assignment is given by

$$E_{e \backslash A}(m) := \begin{cases} e \backslash E_A(m) & (\text{if } m \in \mathcal{M}_A^{\text{init}}) \\ E_A(m) & (\text{otherwise}). \end{cases}$$

For strategies, $e \backslash \mathfrak{s} = \mathfrak{s}$ similarly to $e \otimes (-)$.

Lemma 36. $- \backslash -$ is a bifunctor $\mathbb{E} \times \mathcal{G} \rightarrow \mathcal{G}$. It satisfies $(e_2 \circ e_1) \backslash - = e_1 \backslash (e_2 \backslash -)$ and $\varepsilon \backslash - = \text{id}_{\mathcal{G}}$.

It is easy to see $e \backslash - \dashv e \otimes -$ for every $e \in \mathbb{E}$, and hence they form an \mathbb{E} -parametrised adjunction in the sense of Atkey [1].

VI. TWO-LEVEL GAMES

Based on the effect arena games of Section V, we shall build a game model that can interpret intersection types with effect annotations. The constructions of this section are a straightforward adaptation of the *two-level constructions* introduced in our previous work [16].

A. Two-level effect arena games

Definition 37 (Two-Level Arenas). A *two-level arena* is a triple (A, ϖ, K) , where A is an effect arena, K is an arena without effects, and $\varpi : \mathcal{M}_A \rightarrow \mathcal{M}_K$ is a map of moves that preserves the enabling relation i.e. $\vdash_A m$ implies $\vdash_K \varpi(m)$ and $m \vdash_A m'$ implies $\varpi(m) \vdash_K \varpi(m')$. In case the map ϖ is clear from the context, we abbreviate (A, ϖ, K) to $A :: K$.

The map of moves can be extended naturally to a map of plays, $\varpi(m_1 \cdot m_2 \cdots m_k) := \varpi(m_1) \cdot \varpi(m_2) \cdots \varpi(m_k)$. Notice that the RHS is actually a play of K , since the enabling relations are preserved by ϖ . Take a two-level arena (A, ϖ, K) and strategies \mathfrak{s} of A and \mathfrak{t} of K . We write $\mathfrak{s} \subset_{\varpi} \mathfrak{t}$ just if $s \in \mathfrak{s}$ implies $\varpi(s) \in \mathfrak{t}$ for every s . We omit ϖ and simply write $\mathfrak{s} \subset \mathfrak{t}$ if ϖ is clear from the context.

Definition 38 (Two-Level Strategies). A *two-level strategy* of a two-level arena (A, ϖ, K) is just a pair $s :: t$ of innocent strategies of A and of K respectively such that $s \sqsubset \varpi$. It is *consistent* if s is a consistent strategy of A . It is *relatively total* if $s \cdot \perp \in s$ implies $\varpi(s) \cdot \perp \in t$. A consistent and relatively total two-level strategy is said to be *winning*.

The *product* (resp. *exponential*) of two-level arenas is defined as component-wise products (resp. exponentials) equipped with the obvious map of moves. The category $\mathcal{G} :: \mathcal{Inn}$ has two-level arenas as objects and the maps from $(A :: J)$ to $(B :: K)$ are the two-level winning strategies of $(A :: J) \Rightarrow (B :: K)$. Composition is defined component-wise.

Lemma 39. *Let $s_1 :: t_1$ and $s_2 :: t_2$ be winning two-level strategies. Then the composite $(s_1; s_2) :: (t_1; t_2)$ is winning.*

The category of two-level games is a CCC by the products and exponentials above. Projection on the second component $p : (\mathcal{G} :: \mathcal{Inn}) \rightarrow \mathcal{Inn} : (A :: K) \mapsto K$ is a strict CCC functor. This functor is neither a fibration nor an opfibration, but satisfies a weaker property that suffices for completeness.

Lemma 40 (De-substitution). *Let $(s :: t) : (A :: I) \rightarrow (C :: K)$. Then for any decomposition of t in \mathcal{Inn} , i.e., $I \xrightarrow{t} K = I \xrightarrow{t_1} J \xrightarrow{t_2} K$, we have a decomposition of $s :: t$ in $\mathcal{G} :: \mathcal{Inn}$*

$$(A :: I) \xrightarrow{s :: t} (C :: K) = (A :: I) \xrightarrow{s_1 :: t_1} (B :: J) \xrightarrow{s_2 :: t_2} (C :: K)$$

for some B , s_1 and s_2 .

Two-level arenas (A_1, ϖ_1, K) and (A_2, ϖ_2, K) sharing the same K have another kind of the product, defined by $(A_1, \varpi_1, K) \wedge (A_2, \varpi_2, K) := (A_1 \times A_2, \varpi, K)$, where $\varpi := [\varpi_1, \varpi_2] : \mathcal{M}_{A_1} + \mathcal{M}_{A_2} \rightarrow \mathcal{M}_K$. This two-level arena is the pull-back of a certain diagram.

Positive action $e \otimes -$ is defined by $e \otimes (A :: K) := (e \otimes A) :: K$ on two-level arenas and $e \otimes (s :: t) := s :: t$ on two-level strategies. Negative action is defined similarly.

B. Connection to type-checking games

The results in Section III described by Böhm trees are just a rephrasing of results in this section described by the two-level arena game model, via the bijective correspondence between Böhm trees and innocent strategies.

The game semantics of types mapping types to arenas is straightforwardly defined by using the CCC structure, fibred products and effect actions. Given $q \in Q$, we define the two-level arena $\mathfrak{D}_q = (A, \varpi, K)$, where A consists of a unique O-move of effect ε and ground type q , K consists of a unique O-move and ϖ is the canonical map. The two-level arena interpretation $\llbracket \cdot \rrbracket$ of types is given by:

$$\begin{aligned} \llbracket q :: o \rrbracket &= \mathfrak{D}_q & \llbracket \bigwedge_{i \in I} \langle \tau_i; e_i \rangle :: \kappa \rrbracket &= \bigwedge_{i \in I} (e_i \otimes \llbracket \tau_i :: \kappa \rrbracket) \\ \llbracket (\alpha \rightarrow \tau) :: (\kappa \rightarrow \kappa') \rrbracket &= \llbracket \alpha :: \kappa \rrbracket \Rightarrow \llbracket \tau :: \kappa' \rrbracket. \end{aligned}$$

A type environment is interpreted as the product of its components. We simply write $\llbracket \tau \rrbracket$ leaving the kind implicit.

Assume the map $\langle \cdot \rangle$ from kinds to (ordinary) arenas, and the bijection from the Böhm trees of kind κ to the innocent

strategies of the arena $\langle \kappa \rangle$. By abuse of notation, this bijective map is also written as $\langle \cdot \rangle$. The next lemma bridges the gap between the type-checking games and the two-level game model. See Appendix E-C for more details. Theorem 11 is a corollary of Lemma 39, Lemma 40 and Lemma 41.

Lemma 41. $\widehat{V} \blacktriangleright (\Gamma \vdash U : \tau)$ iff $(\langle \widehat{V} \rangle :: \langle U \rangle) : \llbracket \Gamma \rrbracket \rightarrow \llbracket \tau \rrbracket$.

Further directions

The type-describable properties seem not to be closed under union and negation. The introduction of negation to higher-order types could be of practical significance, since negative ground types have played a key rôle in PREFACE, a state-of-the-art model checker [18]. It would be interesting to understand the general semantic picture when \mathbb{E} is a category, rather than a preorder.

REFERENCES

- [1] R. Atkey. Parameterised notions of computation. *J. Funct. Program.*, 19(3-4):335–376, 2009.
- [2] C. H. Broadbent, A. Carayol, C.-H. L. Ong, and O. Serre. Recursion schemes and logical reflection. In *LICS*, pages 120–129, 2010.
- [3] A. Carayol and O. Serre. Collapsible pushdown automata and labeled recursion schemes: Equivalence, safety and effective selection. In *LICS*, pages 165–174, 2012.
- [4] P. Clairambault and R. Harmer. Totality in arena games. *Ann. Pure Appl. Logic*, 161(5):673–689, 2010.
- [5] P. Clairambault and A. S. Murawski. Böhm trees as higher-order recursive schemes. In *FSTTCS*, pages 91–102, 2013.
- [6] P.-L. Curien. Abstract Böhm trees. *Mathematical Structures in Computer Science*, 8(6):559–591, 1998.
- [7] K. Fujima, S. Ito, and N. Kobayashi. Practical alternating parity tree automata model checking of higher-order recursion schemes. In *APLAS*, 2013.
- [8] A. Haddad. Model checking and functional program transformations. In *FSTTCS*, pages 115–126, 2013.
- [9] M. Hague, A. S. Murawski, C.-H. L. Ong, and O. Serre. Collapsible pushdown automata and recursion schemes. In *LICS*, pages 452–461, 2008.
- [10] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II, and III. *Inf. Comput.*, 163(2):285–408, 2000.
- [11] N. Kobayashi. Model checking higher-order programs. *J. ACM*, 60(3):20, 2013.
- [12] N. Kobayashi and C.-H. L. Ong. A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In *LICS*, pages 179–188, 2009.
- [13] N. Kobayashi, K. Matsuda, and A. Shinohara. Functional programs as compressed data. In *PEPM*, pages 121–130, 2012.
- [14] P.-A. Melliès. Parametric monads and enriched adjunctions. Preprint, 28 pages, 2012.
- [15] C.-H. L. Ong. On model-checking trees generated by higher-order recursion schemes. In *LICS*, pages 81–90, 2006.
- [16] C.-H. L. Ong and T. Tsukada. Two-level game semantics, intersection types, and recursion schemes. In *ICALP (2)*, pages 325–336, 2012.
- [17] D. Perrin and J. Pin. *Infinite Words: Automata, Semigroups, Logic and Games*. Academic Press, 2004.
- [18] S. J. Ramsay, R. P. Neatherway, and C.-H. L. Ong. A type-directed abstraction refinement approach to higher-order model checking. In *POPL*, pages 61–72, 2014.
- [19] S. Salvati and I. Walukiewicz. Krivine machines and higher-order schemes. In *ICALP (2)*, pages 162–173, 2011.
- [20] S. Salvati and I. Walukiewicz. Evaluation is MSOL-compatible. In *FSTTCS*, pages 103–114, 2013.
- [21] C. Stirling. Dependency tree automata. In *FOSSACS*, pages 92–106, 2009.
- [22] S. van Bakel. The heart of intersection type assignment: Normalisation proofs revisited. *Theor. Comput. Sci.*, 398(1-3):82–94, 2008.

This section proves the claims in Section II.

A. Parity condition as a summarisable winning condition

This subsection proves the properties of the parity condition claimed in Section II. Assume the maximum priority $2N \in \omega$. Let $\mathbb{P} = \{0, 1, 2, \dots, 2N\}$ be the set of priorities. The operation \circ on \mathbb{P} is defined by $e_1 \circ e_2 = \max(e_1, e_2)$. The order \preceq is define by:

$$2N \prec \dots \prec 6 \prec 4 \prec 2 \prec 0 \prec 1 \prec 3 \prec 5 \prec \dots \prec 2N - 1.$$

Let Ω_{parity} be the parity condition, i.e. the set of all infinite sequence in which the maximum priority found infinitely often is even.

Proposition 42. Assume that $\langle e_i \rangle_{i \in \omega} \rightsquigarrow \langle e'_i \rangle_{i \in \omega}$. Then $\pi \langle e_i \rangle_{i \in \omega} = \pi \langle e'_i \rangle_{i \in \omega}$.

Proof: It suffices to prove that $\langle e_i \rangle_{i \in \omega} \in \Omega_{\text{parity}}$ iff $\langle e'_i \rangle_{i \in \omega} \in \Omega_{\text{parity}}$.

$\langle e_1, e_2, \dots \rangle \in \Omega_{\text{parity}}$ if and only if there exists a natural number $l \in \omega$ and an even priority $p \in \mathbb{P}$ such that

- For every $j \geq l$, $e_j \leq p$, and
- For every $j \in \omega$, there exists $j' \geq j$ such that $e_{j'} = p$.

Assume that $\langle e_1, e_2, \dots \rangle \in \Omega_{\text{parity}}$ and an infinite sequence $0 < k_1 < k_2 < \dots$. We prove that

$$\langle (e_1 \circ \dots \circ e_{k_1}), (e_{k_1+1} \circ \dots \circ e_{k_2}), \dots \rangle = \langle e'_1, e'_2, \dots \rangle \in \Omega_{\text{parity}}.$$

Since $\langle e_1, e_2, \dots \rangle \in \Omega_{\text{parity}}$, we have l and p that satisfy the above conditions. Since $\lim_{l \rightarrow \infty} k_l = \infty$, we have $l < k_{l'}$ for some l' . Then $l' + 1$ and p satisfy the above conditions for $\langle e'_1, e'_2, \dots \rangle$.

To prove the converse, assume an infinite sequence $0 < k_1 < k_2 < \dots$ and an infinite sequence of priorities $\langle e_i \rangle_{i \in \omega}$. Let

$$\langle e'_i \rangle_{i \in \omega} = \langle (e_1 \circ \dots \circ e_{k_1}), (e_{k_1+1} \circ \dots \circ e_{k_2}), \dots \rangle$$

and suppose that $\langle e'_i \rangle_{i \in \omega} \in \Omega_{\text{parity}}$. So we have l' and p that satisfies the above condition for $\langle e'_i \rangle_{i \in \omega}$. Let $l = k_{l'}$. Then l satisfies the first condition for $\langle e_i \rangle_{i \in \omega}$. Let $j \in \omega$. By the second condition for j and $\langle e'_i \rangle_{i \in \omega}$, there exists $n \geq j$ such that

$$e'_n = e_{k_{n-1}+1} \circ \dots \circ e_{k_n} = p.$$

Hence $e_{j'} = p$ for some $j' \in \{k_{n-1} + 1, \dots, k_n\}$. Since $n - 1 \leq k_{n-1}$, we have $j \leq n = (n - 1) + 1 \leq k_{n-1} + 1 \leq j'$ as desired. ■

Proposition 43. If $e_1 \preceq e'_1$ and $e_2 \preceq e'_2$, then $e_1 \circ e_2 \preceq e'_1 \circ e'_2$.

Proof: Since \circ is symmetric, it suffices to prove that $d_1 \preceq d_2$ implies $e \circ d_1 \preceq e \circ d_2$. Assume that d_2 is the successor of d_1 . Then the claim can be proved by case analysis.

- Case $d_1 \in \{2, 4, 6, \dots, 2N\}$: The successor d_2 is $d_1 - 2$.
 - If $e \leq d_2$, then $e \circ d_1 = d_1 \preceq d_2 = e \circ d_2$.

- If $d_2 < e < d_1$, then $e \circ d_1 = d_1$ and $e \circ d_2 = e$. In this case, e must be an odd number, so $d_1 \preceq e$.
- If $d_1 \leq e$, then $e \circ d_1 = e = e \circ d_2$.
- Case $d_1 = 0$: The successor d_2 is 1.
 - If $e = 0$, then $e \circ d_1 = 0 \preceq 1 = e \circ d_2$.
 - If $e \geq 1$, then $e \circ d_1 = e = e \circ d_2$.
- Case $d_1 \in \{1, 3, \dots, 2N - 3\}$: The successor d_2 is $d_1 + 2$.
 - If $e \leq d_1$, then $e \circ d_1 = d_1 \preceq d_2 = e \circ d_2$.
 - If $d_1 < e < d_2$, then $e \circ d_1 = e$ and $e \circ d_2 = d_2$. In this case, e must be an even number, so $e \preceq d_2$.
 - If $d_2 \leq e$, then $e \circ d_1 = e = e \circ d_2$.

Since \preceq is transitive, we can prove the general case by induction on the “distance” between d_1 and d_2 in the list sorted by \preceq . ■

Proposition 44. Assume that $e'_i \preceq e_i$ for every $i \in \omega$. Then $\pi \langle e'_i \rangle_{i \in \omega} \preceq \pi \langle e_i \rangle_{i \in \omega}$.

Proof: It suffices to prove that $\langle e_i \rangle_{i \in \omega} \in \Omega_{\text{parity}}$ implies that $\langle e'_i \rangle_{i \in \omega} \in \Omega_{\text{parity}}$.

Assume that $\langle e_1, e_2, \dots \rangle \in \Omega$. Let M be the maximum priority found in the sequence and L be the maximum priority found infinitely often in the sequence. Since $\langle e_1, e_2, \dots \rangle \in \Omega$, we know that L is even. Now we have an infinite sequence of indexes $0 < i_1 < i_2 < i_3 < \dots$ which satisfies

$$e_1 \circ e_2 \circ \dots \circ e_{i_1} = M$$

$$e_{i_1+1} \circ \dots \circ e_{i_2} = L$$

$$e_{i_2+1} \circ \dots \circ e_{i_3} = L$$

⋮

Now by Proposition 43, we have

$$e'_1 \circ e'_2 \circ \dots \circ e'_{i_1} = M' \preceq M$$

$$e'_{i_1+1} \circ \dots \circ e'_{i_2} = L'_2 \preceq L$$

$$e'_{i_2+1} \circ \dots \circ e'_{i_3} = L'_3 \preceq L$$

⋮

Since \mathbb{P} is a finite set, there exists L' that appears infinite times in L'_2, L'_3, \dots . Because $L' \preceq L$ and L is even, we know that L' must be even. Hence we have $\langle e'_1, e'_2, \dots \rangle \in \Omega_{\text{parity}}$. ■

Recall that the operation $e \setminus e'$ on priorities is defined by:

$$(e \setminus e') := \begin{cases} e' & \text{if } e < e' \text{ or } (e = e' \text{ and } (e \text{ is odd or } 0)) \\ e + 1 & \text{if } e > e' \text{ and } e \text{ is odd} \\ e - 1 & \text{if } e \geq e' \text{ and } e \text{ is even and } e > 0. \end{cases}$$

We prove that this operation surely gives the left-residual.

Lemma 45. $e \circ d \preceq e'$ iff $d \preceq e \setminus e'$.

Proof: By the case analysis.

(Case $e < e'$) If e' is even, then we have:

$$e \circ d \preceq e'$$

iff $e \circ d \geq e'$ and $e \circ d$ is even

iff $d \geq e'$ and $e \circ d = d$ is even

iff $d \preceq e'$.

If e' is odd, then we have:

- $e \circ d \preceq e'$
- iff $e \circ d \leq e'$ or $e \circ d$ is even
- iff $e \circ d \leq e'$ or ($e \circ d$ is even and $e \circ d > e'$)
- iff $d \leq e'$ or (d is even and $d > e'$)
- iff $d \leq e'$ or d is even
- iff $d \preceq e'$.

(Case $e = e'$ and e is odd)

- $e' \circ d \preceq e'$
- iff $e' \circ d \leq e'$ or $e' \circ d$ is even
- iff $e' \circ d \leq e'$ or ($e' \circ d$ is even and $e' \circ d > e'$)
- iff $d \leq e'$ or (d is even and $d > e'$)
- iff $d \leq e'$ or d is even
- iff $d \preceq e'$.

(Case $e = e' = 0$) It suffices to prove that $0 \circ d \preceq 0$ iff $d \preceq 0 \setminus 0 = 0$. It is trivial.

(Case $e > e'$ and e is odd) Assume that $d \preceq e \setminus e' = e + 1$. Since $e + 1$ is even, d is even and $d \geq e + 1$. Hence $e \circ d = d$. By $d \geq e + 1$ and $e > e'$, we have $d \geq e'$ and hence $d \preceq e'$ as desired.

Assume that $e \circ d \preceq e'$. Note that $e \succ e'$ by the assumption. Therefore $d > e$ and $d \preceq e'$, which implies $d \succeq e$. These conditions implies that d is even and $d > e$. Hence $d \preceq e + 1$.

(Case $e \geq e'$ and e is even and $e > 0$) Note that $e \preceq e'$.

Assume that $d \preceq e \setminus e' = e - 1$. Since $e - 1$ is odd, either (i) $d \leq e - 1$, or (ii) $d > e - 1$ and d is even. In case (i), we have $e \circ d = e \preceq e'$. In case (ii), we have $e \circ d = d \preceq e \preceq e'$.

Assume that $e \circ d \preceq e'$. There are three cases:

- $d \leq e$: Then $e \circ d = e \preceq e'$ always holds. If $d = e$, we have $d \preceq e - 1$ since d is even and $e - 1$ is odd. If $d < e - 1$, then $d \preceq e - 1$ since $e - 1$ is odd.
- $d > e$ and d is even: Then $e \circ d = d \preceq e \preceq e'$ always holds. We have $d \preceq e - 1$ since d is even and $e - 1$ is odd.
- $d > e$ and d is odd: Then $d > e \geq e'$ and d is odd, and hence $d \succ e'$. This contradict to the assumption $e \circ d = d \preceq e'$.

B. Finite summarisable winning condition and ω -regularity

This section studies the connection between summarisable acceptance conditions and ω -regular languages. Let Σ be a finite set and $L \subseteq \Sigma^\omega$ is an ω -language. A summarisable winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$ can recognise L when there exists a map $h : \Sigma \rightarrow \mathbb{E}$ such that $\langle a_1, a_2, \dots \rangle \in L$ iff $\pi(h(a_1), h(a_2), \dots) \in \Omega$. A finite summarisable winning condition coincides with an ω -regular language in the following sense.

Proposition 46. *Let Σ be a finite set and $L \subseteq \Sigma^\omega$ is an ω -language over Σ . Then L is ω -regular iff L can be recognised*

by a finite summarisable winning condition (here Ω is not necessarily stable).

Proof: We start from a well-know result for finite partially-ordered ω -semigroups [17].

