Game Semantics for Nominal Exceptions*

Andrzej S. Murawski[1] and Nikos Tzevelekos[2]

[1] DIMAP and Department of Computer Science, University of Warwick
[2] School of Electronic Engineering and Computer Science, Queen Mary University of London

Abstract. We present a fully abstract denotational model for a higher-order programming language combining call-by-value evaluation and local exceptions. The model is built using nominal game semantics and is the first one to achieve both effective presentability and freedom from "bad exception" constructs.

1 Introduction

Exceptions are a standard programming effect for raising and handling eccentric program behaviour, and more generally for manipulating the flow of control. They are a key feature, for example, of ML, Java and C++. The raising of an exception forces a program to escape out of its context and to the nearest applicable exception-handler. Thus, exceptions provide a means of overriding nested behaviour of pure functional programs. The mechanism that allows handlers to recognise the exceptions to be handled usually relies on the use of names. In the paper we shall focus on modelling such *nominal* exceptions.

The difficulties in modelling (even soundly) nominal exceptions stem from the combination of name-locality and name-mobility with non-local control flow. In particular, traditional approaches do not cope with locality and examine global exceptions only via the exception monad [9]. On the other hand, existing game models [5] rely on Reynolds' principle of modelling references as objects with read/write methods [13], extended to the modelling of exceptions as objects with raise/handle methods. The main defect of this principle is that, in order to achieve full abstraction, "bad" constructors have to be included in the syntax, which means that the language examined will include "bad exceptions", which are terms of exception type that do not correspond to genuine exceptions, but rather to couplings of arbitrary raise/handle methods. These constructs, while solving the full-abstraction problem, distance the languages from the programming features they were set out to capture; in particular, term-equivalence is not conservative with respect to bad constructors. For example, "handle x in (raise x) with skip" is not equivalent to "skip".

Nominal game semantics advocates a departure from Reynolds' modelling rule and stipulates that "nameful" types be modelled by names rather than objects. Nominal games were introduced in [1] and [6] in order to provide the first fully abstract models of the ν-calculus and its extension with pointers (i.e. storable names) respectively. They constitute a 'nominalised' version of game semantics, in which names may appear in

* Research funded by the Engineering and Physical Sciences Research Council (EP/J019577/1) and a Royal Academy of Engineering Research Fellowship (Tzevelekos).

```
class MyExn extends Exception {}

public class Trap {
    public static void main(String [] argv)
    throws Exception {
        Exception e1 = new MyExn();
        Exception e2 = new MyExn();
        try { foo(e1); }
        catch ( Exception x ) {
            System.out.printf("%b, %b",
                              x==e1, x==e2);
        }
    }
    static void foo(Exception e)
    throws Exception {
            throw(e);
    }
}
```

```
exception MyExn
let e1 = MyExn
let e2 = MyExn
let foo(x) = raise x;;

try foo(e1) with x -> (x==e1, x==e2)
```

```
fun new_exn() =
    let exception MyExn
        fun eq(x) = case x of
            MyExn => true
            | _ => false
    in  (MyExn, eq) end
val e1 = new_exn()
val e2 = new_exn()
fun foo(x) = raise x;

foo(#1 e1) handle x => (#2 e1 x, #2 e2 x)
```

Fig. 1. Code samples. Clockwise, from upper-left corner: Java, OCaML and SML. In the Java example, the `catch` clause in method `main` is able to trap the exception `e1` raised by `foo`, extract its name and pass it to `x`. As a result, the program prints `true`, `false`. In OCaML, the same effect is achieved by pattern matching the handled expression. We instigate analogous behaviour in SML, using the generativity of the `exception` constructor to produce local exceptions.

plays as atomic moves. Put differently, they are ordinary games constructed within the universe of *nominal sets* [2]. A first attempt to model exceptions using nominal games was made in [15]. However, the close reliance on the monadic approach led to a model which was too intensional to yield an explicit characterisation of contextual equivalence and the full abstraction result had to be obtained through the intrinsic quotient construction ([15, Proposition 5.23]). The development of a direct model was left as a major challenge for future work in [15]. In the present paper, we meet that challenge by producing two fully abstract and effectively presentable models for higher-order languages with references and exceptions. The fact that our models are not quotiented yields a direct approach to proving program equivalences, with scope for future automation (cf. [11]). In particular, we prove new non-trivial equivalences (cf. Example 28).