Proposition. *An ω -language L is ω -regular if and only if it can be recognised by a finite partially-ordered ω -semigroup (\mathbb{E}, \mathbb{F}) equipped with a lower-closed subset $\Omega \subseteq \mathbb{F}$.*

The right-to-left direction is an immediate consequence of this proposition, since a finite summarisable winning condition is a finite ω -semigroup.

We prove the left-to-right direction. Assume that L is ω -regular. By the above proposition, there exists an ω -semigroup (\mathbb{E}, \mathbb{F}) equipped with a lower-closed subset $\Omega \subseteq \mathbb{F}$ and a map $h : \Sigma \rightarrow \mathbb{E}$ such that $\langle a_1, a_2, \dots \rangle \in L$ if and only if $\pi(h(a_1), h(a_2), \dots) \in \Omega$. We can assume without loss of generality that \mathbb{E} is a monoid. Otherwise add ε to \mathbb{E} and \perp to Ω . These elements are incomparable to other elements. Operations for newly added elements are defined by:

- $\varepsilon \circ e = e \circ \varepsilon = e$ for every $e \in \mathbb{E} \setminus \{\varepsilon\}$.
- $\varepsilon \otimes f = f$ for every $f \in \mathbb{F}$.
- $e \otimes \perp = \perp$ for every $e \in \mathbb{E} \setminus \{\varepsilon\}$.
- $\pi\langle e_1, e_2, \dots \rangle$ is defined by the following rules.
 - Case that the sequence restricted to \mathbb{E} is infinite: Let $\langle e'_1, e'_2, \dots \rangle$ be the infinite sequence removing all ε s from $\langle e_1, e_2, \dots \rangle$. Then $\pi\langle e_1, e_2, \dots \rangle := \pi\langle e'_1, e'_2, \dots \rangle$.
 - Case that the sequence restricted to \mathbb{E} is finite: Then $\pi\langle e_1, e_2, \dots \rangle := \perp$.

Now we have an ω -monoid (\mathbb{E}, \mathbb{F}) equipped with a lower-closed subset $\Omega \subseteq \mathbb{F}$ and a map $h : \Sigma \rightarrow \mathbb{E}$ such that $\langle a_1, a_2, \dots \rangle \in L$ iff $\pi(h(a_1), h(a_2), \dots) \in \Omega$. It suffices to prove that \mathbb{E} has left-residuals, but it is not the case in general. We shall define an extension $(\mathbb{E}', \mathbb{F}', \Omega')$ such that $\mathbb{E}' \supseteq \mathbb{E}$, $\mathbb{F}' \supseteq \mathbb{F}$ and $\Omega' \supseteq \Omega$ (in a certain sense) and prove that \mathbb{E}' has left-residuals. They are defined by:

- $\mathbb{E}' = \mathcal{P}(\mathbb{E})$,
- $\mathbb{F}' = \mathcal{P}(\mathbb{F})$, and
- $\Omega' = \{O \in \mathcal{P}(\mathbb{F}) \mid O \subseteq \Omega\}$.

Here $\mathcal{P}(A)$ means the powerset of A , and \mathbb{E}' and \mathbb{F}' are order by set-inclusion. By identifying an elements e of \mathbb{E} to the singleton set $\{e\} \in \mathbb{E}'$, \mathbb{E} can be viewed as a subset of \mathbb{E}' . Operations are defined by:

- For $E_1, E_2 \in \mathbb{E}'$ (i.e. $E_1, E_2 \subseteq \mathbb{E}$), $E_1 \circ E_2 := \{e_1 \circ e_2 \mid e_1 \in E_1, e_2 \in E_2\}$.
- For $E \in \mathbb{E}'$ and $F \in \mathbb{F}'$ (i.e. $E \subseteq \mathbb{E}$ and $F \subseteq \mathbb{F}$), $E \otimes F := \{e \otimes f \mid e \in E, f \in F\}$.
- For $\langle A_i \rangle_{i \in \omega}$, $\pi\langle A_i \rangle_{i \in \omega} := \{\pi\langle e_i \rangle_{i \in \omega} \mid \langle e_i \rangle_{i \in \omega} \in \prod_{i \in \omega} A_i\}$, where $\prod_{i \in \omega} A_i$ is the set of infinite sequences of \mathbb{E} such that i th element is in A_i for every $i \in \omega$.

It is easy to see that $E_1 \circ E_2 \subseteq E'_1 \circ E'_2$ if $E_1 \subseteq E'_1$ and $E_2 \subseteq E'_2$. The ordered monoid \mathbb{E} has all joins defined by the set union. Furthermore $E \circ \emptyset = \emptyset \subseteq E'$ for every $E, E' \subseteq \mathbb{E}$.

Now we define the left-residual by

$$E_1 \setminus E_2 := \bigcup \{E' \subseteq \mathbb{E} \mid E_1 \circ E' \subseteq E_2\}.$$

It is easy to prove that $E_1 \circ E' \subseteq E_2$ iff $E' \subseteq E_1 \setminus E_2$. The map $h' : \Sigma \rightarrow \mathbb{E}'$ is defined by $h'(a) := \{h(a)\}$. Then $(\mathbb{E}', \mathbb{F}', \Omega')$ with h' satisfies all the requirements. ■

For example, a Muller condition can be viewed as a finite and stable summarisable winning condition.

Example 47. A Muller condition can be considered as a finite summarisable winning condition as follows. Let Q be a finite set and $P \subseteq \mathcal{P}(Q)$. We define $\mathbb{E} = \mathcal{P}(\mathcal{P}(Q))$ with product $e_1 \circ e_2 = \{A_1 \cup A_2 \mid A_1 \in e_1 \text{ and } A_2 \in e_2\}$. The order is given by $e_1 \preceq e_2$ if and only if $e_1 \subseteq e_2$. The left-residuals can be defined by

$$e_1 \setminus e_2 := \bigcup \{e' \in \mathcal{P}(\mathcal{P}(Q)) \mid e_1 \circ e' \subseteq e_2\}.$$

For an infinite sequence $\langle A_i \rangle_{i \in \omega}$ of subsets of Q , we define $\inf \langle A_i \rangle_{i \in \omega} \subseteq Q$ by

$$q \in \inf \langle A_i \rangle_{i \in \omega} \text{ iff } \#\{i \mid q \in A_i\} = \infty.$$

Now we write $\langle B_i \rangle_{i \in \omega} \in \bar{\Omega} \subseteq \mathcal{P}(\mathcal{P}(Q))^\omega$ when

$$\forall \varphi : \omega \rightarrow \mathcal{P}(Q). (\exists i. \varphi(i) \notin B_i) \text{ or } (\inf \langle \varphi(1), \varphi(2), \dots \rangle \in P).$$

We define $\mathbb{F} = \{\mathbf{t}, \mathbf{f}\}$ with $\mathbf{t} \prec \mathbf{f}$ and $\Omega = \{\mathbf{t}\}$. The infinite product is defined as $\pi \langle B_1, B_2, \dots \rangle = \mathbf{t}$ if and only if $\langle B_1, B_2, \dots \rangle \in \bar{\Omega}$. A state $q \in Q$ can be embedded into $\mathbb{E} = \mathcal{P}(\mathcal{P}(Q))$ by $h : q \mapsto \{\{q\}\}$. This winning condition with the map h recognises the Muller condition, i.e. $\langle q_1, q_2, \dots \rangle$ satisfies the Muller condition determined by P if and only if $\pi \langle h(q_1), h(q_2), \dots \rangle = \mathbf{t}$. This winning condition is finite and stable.

APPENDIX B

SUPPLEMENTARY MATERIALS FOR SECTION III

A. Definition of Böhm trees

This subsection gives the definition of Böhm trees, which are fairly standard.

We call the simple types *kinds* in order to avoid confusion with (intersection) types. The set of kinds is defined by the following grammar:

$$\kappa ::= \circ \mid \kappa \rightarrow \kappa.$$

The unique ground kind \circ is the kind of λ , and $\kappa \rightarrow \kappa'$ is the kind of functions from κ to κ' as usual.

A *Böhm tree* is a possibly infinite tree, each node of which has a label of the form $\lambda x_1 \dots \lambda x_k. y$, which we abbreviate to $\lambda x_1 \dots x_k. y$, or $\lambda x_1 \dots x_k. \perp$. Böhm trees are co-inductively defined by the grammar:

$$\begin{aligned} T &::= \perp \mid x U_1 \dots U_k \\ U &::= \lambda x_1 \dots x_k. T \end{aligned}$$

where $k \geq 0$. We consider only well-kinded Böhm trees in η -long form. A judgement is of the form $\Delta \Vdash T :: \circ$ or $\Delta \Vdash U :: \kappa$. Here \Vdash is used to distinguish between judgements for terms

and Böhm trees. We write $\Delta(x) = \kappa$ if $\Delta = \Delta_0, x :: \kappa, \Delta_1$. The kinding rules are given by:

$$\begin{aligned} &\overline{\Delta \Vdash \perp :: \circ} \\ &\frac{\Delta(x) = \kappa_1 \rightarrow \dots \rightarrow \kappa_k \rightarrow \circ \quad \forall i \leq k. \Delta \Vdash U_i :: \kappa_i}{\Delta \Vdash x U_1 \dots U_k :: \circ} \\ &\frac{\Delta, x_1 :: \kappa_1, \dots, x_k :: \kappa_k \Vdash T :: \circ}{\Delta \Vdash \lambda x_1 \dots x_k. T :: \kappa_1 \rightarrow \dots \rightarrow \kappa_k \rightarrow \circ} \end{aligned}$$

Here the rules should be interpreted co-inductively, and thus a derivation tree can be infinite. Note that, even though T may be an infinite tree having infinitely many lambda abstractions, each subterm of T is equipped with a finite kind environment.

For the definition of composition, see Appendix B-C.

B. Böhm trees are innocent strategies

In this subsection, we give the well-known bijective correspondence between Böhm trees of a given kind and innocent strategies of the corresponding arena [10, 6]. There is an obvious bijection between kinds and finite tree arenas. Henceforth we shall use kinds and arenas interchangeably.

Notation. In the following, we use A, B, C , etc., to range over kinds. For ease of type-setting, we abbreviate $A_1 \rightarrow \dots \rightarrow A_n \rightarrow \circ$ to (A_1, \dots, A_n, \circ) .

Recall that innocent strategies \mathfrak{s} can be represented as prefix-closed, P-deterministic (i.e. whenever P-views $p \cdot m^O \cdot n_1^P$ and $p \cdot m^O \cdot n_2^P \in \mathfrak{s}$ then $n_1^P = n_2^P$) and O-full (i.e. for every P-view $p \cdot m^O$, if $p \in \mathfrak{s}$ then $p \cdot m^O \in \mathfrak{s}$) sets of even-length P-views (here $p \cdot \perp \in \mathfrak{s}$ is considered as an even-length P-view).

Lemma 48. *Let $\Delta = z_1 : C_1, \dots, z_m : C_m$ be a kind environment. There is a bijective correspondence between Böhm trees $\Delta \Vdash U :: A$ and innocent strategies $\langle U \rangle : \prod_{i=1}^m C_i \rightarrow A$. Further there is a bijective correspondence between finite paths (from the root) of a given Böhm tree U and even-length P-views of the strategy $\langle U \rangle$.*

First we introduce a naming scheme for moves of a given arena. Since each move m of an arena A is the root of a unique subarena C (say), we shall name m by referring to C , and say that m has kind C .

An arena may be viewed as a graph whose edge-set is the enabling relation. It follows from the definition that this graph is a forest, whose roots (which are on level 0) are the initial moves; further, moves on level 0, 2, 4, etc., are O-moves, and those on levels 1, 3, 5, etc., are P-moves.

Assume a denumerable set of kinded variables. Fix an arena A . A P-move of A of kind C is named by a (kinded) variable $x : C$. An O-move of kind (A_1, \dots, A_n, \circ) is named by an expression of the form $\lambda x_1 : A_1 \dots \lambda x_n : A_n$, which we call a *lambda*, such that its i -child (in the arena A) is named $x_i : A_i$, for each $i \in \{1, \dots, n\}$.

To exhibit their correspondence with innocent strategies, we present Böhm trees in the style of Stirling, as a kind of node-labelled trees such that nodes on even (respectively odd) levels are labelled by lambdas (respectively variables). Formally, Böhm trees, with free variables from a given kind environment Δ , are defined as well-founded, finitely-branching, ranked trees whose nodes are labelled according to the following rules.

- (i) Nodes on levels 0, 2, 4, etc., are labelled by lambdas, and nodes on levels 1, 3, 5, etc., are labelled by variables or \perp .
- (ii) If a node has a lambda label $\lambda x_1 : A_1 \dots x_n : A_n$, then it has a unique child which is labelled by either \perp or a variable $y : B$. If the latter, then either $y : B$ occurs in Δ , or it is bound by the lambda label of one of its ancestors (where necessary, the bindee-to-binder relationship is indicated by pointers).
- (iii) If a node has a variable label $y : (B_1, \dots, B_m, o)$, then it has m children; for each $i \in \{1, \dots, m\}$, its i -child has a lambda label of the form $\lambda x_1 : A_1 \dots x_n : A_n$ where $B_i = (A_1, \dots, A_n, o)$.
- (iv) A \perp -labelled node is a leaf.

Take a Böhm tree $\Delta \Vdash U :: A$, and let p be a finite path from the root

$$\ell_1 \cdot v_1 \cdot \ell_2 \cdot v_2 \cdot \dots \cdot \ell_n \cdot v_n$$

where the ℓ_i are lambdas, and the v_j are variables (except v_n may be a variable or \perp). Then p corresponds to a P-view of the arena $(\prod_{i=1}^m C_i) \rightarrow A$ where the ℓ_i and v_j denote O-moves and P-moves respectively. In this reading, each ℓ_{i+1} is explicitly justified by v_i , and each variable v_j is explicitly justified by the ancestor whose lambda binds it.

C. Böhm tree application is strategy composition

We define a *tree* T to be a prefix-closed and order-closed subset of $(\mathbb{N}_+)^*$: formally for every $s \in (\mathbb{N}_+)^*$ and $n \geq 1$, if $s \cdot n \in T$ then $s \in T$, and if $s \cdot (n+1) \in T$ then $s \cdot n \in T$. Let Σ be the set of expressions of the form $\lambda x_1 \dots x_k. y$ or $\lambda x_1 \dots x_k. \perp$ where $k \geq 0$. We can represent a Böhm tree U as a function $\lambda_U : \text{dom}(\lambda_U) \rightarrow \Sigma$ where $\text{dom}(\lambda_U)$ is a tree. Given Böhm trees U and V , we define $U \sqsubseteq V$ just if (i) $\text{dom}(\lambda_U) \subseteq \text{dom}(\lambda_V)$, and (ii) modulo renaming of bound variables in U , for each $\alpha \in \text{dom}(\lambda_U)$, either $\lambda_U(\alpha) = \lambda_V(\alpha)$ or $\lambda_U(\alpha) = \lambda x_1 \dots x_k. \perp$. It is straightforward to see that the set of Böhm trees is a complete partial order with respect to \sqsubseteq .

Given Böhm trees $\Delta \Vdash U :: \kappa \rightarrow \kappa'$ and $\Delta \Vdash V :: \kappa$, we define their application

$$U @ V := \bigsqcup \{ \text{BT}(FG) \mid F, G \in \mathcal{BT}_{\text{fin}}, F \sqsubseteq U, G \sqsubseteq V \}$$

where $\mathcal{BT}_{\text{fin}}$ is the set of finite Böhm trees. Note that $\text{BT}(FG)$ is well-defined because finite Böhm trees are just $\lambda\perp$ -terms (i.e. generated from a distinguished constant \perp) in β -normal η -long form. It is straightforward to prove that for $\lambda\perp$ -terms M and N , if $M \sqsubseteq N$ and $M \rightarrow^* \lambda x_1 \dots x_n. y P_1 \dots P_n$ then

$N \rightarrow^* \lambda x_1 \dots x_n. y P'_1 \dots P'_n$ and $P_i \sqsubseteq P'_i$ for each i . Here $M \sqsubseteq N$ on $\lambda\perp$ -terms is defined by the following rules:

$$\begin{aligned} \perp &\sqsubseteq M \\ M &\sqsubseteq M \\ \lambda x. M &\sqsubseteq \lambda x. M' && \text{if } M \sqsubseteq M' \\ M_1 M_2 &\sqsubseteq M'_1 M'_2 && \text{if } M_1 \sqsubseteq M'_1 \text{ and } M_2 \sqsubseteq M'_2. \end{aligned}$$

For finite $\lambda\perp$ -terms in β -normal η -long form, the two definitions of \sqsubseteq coincide. It follows that the set of trees, $\text{BT}(FG)$, where F and G range respectively over finite approximants of U and V , is directed.

Alternatively, one can define $U @ V$ by appealing to the bijective correspondence between Böhm trees and innocent strategies. Using the notation of Section B-B, we simply define $\Delta \Vdash U @ V :: \kappa'$ to be the composite strategy

$$\prod_{i=1}^n C_i \xrightarrow{\langle \langle U \rangle, \langle V \rangle \rangle} (\kappa \rightarrow \kappa') \times \kappa \xrightarrow{ev_{\kappa, \kappa'}} \kappa'.$$

D. Properties of types

Lemma 49. *Let α and β be intersection types of the same kind and e be an effect.*

- $e \Vdash \alpha \preceq \beta$ if and only if $\alpha \preceq e \otimes \beta$.
- $\alpha \preceq e \otimes (e \Vdash \alpha)$.
- $e \Vdash (e \otimes \alpha) \preceq \alpha$.

Similar propositions hold for type environments.

Proof: The second and the third propositions are easy consequence of the first. We prove the first proposition. Let $\alpha = \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$ and $\beta = \bigwedge_{j \in J} \langle \sigma_j; d_j \rangle$.

Assume $e \Vdash \alpha \preceq \beta$. Then, for every $j \in J$, there exists $k_j \in I$ such that $\tau_{k_j} \preceq \sigma_j$ and $e \Vdash e_{k_j} \succeq d_j$. Hence $e_{k_j} \succeq e \otimes d_j$. Therefore $\alpha \preceq e \otimes \beta$. The converse can be proved similarly. \blacksquare

E. Properties of type-checking games

Lemma 50 (Identity). *P wins for $x : \alpha \models \text{BT}(x) : \beta$ iff $\alpha \preceq \beta$.*

Proof: By induction on the kind κ for x .

(Case $\kappa = o$): In this case, $\text{BT}(x) = x$. Let $\alpha = \bigwedge_{i \in I} \langle q_i; e_i \rangle$ and $\beta = \bigwedge_{j \in J} \langle p_j; d_j \rangle$. Assume that P wins for the game. For each $j \in J$, O can move to the position $x : d_j \Vdash \alpha \models x : p_j$. Since P wins for the game, there exists k_j such that $p_{k_j}(d_j \Vdash \alpha) = \langle p_j; e' \rangle$ with $e' \succeq e$. By definition of $d_j \Vdash \alpha$, we have $q_{k_j} = p_j$ and $d_j \Vdash e_{k_j} \succeq e$, which implies $e_{k_j} \succeq d_j$. Therefore $\alpha \preceq \beta$. The converse can be proved similarly.

(Case $\kappa = \kappa_1 \rightarrow \dots \rightarrow \kappa_n \rightarrow o$): In this case,

$$\text{BT}(x) = \lambda y_1 \dots y_n. x \text{BT}(y_1) \dots \text{BT}(y_n).$$

Let $\alpha = \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$ and $\beta = \bigwedge_{j \in J} \langle \sigma_j; d_j \rangle$. Assume that P wins for the game. Let $j \in J$ and assume $\sigma_j = \delta_1 \rightarrow \dots \rightarrow \delta_n \rightarrow q$. Then O can move to the position

$$x : (d_j \Vdash \alpha), y_1 : \delta_1, \dots, y_n : \delta_n \models x \text{BT}(y_1) \dots \text{BT}(y_n) : q.$$

Since P wins for the game, there exists $k_j \in I$ such that $\mathbf{p}_{k_j}(d_j \backslash \alpha) = \langle \delta'_1 \rightarrow \dots \rightarrow \delta'_n \rightarrow q; e' \rangle$ with $e' \succeq \varepsilon$. Further P must win for $y_i : \delta_i \models \text{BT}(y_i) : \delta'_i$ for every i . By the induction hypothesis, we have $\delta_i \preceq \delta'_i$ for every i and hence

$$\delta'_1 \rightarrow \dots \rightarrow \delta'_n \rightarrow q \preceq \delta_1 \rightarrow \dots \rightarrow \delta_n \rightarrow q.$$

By $d_j \backslash e_{k_j} = e' \succeq \varepsilon$, we have $e_{k_j} \succeq d_j$. Since this holds for every $j \in J$, we have $\alpha \preceq \beta$ as desired. The converse can be proved similarly. ■

Lemma 51 (Subtyping). *If P wins for $\Gamma \models T : q$ and $\Gamma' \preceq \Gamma$, then he wins for $\Gamma' \models T : q$. Similarly, if P wins for $\Gamma \models (U_1, \dots, U_k) : (\alpha_1, \dots, \alpha_k)$ and $\Gamma' \preceq \Gamma$ and $\alpha_i \preceq \alpha'_i$ for every $i \leq k$, then he wins for $\Gamma' \models (U_1, \dots, U_k) : (\alpha'_1, \dots, \alpha'_k)$.*

Proof: For the notational convenience, we write $\Gamma \models (U_i)_i : (\alpha_i)_i$ for $\Gamma \models (U_1, \dots, U_k) : (\alpha_1, \dots, \alpha_k)$. We write $(\Gamma' \models T : q) \mathcal{S} (\Gamma \models T : q)$ just if $\Gamma' \preceq \Gamma$, and $(\Gamma' \models (U_i)_i : (\alpha'_i)_i) \mathcal{S} (\Gamma \models (U_i)_i : (\alpha_i)_i)$ just if $\Gamma' \preceq \Gamma$ and $\alpha'_i \succeq \alpha_i$ for every i . The proposition says that, if P wins for the right-hand-side of \mathcal{S} , then he wins for the left-hand-side.

The key observations are:

- **(P-positions):** $(\Gamma' \models T : q) \mathcal{S} (\Gamma \models T : q)$ and

$$(\Gamma \models T : q) \xrightarrow{\varepsilon} (\Xi \models (U_i)_i : (\alpha_i)_i)$$

implies

$$(\Gamma' \models T : q) \xrightarrow{\varepsilon} (\Xi' \models (U_i)_i : (\alpha'_i)_i)$$

for some $(\Xi' \models (U_i)_i : (\alpha'_i)_i) \mathcal{S} (\Xi \models (U_i)_i : (\alpha_i)_i)$.

- **(O-positions):** $(\Gamma' \models (U_i)_i : (\alpha'_i)_i) \mathcal{S} (\Gamma \models (U_i)_i : (\alpha_i)_i)$ and

$$(\Gamma' \models (U_i)_i : (\alpha'_i)_i) \xrightarrow{e'} (\Xi' \models T : q)$$

implies

$$(\Gamma \models (U_i)_i : (\alpha_i)_i) \xrightarrow{e} (\Xi \models T : q)$$

for some $\Xi \models T : q$ and e such that $(\Xi' \models T : q) \mathcal{S} (\Xi \models T : q)$ and $e' \preceq e$.

It follows from these observations that a strategy for the right-hand-side of \mathcal{S} defines a strategy for the left-hand-side. Furthermore the condition $e' \preceq e$ in the second observation ensures that, if the effects for an infinite play for the right-hand-side is $\langle e_1, e_2, \dots \rangle$, then the effects for the corresponding infinite play for the left-hand-side is $\langle e'_1, e'_2, \dots \rangle$ which satisfies $e'_i \preceq e_i$. Therefore the corresponding strategy for the left-hand-side game is winning if that for the right-hand-side is winning. ■

Lemma 52 (Abstraction). $\Gamma \models \lambda x \lambda y_1 \dots y_n. T : \alpha \rightarrow \tau$ iff $\Gamma, x : \alpha \models \lambda y_1 \dots y_n. T : \tau$.

Proof: Assume

$$\tau = \beta_1 \rightarrow \beta_2 \rightarrow \dots \rightarrow \beta_n \rightarrow q.$$

Then both O-positions have the same unique successor position, namely,

$$\Gamma, x : \alpha, y_1 : \beta_1, \dots, y_n : \beta_n \models T : q.$$

Hence the winners of these positions coincide. ■

The next lemma relies on the stability of Ω .

Lemma 53 (Adjoint). $e \backslash \Gamma \models U : \alpha$ iff $\Gamma \models U : e \circ \alpha$.

Proof: By the stability of Ω , we have

$$\pi \langle e_1, e_2, \dots \rangle \in \Omega \text{ iff } \pi \langle e_0, e_1, e_2, \dots \rangle \in \Omega.$$

Assume $U = \lambda x_1 \dots x_n. T$ and

$$\alpha = \bigwedge_{i \in I} \langle \beta_{i1} \rightarrow \dots \rightarrow \beta_{in} \rightarrow q_i; e_i \rangle.$$

Then, for each $i \in I$, O can move to $e_i \backslash (e \backslash \Gamma), x_1 : \beta_{i1}, \dots, x_n : \beta_{in} \models T : q_i$. Recall that $e_i \backslash (e \backslash \Gamma) = (e \circ e_i) \backslash \Gamma$ and

$$e \circ \alpha = \bigwedge_{i \in I} \langle \beta_{i1} \rightarrow \dots \rightarrow \beta_{in} \rightarrow q_i; (e \circ e_i) \rangle.$$

So the two positions in the statement have the same set of positions that O can reach by the first move. So there exists a bijective correspondence between plays from $e \backslash \Gamma \models U : \alpha$ and plays from $\Gamma \models U : e \circ \alpha$. In particular, a strategy for one position can be viewed as a strategy for the other. It suffices to check that the bijection preserves the winning condition for infinite plays.

If the effects along an infinite play from $e \backslash \Gamma \models U : \alpha$ is

$$\langle e_i, d_1, d_2, \dots \rangle$$

(this means that O choose the i th position as the first move), then the effects along the corresponding play from $\Gamma \models U : e \circ \alpha$ is

$$\langle e \circ e_i, d_1, d_2, \dots \rangle.$$

By the stability of Ω , we have

$$\begin{aligned} \pi \langle e_i, d_1, d_2, \dots \rangle &\in \Omega \\ \text{iff } \pi \langle d_1, d_2, \dots \rangle &\in \Omega \\ \text{iff } \pi \langle e \circ e_i, d_1, d_2, \dots \rangle &\in \Omega \end{aligned}$$

as desired. ■

Similarly, the next lemma follows from the stability of Ω .

Lemma 54 (Positive/negative actions).

- $\Gamma \models U : \alpha$ implies $e \circ \Gamma \models U : e \circ \alpha$.
- $\Gamma \models U : \alpha$ implies $e \backslash \Gamma \models U : e \backslash \alpha$.

Proof: Assume that $\Gamma \models U : \alpha$. Since $e \backslash (e \circ \Gamma) \preceq \Gamma$ by Lemma 49, we have $e \backslash (e \circ \Gamma) \models U : \alpha$ by Lemma 51. By Lemma 53, we have $e \circ \Gamma \models U : e \circ \alpha$.

Assume that $\Gamma \models U : \alpha$. Since $\alpha \preceq e \circ (e \backslash \alpha)$ by Lemma 49, we have $\Gamma \models U : e \circ (e \backslash \alpha)$ by Lemma 51. By Lemma 53, we have $e \backslash \Gamma \models U : e \backslash \alpha$. ■

F. Definitions of extended kinds, types and Böhm trees

An *extended kind* is defined by the following grammar:

$$\widehat{\kappa} ::= \circ \mid \left(\prod_{i \in I} \widehat{\kappa}_i \right) \rightarrow \widehat{\kappa},$$

where $I = \emptyset, \omega$ or $\{1, 2, \dots, k\}$ for some $k \in \omega$. Like intersection types, we only consider extended kinds that refine a simple kind. The refinement relation is given by:

$$\frac{}{\circ :: \circ} \quad \frac{\widehat{\kappa}_i :: \kappa \quad \widehat{\kappa}' :: \kappa'}{\prod_{i \in I} \widehat{\kappa}_i \rightarrow \widehat{\kappa}' :: \kappa \rightarrow \kappa'}$$

The set of extended kinds is formally defined by induction on the structure of the simple kinds. We write $\mathsf{p}_j(\prod_{i \in I} \widehat{\kappa}_i) = \kappa_j$ if $j \in I$.

The set of extended Böhm trees are defined (co-inductively) by the following grammar:

$$\begin{aligned} \widehat{T}, \widehat{S} &::= \perp \mid (\mathsf{p}_i x) \left(\prod_{i \in I_1} \widehat{U}_i^1 \right) \dots \left(\prod_{i \in I_k} \widehat{U}_i^k \right) \\ \widehat{U}, \widehat{V} &::= \lambda x^1 \dots x^k. T \end{aligned}$$

where $k \geq 0$. We consider only well-kinded Böhm trees. An extended kind environment $\widehat{\Delta}$ is a finite sequence of bindings of the form

$$x^1 :: \prod_{i \in I_1} \widehat{\kappa}_i^1, x^2 :: \prod_{i \in I_2} \widehat{\kappa}_i^2, \dots, x^k :: \prod_{i \in I_k} \widehat{\kappa}_i^k.$$

Kinding rules are given by:

$$\begin{aligned} &\frac{}{\widehat{\Delta} \Vdash \perp :: \circ} \\ &\frac{\forall l \leq k. \forall i \in I_l. \widehat{\Delta} \Vdash \widehat{U}_i^l :: \widehat{\kappa}_i^l \quad \mathsf{p}_j(\widehat{\Delta}(x)) = \prod_{i \in I_1} \widehat{\kappa}_i^1 \rightarrow \dots \rightarrow \prod_{i \in I_k} \widehat{\kappa}_i^k \rightarrow \circ}{\widehat{\Delta} \Vdash (\mathsf{p}_j x) \left(\prod_{i \in I_1} \widehat{U}_i^1 \right) \dots \left(\prod_{i \in I_k} \widehat{U}_i^k \right) :: \circ} \\ &\frac{\widehat{\Delta}, x^1 :: \prod_{i \in I_1} \widehat{\kappa}_i^1, \dots, x^k :: \prod_{i \in I_k} \widehat{\kappa}_i^k \Vdash T :: \circ}{\widehat{\Delta} \Vdash \lambda x^1 \dots x^k. T :: \prod_{i \in I_1} \widehat{\kappa}_i^1 \rightarrow \dots \rightarrow \prod_{i \in I_k} \widehat{\kappa}_i^k \rightarrow \circ} \end{aligned}$$

Similar to Böhm trees, well-kinded extended Böhm trees bijectively correspond to innocent strategies of a certain arena. The only difference is the number of moves in the arena: an arena corresponds to a kind must have finite moves, but an arena corresponding to an extended kind may have infinite moves.

The composition of extended Böhm trees is defined through the composition of innocent strategies.

The set of *extended types* is defined by:

$$\begin{aligned} \widehat{\tau} &::= q \mid \widehat{\alpha} \rightarrow \tau \\ \widehat{\alpha} &::= \prod_{i \in I} \langle \widehat{\tau}_i; e_i \rangle. \end{aligned}$$

Notice that this grammar is the same as that of (usual) types, expect that the product is used instead of the intersection. It would be worth emphasising here that extended types do not have intersection. Similar to types, the set of extended types is formally defined by induction on the structure of extended kinds (or equivalently, by induction on the structure of kinds, since extended kinds are defined by induction on kinds).

$$\frac{}{\overline{q} :: \circ} \quad \frac{\widehat{\alpha} :: \prod_{i \in I} \widehat{\kappa}_i \quad \widehat{\tau} :: \widehat{\kappa}'}{\widehat{\alpha} \rightarrow \widehat{\tau} :: \left(\prod_{i \in I} \widehat{\kappa}_i \right) \rightarrow \widehat{\kappa}'} \quad \frac{\forall i \in I. \widehat{\tau}_i :: \widehat{\kappa}_i}{\prod_{i \in I} \langle \widehat{\tau}_i; e_i \rangle :: \prod_{i \in I} \widehat{\kappa}_i}$$

For an extended type $\widehat{\tau}$, we write $|\widehat{\tau}|$ for the unique extended kind $\widehat{\kappa}$ such that $\widehat{\tau} :: \widehat{\kappa}$. Intuitively, $|\cdot|$ forgets ground types and effects. Similarly, $|\widehat{\alpha}| = \prod_{i \in I} \widehat{\kappa}_i$ means that $\widehat{\alpha} :: \prod_{i \in I} \widehat{\kappa}_i$. The projection is define by $\mathsf{p}_j(\prod_{i \in I} \langle \widehat{\tau}_i; e_i \rangle) := \langle \widehat{\tau}_j; e_j \rangle$.

The map \flat from types to extended types is defined by:

$$\begin{aligned} \flat(\alpha \rightarrow \tau) &= \flat(\alpha) \rightarrow \flat(\tau) \\ \flat\left(\bigwedge_{i \in I} \langle \tau_i; e_i \rangle\right) &= \prod_{i \in I} \flat(\tau_i); e_i. \end{aligned}$$

An *extended type environment* $\widehat{\Gamma}$ is a finite sequence of extended intersection type bindings of the form:

$$x^1 : \widehat{\alpha}^1, \dots, x^n : \widehat{\alpha}^n.$$

The type-checking games can naturally be extended to extended Böhm trees. A position of an extended game is of the form

$$\widehat{\Gamma} \models \widehat{T} : q$$

or

$$\widehat{\Gamma} \models \left(\prod_{i \in I_1} \widehat{U}_i^1, \dots, \prod_{i \in I_n} \widehat{U}_i^n \right) : \left(\prod_{i \in I_1} \langle \widehat{\tau}_i^1; e_i^1 \rangle, \dots, \prod_{i \in I_n} \langle \widehat{\tau}_i^n; e_i^n \rangle \right)$$

where $n \geq 0$. The former is a *P-position* and the latter is an *O-position*. We write $\widehat{\Gamma} \models (\prod_{i \in I_k} \widehat{U}_i^k)_k : (\prod_{i \in I_k} \langle \widehat{\tau}_i^k; e_i^k \rangle)_k$ for the latter position. Positions must be kind-respecting: for the former, $\widehat{\Gamma} :: \widehat{\Delta}$ for some $\widehat{\Delta}$, and $\widehat{\Delta} \Vdash \widehat{T} :: \circ$; and for the latter, $\widehat{\Gamma} :: \widehat{\Delta}$ and $\widehat{\Delta} \Vdash U_i^k :: \widehat{\kappa}_i^k$ and $\widehat{\tau}_i^k :: \widehat{\kappa}_i^k$ for every i and k . There are three kinds of edges. The first kind has the form

$$\begin{aligned} &(\widehat{\Gamma} \models (\mathsf{p}_j x) \left(\prod_{i \in I_1} \widehat{U}_i^1 \right) \dots \left(\prod_{i \in I_n} \widehat{U}_i^n \right) : q) \\ &\xrightarrow{\varepsilon} (\widehat{\Gamma} \models \left(\prod_{i \in I_k} \widehat{U}_i^k \right)_k : \left(\prod_{i \in I_k} \langle \widehat{\tau}_i^k; e_i^k \rangle \right)_k) \end{aligned}$$

where

$$\mathsf{p}_j(\widehat{\Gamma}(x)) = \langle \prod_{i \in I_1} \langle \widehat{\tau}_i^1; e_i^1 \rangle \rightarrow \dots \rightarrow \prod_{i \in I_n} \langle \widehat{\tau}_i^n; e_i^n \rangle \rightarrow q; e \rangle$$

with $e \succeq \varepsilon$. Note that the choice of P is completely determined by the extended Böhm tree, unlike the type-checking game in Section III. This is because the extended types do not have intersection. The second kind of edges has the form

$$(\widehat{\Gamma} \models \left(\prod_{i \in I_k} \widehat{U}_i^k \right)_k : \left(\prod_{i \in I_k} \langle \widehat{\tau}_i^k; e_i^k \rangle \right)_k) \xrightarrow{e_i^k} ((e \Vdash \widehat{\Gamma}), \widetilde{x} : \widetilde{\beta} \models \widehat{T} : q)$$

where, for some k and $i \in I_k$,

$$\widehat{U}_i^k = \lambda x_1 \dots x_l. \widehat{T}$$

and $\widehat{\tau}_i^k = \widehat{\beta}_1 \rightarrow \dots \rightarrow \widehat{\beta}_l \rightarrow q$ and $\widetilde{x} : \widetilde{\beta}$ means $x_1 : \widehat{\beta}_1, \dots, x_l : \widehat{\beta}_l$. The third kind of edges is for divergence,

$$(\widehat{\Gamma} \models \perp : q) \xrightarrow{e_j} (\widehat{\Gamma} \models \perp : q).$$

The winner of an infinite play is determined by the effects of the path: P wins just if $\pi \langle e_1, e_2, \dots \rangle \in \Omega$, where $\langle e_1, e_2, \dots \rangle$ is the sequence of effects along the path. If a finite maximal play ends at a P-position (resp. O-position) then O (resp. P) wins.

Specifying the initial position determines a game. We employ the same convention as in the type-checking games, e.g. $\widehat{\Gamma} \models \widehat{T} : q$ means that P wins for the game starting from $\widehat{\Gamma} \models \widehat{T} : q$.

G. Proof of Lemma 10

Before the proof, we define some notations. The relation $\widehat{V} \in U$ and $\widehat{S} \in T$ is co-inductively defined by the following rules:

- If $\lambda x_1 \dots x_n. \widehat{S} \in \lambda x_1 \dots x_n. T$, then
 - 1) $n' = n$, and
 - 2) $\widehat{S} \in T$.
- If $\perp \in T$, then
 - 1) $T = \perp$.
- If $(p_j y) (\prod_{i \in I_1} \widehat{V}_i^1) \dots (\prod_{i \in I_n} \widehat{V}_i^n) \in y' U^1 \dots U^{n'}$, then
 - 1) $y = y'$,
 - 2) $n = n'$, and
 - 3) $\widehat{V}_i^k \in U^k$ for every $k \leq n$ and $i \in I_k$.

We write $(\prod_{i \in I_k} \widehat{V}_i^k) \blacktriangleright (\Gamma \models (U^k)_k : \alpha^k)$ if $\widehat{V}_i^k \in U^k$ and $b(\Gamma) \models V_i^k : b(p_i \alpha^k)$. Similarly $\widehat{S} \blacktriangleright (\Gamma \models T : q)$ if $\widehat{S} \in T$ and $b(\Gamma) \models \widehat{S} : q$.

Note that, if v_0 is a position that P wins and $v_0 v_1 v_2 \dots v_n$ is a play following a winning strategy, then P wins for v_i (for every $i \leq n$). Furthermore a winning strategy for v_i is given by the restriction of the winning strategy for v_0 . This observation is a consequence of stability. We implicitly use this observation in the proof.

First we prove the right-to-left direction. Let U be a Böhm tree and \widehat{V} be an extended Böhm tree. Assume that $\widehat{V} \blacktriangleright (\Gamma \models U : \alpha)$. We define the strategy of $\Gamma \models U : \alpha$ as follows. Formally it is defined by induction on the length of the play.

- For O-positions: Assume $(\prod_{i \in I_k} \widehat{V}_i^k) \blacktriangleright (\Gamma \models (U^k)_k : (\alpha^k)_k)$. A move from the position $\Gamma \models (U^k)_k : (\alpha^k)_k$ is defined by a pair l and j , where $\alpha^l = \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$ and $j \in I$. Suppose that

$$U^l = \lambda x^1 \dots x^n. T$$

$$\tau_j = \beta^1 \rightarrow \dots \rightarrow \beta^n \rightarrow q.$$

Then the move is illustrated as

$$(\Gamma \models (U^k)_k : (\alpha^k)_k) \xrightarrow{e_j} ((e_j \parallel \Gamma), x^1 : \beta^1, \dots, x^n : \beta^n \models T : q).$$

The strategy after this transition is defined as follows. Since $V_j^k \in U^k$, we have

$$V_j^k = \lambda x^1 \dots x^n. \widehat{S}$$

with $\widehat{S} \in T$. By the definition of $b(\Gamma) \models (\prod_{i \in I_k} \widehat{V}_i^k) : (b(\alpha^k))_k$, we have

$$(b(\Gamma) \models (\prod_{i \in I_k} \widehat{V}_i^k) : (b(\alpha^k))_k) \xrightarrow{e_j} ((e_j \parallel b(\Gamma)), x^1 : b(\beta^1), \dots, x^n : b(\beta^n) \models \widehat{S} : q).$$

By the assumption, the right-hand-side position is P-winning. Hence

$$\widehat{S} \blacktriangleright ((e_j \parallel \Gamma), x^1 : \beta^1, \dots, x^n : \beta^n \models T : q).$$

The strategy for the following plays is determined by \widehat{S} .

- For P-positions: Assume $\widehat{S} \blacktriangleright (\Gamma \models T : q)$. If $\widehat{S} = \perp$, then we have $T = \perp$, in which case P has a unique strategy. Otherwise we have

$$\widehat{S} = (p_j x) (\prod_{i \in I_1} \widehat{V}_i^1) \dots (\prod_{i \in I_n} \widehat{V}_i^n)$$

$$T = x U^1 \dots U^n.$$

Because $b(\Gamma) \models \widehat{S} : q$, we have

$$p_j(\Gamma(x)) = \langle \beta^1 \rightarrow \dots \rightarrow \beta^n \rightarrow q; e \rangle$$

for some $e \succeq \varepsilon$ and

$$b(\Gamma) \models (\prod_{i \in I_k} \widehat{V}_i^k) : (b(\beta^k))_k.$$

So P can move to

$$\Gamma \models T : q \xrightarrow{e} \Gamma \models (U^k)_k : (\beta^k)_k.$$

Since $\widehat{S} \in T$, we have $\widehat{V}_i^k \in U^k$ for every k and $i \in I_k$, and hence

$$(\prod_{i \in I_k} \widehat{V}_i^k) \blacktriangleright (\Gamma \models (U^k)_k : (\beta^k)_k).$$

The strategy following the P-move is defined by $(\prod_{i \in I_k} \widehat{V}_i^k)_k$.

It is easy to see that P does not get stuck since $b(\Gamma) \models \widehat{S} : q$ in every P-positions. The effects for an infinite play of $\Gamma \models T : q$ is the same as those of $b(\Gamma) \models \widehat{S} : q$. Since the latter is P-winning, we know that every infinite play following the strategy defined above is P-winning.

We prove the left-to-right direction. For a pair of a position and a winning strategy, we define an extended Böhm tree (or a tuple of extended Böhm trees) as follows.

- O-positions: Assume a winning strategy for $\Gamma \models (U^k)_k : (\bigwedge_{i \in I_k} \langle \tau_i^k; e_i^k \rangle)_k$. For each k and $i \in I_k$, we define an extended Böhm tree \widehat{V}_i^k as follows. Suppose

$$U^k = \lambda x^1 \dots x^n. T$$

and

$$\tau_i^k = \beta^1 \rightarrow \dots \rightarrow \beta^n \rightarrow q_i^k,$$

then we have

$$(\Gamma \models (U^k)_k : (\alpha^k)_k) \xrightarrow{e_i^k} ((e_i^k \parallel \Gamma), x^1 : \beta^1, \dots, x^n : \beta^n \models T : q_i^k).$$

Now we have a winning strategy for the right-hand-side position. Set \widehat{S}_i^k be the extended Böhm tree corresponding the winning strategy. Then we have $((e_i^k \parallel \Gamma), x^1 : \beta^1, \dots, x^n : \beta^n) \models \widehat{S}_i^k : q_i^k$. Let $\widehat{V}_i^k = \lambda x^1 \dots x^n. \widehat{S}_i^k$. Then we have $(\prod_{i \in I_k} \widehat{V}_i^k)_k \blacktriangleright (\Gamma \models (U^k)_k : (\alpha^k)_k)$ as desired.

- P-positions: For the game $\Gamma \models \perp : q$, the corresponding Böhm tree is \perp . Assume a winning strategy of the position $\Gamma \models x U^1 \dots U^n : q$. The winning condition determines a move

$$\Gamma \models x U^1 \dots U^n : q \xrightarrow{\varepsilon} \Gamma \models (U^k)_k : (\beta^k)_k.$$

Then, for some j , we have $p_j(\Gamma(x)) = \langle \beta^1 \rightarrow \dots \rightarrow \beta^n \rightarrow q; e \rangle$ with $q \succeq \varepsilon$. Since P wins for the right-hand-side position, we have

$$(\prod_{i \in I} \widehat{V}_i^k)_k \blacktriangleright (\Gamma \models (U^k)_k : (\beta^k)_k).$$

Then we define

$$\widehat{S} = (p_j x) (\prod_{i \in I_1} \widehat{V}_i^1) \dots (\prod_{i \in I_n} \widehat{V}_i^n).$$

Then $\widehat{S} \blacktriangleright (\Gamma \models T : q)$.

APPENDIX C

SUPPLEMENTARY MATERIALS FOR SECTION IV

A. Definition of λY -calculus and Böhm trees

This subsection gives definitions of λY -calculus and of the Böhm tree of a λY -term, which are fairly standard.

1) λY -calculus: The syntax of *terms* is given by:

$$M, N ::= x \mid M M \mid \lambda x. M \mid \mathbf{Y}_\kappa.$$

We allow tacit renaming of bound variables and identify α -equivalent terms. The kind κ of \mathbf{Y}_κ is often omitted.

The calculus is simply typed. A *kind environment* is a finite sequence of bindings of the form $x :: \kappa$. A kind environment can have at most one type binding for each variable. We use $\Delta, \Delta', \Delta_i$, etc., to denote kind environments. A *kind judgement* is of the form $\Delta \vdash M :: \kappa$. The kinding rules are given by:

$$\begin{array}{c} \hline \Delta, x :: \kappa, \Delta' \vdash x :: \kappa \\ \hline \Delta \vdash M :: \kappa' \rightarrow \kappa \quad \Delta \vdash M' :: \kappa' \\ \hline \Delta \vdash M M' :: \kappa \\ \hline \Delta, x :: \kappa \vdash M :: \kappa' \\ \hline \Delta \vdash \lambda x. M :: \kappa \rightarrow \kappa' \\ \hline \Delta \vdash \mathbf{Y}_\kappa :: (\kappa \rightarrow \kappa) \rightarrow \kappa \end{array}$$

Hereafter we assume that terms are equipped with their kind derivations. Therefore all terms (including variables) are assumed to have their kinds.

The calculus has three rewriting rules: β -reduction, η -expansion and expanding the fixed-point combinator. Any subterm can be rewritten without depending on its context. Formally the (small step) rewriting relation $M \longrightarrow M'$ is defined as the smallest relation that satisfies the following rules:

- $(\lambda x. M) N \longrightarrow M[N/x]$, where $M[N/x]$ is the standard capture-avoiding substitution.
- $M \longrightarrow \lambda x. M x$ if M has a function kind and x is a fresh variable.
- $\mathbf{Y} M \longrightarrow M (\mathbf{Y} M)$.
- $M N \longrightarrow M' N$ if $M \longrightarrow M'$.
- $M N \longrightarrow M N'$ if $N \longrightarrow N'$.
- $\lambda x. M \longrightarrow \lambda x. M'$ if $M \longrightarrow M'$.

We write \longrightarrow^* for the reflexive and transitive closure of \longrightarrow .

2) *Böhm trees and value trees*: A term M with $\Delta \vdash M :: \kappa$ is associated with a Böhm tree $U = \text{BT}(M)$ with $\Delta \Vdash U :: \kappa$ as below. Term M is *head-normalisable* just if $M \longrightarrow^* \lambda x_1 \dots x_k. y N_1 \dots N_l$. The map $\text{BT}(-)$ is defined by the following rules.

- If M is not head-normalisable then $\text{BT}(M) = \lambda x_1 \dots x_k. \perp$ where k is the arity of κ , i.e. $\kappa = \kappa_1 \rightarrow \dots \rightarrow \kappa_k \rightarrow \circ$.
- Assume $M \longrightarrow^* \lambda x_1 \dots x_k. y N_1 \dots N_l$. We can assume without loss of generality that $y :: \kappa_1 \rightarrow \dots \rightarrow \kappa_l \rightarrow \circ$ (otherwise apply η -expansion). Then $\text{BT}(M) = \lambda x_1 \dots x_k. y U_1 \dots U_l$ where each $U_i = \text{BT}(N_i)$.

This is an extension of the notion of *value trees* of recursion schemes. From this viewpoint, a tree constructor (or terminal symbol) of arity k is just a free variable of kind

$\overbrace{\circ \rightarrow \dots \rightarrow \circ}^k \rightarrow \circ$, which we shall abbreviate as $\circ^k \rightarrow \circ$; thus a ranked alphabet is just an order-1 kind environment.³ Given an order-1 kind environment $\Delta = (\mathbf{a}_1 :: \circ^{ar_1} \rightarrow \circ, \dots, \mathbf{a}_k :: \circ^{ar_k} \rightarrow \circ)$, a possibly-infinite Δ_\perp -labelled tree (i.e. ranked tree with node labels taken from the set Δ_\perp consisting of \perp and the symbols from Δ) is just a Böhm tree T with $\Delta \Vdash T :: \circ$. For the purpose of generating possibly-infinite Δ_\perp -labelled trees, deterministic recursion schemes are equivalent to order-2 terms, and the value tree of such a term is just its Böhm tree.

B. Basic properties of the type system

The subtyping rule for prime effect judgements, which is omitted in the main text, is given by:

$$\frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash M : \langle \tau; e \rangle}{\Gamma' \vdash M : \langle \tau; e \rangle}$$

³The *order* of kind is defined by: $\text{order}(\circ) = 0$ and $\text{order}(\kappa_1 \rightarrow \dots \rightarrow \kappa_k \rightarrow \circ) = 1 + \max\{\text{order}(\kappa_i) \mid 1 \leq i \leq k\}$. The order of kind environment is given by $\text{order}(\Delta) = \max\{\text{order}(\kappa) \mid x :: \kappa \in \Delta\}$.

Lemma 55. *The following rules are admissible:*

$$\frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash M : \langle \tau; e \rangle \quad \langle \tau; e \rangle \preceq \langle \tau'; e' \rangle}{\Gamma' \vdash M : \langle \tau'; e' \rangle}$$

and

$$\frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash M : \alpha \quad \alpha \preceq \alpha'}{\Gamma' \vdash M : \alpha}.$$

Proof: Easy. ■

The next lemma is used in the (sketch of) the proof of Theorem 18.

Lemma 56 (Inversion).

- 1) If $\Gamma \vdash x : \tau$, then there exists an index k such that $p_k(\Gamma(x)) = \langle \sigma; e \rangle$ and $\sigma \preceq \tau$ and $e \succeq \varepsilon$.
- 2) If $\Gamma \vdash M_1 M_2 : \tau$, then $\Gamma \vdash M_1 : \alpha \rightarrow \tau$ and $\Gamma \vdash M_2 : \alpha$ for some α .
- 3) If $\Gamma \vdash \lambda x.M : \alpha \rightarrow \tau$, then $\Gamma, x : \alpha \vdash M : \tau$.
- 4) If $\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$, then $\Gamma \vdash M : \langle \tau_i; e_i \rangle$ for every i .
- 5) If $\Gamma \vdash M : \langle \tau; e \rangle$, then $e \Vdash \Gamma \vdash M : \tau$.

Proof: By induction on the structure of the derivation.

(Proof of 1) If the last rule used to derive $\Gamma \vdash x : \tau$ is the rule for variables, then the claim is trivial. Assume that the last rule is the subtyping rule. Then we have $\Gamma' \vdash x : \tau'$ for some $\Gamma \preceq \Gamma'$ and $\tau' \preceq \tau$. By the induction hypothesis, there exists k' such that $p_{k'}(\Gamma'(x)) = \langle \sigma'; e' \rangle$ and $\sigma' \preceq \tau'$ and $e' \succeq \varepsilon$. We have $\Gamma(x) \preceq \Gamma'(x)$ by the subtyping rule for type environments, and hence there exists k such that $p_k(\Gamma(x)) \preceq p_{k'}(\Gamma'(x))$. Let $p_k(\Gamma(x)) = \langle \sigma; e \rangle$. Then we have $\sigma \preceq \sigma'$ and $e \succeq e'$, hence $\sigma \preceq \tau$ and $e \succeq \varepsilon$ as desired.

(Proof of 2) If the last rule used to derive $\Gamma \vdash M_1 M_2 : \tau$ is the rule for applications, then the claim is trivial. Assume that the last rule is the subtyping rule. Then we have $\Gamma' \vdash M_1 M_2 : \tau'$ for some $\Gamma \preceq \Gamma'$ and $\tau' \preceq \tau$. By the induction hypothesis, we have $\Gamma' \vdash M_1 : \alpha \rightarrow \tau'$ and $\Gamma' \vdash M_2 : \alpha$ for some α . By applying subtyping rule and Lemma 55, we have $\Gamma \vdash M_1 : \alpha \rightarrow \tau$ and $\Gamma \vdash M_2 : \alpha$ as required.

(Proof of 3) If the last rule used to derive $\Gamma \vdash \lambda x.M : \alpha \rightarrow \tau$ is the rule for abstraction, then the claim is trivial. If the last rule is the subtyping rule, then we can prove the claim by a way similar to the above case.

(Proof of 4) If the last rule used to derive $\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$ is the rule for intersection, the claim is trivial. Notice that the other rule cannot conclude the judgement.

(Proof of 5) If the last rule used to derive $\Gamma \vdash M : \langle \tau; e \rangle$ is the rule for positive actions, the claim is trivial. Assume that the last rule is the subtyping rule. Then we have $\Gamma' \vdash M : \langle \tau'; e' \rangle$ and $\Gamma \preceq \Gamma'$. By the induction hypothesis, we have $e \Vdash \Gamma' \vdash M : \tau'$. Since $e \Vdash \Gamma \preceq e \Vdash \Gamma'$, we have $e \Vdash \Gamma \vdash M : \tau$ as required. ■

The inversion for $\Gamma \vdash x : \tau$ can be described as $\Gamma(x) \preceq \langle \tau; \varepsilon \rangle$.

C. Proof of Lemma 15

Assume that $\kappa = \kappa_1 \rightarrow \dots \rightarrow \kappa_n \rightarrow \circ$ and let $U = \text{BT}(\mathbf{Y}_{(\kappa \rightarrow \kappa) \rightarrow \kappa})$. Then U is given by

$$U = \lambda f.V$$

$$V = \lambda x_1 \dots x_n. f V \text{BT}(x_1) \dots \text{BT}(x_n),$$

where U is a Böhm tree of kind $(\kappa \rightarrow \kappa) \rightarrow \kappa$ and V is a Böhm tree of kind κ (with the free variable $f : \kappa \rightarrow \kappa$). Recall that $x : \alpha \models \text{BT}(x) : \beta$ if and only if $\alpha \preceq \beta$ (Lemma 50). The game $\models U : \alpha_0 \rightarrow \tau_0$, which is equivalent to $f : \alpha_0 \models V : \tau_0$, has the following edges. An edge from a P-position is

$$\begin{aligned} & (f : \alpha, x_1 : \beta_1, \dots, x_n : \beta_n \models f V \text{BT}(x_1) \dots \text{BT}(x_n) : q) \\ & \xrightarrow{\varepsilon} \\ & (f : \alpha \models (V, \text{BT}(x_1), \dots, \text{BT}(x_n)) : (\gamma, \beta'_1, \dots, \beta'_n)) \end{aligned}$$

whenever $p_i(\alpha) = \langle \gamma \rightarrow \beta'_1 \rightarrow \dots \rightarrow \beta'_n \rightarrow q; e \rangle$ with $e \succeq \varepsilon$. To win this position, P must have a winning strategy for $x_k : \beta_k \models \text{BT}(x_k) : \beta'_k$ for each k . Hence $\beta_k \preceq \beta'_k$ is required to win. This implies that

$$\beta'_1 \rightarrow \dots \rightarrow \beta'_n \rightarrow q \preceq \beta_1 \rightarrow \dots \rightarrow \beta_n \rightarrow q.$$

This edge corresponds to the edge in $G(\alpha_0 \rightarrow \tau_0)$

$$\begin{aligned} & \alpha \models_P \beta_1 \rightarrow \dots \rightarrow \beta_n \rightarrow q \\ & \xrightarrow{\varepsilon} \\ & \alpha \models_O \gamma. \end{aligned}$$

There is an obvious correspondence between edges from O-positions. Therefore P wins for $\models f : \alpha_0 \models V : \tau_0$ if and only if he wins for $G(\alpha_0 \rightarrow \tau_0)$.

D. Proof of Theorem 18

We prove:

- $\Gamma \vdash M : \tau$ iff $\Gamma \models \text{BT}(M) : \tau$, and
- $\Gamma \vdash M : \alpha$ iff $\Gamma \models \text{BT}(M) : \alpha$.

Recall that $\Gamma \models \text{BT}(M) : \tau$ is an abbreviation for $\Gamma \models \text{BT}(M) : \langle \tau; \varepsilon \rangle$.

Assume that $\Gamma \vdash M : \tau$. We prove $\Gamma \models \text{BT}(M) : \tau$ by induction on the structure of the derivation. We do case analysis on the last rule used in the derivation.

- Variable rule: It follows from Lemma 50 and Lemma 51.
- Abstraction rule: Then we have $M = \lambda x_1.M_0$ and

$$\tau = \alpha_1 \rightarrow \alpha_2 \rightarrow \dots \rightarrow \alpha_n \rightarrow q$$

and

$$\Gamma, x_1 : \alpha_1 \vdash M_0 : \alpha_2 \rightarrow \dots \rightarrow \alpha_n \rightarrow q.$$

Assume that $\text{BT}(M_0) = \lambda x_2 \dots x_n.T$. Then $\text{BT}(M) = \lambda x_1 \dots x_n.T$. From the induction hypothesis, we have $\Gamma, x_1 : \alpha_1 \models \lambda x_2 \dots x_n.T : \alpha_2 \rightarrow \dots \rightarrow \alpha_n \rightarrow q$. By Lemma 52, we have $\Gamma \models \lambda x_1 \lambda x_2 \dots x_n.T : \alpha_1 \rightarrow \alpha_2 \rightarrow \dots \rightarrow \alpha_n \rightarrow q$ as desired.

- Application rule: Then we have $M = M_0 M_1$, $\Gamma \vdash M_0 : \alpha \rightarrow \tau$ and $\Gamma \vdash M_1 : \alpha$. By the induction hypothesis, we have $\Gamma \models \text{BT}(M_0) : \alpha \rightarrow \tau$ and $\Gamma \models \text{BT}(M_1) : \alpha$. By

Theorem 8, we have $\Gamma \models \text{BT}(M_0) @ \text{BT}(M_1) : \tau$. By Proposition 12, we have $\Gamma \models \text{BT}(M_0 M_1) : \tau$ as required.

- Subtyping rule: A consequence of Lemma 51 and the induction hypothesis.
- Positive action rule: A consequence of Lemma 54.
- Intersection rule: Easy.
- Recursion: Trivial from the typing rule.

We prove the converse. Assume $\Gamma \models \text{BT}(M) : \tau$ or $\Gamma \models \text{BT}(M) : \alpha$. We prove the corresponding type judgement by induction on M . Here the latter style judgements are considered “bigger” than the former style if the subjects are the same. First consider the former style judgements.

- Case $M = x$: It follows from Lemma 50 and the subtyping rule.
- Case $M = M_0 M_1$: By Proposition 12, $\text{BT}(M_0 M_1) = \text{BT}(M_0) @ \text{BT}(M_1)$. So we have $\Gamma \models \text{BT}(M_0) @ \text{BT}(M_1) : \tau$. By Theorem 8, we have $\Gamma \models \text{BT}(M_0) : \alpha \rightarrow \tau$ and $\Gamma \models \text{BT}(M_1) : \alpha$ for some α . By the induction hypothesis, we have $\Gamma \vdash M_0 : \alpha \rightarrow \tau$ and $\Gamma \vdash M_1 : \alpha$. By the application rule, we have $\Gamma \vdash M_0 M_1 : \tau$.
- Case $M = \lambda x.M_0$: Assume $\text{BT}(M_0) = \lambda y_1 \dots y_n.T$ and $\tau = \alpha \rightarrow \sigma$. Then $\text{BT}(M) = \lambda x y_1 \dots y_n.T$. By Lemma 52 and the assumption, we know that P wins for $\Gamma, x : \alpha \models \lambda y_1 \dots y_n.T : \sigma$. By the induction hypothesis, we have $\Gamma, x : \alpha \vdash M_0 : \sigma$. By applying the application rule, we have $\Gamma \vdash \lambda x.M_0 : \sigma$.
- Case $M = \mathbf{Y}$: Trivial from the typing rule.

Assume that $\Gamma \models \text{BT}(M) : \alpha$. We prove that $\Gamma \vdash M : \alpha$. Note that we can use the induction hypothesis for judgements of the form $\Gamma \models \text{BT}(M) : \tau$. Let $\alpha = \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$. Since $\Gamma \models \text{BT}(M) : \alpha$ is an O-position, we know that P wins for $e_i \parallel \Gamma \models \text{BT}(M) : \tau_i$ for all $i \in I$. By the induction hypothesis, we have $e_i \parallel \Gamma \vdash M : \tau_i$ for every $i \in I$. By applying the positive action rule, we have $e_i \otimes (e_i \parallel \Gamma) \vdash M : \langle \tau_i; e_i \rangle$ for every i . By Lemma 49, we have $\Gamma \preceq e_i \otimes (e_i \parallel \Gamma)$. So by the subtyping rule, we have $\Gamma \vdash M : \langle \tau_i; e_i \rangle$ for all $i \in I$. By applying the intersection rule, we have $\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle$.

E. Rules for term representation and proof of Theorem 20

This subsection defines the representation D of a derivation, where D is a term of λY -calculus extended to have products. The syntax of the extended calculus is given by:

$$D, E, F ::= \mathbf{p}_i x \mid \lambda x.D \mid D \left(\prod_{i \in I} D_i \right) \mid \mathbf{Y}.$$

Here I is a *finite* set. The construction $\prod_{i \in \{1\}} D_i$ is simply written as D_1 . Similarly, if the kind of x is $\prod_{i \in \{1\}} \hat{\kappa}_i$ (i.e. a 1-tuple), the projection $\mathbf{p}_1 x$ is simply written as x . We also write $\prod_{i \in I} \mathbf{p}_i x$ as x .

Remark 57. Note that the extended λY -calculus (with *finite* products) is just a reformulation of λY -calculus. The following

table (informally) shows the correspondence.

Extended term	Term
λx	$\longleftrightarrow \lambda x_1 x_2 \dots x_n$
$\mathbf{p}_i x$	$\longleftrightarrow x_i$
$D \left(\prod_{i \in I} E_i \right)$	$\longleftrightarrow D E_1 E_2 \dots E_n$ (where $I = \{1, 2, \dots, n\}$)

Henceforth, by λY -representation, we means a representation by an extended λY -term.

First we give the rules of the representations of subtyping judgements.

$$\overline{\lambda x.x \triangleright (q \preceq q)}$$

$$\frac{D_0 \triangleright (\tau \preceq \tau') \quad (\prod_{i \in I} E_i) \triangleright (\bigwedge_{j \in J} \langle \delta_j; d_j \rangle \preceq \bigwedge_{i \in I} \langle \sigma_i; e_i \rangle)}{(\lambda f. \lambda x. D(f(\prod_{i \in I} (E_i(\prod_{j \in J} \mathbf{p}_j x)))) \triangleright (\bigwedge_{i \in I} \langle \sigma_i; e_i \rangle \rightarrow \tau \preceq \bigwedge_{j \in J} \langle \delta_j; d_j \rangle \rightarrow \tau')}$$

$$\exists \varphi : J \rightarrow I. \forall j \in J.$$

$$\frac{D_j \triangleright (\tau_{\varphi(j)} \preceq \sigma_j) \text{ and } e_i \succeq d_j}{(\prod_{j \in J} D_j(\mathbf{p}_{\varphi(j)} x)) \triangleright \bigwedge_{i \in I} \langle \tau_i; e_i \rangle \preceq \bigwedge_{j \in J} \langle \sigma_j; d_j \rangle}$$

Lemma 58. *If $D \triangleright (\tau \preceq \sigma)$, then $\text{BT}(D) \blacktriangleright (\models \text{BT}(\lambda x.x) : \langle \tau; \varepsilon \rangle \rightarrow \sigma)$. If $\prod_{i \in I} D_i \triangleright (\alpha \preceq \bigwedge_{i \in I} \langle \tau_i; e_i \rangle)$, then $\text{BT}(D_i) \blacktriangleright (\models \text{BT}(\lambda x.x) : (e \parallel \alpha) \rightarrow \tau)$.*

Proof: By easy induction on the structure of the kind of types. Note that to prove $\models \text{BT}(D) : \langle \tau; \varepsilon \rangle \rightarrow \sigma$, we can appeal to the type system by Theorem 18, that means, it suffices to prove that $\vdash D : \langle \tau; \varepsilon \rangle \rightarrow \sigma$. ■

The representation of a type derivation is defined by the following rules.

$$\frac{\langle \tau; e \rangle = \mathbf{p}_i(\Gamma(x)) \text{ for some } i \text{ and } e \succeq \varepsilon}{\mathbf{p}_i x \triangleright (\Gamma \vdash x : \tau)}$$

$$\frac{D \triangleright (\Gamma, x : \alpha \vdash M : \tau)}{\lambda x.D \triangleright (\Gamma \vdash \lambda x.M : \alpha \rightarrow \tau)}$$

$$\frac{D \triangleright (\Gamma \vdash M : \alpha \rightarrow \tau) \quad \prod_{i \in I} E_i \triangleright (\Gamma \vdash N : \alpha)}{D \left(\prod_{i \in I} E_i \right) \triangleright (\Gamma \vdash M N : \tau)}$$

$$\frac{\begin{array}{l} \Gamma = x_1 : \alpha_1, \dots, x_n : \alpha_n \\ \Gamma' = x_1 : \alpha'_1, \dots, x_n : \alpha'_n \\ \prod_{j \in J_i} E_i^j \triangleright (\alpha'_i \preceq \alpha_i) \text{ for each } i \leq n \\ D \triangleright (\Gamma \vdash M : \tau) \quad F \triangleright (\tau \preceq \tau') \end{array}}{F \left((\lambda x_1 \dots \lambda x_n. D) \left(\prod_{j \in J_1} E_1^j x_1 \right) \dots \left(\prod_{j \in J_n} E_n^j x_n \right) \right) \triangleright (\Gamma' \vdash M : \tau')}$$

$$\frac{\begin{array}{l} \Gamma = x_1 : \alpha_1, \dots, x_n : \alpha_n \\ \Gamma' = x_1 : \alpha'_1, \dots, x_n : \alpha'_n \\ \prod_{j \in J_i} E_i^j \triangleright (\alpha'_i \preceq \alpha_i) \text{ for each } i \leq n \\ D \triangleright (\Gamma \vdash M : \langle \tau; e \rangle) \end{array}}{(\lambda x_1 \dots \lambda x_n. D) \left(\prod_{j \in J_1} E_1^j x_1 \right) \dots \left(\prod_{j \in J_n} E_n^j x_n \right) \triangleright (\Gamma' \vdash M : \langle \tau; e \rangle)}$$

$$\begin{array}{c}
\frac{D \triangleright (\Gamma \vdash M : \tau)}{D \triangleright (e \otimes \Gamma \vdash M : \langle \tau; e \rangle)} \\
\\
\frac{\forall i \in I. D_i \triangleright (\Gamma \vdash M : \langle \tau_i; e_i \rangle)}{\prod_{i \in I} D_i \triangleright (\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle)} \\
\\
\frac{\text{BT}(D) \blacktriangleright (\emptyset \models \text{BT}(\mathbf{Y}) : \tau)}{D \triangleright (\Gamma \vdash \mathbf{Y} : \tau)}
\end{array}$$

The following lemma is useful to prove Theorem 20.

Lemma 59. $\widehat{V}_1 \in U_1$ and $\widehat{V}_2 \in U_2$ implies $(\widehat{V}_1 @ \widehat{V}_2) \in (U_1 @ U_2)$.

Proof: It follows from the game-semantic counterpart of this result. That is:

$$\mathfrak{s}_1 \subset \mathfrak{t}_1 \text{ and } \mathfrak{s}_2 \subset \mathfrak{t}_2 \text{ implies } (\mathfrak{s}_1; \mathfrak{s}_2) \subset (\mathfrak{t}_1; \mathfrak{t}_2).$$

It is also possible to prove this lemma syntactically. ■

Lemma 60.

- If $D \triangleright (\Gamma \vdash M : \tau)$, then $\text{BT}(D) \blacktriangleright (\Gamma \models \text{BT}(M) : \tau)$.
- If $\prod_{i \in I} D_i \triangleright (\Gamma \vdash M : \alpha)$, then $\prod_{i \in I} \text{BT}(D_i) \blacktriangleright (\Gamma \vdash \text{BT}(M) : \alpha)$.

Proof: By (mutual) induction on the derivation of $D \triangleright (\Gamma \vdash M : \tau)$ and $\prod_{i \in I} D_i \triangleright (\Gamma \vdash M : \alpha)$.

(Variable rule) Trivial.

(Abstraction rule) By the induction hypothesis, we have $\text{BT}(D) \blacktriangleright (\Gamma, x : \alpha \models \text{BT}(M) : \tau)$. Hence $\text{BT}(D) \in \text{BT}(M)$ and $b(\Gamma), x : b(\alpha) \models \text{BT}(D) : b(\tau)$. Hence $\text{BT}(\lambda x.D) = \lambda x.\text{BT}(D) \in \lambda x.\text{BT}(M) = \text{BT}(\lambda x.M)$ and $b(\Gamma) \models \lambda x.\text{BT}(D) : b(\alpha) \rightarrow b(\tau)$ as desired.

(Subtyping for prime types) By the induction hypothesis, we have $\text{BT}(D) \blacktriangleright (\Gamma \models \text{BT}(M) : \tau)$, and hence $\text{BT}(D) \in \text{BT}(M)$ and $b(\Gamma) \models \text{BT}(D) : b(\tau)$. By Lemma 58, we have $\text{BT}(E_i^j) \in \text{BT}(\lambda z.z)$ for every i and j . Hence by Lemma 59,

$$\text{BT}(E_i^j x_i) \in \text{BT}((\lambda z.z)x_i) = \text{BT}(x_i).$$

Since $\text{BT}(\lambda x_1 \dots x_n.D) \in \text{BT}(\lambda x_1 \dots x_n.M)$, we have

$$\begin{aligned}
& \text{BT}((\lambda x_1 \dots \lambda x_n.D) (\prod_{j \in J_1} E_1^j x_1) \dots (\prod_{j \in J_n} E_n^j x_n)) \\
& \in \text{BT}((\lambda x_1 \dots \lambda x_n.M) x_1 \dots x_n) \\
& = \text{BT}(M)
\end{aligned}$$

by Lemma 59. Because $\text{BT}(F) \in \text{BT}(\lambda z.z)$, we obtain the expected \in -judgement. To prove that the strategy given by the term is winning, we can appeal to the type system.

(Subtyping for intersection types) Similar to the above case.

(Positive action) By the induction hypothesis, we have $\text{BT}(D) \blacktriangleright (\Gamma \models \text{BT}(M) : \tau)$. Hence $\text{BT}(D) \in \text{BT}(M)$ and $b(\Gamma) \models \text{BT}(D) : b(\tau)$. It suffices to prove that $e \otimes b(\Gamma) \models \text{BT}(D) : \langle b(\tau); e \rangle$. It follows from the proof-relevant version of Lemma 54, which is easy to prove.

(Intersection rule) Trivial.

(Y) Trivial. ■

Theorem 20 is a consequence of the above lemma.

F. Proof of Lemma 21

Fix a kind κ . Since the winning condition is assume to be finite, there exists finitely many equivalence classes of (prime) types of kind κ . We assume a set $K = \{\tau_1, \tau_2, \dots, \tau_n\}$ of representatives. With out loss of generality, we can assume that τ_i is finite for every i .

Definition 61 (Alternative game for \mathbf{Y}). Given a prime type τ of kind $(\kappa \rightarrow \kappa) \rightarrow \kappa$, we define the game $H(\tau)$ as follows. Here types are considered modulo the equivalence \approx induced by the subtyping relation. A P-position has the form $\alpha \models_P \sigma$, where $\alpha :: \kappa \rightarrow \kappa$ and $\sigma :: \kappa$, and an O-position has the form $\alpha \models_O \beta$, where $\alpha :: \kappa \rightarrow \kappa$ and $\beta :: \kappa$. Recall that σ, τ (resp. α, β) range over prime (resp. intersection) types. Edges are defined by:

$$(\alpha \models_P \tau) \xrightarrow{\varepsilon} (\alpha \models_O \beta),$$

whenever $p_k(\alpha) = \langle \beta \rightarrow \tau; e \rangle$ with $e \succeq \varepsilon$ for some k , and

$$(\alpha \models_O \beta) \xrightarrow{e} (e \parallel \alpha \models_P \tau)$$

whenever $p_k(\beta) = \langle \tau; e \rangle$ for some k . The initial position of $H(\alpha \rightarrow \tau)$ is $\alpha \models_P \tau$. The winner of an infinite play is determined by the (infinite sequence of) effects along the play.

The difference between $G(-)$ and $H(-)$ is the moves from P-positions. The game $G(-)$ requires that $p_k(\alpha) = \langle \beta \rightarrow \tau'; e \rangle$ with $e \succeq \varepsilon$ and $\tau' \preceq \tau$ for some k , but the game $H(-)$ further requires that $\tau = \tau'$. Hence P-strategies for $H(-)$ is more restrictive than those for $G(-)$.

If P wins for $H(\tau)$, then he wins for $G(\tau)$ by definition. The converse holds in the following sense.

Lemma 62. P wins for $G(\tau)$ if and only if P wins for $H(\tau')$ for some $\tau' \preceq \tau$.

Proof: We first define τ' . Let

$$\tau = (\bigwedge_{i \in I} \langle \alpha_i \rightarrow \sigma_i; e_i \rangle) \rightarrow \tau_0.$$

We can assume without loss of generality that $\sigma_i \in K$ for every i . Then τ' is defined by:

$$\tau' = (\bigwedge \{ \langle \alpha_i \rightarrow \sigma'; e_i \rangle \mid i \in I \text{ and } \sigma \preceq \sigma' \}) \rightarrow \tau_0.$$

(Note that the intersection of a set of types is uniquely determined modulo the equivalence of the subtyping relation.) Then $\tau' \preceq \tau$.

Notice that a P-position of $G(\tau)$ that is reachable from the initial position has the form

$$\bigwedge_{i \in I} \langle \alpha_i \rightarrow \sigma_i; e'_i \rangle \models_P \delta.$$

The game $H(\tau')$ has a corresponding position, say,

$$\bigwedge \{ \langle \alpha_i \rightarrow \sigma'; e'_i \rangle \mid i \in I \text{ and } \sigma_i \preceq \sigma' \} \models \delta.$$

So every strategy for $G(\tau)$ has a corresponding strategy for $H(\tau')$. ■

Lemma 63. Let τ be a finite prime type of kind $(\kappa \rightarrow \kappa) \rightarrow \kappa$. Assume that P wins for $H(\tau)$. Then one can effectively construct a λY -term D such that $\text{BT}(D) \blacktriangleright (\emptyset \vdash Y : \tau)$.

Proof: Let

$$\tau = \left(\bigwedge_{i \in I} \left\langle \bigwedge_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle \rightarrow \sigma_i; e_i \right\rangle \right) \rightarrow \tau_0.$$

We can assume without loss of generality that τ_0, σ_i (for every $i \in I$) and $\delta_{i,j}$ (for every $i \in I$ and $j \in J_i$) are elements of K .

Every position of $H(\tau)$ reachable from the initial position is of the form:

$$d \parallel \left(\bigwedge_{i \in I} \left\langle \bigwedge_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle \rightarrow \sigma_i; e_i \right\rangle \right) \models_P \delta_{i,j}$$

or

$$d \parallel \left(\bigwedge_{i \in I} \left\langle \bigwedge_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle \rightarrow \sigma_i; e_i \right\rangle \right) \models_O \bigwedge_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle$$

or the initial node

$$d \parallel \left(\bigwedge_{i \in I} \left\langle \bigwedge_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle \rightarrow \sigma_i; e_i \right\rangle \right) \models_P \tau_0.$$

Hence a P-position (except for the initial position) can be represented by (d, i, j) where $d \in \mathbb{E}$, $i \in I$ and $j \in J$.

Since $H(\tau)$ is an ω -regular game over a finite graph, one can effectively construct a finite memory strategy for the game. Since an edge from a P-position can be specified by $i \in I$, the winning strategy can be expressed by the pair of functions:

$$\begin{aligned} w : (d, i, j, a) &\mapsto i' \in I \\ v : (d, i, j, a) &\mapsto a' \in A \end{aligned}$$

together with the decision on the initial position, say $i_0 \in I$ and $a_0 \in A$.

The expected term D is defined using mutual recursion on variables $x_{(d,i,j,a)}$, where (d, i, j) indicates a P-position (except for the initial position) and $a \in A$ is the memory the winning strategy uses. Note that mutual recursion can be implemented by the unary recursion.

$$D = \lambda f : \prod_{i \in I} \left\langle \prod_{j \in J_i} \langle \delta_{i,j}; d_{i,j} \rangle \rightarrow \sigma_i; e_i \right\rangle.$$

let rec $\forall d \in \mathbb{E}, i \in I, j \in J_i, a \in A$

$$x_{(d,i,j,a)} = (\text{p}_{i'} f) \left(\prod_{j' \in J_{i'}} x_{(d,i',j',od, i', j', a')} \right)$$

where $i' = w(d, i, j, a)$ **and** $a' = v(d, i, j, a)$

in

$$(\text{p}_{i_0} f) \left(\prod_{j \in J_{i_0}} x_{(d_{i_0}, j, i_0, j, a_0)} \right).$$

Then $\text{BT}(D) \blacktriangleright (\emptyset \vdash Y : \tau)$. ■

Now we assume that $\emptyset \vdash Y : \tau$. Then by Lemma 15, we know that P wins for $G(\tau)$. Then by Lemma 62, P wins for $H(\tau')$ for some $\tau' \preceq \tau$. By Lemma 63, one can effectively construct D_1 such that $\text{BT}(D_1) \blacktriangleright (\emptyset \vdash Y : \tau')$. Since $\tau' \preceq \tau$, one can effectively construct D_0 such that $\text{BT}(D_0) \blacktriangleright (\emptyset \vdash \text{BT}(\lambda x.x) : \langle \tau'; \varepsilon \rangle \rightarrow \tau)$ by Lemma 58. Let $D = D_0 D_1$. By Theorem 11 and Proposition 12, we have $\text{BT}(D) \blacktriangleright (\emptyset \vdash Y : \tau)$ as desired.

APPENDIX D

SUPPLEMENTARY MATERIALS FOR SECTION V

A. Definitions of products and exponentials

Assume arenas $A = (\mathcal{M}_A, \lambda_A, \vdash_A, \vartheta_A, E_A)$ and $B = (\mathcal{M}_B, \lambda_B, \vdash_B, \vartheta_B, E_B)$.

1) *Product:* $A \times B = (\mathcal{M}, \lambda, \vdash, \vartheta, E)$ is defined by:

- $\mathcal{M} = \mathcal{M}_A + \mathcal{M}_B$.
- $\lambda(m) = \begin{cases} \lambda_A(m) & (\text{if } m \in \mathcal{M}_A) \\ \lambda_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.
- $m \vdash m'$ just if $m \vdash_A m'$ or $m \vdash_B m'$.
- $\vartheta(m) = \begin{cases} \vartheta_A(m) & (\text{if } m \in \mathcal{M}_A) \\ \vartheta_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.
- $E(m) = \begin{cases} E_A(m) & (\text{if } m \in \mathcal{M}_A) \\ E_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.

2) *Exponential:* $A \Rightarrow B = (\mathcal{M}, \lambda, \vdash, \vartheta, E)$ is defined by:

- $\mathcal{M} = \mathcal{M}_B^{\text{init}} \times \mathcal{M}_A + \mathcal{M}_B$.
- $\lambda(m) = \begin{cases} \lambda_A(m_1) & (\text{if } m = (m_0, m_1) \in \mathcal{M}_B^{\text{init}} \times \mathcal{M}_A) \\ \lambda_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.
- The enabling relation \vdash is defined by the following rules.
 - If $m \vdash_B n$, then $m \vdash n$.
 - If $m \in \mathcal{M}_B^{\text{init}}$ and $n \in \mathcal{M}_A^{\text{init}}$, then $m \vdash (m, n)$.
 - If $m \in \mathcal{M}_B^{\text{init}}$ and $n \vdash_A n'$, then $(m, n) \vdash (m, n')$.
- $\vartheta(m) = \begin{cases} \vartheta_A(m_1) & (\text{if } m = (m_0, m_1) \in \mathcal{M}_B^{\text{init}} \times \mathcal{M}_A) \\ \vartheta_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.
- $E(m) = \begin{cases} E_B(m_0) \setminus E_A(m_1) & (\text{if } m = (m_0, m_1) \text{ and } m_1 \in \mathcal{M}_A^{\text{init}}) \\ E_A(m_1) & (\text{if } m = (m_0, m_1) \text{ and } m_1 \notin \mathcal{M}_A^{\text{init}}) \\ E_B(m) & (\text{if } m \in \mathcal{M}_B) \end{cases}$.

B. Proof of Lemma 28

Let A be an effect arena and assume that A is accepted. Consider the arena $A \Rightarrow A$. Given a move m of A , we write m^l (resp. m^r) for the corresponding move in the left (resp. right) component of $A \Rightarrow A$.

Every (possibly infinite) P-view $p \in \text{id}_A : A^l \Rightarrow A^r$ is of the form

$$\begin{aligned} &m_1^r \cdot (m_1^r, m_1^l) \cdot (m_1^r, m_2^l) \cdot m_2^r \cdot m_3^r \cdot \dots \cdot \\ &m_{2n-1}^r \cdot (m_1^r, m_{2n-1}^l) \cdot (m_1^r, m_{2n}^l) \cdot m_{2n}^r \cdot \dots, \end{aligned}$$

where m_{2n+1}^r and m_{2n}^l are O-moves, m_{n+1}^r is justified by m_n^r and m_{n+1}^l is justified by m_n^l . (All the moves from the left component are associated with the initial move of the right component.) It is easy to see that p has correct summary. Note that, for every initial A -move m , we have

$$\varepsilon \preceq E_A(m) \setminus E_A(m) = E_{A^l \Rightarrow A^r}((m, m)).$$

To prove that p satisfies the winning condition, it suffices to show that

$$\pi\langle E_{A \Rightarrow A}(m_1^r), \dots, E_{A \Rightarrow A}(m_{2n-1}^r), E_{A \Rightarrow A}((m_1^r, m_{2n}^l)), \dots \rangle \in \Omega. m.$$

By the definition of the effect assignment for $A \Rightarrow A$, the above sequence is equivalent to

$$\pi\langle E_A(m_1), E_A(m_2), \dots, E_A(m_{2n-1}), E_A(m_{2n}), \dots \rangle \in \Omega.$$

This condition holds because $m_1 \vdash_A m_2 \vdash_A m_3 \vdash_A \dots$ and A satisfies Ω .

It is easy to prove that id_A is ground-type reflecting.

C. Proof of Lemma 31

Our starting point is the following result proved by Clairambault and Harmer [4]. An innocent strategy \mathfrak{s} is said to be *total* if $s \cdot \perp \notin \mathfrak{s}$ for every s , and *Noetherian* just if it has no infinite view.

Lemma 64 (Clairambault and Harmer [4]). *Let $\mathfrak{s} : |A| \Rightarrow |B|$ and $\mathfrak{t} : |B| \Rightarrow |C|$ be innocent strategies. If \mathfrak{s} and \mathfrak{t} are total and Noetherian, then so is $(\mathfrak{s}; \mathfrak{t}) : |A| \Rightarrow |C|$.*

Assume that $\mathfrak{s} = m_1 \cdot m_2 \cdot \dots$. We first construct a “linearised” version of arenas, strategies and the interaction sequence. Let A' be the arena having as moves the set of all occurrences of A -moves in \mathfrak{s} , formally defined by:

$$\mathcal{M}_{A'} = \{i \in \omega \mid m_i \in \mathcal{M}_A\}.$$

For every $i, j \in \mathcal{M}_{A'}$, $i \vdash j$ if and only if m_j points to m_i in \mathfrak{s} . The arena B' and C' are define similarly, e.g. sets of moves are

$$\mathcal{M}_{B'} = \{i \in \omega \mid m_i \in \mathcal{M}_B\} \quad \mathcal{M}_{C'} = \{i \in \omega \mid m_i \in \mathcal{M}_C\}.$$

By abuse of notation, for every $i \in \mathcal{M}_{B'}$, we also write i for the move (j, i) in $(A' \Rightarrow B') \Rightarrow C' = \text{Int}(A', B', C')$, where j is the unique initial C' move. Similarly, for every $i \in \mathcal{M}_{A'}$, we write i for the move $(j, (k, i))$ in $(A' \Rightarrow B') \Rightarrow C'$, where j is the unique initial C' move and k is the index of the unique initial B move m_k that hereditary justifies m_i . Then for every $n \in \omega$, we have $1 \cdot 2 \cdot \dots \cdot n \in \text{Int}(A', B', C')$, where j points to i just if m_j points to m_i in \mathfrak{s} . So $\mathfrak{s}' = 1 \cdot 2 \cdot \dots \cdot n \cdot \dots$ is a well-defined infinite interaction sequence. Consider the minimum strategies such that their interaction sequence contains \mathfrak{s}' , which is defined by:

$$\mathfrak{s}' = \{\ulcorner (1 \cdot 2 \cdot \dots \cdot k) \urcorner \upharpoonright A' \Rightarrow B' \urcorner \mid k \in \omega\}$$

$$\mathfrak{t}' = \{\ulcorner (1 \cdot 2 \cdot \dots \cdot k) \urcorner \upharpoonright B' \Rightarrow C' \urcorner \mid k \in \omega\}.$$

Then $\mathfrak{s}' = 1 \cdot 2 \cdot \dots \in \text{Int}(\mathfrak{s}', \mathfrak{t}')$ as expected. Since $\mathfrak{s} \upharpoonright A \Rightarrow C \in (\mathfrak{s}; \mathfrak{t})$, $\mathfrak{s}; \mathfrak{t}$ is non-Noetherian (if \mathfrak{s} generates an infinite P-view) or non-total (if \mathfrak{s} is an infinite chattering). Hence by Lemma 64 and totality of \mathfrak{s}' and \mathfrak{t}' , either \mathfrak{s}' or \mathfrak{t}' has an infinite P-view, say

$$i_1 \cdot i_2 \cdot \dots \cdot i_j \cdot \dots$$

Then $m_{i_1} m_{i_2} m_{i_3} \dots m_{i_j} \dots$ is a subsequence of \mathfrak{s} that is a P-view in the strategy. Furthermore, by definition of the strategies, we have $i_{2k+2} = 1 + i_{2k+1}$ for every k . So the sequence is actually a sub-view.

D. Proof of Lemma 29

By induction on the length of s_2 . We do the case analysis

If m is an O-move of $A \Rightarrow C$, then s_2 must be the empty play since $s \upharpoonright_{A \Rightarrow C}$ is a P-view. In this case, the equation trivially holds.

Assume otherwise. Then m is a P-move of $A \Rightarrow B$ or of $B \Rightarrow C$. Assume that m is a P-move of $A \Rightarrow B$. Note that $s \upharpoonright_{A \Rightarrow B} \in \mathfrak{s}$.

- Case $s_2 = \epsilon$: Then $(s_1 \cdot n \cdot m) \upharpoonright_{A \Rightarrow B} = (s_1 \upharpoonright_{A \Rightarrow B}) \cdot n \cdot m \in \mathfrak{s}$. Because \mathfrak{s} has correct summary and m points to n , we have $E^*(\epsilon) = \epsilon \preceq E_{A \Rightarrow B}(m)$. Because m is not an initial $A \Rightarrow B$ move, $E_{A \Rightarrow B}(m) = E_{(A \Rightarrow B) \Rightarrow C}(m)$. Therefore we have

$$E_{(A \Rightarrow B) \Rightarrow C}^*((s_2 \cdot m) \upharpoonright_{A \Rightarrow C}^O) = \epsilon \preceq E_{(A \Rightarrow B) \Rightarrow C}(m)$$

as desired.

- Case s_2 is not empty: Let $s' = s_1 \cdot n \cdot s_2 \cdot m$ and assume

$$s' = s_1 \cdot n_1 \cdot n_2 \cdot t_2 \cdot n_3 \cdot \dots \cdot n_{2l} \cdot t_{2l} \cdot n_{2l+1} \cdot m.$$

Here for every $k \leq l$, n_{2k+1} is an O-move of $A \Rightarrow B$ and n_{2k} is a P-move of $A \Rightarrow B$ (and $n_1 = n$). The P-view of the $A \Rightarrow B$ component of s' (that is, $\ulcorner s' \upharpoonright_{A \Rightarrow B} \urcorner$) is given by:

$$\ulcorner s_1 \upharpoonright_{A \Rightarrow B} \urcorner \cdot n_1 \cdot n_2 \cdot n_3 \cdot \dots \cdot n_{2l} \cdot n_{2l+1} \cdot m.$$

Because \mathfrak{s} has correct summary and m points to n , we have

$$E_{A \Rightarrow B}^*(n_3 \cdot n_5 \cdot \dots \cdot n_{2l+1}) \preceq E_{A \Rightarrow B}(m).$$

Now we have

$$\begin{aligned} & E_{A \Rightarrow B}^*(n_3 \cdot \dots \cdot n_{2l+1}) \\ &= E_{A \Rightarrow B}(n_3) \circ \dots \circ E_{A \Rightarrow B}(n_{2l+1}) \\ &= E_{(A \Rightarrow B) \Rightarrow C}(n_3) \circ \dots \circ E_{(A \Rightarrow B) \Rightarrow C}(n_{2l+1}) \\ &\succeq E_{(A \Rightarrow B) \Rightarrow C}^*((t_2 \cdot n_3) \upharpoonright_{A \Rightarrow C}^O) \circ \dots \\ &\quad \circ E_{(A \Rightarrow B) \Rightarrow C}^*((t_{2l} \cdot n_{2l+1}) \upharpoonright_{A \Rightarrow C}^O) \\ &= E_{(A \Rightarrow B) \Rightarrow C}^*((n_2 \cdot t_2 \cdot n_3) \upharpoonright_{A \Rightarrow C}^O) \circ \dots \\ &\quad \circ E_{(A \Rightarrow B) \Rightarrow C}^*((n_{2l} \cdot t_{2l} \cdot n_{2l+1}) \upharpoonright_{A \Rightarrow C}^O) \\ &= E_{(A \Rightarrow B) \Rightarrow C}^*(s_2 \upharpoonright_{A \Rightarrow C}^O) \\ &= E_{(A \Rightarrow B) \Rightarrow C}^*((s_2 \cdot m) \upharpoonright_{A \Rightarrow C}^O). \end{aligned}$$

For the second equation, note that n_{2k+1} (for $k > 0$) is not an initial move of $A \Rightarrow B$ and thus $E_{A \Rightarrow B}(n_{2k+1}) = E_{(A \Rightarrow B) \Rightarrow C}(n_{2k+1})$. The inequation in the fourth line come from the induction hypothesis. Other equations are consequences of the fact that m and n_{2k} (for $k \leq l$) are not O-moves of $A \Rightarrow C$. In summary, we have

$$\begin{aligned} & E_{(A \Rightarrow B) \Rightarrow C}^*((s_2 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \preceq E_{A \Rightarrow B}^*(n_3 \cdot \dots \cdot n_{2l+1}) \\ &\preceq E_{A \Rightarrow B}(m) = E_{(A \Rightarrow B) \Rightarrow C}(m) \end{aligned}$$

as required.

The other case can be proved by a similar way. Note that $E_{B \Rightarrow C}(m) = E_{(A \Rightarrow B) \Rightarrow C}(m)$ for every move m in the component $B \Rightarrow C$.

E. Proof of corollary 30

Assume effect arenas A , B and C and summarising innocent strategies $\mathfrak{s} : A \Rightarrow B$ and $\mathfrak{t} : B \Rightarrow C$. Let $p \in (\mathfrak{s}; \mathfrak{t})$ be a P-view and assume

$$p = p_0 \cdot \overset{\curvearrowright}{n} \cdot p_1 \cdot m.$$

It suffices to prove that $E_{A \Rightarrow C}^*(p_1 \upharpoonright^O) \preceq E_{A \Rightarrow C}(m)$.

Let $s \in \text{Int}(\mathfrak{s}, \mathfrak{t})$ be the interaction sequence that generates p , i.e. $p = s \upharpoonright_{A \Rightarrow C}$. Let

$$s = s_0 \cdot \overset{\curvearrowright}{n'} \cdot s_1 \cdot m.$$

If m is not an initial move of A , then $n' = n$ and $p_1 = s_1 \upharpoonright_{A \Rightarrow C}$. By Lemma 29, we have

$$E_{(A \Rightarrow B) \Rightarrow C}^*((s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \preceq E_{(A \Rightarrow B) \Rightarrow C}(m).$$

Since m and moves in $(s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O$ are not initial B - or A -moves, we have

$$\begin{aligned} E_{(A \Rightarrow B) \Rightarrow C}^*((s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) &= E_{A \Rightarrow C}^*((s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \\ &= E_{A \Rightarrow C}^*(p_1 \upharpoonright^O) \end{aligned}$$

and

$$E_{(A \Rightarrow B) \Rightarrow C}(m) = E_{A \Rightarrow C}(m).$$

Hence $E_{A \Rightarrow C}^*(p_1 \upharpoonright^O) \preceq E_{A \Rightarrow C}(m)$ as required.

If m is an initial A -move, then p_0 is the empty and n is an initial C -move. Then we have

$$\begin{aligned} p &= \overset{\curvearrowright}{n} \cdot p_1 \cdot m \\ s &= \overset{\curvearrowright}{n} \cdot s'_0 \cdot \overset{\curvearrowright}{n'} \cdot s_1 \cdot m, \end{aligned}$$

where $n \cdot s'_0 = s_0$ and n' is an initial B -move. By Lemma 29, we have

$$\begin{aligned} E_{(A \Rightarrow B) \Rightarrow C}^*((s'_0 \cdot n') \upharpoonright_{A \Rightarrow C}^O) &\preceq E_{(A \Rightarrow B) \Rightarrow C}(n') \\ E_{(A \Rightarrow B) \Rightarrow C}^*((s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) &\preceq E_{(A \Rightarrow B) \Rightarrow C}(m). \end{aligned}$$

By definition of $E_{(A \Rightarrow B) \Rightarrow C}$, we have

$$\begin{aligned} E_{(A \Rightarrow B) \Rightarrow C}(n') &= E_C(n) \setminus E_B(n') \\ E_{(A \Rightarrow B) \Rightarrow C}(m) &= E_B(n') \setminus E_A(m). \end{aligned}$$

Here we identify an A -move and a move in the A -component of $(A \Rightarrow B) \Rightarrow C$ and so on. Therefore

$$\begin{aligned} E_C(n) \circ E_{(A \Rightarrow B) \Rightarrow C}^*((s'_0 \cdot n') \upharpoonright_{A \Rightarrow C}^O) &\preceq E_B(n') \\ E_B(n') \circ E_{(A \Rightarrow B) \Rightarrow C}^*((s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) &\preceq E_A(m). \end{aligned}$$

By monotonicity of \circ and the definition of E^* , we have

$$E_C(n) \circ E_{(A \Rightarrow B) \Rightarrow C}^*((s'_0 \cdot n' \cdot s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \preceq E_A(m)$$

and hence

$$E_{(A \Rightarrow B) \Rightarrow C}^*((s'_0 \cdot n' \cdot s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \preceq (E_C(n) \setminus E_A(m)).$$

Since $(s'_0 \cdot n' \cdot s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O$ does not contain any initial A -move nor initial B -move,

$$\begin{aligned} E_{(A \Rightarrow B) \Rightarrow C}^*((s'_0 \cdot n' \cdot s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) \\ = E_{A \Rightarrow C}^*((s'_0 \cdot n' \cdot s_1 \cdot m) \upharpoonright_{A \Rightarrow C}^O) = E_{A \Rightarrow C}^*(p_1 \upharpoonright^O). \end{aligned}$$

Hence $E_{A \Rightarrow C}^*(p_1 \upharpoonright^O) \preceq E_{A \Rightarrow C}(m)$ as desired.

F. Proof of Lemma 32

Firstly we check that the winning condition for infinite P-views. Let $\mathbf{p} \in (\mathfrak{s}; \mathfrak{t})$ be an infinite P-view in the composite, and $s \in \text{Int}(\mathfrak{s}, \mathfrak{t})$ be the interaction sequence generating \mathbf{p} . By Lemma 31, s has an infinite P-view of \mathfrak{s} or \mathfrak{t} as its sub-view. Consider the case that it is an infinite P-view \mathbf{q} of \mathfrak{s} . The other case can be proved similarly. Assume that

$$s = m_1 \cdot m_2 \cdot \dots \text{ and } \mathbf{q} = n_1 \cdot n_2 \cdot \dots$$

Since \mathbf{q} is a sub-view of s , we have

$$s = \overset{\curvearrowright}{m_1} \cdot s_1 \cdot \overset{\curvearrowright}{n_1} \cdot \overset{\curvearrowright}{n_2} \cdot s_3 \cdot \overset{\curvearrowright}{n_3} \cdot \overset{\curvearrowright}{n_4} \cdot s_5 \cdot \overset{\curvearrowright}{n_5} \cdot n_6 \cdot \dots$$

Note that n_{2k} cannot be an O-move of $A \Rightarrow C$ and hence

$$(\mathbf{p} \upharpoonright^O) = m_1 \cdot ((s_1 \cdot n_1) \upharpoonright_{A \Rightarrow C}^O) \cdot ((s_3 \cdot n_3) \upharpoonright_{A \Rightarrow C}^O) \cdot \dots$$

Let us define $t_{2k+1} = (s_{2k+1} \cdot n_{2k+1}) \upharpoonright_{A \Rightarrow C}^O$. By Lemma 29, for every k , we have

$$E_{(A \Rightarrow B) \Rightarrow C}^*(t_{2k+1}) \preceq E_{(A \Rightarrow B) \Rightarrow C}(n_{2k+1}).$$

By definition of $E_{(A \Rightarrow B) \Rightarrow C}(n_1)$,

$$E_{(A \Rightarrow B) \Rightarrow C}(n_1) = E_C(m_1) \setminus E_{A \Rightarrow B}(n_1).$$

Therefore

$$E_C(m_1) \circ E_{(A \Rightarrow B) \Rightarrow C}(n_1) \preceq E_{A \Rightarrow B}(n_1).$$

Now we have

$$\begin{aligned} \pi \langle E_{A \Rightarrow B}(n_1), E_{A \Rightarrow B}(n_3), E_{A \Rightarrow B}(n_5), \dots \rangle \\ = \pi \langle E_{A \Rightarrow B}(n_1), E_{(A \Rightarrow B) \Rightarrow C}(n_3), E_{(A \Rightarrow B) \Rightarrow C}(n_5), \dots \rangle \\ \succeq \pi \langle E_C(m_1) \circ E_{(A \Rightarrow B) \Rightarrow C}(n_1), E_{(A \Rightarrow B) \Rightarrow C}(n_3), \dots \rangle \\ \succeq \pi \langle E_C(m_1) \circ E_{(A \Rightarrow B) \Rightarrow C}^*(t_1), E_{(A \Rightarrow B) \Rightarrow C}^*(t_3), \dots \rangle \end{aligned}$$

Because $\pi \langle E_{A \Rightarrow B}(n_1), E_{A \Rightarrow B}(n_3), E_{A \Rightarrow B}(n_5), \dots \rangle \in \Omega$ by the assumption and Ω is lower-closed, we have $\pi \langle E_C(m_1) \circ E_{(A \Rightarrow B) \Rightarrow C}^*(t_1), E_{(A \Rightarrow B) \Rightarrow C}^*(t_3), \dots \rangle \in \Omega$. By the property of the infinite product and $E_C(m_1) = E_{(A \Rightarrow B) \Rightarrow C}(m_1)$, this is equivalent to $E_{(A \Rightarrow B) \Rightarrow C}^\omega(\mathbf{p} \upharpoonright^O) \in \Omega$. Observe that $\mathbf{p} \upharpoonright^O$ does not contain any initial A -move that must be a P-move of $A \Rightarrow C$. Hence $E_{A \Rightarrow C}^\omega(\mathbf{p} \upharpoonright^O) = E_{(A \Rightarrow B) \Rightarrow C}^\omega(\mathbf{p} \upharpoonright^O) \in \Omega$.

Now we check the winning condition for infinite chattering. Since $p \cdot \perp \in (\mathfrak{s}; \mathfrak{t})$, the interaction of \mathfrak{s} and \mathfrak{t} either goes into infinite chattering or reaches an unanswered play of \mathfrak{s} or \mathfrak{t} . For the former case, an argument similar to above proves the claim. A point is that there exists $l \in \omega$ such that $l < k$ implies emptiness of t_{2k+1} . The latter case can be easily proved.

G. Proofs of claims in Section V-D

To prove $A \times B$ is a product, it suffices to give a natural bijection $\mathcal{G}(C, A \times B) \cong \mathcal{G}(C, A) \times \mathcal{G}(C, B)$. Recall that we have a natural bijection $\varphi : \text{Inn}(|C|, |A| \times |B|) \cong \text{Inn}(|C|, |A|) \times \text{Inn}(|C|, |B|)$. It is easy to prove that φ maps a consistent strategy to a pair of consistent strategies and vice versa. Similarly, we can prove that the natural bijection $\text{Inn}(|A| \times |B|, |C|) \cong \text{Inn}(|A|, |B| \Rightarrow |C|)$ maps a consistent strategy of $\mathcal{G}(A \times B, C)$ to a consistent strategy of $\mathcal{G}(A, B \Rightarrow C)$ and vice versa. Therefore \mathcal{G} is a CCC.

To prove $- \otimes -$ is a bifunctor, it suffices to prove that $\text{id}_{|A|} : e \otimes A \rightarrow e' \otimes A$ is consistent for every $e \succeq e'$. First we check the winning condition for infinite plays. Assume an infinite P-view $s = m_1 \cdot m_2 \cdot \dots \in \text{id}_{|A|}$. Since A satisfies Ω , we have

$$\pi \langle E_{A \Rightarrow A}(m_1), E_{A \Rightarrow A}(m_3), E_{A \Rightarrow A}(m_5), \dots \rangle \in \Omega.$$

By definition of $e \otimes A \Rightarrow e' \otimes A$, we have

$$\begin{aligned} \pi \langle E_{e \otimes A \Rightarrow e' \otimes A}(m_1), E_{e \otimes A \Rightarrow e' \otimes A}(m_3), E_{e \otimes A \Rightarrow e' \otimes A}(m_5), \dots \rangle \\ = \pi \langle e' \circ E_{A \Rightarrow A}(m_1), E_{A \Rightarrow A}(m_3), E_{A \Rightarrow A}(m_5), \dots \rangle \\ = \pi \langle e', E_{A \Rightarrow A}(m_1), E_{A \Rightarrow A}(m_3), E_{A \Rightarrow A}(m_5), \dots \rangle. \end{aligned}$$

Since Ω is stable, we have $\pi \langle e', E_{A \Rightarrow A}(m_1), E_{A \Rightarrow A}(m_3), E_{A \Rightarrow A}(m_5), \dots \rangle \in \Omega$ from $\pi \langle E_{A \Rightarrow A}(m_1), E_{A \Rightarrow A}(m_3), E_{A \Rightarrow A}(m_5), \dots \rangle \in \Omega$. Second we prove that $\text{id}_{|A|}$ is summarising. The only problematic case is $m \cdot (m, m) \in \text{id}_{|A|}$ for $m \in \mathcal{M}_{|A|}^{\text{init}}$, which requires $E_{e \otimes A \Rightarrow e' \otimes A}((m, m)) \succeq \varepsilon$. We have

$$\begin{aligned} E_{e \otimes A \Rightarrow e' \otimes A}((m, m)) &= E_{e' \otimes A}(m) \setminus E_{e \otimes A}(m) \\ &= (e' \circ E_A(m)) \setminus (e \circ E_A(m)). \end{aligned}$$

Since $e' \preceq e$, we have $e' \circ E_A(m) \preceq e \circ E_A(m)$ and hence $\varepsilon \preceq (e' \circ E_A(m)) \setminus (e \circ E_A(m))$ as desired.

The fact that $- \parallel -$ is a bifunctor $\mathbb{E} \times \mathcal{G} \rightarrow \mathcal{G}$ can be proved similarly.

To prove that $- \parallel -$ and $- \otimes -$ form an \mathbb{E} -parametrised adjunction, it suffices to prove that $(e \parallel -) \dashv (e \otimes -)$ for every $e \in \mathbb{E}$. We prove that the identity map $\text{Inn}(|A|, |B|) \cong \text{Inn}(|A|, |B|)$ maps a consistent strategy $\mathcal{G}(e \parallel A, B)$ to a consistent strategy $\mathcal{G}(A, e \otimes B)$ and vice versa. The preservation of the winning condition is a consequence of the stability of Ω as above. We prove that the preservation of the correctness of summary. Assume a summarising strategy $s : (e \parallel A) \Rightarrow B$. The only problematic case is the effect of the initial A move, i.e.,

$$s = \overset{\curvearrowright}{m \cdot s_0 \cdot (m, n)}$$

where m is an initial B move and n is an initial A move (and hence (m, n) is a move of $(e \parallel A) \Rightarrow B$). Since s is summarising, we have

$$E_{(e \parallel A) \Rightarrow B}(s_0 \upharpoonright^O) \preceq E_{(e \parallel A) \Rightarrow B}((m, n)) = E_B(m) \setminus E_{e \parallel A}(n).$$

Since $s_0 \upharpoonright^O$ cannot contain any initial A -move nor initial B -move, we have

$$E_{(e \parallel A) \Rightarrow B}(s_0 \upharpoonright^O) = E_{A \Rightarrow B}(s_0 \upharpoonright^O) = E_{A \Rightarrow (e \otimes B)}(s_0 \upharpoonright^O).$$

We write this effect as d . Now we have

$$\begin{aligned} d &\preceq E_B(m) \setminus E_{e \parallel A}(n) \\ \text{iff} \quad E_B(m) \circ d &\preceq E_{e \parallel A}(n) \\ \text{iff} \quad E_B(m) \circ d &\preceq e \setminus E_A(n) \\ \text{iff} \quad e \circ E_B(m) \circ d &\preceq E_A(n) \\ \text{iff} \quad E_{e \otimes B}(m) \circ d &\preceq E_A(n) \\ \text{iff} \quad d &\preceq E_{e \otimes B}(m) \setminus E_A(n) \\ \text{iff} \quad d &\preceq E_{A \Rightarrow (e \otimes B)}((m, n)). \end{aligned}$$

APPENDIX E

SUPPLEMENTARY MATERIALS FOR SECTION VI

A. Proof of Lemma 39

$(s_1; s_2) \subset (t_1; t_2)$ follows from $s_1 \subset t_1$ and $s_2 \subset t_2$ (by induction on the length of the interaction sequence). Consistency of $s_1; s_2$ follows from Theorem 33. To prove relative totality, a key fact is that $s \cdot \perp \in (s_1; s_2)$ means that one reaches an unanswered play of s_1 or s_2 or infinite chattering. For the former case one reaches the corresponding an unanswered play of s_1 or s_2 and for the latter case one find infinite chattering of t_1 and t_2 .

B. Proof of Lemma 40

Let $(s::t) : (A::I) \rightarrow (C::K)$ and assume that

$$I \xrightarrow{t} K = I \xrightarrow{t_1} J \xrightarrow{t_2} K.$$

First we construct $B :: J$ from analysis of the interaction sequence $\text{Int}(t_1, t_2)$. Effect arena $B = (\mathcal{M}, \vdash, \lambda, \vartheta, E)$ is defined as follows.

- $M = \left\{ \binom{p}{u} \mid \begin{array}{l} u \in \text{Int}(t_1, t_2), u \text{ ends with a } J\text{-move} \\ p \in \mathfrak{s}, \ulcorner p \urcorner = p, \varpi(p) = u \upharpoonright_{I \Rightarrow K} \end{array} \right\}$
- $\binom{p}{u} \vdash \binom{p'}{u'}$ iff
 - 1) p is a prefix of p' ,
 - 2) u is a prefix of u' , and
 - 3) the last move of u' is explicitly justified by the last move of u .
- $\lambda(\binom{p}{u}) = \lambda_J(m)$ where m is the last move of u .
- $\vartheta(\binom{p}{u}) := \vartheta_{A \Rightarrow C}(m)$ where m is the last move of p .
- $E(\binom{p}{u})$ will be defined below.

If $\binom{p}{u}$, then each move m in the $(I \Rightarrow K)$ -component of u has an unique corresponding move in $A \Rightarrow C$, say m' . We define $E(m) = E_{A \Rightarrow C}(m')$. The effect for $\binom{p}{u}$ is defined by:

- If the last move of u is an initial J -move, then $E_B(\binom{p}{u}) := E^*(u \upharpoonright_{I \Rightarrow K}^O) = E_{A \Rightarrow C}^*(p)$.
- Otherwise. Let

$$u = u_1 \cdot \overset{\curvearrowright}{n \cdot u_2 \cdot m},$$

$$\text{then } E_B(\binom{p}{u}) := E^*(u_2 \upharpoonright_{I \Rightarrow K}^O).$$

Intuitively the effect assignments is defined so that it satisfies Lemma 29. However, note that $\binom{p}{u}$ is a move, not an interaction sequence.

Second we define “interaction sequences”. Let p be a P-view of $A \Rightarrow C$ and $u \in \text{Int}(I, J, K)$ be an interaction

sequence and assume that $\varpi(p) = u \upharpoonright_{I \Rightarrow K}$. We define a justified sequence $(A \Rightarrow B) \Rightarrow C$ by:

- If $u = u_0 \cdot m$ with $m \in J$, then $[u_0 \cdot m \mid p] = [u_0 \mid p] \cdot m$.
- Otherwise m is in the $(I \Rightarrow K)$ -component. Then $p = p_0 \cdot m'$ such that $\varpi(m') = m$. We define $[u_0 \cdot m \mid p_0 \cdot m'] = [u_0 \mid p_0] \cdot m'$.

Notice that $[u \mid p]$ has the same length as u . The pointing structure of $[u \mid p]$ inherits from that of u .

Lemma 65. *Let p be a P-view of $A \Rightarrow C$ and $u \in \text{Int}(I, J, K)$ be an interaction sequence and assume that $\varpi(p) = u \upharpoonright_{I \Rightarrow K}$. Then $[u \mid p] \in \text{Int}(A, B, C)$.*

Proof: By induction on the length of u .

We define $\mathcal{I} \subseteq \text{Int}(A, B, C)$ by

$$\mathcal{I} = \left\{ [u \mid p] \mid \begin{array}{l} u \in \text{Int}(t_1, t_2), p \in \mathfrak{s}, \\ \upharpoonright p^\top = p, \varpi(p) = u \upharpoonright_{I \Rightarrow K} \end{array} \right\}.$$

This should be the set of all interactions between \mathfrak{s}_1 and \mathfrak{s}_2 .

Now we define strategies. Let f_1 be a view function of the arena $A \Rightarrow B$ determined by the set of P-views $\{\upharpoonright s \upharpoonright_{A \Rightarrow B}^\top \mid s \in \mathcal{I}\}$ and f_2 be a view function of an arena $B \Rightarrow C$ determined by $\{\upharpoonright s \upharpoonright_{B \Rightarrow C}^\top \mid s \in \mathcal{I}\}$. The strategy \mathfrak{s}_1 is induced from f_1 and \mathfrak{s}_2 from f_2 .

We show that \mathfrak{s}_1 and \mathfrak{s}_2 are well-defined. We need an auxiliary lemma.

Lemma 66. *Let p be a P-view of $I \Rightarrow K$ and $u \in \text{Int}(I, J, K)$ be an interaction sequence and assume that $\varpi(p) = u \upharpoonright_{I \Rightarrow K}$.*

- If u ends with an O-move of $I \Rightarrow J$, then $[u \mid p]$ can be determined by $\upharpoonright [u \mid p] \upharpoonright_{A \Rightarrow B}^\top$.*
- If u ends with an O-move of $J \Rightarrow K$, $[u \mid p]$ can be determined by $\upharpoonright [u \mid p] \upharpoonright_{B \Rightarrow C}^\top$.*

Proof: We prove (i). (ii) is shown by a similar way. We prove the claim by induction on the length of u .

If u ends with a move of J , then the last move contains as much information as the pair (u, p) . Assume that u ends with a move of I . The last move of u is an O-move of $I \Rightarrow J$, and hence an O-move of $I \Rightarrow K$. By the assumption, $(I \Rightarrow K)$ -component of u is a P-view. So u is of the form

$$u = u' \overset{P}{\overbrace{m_1}^{\curvearrowright}} \cdot m_2^O.$$

(Note that m_2^O cannot be an initial I -move.) Since

$$\varpi(p) = u \upharpoonright_{I \Rightarrow K} = (u' \upharpoonright_{I \Rightarrow K}) \cdot m_1^P \cdot m_2^O,$$

there are some moves $n_1^P, n_2^O \in \mathcal{M}_{A \Rightarrow C}$ and a P-view p' of $A \Rightarrow C$ such that $p = p' \cdot n_1^P \cdot n_2^O$ and $\varpi(n_1^P) = m_1^P$ and $\varpi(n_2^O) = m_2^O$. Therefore we have

$$[u \mid p] = [u' \mid p'] \cdot n_1^P \cdot n_2^O.$$

Since n_2^O is an O-move of A ,

$$\upharpoonright [u \mid p]^\top = \upharpoonright [u' \mid p'] \cdot n_1^P \cdot n_2^O^\top = \upharpoonright [u' \mid p']^\top \cdot m_1^P \cdot m_2^O.$$

By the induction hypothesis, we can compute $[u' \mid p']$ from $\upharpoonright [u' \mid p']^\top$. Thus (i) holds. \blacksquare

Lemma 67. *f_1 and f_2 are well-defined view functions.*

Proof: We prove that f_1 is well-defined. Well-definedness of f_2 is shown by the same way.

Let $s \in f_1$ be an play ending with an O-move. It suffices to show that:

$$\text{If } s \cdot m \in \varphi_1 \text{ and } s \cdot m' \in \varphi_1, \text{ then } s \cdot m = s \cdot m'.$$

Assume that $s \cdot m \in f_1$ and $s \cdot m' \in f_1$. By definition of f_1 , we have $u \cdot m, u' \cdot m' \in \mathcal{I}$ such that

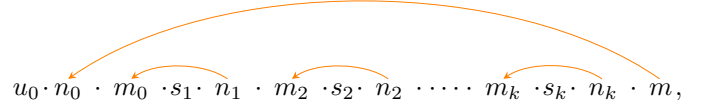
$$\begin{aligned} s \cdot m &= \upharpoonright (u \cdot m) \upharpoonright_{A \Rightarrow B}^\top \\ s \cdot m' &= \upharpoonright (u' \cdot m') \upharpoonright_{A \Rightarrow B}^\top. \end{aligned}$$

Since s ends with an O-move of $A \Rightarrow B$, by Lemma 66, s completely determines u and u' . Thus $u = u'$ and $u \cdot m' \in f_1$. By determinacy of \mathfrak{s} , t_1 and t_2 , we have $u \cdot m = u \cdot m'$ as required. \blacksquare

Hence f_1 and f_2 define innocent strategies \mathfrak{s}_1 and \mathfrak{s}_2 of effect arenas $A \Rightarrow B$ and $B \Rightarrow C$. We now prove that $\mathfrak{s}_1 :: t_1$ and $\mathfrak{s}_2 :: t_2$ are winning. Trivially, $\varpi(\mathfrak{s}_1) \subset t_1$ and $\varpi(\mathfrak{s}_2) \subset t_2$. It is easy to see that \mathfrak{s}_1 and \mathfrak{s}_2 are Ground-Type Reflecting.

Lemma 68. *\mathfrak{s}_1 and \mathfrak{s}_2 are summarising.*

Proof: Assume a P-view of \mathfrak{s}_1 ending with a P-move. By definition, it is of the form $\upharpoonright [u \cdot m \mid p] \upharpoonright_{A \Rightarrow B}^\top$. Let u be

$$u_0 \cdot n_0 \cdot m_0 \cdot s_1 \cdot n_1 \cdot m_2 \cdot s_2 \cdot n_2 \cdots m_k \cdot s_k \cdot n_k \cdot m,$$


where m and m_i are P-moves and n_i is an O-move (of $I \Rightarrow J$). Since this interaction sequence is associated with p such that $\varpi(p) = u \upharpoonright_{I \Rightarrow K}$, an effect is assigned for each move.

Then we have

$$\begin{aligned} E(n_i) &= E^*((s_i \cdot n_i) \upharpoonright_{I \Rightarrow K}^O) \quad (\text{for every } i) \\ E(m) &= E^*((m_0 \cdot s_1 \cdot n_1 \cdots n_k \cdot m) \upharpoonright_{I \Rightarrow K}^O). \end{aligned}$$

So

$$E(n_1) \circ \cdots \circ E(n_k) = E(m)$$

as desired. So \mathfrak{s}_1 is summarising.

Assume a P-view of \mathfrak{s}_2 . Then we have an interaction sequence u similar to the above case. If m is not an initial J -move, we obtain the expected result by the same argument as above. Assume that m is an initial J -move and hence u_0 is empty. Then

$$\begin{aligned} E(n_i) &= E^*((s_i \cdot n_i) \upharpoonright_{I \Rightarrow K}^O) \quad (\text{for every } i) \\ E_B(m) &= E^*((n_0 \cdot m_0 \cdot s_1 \cdot n_1 \cdots n_k \cdot m) \upharpoonright_{I \Rightarrow K}^O) \end{aligned}$$

and hence

$$E(n_1) \circ \cdots \circ E(n_k) \preceq E(n_0) \setminus E_B(m).$$

By the definition of the effect assignment for the exponent, we have

$$E_{B \Rightarrow C}(m) = E_C(n_0) \setminus E_B(m),$$

which implies

$$E(n_1) \circ \dots \circ E(n_k) \preceq E_{B \Rightarrow C}$$

as expected. \blacksquare

Lemma 69. \mathfrak{s}_1 and \mathfrak{s}_2 satisfies Ω .

Proof: We prove that \mathfrak{s}_1 satisfies Ω . Assume an infinite P-view of $p = \mathfrak{s}_1$. By the definition of B and \mathfrak{s}_1 , we have an infinite interaction sequence $u \in \text{Int}(t_1, t_2)$ such that $\varpi(p) = u|_{I \Rightarrow K}$. Hence $\varpi(p)$ is a sub-view of u , i.e.,

$$u = m_1 \cdot s_1 \cdot n_1 \cdot n_2 \cdot s_3 \cdot n_3 \cdot n_4 \cdot s_5 \cdot n_5 \cdot n_6 \cdot \dots,$$

where $\varpi(p) = n_1 \cdot n_2 \cdot \dots$. Note that n_{2k+1} is an O-move and n_{2k} is a P-move (of $J \Rightarrow K$). Since u is associated with p , each move in u can be considered as having its effect. It suffices to prove that

$$\pi(E(n_1), E(n_3), \dots) \in \Omega.$$

By definition,

$$\begin{aligned} E(n_1) &= E^*((m_1 \cdot s_1 \cdot n_1)|_{I \Rightarrow K}^O) \\ E(n_3) &= E^*((s_3 \cdot n_3)|_{I \Rightarrow K}^O) \\ E(n_5) &= E^*((s_5 \cdot n_5)|_{I \Rightarrow K}^O) \\ &\vdots \end{aligned}$$

Since \mathfrak{s} satisfies Ω , we have

$$\pi(E^*((m_1 \cdot s_1 \cdot n_1)|_{I \Rightarrow K}^O), E^*((s_3 \cdot n_3)|_{I \Rightarrow K}^O), \dots) \in \Omega.$$

Hence we obtain the claim. \blacksquare

Lemma 70. $(\mathfrak{s}_1; \mathfrak{s}_2) = \mathfrak{s}$.

Proof: We first prove that $\mathfrak{s} \subseteq (\mathfrak{s}_1; \mathfrak{s}_2)$. Since \mathfrak{s} is innocent, it suffices to show that $\mathfrak{s}_1; \mathfrak{s}_2$ contains every P-view $p \in \mathfrak{s}$. Let $p \in \mathfrak{s}$ be a P-view. Then $\varpi(p) \in \mathfrak{t} = (t_1; t_2)$. Thus there is $u \in \text{Int}(t_1, t_2)$ such that $\varpi(p) = u|_{I \Rightarrow K}$. By definition, $[u | p] \in I$. We can prove $[u | p] \in \text{Int}(\mathfrak{s}_1, \mathfrak{s}_2)$ by induction on the length of u . So $p = [u | p]|_{A \Rightarrow C} \in (\mathfrak{s}_1; \mathfrak{s}_2)$.

Second we prove that $(\mathfrak{s}_1; \mathfrak{s}_2) \subseteq \mathfrak{s}$. It suffices to show that $p \in \mathfrak{s}$ for every P-view $p \in (\mathfrak{s}_1; \mathfrak{s}_2)$. Since $p \in (\mathfrak{s}_1; \mathfrak{s}_2)$, we have its uncovering $u \in \text{Int}(\mathfrak{s}_1, \mathfrak{s}_2)$ (so $p = u|_{A \Rightarrow C}$). Then by induction on the length of u , we can prove that $u|_{A \Rightarrow C} \in \mathfrak{s}$. \blacksquare

Lemmas above prove that the strategies \mathfrak{s}_1 and \mathfrak{s}_2 satisfy all the requirements.

C. Proof of Lemma 41

First we give a game semantics of extended types. Since extended types do not have intersections, they can be viewed as effect arenas.

$$\begin{aligned} \langle q \rangle_E &= A_q \\ \langle \hat{\alpha} \rightarrow \tau \rangle_E &= \langle \hat{\alpha} \rangle_E \Rightarrow \langle \hat{\tau} \rangle_E \\ \langle \prod_{i \in I} \langle \hat{\tau}_i; e_i \rangle \rangle_E &= \prod_{i \in I} e_i \otimes \langle \hat{\tau}_i \rangle_E \end{aligned}$$

where A_q is the arena consisting of the unique O-move of ground-type q and of effect ε . Notice that, since the unique initial move of $\langle \hat{\tau} \rangle_E$ has effect ε , the effect assignment for $\langle \hat{\alpha} \rightarrow \hat{\tau} \rangle_E = \langle \hat{\alpha} \rangle_E \Rightarrow \langle \hat{\tau} \rangle_E$ is simply written as:

$$\begin{aligned} E_{\langle \hat{\alpha} \rightarrow \hat{\tau} \rangle_E}(m) &= E_{\langle \hat{\tau} \rangle_E}(m) & \text{if } m \in \mathcal{M}_{\langle \hat{\tau} \rangle_E} \\ E_{\langle \hat{\alpha} \rightarrow \hat{\tau} \rangle_E}((m, n)) &= E_{\langle \hat{\alpha} \rangle_E}(n) & \text{if } (m, n) \in \mathcal{M}_{\langle \hat{\tau} \rangle_E}^{\text{init}} \times \mathcal{M}_{\langle \hat{\alpha} \rangle_E} \end{aligned}$$

The game semantics of extended kinds is defined by:

$$\begin{aligned} \langle o \rangle &= |A|_o \\ \langle \hat{\alpha} \rightarrow \tau \rangle &= \langle \hat{\alpha} \rangle \Rightarrow \langle \hat{\tau} \rangle \\ \langle \prod_{i \in I} \langle \hat{\tau}_i; e_i \rangle \rangle &= \prod_{i \in I} e_i \otimes \langle \hat{\tau}_i \rangle \end{aligned}$$

where $|A|_o$ is the arena consisting of the unique O-move. Then we have

$$|\langle \hat{\tau} \rangle_E| = |\langle \hat{\tau} \rangle|$$

where $|\cdot|$ in the left-hand-side is the forgetful functor and $|\cdot|$ is the map from extended types to their kinds.

Assume a Böhm tree $\Delta \Vdash U :: \kappa$, a type environment $\Gamma :: \Delta$, a prime type $\tau :: \kappa$ and an extended Böhm tree $|\mathfrak{b}(\Gamma)| \Vdash \hat{V} :: |\mathfrak{b}(\tau)|$. The game semantics for extended types gives the effect arena component of the two-level arena semantics of types.

Lemma 71.

- $\llbracket \tau \rrbracket = (\langle \mathfrak{b}(\tau) \rangle_E, \varpi, \langle \kappa \rangle)$ for some ϖ .
- $\llbracket \Gamma \rrbracket = (\langle \mathfrak{b}(\Gamma) \rangle_E, \varpi', \langle \Delta \rangle)$ for some ϖ' .

Proof: The first claim can be proved by induction on the structure of κ . The second claim follows from the first one. \blacksquare

Lemma 41 is a consequence of the following lemmas.

Lemma 72. $\hat{V} \in U$ iff $\langle \hat{V} \rangle \subset \langle U \rangle$.

Lemma 73. Let \hat{U}_0 be an extended Böhm tree. Then $\hat{\Gamma} \models \hat{U}_0 : \hat{\tau}$ iff $\langle \hat{U}_0 \rangle$ is a consistent strategy of $\langle \hat{\Gamma} \rangle_E \Rightarrow \langle \hat{\tau} \rangle_E$.

Proof: We first prove that, if P does not get stuck in the game $\hat{\Gamma} \models \hat{U}_0 : \hat{\tau}$, then $\langle \hat{U}_0 \rangle$ is ground-type reflecting and summarising, by defining a bijective correspondence between plays (i.e. finite sequences of P/O-positions) of $\hat{\Gamma} \models \hat{U}_0 : \hat{\tau}$ and finite P-views in $\langle \hat{U}_0 \rangle$ that is ground-type reflecting and summarising.

Let $\hat{U}_0 = \lambda x_1 \dots x_l. \hat{T}$. Then the initial P-position is

$$(\Gamma, x_1 : \beta_1, \dots, x_l : \beta_l \models T : q_0)$$

which corresponds to the P-view consisting of the initial move of the effect arena.

Suppose the play $s \cdot u$ corresponds to the P-view $p \cdot m$. We consider, by a case analysis, the correspondence between the respective extensions $s \cdot u \cdot v$ and $p \cdot m \cdot n$.

- Suppose $u \mapsto v$ is the following Type-1 transition

$$\begin{aligned} (\hat{\Gamma} \models (p_j x) (\prod_{i \in I_1} \hat{U}_i^1) \dots (\prod_{i \in I_K} U_i^K) : q) \\ \xrightarrow{\varepsilon} (\hat{\Gamma} \models (\prod_{i \in I_1} U_i^1)_l : (\alpha^1)_l) \end{aligned}$$

where $p_j(\hat{\Gamma}(x)) = \langle \alpha^1 \rightarrow \dots \rightarrow \alpha^k \rightarrow q; e \rangle$ and $\varepsilon \preceq e$; further $\hat{\Gamma} = d \setminus \hat{\Gamma}', x_1 : \beta_1, \dots, x_l : \beta_l$ (as given by the transition preceding $u \mapsto v$). The O-move m^O of the effect arena that corresponds to u is $\langle \beta_1 \rightarrow \dots \rightarrow \beta_l \rightarrow q; d \rangle$, which lies over the lambda-move $\lambda x_1 : \beta_1 \dots x_m : \beta_l$ of the subtree $\lambda x_1 \dots x_m. (p_j x) (\prod_{i \in I_1} \hat{U}_i^1) \dots (\prod_{i \in I_k} \hat{U}_i^k)$ of \hat{U}_0 (here we appeal to the correspondence between Böhm trees and innocent strategies as set out in Appendix B-B).

Suppose the variable x (via binding $x : \prod_{i \in I} \langle \tau_i; g_i \rangle$) was introduced to the environment $\hat{\Gamma}_1$ of the P-position $u_1 = (\hat{\Gamma}_1 \models \hat{T}_1 : q_1)$ in s ; by the induction hypothesis, let $m_1^O = \langle \dots \rightarrow \prod_{i \in I} \langle \tau_i; g_i \rangle \rightarrow \dots \rightarrow q'; g \rangle$ be the O-move in p that corresponds to u_1 . Then the P-move n^P that corresponds to the O-node v is $\langle \tau_j; g_j \rangle$, which lies over the variable-move x of the subtree $(p_j x) (\prod_{i \in I_1} \hat{U}_i^1) \dots (\prod_{i \in I_k} \hat{U}_i^k)$ of \hat{U}_0 . Notice that n^P is explicitly justified by the O-move m_1^O in p .

We need to show that the effect of the P-move $\langle \tau_j; g_j \rangle$ is a correct summary. Suppose the effects of the O-moves of the P-view p after m_1^O are e_1, \dots, e_n respectively. By the induction hypothesis, they correspond to the respective effects of the Type-2 transitions of s after the P-node u_1 . Thanks to the cumulative effects of the Type-2 transitions, we have $\hat{\Gamma}(x) = e_n \setminus \dots \setminus e_1 \setminus \hat{\Gamma}_1(x) = \prod_{i \in I} \langle \tau_i; e_n \setminus \dots \setminus e_1 \setminus g_i \rangle$, and so, $e = e_n \setminus \dots \setminus e_1 \setminus g_j$. Since $\varepsilon \preceq e$ by assumption, we have $\varepsilon \preceq e_n \setminus \dots \setminus e_1 \setminus g_j$. Since each $e_i \setminus -$ is a left-residual, we have $e_1 \circ \dots \circ e_n \preceq g_j$ as desired.

It is easy to prove that the P-move reflects the gourd-type.

- Suppose $u \mapsto v$ is the following Type-2 transition

$$\begin{aligned} & (\hat{\Gamma} \models (\prod_{i \in I_k} \hat{U}_i^k) : (\hat{\alpha}^k)_k) \\ & \xrightarrow{e} ((e \setminus \hat{\Gamma}), x_1 : \hat{\beta}_1, \dots, x_l : \hat{\beta}_l \models \hat{T} : q) \end{aligned}$$

where, for some k and $i \in I_k$, $\hat{U}_i^k = \lambda x_1 \dots \lambda x_l. \hat{T}$ and $p_i(\alpha_k) = \langle \beta_1 \rightarrow \dots \rightarrow \beta_l \rightarrow q; e \rangle$. Suppose the O-node u corresponds to the P-move $m^P = \langle \alpha_1 \rightarrow \dots \rightarrow \alpha_k \rightarrow q_1; f \rangle$ which lies over the variable x of the subtree $(p_j x) (\prod_{i \in I_k} \hat{U}_i^1) \dots (\prod_{i \in I_k} \hat{U}_i^k)$. Then the P-position v corresponds to the O-move $n^O = \langle \beta_1 \rightarrow \dots \rightarrow \beta_l \rightarrow q; e \rangle$, which lies over the lambda-move $\lambda x_1 \dots x_l$ of the subtree $\hat{U}_i^k = \lambda x_1 \dots x_l. \hat{T}$ of \hat{U}_0 . Notice that n^O is explicitly justified by the preceding move m^P , extending the P-view $p \cdot m^P$.

- In the case of Type-3 transitions

$$(\Gamma \models \perp : q) \xrightarrow{\varepsilon} (\Gamma \models \perp : q)$$

suppose the P-node of the last Type-2 transition in $s \cdot u$ is $(e \setminus \hat{\Gamma}, x_1 : \hat{\beta}_1, \dots, x_l : \hat{\beta}_l \models \hat{T} : q)$, and let $m^O = \lambda x_1 \dots x_l$ be the O-move of the arena-with-effects corresponding to it. Then the Type-3 transition corresponds to the P-view $p \cdot m^O \cdot \perp$ which extends the P-view $p \cdot m^O$.

It remains to observe that the (unique) P-strategy of $\hat{\Gamma} \models \hat{U}_0 : \hat{\tau}$ is winning if and only if the corresponding innocent strategy $\langle \hat{U}_2 \rangle$ satisfies Ω . This is obvious because the above bijection of plays maps

$$v_2 \xrightarrow{\varepsilon} v_3 \xrightarrow{e_3} v_4 \xrightarrow{\varepsilon} v_5 \xrightarrow{e_5} \dots$$

to a P-view

$$m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 \cdot \dots$$

such that $E(m_{2k+3}) = e_{2k+3}$ (notice that $E(m_1) = \varepsilon$). ■

APPENDIX F

UNSTABLE WINNING CONDITIONS

This section discuss the case in which Ω is not stable. Fix an ω -monoid (\mathbb{E}, \mathbb{F}) equipped with left-residuals.

A. The set of all lower closed set

We write \mathbb{I} for the set of all lower-closed subsets of \mathbb{F} (stands for order Ideals). It is ordered by set-inclusion. An element of \mathbb{I} is written as Ω . The positive action to \mathbb{I} is defined by:

$$e \otimes \Omega := \{e \otimes f \mid f \in \Omega\}^{\preceq},$$

where A^{\preceq} is the lower closure of A . It is easy to see that $e \otimes (e' \otimes \Omega) = (e \circ e') \otimes \Omega$.

Negative action is defined by:

$$e \parallel \Omega := \{f \in \mathbb{F} \mid e \otimes f \in \Omega\}.$$

Another (perhaps more suggestive) definition of the negative action is:

$$e \parallel \Omega := \{(e \setminus e') \otimes f \mid e' \otimes f \in \Omega\}^{\preceq}.$$

Proposition 74. *The two definitions of the negative action are equivalent.*

Proof: Suppose that $f \in e \parallel \Omega$ in the first definition. Then $e \otimes f \in \Omega$. So $(e \setminus e) \otimes f \in e \parallel \Omega$ in the second definition. Note that $f = \varepsilon \otimes f \preceq (e \setminus e) \otimes f$ by $\varepsilon \preceq e \setminus e$ and the monotonicity of the action. Since $e \parallel \Omega$ in the second definition is lower-closed by its definition, we have $f \in e \parallel \Omega$ in the definition, as desired.

Suppose that $f \in e \parallel \Omega$ in the second definition. Then $f \preceq (e \setminus e') \otimes f'$ and $e' \otimes f' \in \Omega$ for some $e' \in \mathbb{E}$ and $f' \in \mathbb{F}$. Now we have

$$\begin{aligned} e \otimes f &= e \otimes ((e \setminus e') \otimes f') \\ &= (e \circ (e \setminus e')) \otimes f' \\ &\preceq e' \otimes f' \end{aligned}$$

since $e \circ (e \setminus e') \preceq e'$. Because Ω is lower-closed and $e' \otimes f' \in \Omega$, we have $e \otimes f \in \Omega$. So $f \in e \parallel \Omega$ in the first definition. ■

Stability can be characterised by using positive and negative actions: Ω is stable just if

- $e \otimes \Omega \subseteq \Omega$ for every $e \in \mathbb{E}$, and
- $e \parallel \Omega \subseteq \Omega$ for every $e \in \mathbb{E}$.

Lemma 75. $e \otimes \Omega_1 \subseteq \Omega_2$ if and only if $\Omega_1 \subseteq e \parallel \Omega_2$.

Proof: (\Rightarrow) Assume $e \otimes \Omega_1 \subseteq \Omega_2$ and $f \in \Omega_1$. Then $e \otimes f \in \Omega_2$. So by the definition of the negative action, $f \in e \parallel \Omega_2$.

(\Leftarrow) Assume $\Omega_1 \subseteq e \parallel \Omega_2$ and $f \in e \otimes \Omega_1$. Then $f = e \otimes f'$ for some $f' \in \Omega_1$. Then $f' \in e \parallel \Omega_2$. By the definition of the negative action, we know that $e \otimes f' \in \Omega_2$, as required. ■

$\Omega \in \mathbb{I}$ is stable if and only if $e \otimes \Omega \subseteq \Omega$ and $e \parallel \Omega \subseteq \Omega$ for every $e \in \mathbb{E}$.

B. Type-checking games

There is no change in the definition of the game graph of type-checking games. A game is now determined by a pair of a position and a winning condition $\Omega \in \mathbb{I}$. We write $\Gamma \models^\Omega U : \alpha$ for the game starting from $\Gamma \models U : \alpha$ whose winning condition is Ω .

In the case that Ω is stable, we have

$$\Gamma \models^\Omega U : e \otimes \alpha \text{ iff } e \Vdash \Gamma \models^\Omega U : \alpha.$$

This proposition is used in the proof of Theorem 18 to prove soundness of the effect action rule. This proposition is no longer true if Ω is not stable. Instead, we have:

$$\Gamma \models^{e \otimes \Omega} U : e \otimes \alpha \text{ iff } e \Vdash \Gamma \models^\Omega U : \alpha.$$

As we will see, the category $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$ can be defined even if Ω is not an idea of the monoid action. So by the same argument as in Section VI-B, we have the following compositinality result.

Theorem 76.

- $\widehat{V}_1 \blacktriangleright (\Gamma \models^\Omega U_1 : \alpha \rightarrow \tau)$ and $\widehat{V}_2 \blacktriangleright (\Gamma \models^\Omega U_2 : \alpha)$ implies $(\widehat{V}_1 @ \widehat{V}_2) \blacktriangleright (\Gamma \models^\Omega U_1 @ U_2 : \tau)$.
- $\widehat{W} \blacktriangleright (\Gamma \models^\Omega U_1 @ U_2 : \tau)$ implies $\widehat{V}_1 \blacktriangle (\Gamma \models^\Omega U_1 : \alpha \rightarrow \tau)$ and $\widehat{V}_2 \blacktriangleright (\Gamma \models^\Omega U_2 : \alpha)$ and $\widehat{W} = \widehat{V}_1 @ \widehat{V}_2$ for some α , \widehat{V}_1 and \widehat{V}_2 .

C. Type system

The sets of types and type environments are the same as in Section III. A type judgement is explicitly annotated by Ω , like $\Gamma \vdash^\Omega M : \tau$. The typing rules are listed as follows. Notice that the positive action rule changes the winning condition Ω .

$$\frac{\langle \tau; e \rangle = \mathbf{p}_i(\Gamma(x)) \text{ for some } i \text{ and } e \succeq \varepsilon}{\Gamma \vdash^\Omega x : \tau} \quad \frac{\Gamma, x : \alpha \vdash^\Omega M : \tau}{\Gamma \vdash^\Omega \lambda x. M : \alpha \rightarrow \tau}$$

$$\frac{\Gamma \vdash^\Omega M : \alpha \rightarrow \tau \quad \Gamma \vdash^\Omega N : \alpha}{\Gamma \vdash^\Omega M N : \tau} \quad \frac{\Gamma' \preceq \Gamma \quad \Gamma \vdash^\Omega M : \tau \quad \tau \preceq \tau'}{\Gamma' \vdash^\Omega M : \tau'}$$

$$\frac{\Gamma \vdash^\Omega M : \tau}{e \otimes \Gamma \vdash^{e \otimes \Omega} M : \langle \tau; e \rangle} \quad \frac{\forall i \in I. \Gamma \vdash M : \langle \tau_i; e_i \rangle}{\Gamma \vdash M : \bigwedge_{i \in I} \langle \tau_i; e_i \rangle} \quad \frac{\models \text{BT}(\mathbf{Y}) : \tau}{\Gamma \vdash^\Omega \mathbf{Y} : \tau}$$

Transfer theorem for the extended type system can be proved by the same way as that for the plain type system.

Theorem 77. $\Gamma \models^\Omega M : \alpha$ iff $\Gamma \vdash^\Omega \text{BT}(M) : \alpha$.

Assume that \mathbb{E} and \mathbb{F} are finite. Then \mathbb{I} is finite. Hence, for every M , the judgements for M is finite. So inductive enumeration of all derivable judgements provides a decision procedure.

Theorem 78. If (\mathbb{E}, \mathbb{F}) is finite, then $\Gamma \vdash^\Omega M : \tau$ is decidable.

By the same way as in the plain type system, the representation of a finite derivation can be defined. The next result can be proved by the same way as in the plain system.

Theorem 79. $D \triangleright (\Gamma \vdash^\Omega M : \tau)$ implies $\text{BT}(D) \blacktriangleright (\Gamma \models^\Omega \text{BT}(M) : \tau)$.

D. Game model

Notice that the definition of the game model in Section V does not rely on the fact that Ω is stable. Hence, for every $\Omega \in \mathbb{I}$, we have an arena game model $\mathcal{G}_{(\mathbb{E}, \mathbb{F}, \Omega)}$. We abbreviate it to \mathcal{G}_Ω . The two-level construction works well for this setting.

The effect action is no longer be an endofunctor if Ω is not stable. Instead, it defined a functor

$$e \otimes - : \mathcal{G}_\Omega \longrightarrow \mathcal{G}_{e \otimes \Omega}.$$

Similarly, the negative action is a functor

$$e \Vdash - : \mathcal{G}_\Omega \longrightarrow \mathcal{G}_{e \Vdash \Omega}.$$

Assume an effect arena A and $\Omega_1, \Omega_2 \in \mathbb{I}$. Let \mathfrak{s} be a strategy of $|A|$. If \mathfrak{s} satisfies Ω_1 and $\Omega_1 \subseteq \Omega_2$, then \mathfrak{s} also satisfies Ω_2 . This means that there is an inclusion $\mathcal{G}_{\Omega_1} \longrightarrow \mathcal{G}_{\Omega_2}$ if $\Omega_1 \subseteq \Omega_2$. This defines a functor $\mathbb{I} \longrightarrow \mathbf{Cat}$, where \mathbf{Cat} is the category of categories.

APPENDIX G

A REMARK ON TRANSFER THEOREM

A *order- n transfer theorem* for a set \mathcal{C} of properties asserts the existence of an effective transformation T on \mathcal{C} such that for every order- n tree-generator M and property φ , $\text{BT}(M)$ satisfies property φ iff M (or some representation thereof) satisfies $T(\varphi)$. Thus, such a theorem says that the \mathcal{C} -definable properties of order- n $\lambda\mathbf{Y}$ -definable Böhm trees are effectively the \mathcal{C} -definable properties of the term-generators themselves.

More formally an *order- n transfer theorem* for a simply-kinded language \mathcal{L} w.r.t. a set \mathcal{C} of properties is an assertion of the following kind. Let φ range over \mathcal{C} .

There is an effective transformation on \mathcal{C} , $T_{\mathcal{C}}$, such that for every order- n term-in-context of \mathcal{L} (i.e. kinding judgement) $\Gamma \vdash M :: \kappa$, and for every φ in \mathcal{C} , we have:

$$\Gamma \models \text{BT}(M) : \varphi \iff \Gamma \models' G_M : T_{\mathcal{C}}(\varphi)$$

where G_M is a suitable representation (e.g. the abstract syntax graph) of the term M .

Sylvain and Walukiewicz [20] have proved an order-2 transfer theorem where \mathcal{L} is the infinitary $\lambda\mathbf{Y}$ -calculus (definable using a bounded set of kinds) and \mathcal{C} is the set of MSO formulas (or alternating parity automata).

Our result (Theorem 18) is an order- n transfer theorem where \mathcal{L} is the $\lambda\mathbf{Y}$ -calculus, and \mathcal{C} is the set of intersection types; the transformation $T_{\mathcal{C}}$ is the identity operation and $G_M = M$.

A. Typing is MSO-definable

Let us fix a *finite* set of (kinded) variables as in [20], say

$$\mathcal{X} = \{z_1 :: \iota_1, z_2 :: \iota_2, \dots, z_n :: \iota_n\},$$

and consider $\lambda\mathbf{Y}$ -terms with concrete variable names (i.e. terms $\lambda z_1. z_1$ and $\lambda z_2. z_2$ are considered as different terms, even if $\iota_1 = \iota_2$). We further assume a finite set of kinds \mathcal{K}

and requires that all kinds in a term should be in \mathcal{K} . In this setting, a term can be considered as a tree over the alphabet

$$\begin{aligned}\Sigma(\mathcal{X}, \mathcal{K}) = & \{x \mid x \in \mathcal{X}\} \\ & \cup \{\lambda x \mid x \in \mathcal{X}\} \\ & \cup \{\textcircled{\lambda}_{\kappa_1, \kappa_2} \mid \kappa_1 \rightarrow \kappa_2 \in \mathcal{K}\} \\ & \cup \{\mathbf{Y}_\kappa \mid (\kappa \rightarrow \kappa) \rightarrow \kappa \in \mathcal{K}\},\end{aligned}$$

where x and \mathbf{Y}_κ are of arity 0 (i.e. they are leaves of a tree), λx is of arity 1 and $\textcircled{\lambda}_{\kappa_1, \kappa_2}$ is of arity 2. The first branch of $\textcircled{\lambda}_{\kappa_1, \kappa_2}$ is a tree representing a term of kind $\kappa_1 \rightarrow \kappa_2$ and the second branch if of kind κ_1 , and the tree rooted by $\textcircled{\lambda}_{\kappa_1, \kappa_2}$ represents a term of kind κ_2 . Let $\mathcal{T}(\mathcal{X}, \mathcal{K})$ be the set of $\lambda\mathbf{Y}$ -terms in this signature.

As observed in [20], $\mathcal{T}(\mathcal{X}, \mathcal{K})$ is not closed under reduction since capture-avoiding substitution may require a fresh variable. The set $\mathcal{T}(\mathcal{X}, \mathcal{K})$ is a proper subset of that considered in [20], since (i) we only focus on the *finite* $\lambda\mathbf{Y}$ -terms, whereas [20] studies the *infinite* $\lambda\mathbf{Y}$ -calculus, and (ii) [20] assumes *infinite* supply of \mathbf{Y} -variables, variables introduced by recursion, but we do not distinguish them from usual variables (as our syntax suggests).

This subsection proves the following result. Note that, since $M \in \mathcal{T}(\mathcal{X}, \mathcal{K})$ is just a tree in this setting, we have an established notion of MSO formulas and the satisfaction relation.

Proposition 80. *Assume a finite set of kinded variables \mathcal{X} and a finite set of kinds \mathcal{K} . Given a type δ , the set $\{M \in \mathcal{T}(\mathcal{X}, \mathcal{K}) \mid \vdash M : \delta\}$ is effectively MSO-definable, i.e., one can effectively compute a MSO-formula φ_δ such that $\vdash M : \delta$ coincides with $M \models \varphi_\delta$.*

Since M is a tree over a finite ranked alphabet, we can appeal to (effective) equivalence between MSOL and alternating parity tree automata. So our goal is to construct an alternating parity tree automata \mathcal{A}_δ such that $\vdash M : \delta \iff M \in \mathcal{L}(\mathcal{A}_\delta)$, where $\mathcal{L}(\mathcal{A})$ is the language recognised by \mathcal{A} .

Assume a finite set Q of ground types and a finite and stable winning condition $(\mathbb{E}, \mathbb{F}, \Omega)$ as in Section IV. In this subsection, we consider types modulo \approx . Therefore, for every $\kappa \in \mathcal{K}$, the set $\{\tau \mid \tau :: \kappa\}$ is viewed as a finite set. We regard $\mathcal{X} = \{z_1 :: \iota_1, \dots, z_n :: \iota_n\}$ as a kind environment. Then the set of types environments $\{\Gamma \mid \Gamma :: \mathcal{X}\}$ (modulo \approx) is also a finite set.

Let $\Gamma = (z_1 : \alpha_1, \dots, z_n : \alpha_n) :: \mathcal{X}$, $x = z_i \in \mathcal{X}$ and $\alpha'_i :: \iota_i$. We write $\Gamma[z_i \mapsto \alpha]$ for the type environment

$$z_1 : \alpha_1, \dots, z_{i-1} : \alpha_{i-1}, z_i : \alpha'_i, z_{i+1} : \alpha_{i+1}, \dots, z_n : \alpha_n.$$

The family of alternating parity tree automata \mathcal{A}_δ parametrised on δ share the same set of states and the same transition relation. In other words, they differ only on the initial states. In the following, we define an alternating parity tree automaton \mathcal{A} without the initial state.

The set of states is:

$$\mathcal{J} := \bigcup_{\kappa \in \mathcal{K}} \{(\Gamma, \tau) \mid \Gamma :: \mathcal{X} \text{ and } \tau :: \kappa\}.$$

We write $\Gamma \vdash \square : \tau$ for the state (Γ, τ) . Note that \mathcal{J} is a finite set. We write $\mathcal{A}(\Gamma \vdash \square : \tau)$ for the alternating parity tree automaton with the initial state $\Gamma \vdash \square : \tau$ that has the same states and transition relation is \mathcal{A} .

For the notational convenience, we regard $\alpha :: \kappa$ as a subset of $\{\tau \mid \tau :: \kappa\} \times \mathbb{E}$ and write $\langle \tau; e \rangle \in \alpha$ if $\mathbf{p}_k(\alpha) = \langle \tau; e \rangle$ for some k .

We define the transition relation $R \subseteq \mathcal{P}(\Sigma \times \mathcal{J} \times \mathcal{P}(\mathcal{J} \times \{1, 2\}))$, where $\mathcal{P}(A)$ means the powerset of A and $\{1, 2\}$ is the set of directions.

- $(z_i, (\Gamma \vdash \square : \tau), \emptyset) \in R$ iff $\Gamma(z_i) \preceq \langle \tau; \varepsilon \rangle$.
- $(\mathbf{Y}_\kappa, (\Gamma \vdash \square : \tau), \emptyset) \in R$ iff $\vdash \mathbf{Y} : \tau$.
- $(\lambda z_i, (\Gamma \vdash \square : \alpha \rightarrow \tau), \{(\Gamma[z_i \mapsto \alpha] \vdash \square : \tau, 1)\}) \in R$ for every z_i, Γ, α and τ .
- $(\textcircled{\lambda}_{\kappa_1, \kappa_2}, (\Gamma \vdash \square : \tau), \{(\Gamma \vdash \square : \alpha \rightarrow \tau, 1)\} \cup \{(e \parallel \Gamma \vdash \square : \sigma, 2) \mid \langle \sigma; e \rangle \in \alpha\}) \in R$ for every Γ, τ , and $\alpha :: \kappa_1$.

Lemma 81. *Let $M \in \mathcal{T}(\mathcal{X}, \mathcal{K})$. Then $\Gamma \vdash M : \tau$ if and only if $M \in \mathcal{L}(\mathcal{A}(\Gamma \vdash \square : \tau))$.*

Proof: By induction on the structure of M , using Lemma 56. \blacksquare

By the above lemma, we know that $\{M \in \mathcal{T}(\mathcal{X}, \mathcal{K}) \mid \vdash M : \tau\}$ is the language accepted by $\mathcal{A}_\delta := \mathcal{A}(\Gamma \vdash \square : \delta)$. By effective equivalence between alternating parity tree automata and MSO-formulas, we have Proposition 80.