We consider two kinds of exception-handling mechanisms, in Sections 2-4 and 5-6 respectively. In the first one, illustrated by the code samples in Figure 1, the handler is given explicit access to the exception names that it encounters. Another, less invasive approach, is to require the handler to specify which exception is to be intercepted, under the assumption that all the others will be propagated by default. This approach respects privacy of exceptions in that no handler may react to a freshly generated exception. The latter kind of exceptions turns out to lead to a slightly more complicated game model.[3]

At the technical level our full abstraction results are obtained by modelling the exception type by an arena whose moves belong to a countable set of *names*. Additionally, players are allowed use moves of the form $e!$ (where e is an exception name) as answers to *arbitrary* questions. Uses of $e!$ can be taken to correspond to raising an exception. These two relatively simple enrichments, along with standard game semantic conditions such as alternation and well-bracketing, already give rise to a fully abstract model of the

[3] This is a common pattern in game semantics: fewer conditions are needed to describe models of richer languages, because the corresponding interactions are less constrained.

$$\frac{}{\mathsf{u}, \Gamma \vdash (), \Omega : \mathsf{unit}} \qquad \frac{i \in \mathbb{Z}}{\mathsf{u}, \Gamma \vdash i : \mathsf{int}} \qquad \frac{l \in \mathsf{u} \cap \mathcal{L}_\beta}{\mathsf{u}, \Gamma \vdash l : \mathsf{ref}\,\beta} \qquad \frac{e \in \mathsf{u} \cap \mathcal{E}}{\mathsf{u}, \Gamma \vdash e : \mathsf{exn}}$$

$$\frac{\mathsf{u}, \Gamma \vdash M : \mathsf{int} \quad \mathsf{u}, \Gamma \vdash N_0, N_1 : \theta}{\mathsf{u}, \Gamma \vdash \mathsf{if0}\, M \,\mathsf{then}\, N_0 \,\mathsf{else}\, N_1 : \theta} \qquad \frac{\mathsf{u}, \Gamma \vdash M, N : \mathsf{int}}{\mathsf{u}, \Gamma \vdash M \oplus N : \mathsf{int}} \qquad \frac{\mathsf{u}, \Gamma \vdash M, N : \mathsf{exn}, \mathsf{ref}\,\beta}{\mathsf{u}, \Gamma \vdash M = N : \mathsf{int}}$$

$$\frac{(x : \theta) \in \Gamma}{\mathsf{u}, \Gamma \vdash x : \theta} \qquad \frac{\mathsf{u}, \Gamma, x : \theta \vdash M : \theta'}{\mathsf{u}, \Gamma \vdash \lambda x^{\theta}.M : \theta \to \theta'} \qquad \frac{\mathsf{u}, \Gamma \vdash M : \theta \to \theta' \quad \mathsf{u}, \Gamma \vdash N : \theta}{\mathsf{u}, \Gamma \vdash MN : \theta'}$$

$$\frac{\mathsf{u}, \Gamma \vdash M : \beta}{\mathsf{u}, \Gamma \vdash \mathsf{ref}_\beta(M) : \mathsf{ref}\,\beta} \qquad \frac{\mathsf{u}, \Gamma \vdash M : \mathsf{ref}\,\beta}{\mathsf{u}, \Gamma \vdash !M : \beta} \qquad \frac{\mathsf{u}, \Gamma \vdash M : \mathsf{ref}\,\beta \quad \mathsf{u}, \Gamma \vdash N : \beta}{\mathsf{u}, \Gamma \vdash M := N : \mathsf{unit}}$$

$$\frac{}{\mathsf{u}, \Gamma \vdash \mathsf{exn}() : \mathsf{exn}} \qquad \frac{\mathsf{u}, \Gamma \vdash M : \mathsf{exn}}{\mathsf{u}, \Gamma \vdash \mathsf{raise}\, M : \theta} \qquad \frac{\mathsf{u}, \Gamma \vdash M : \theta \quad \mathsf{u}, \Gamma, x : \mathsf{exn} \vdash N : \theta}{\mathsf{u}, \Gamma \vdash M \,\mathsf{handle}\, x \,\mathsf{=>}\, N : \theta}$$

Fig. 2. Syntax of ExnML.

first kind of exceptions, i.e. handlers have direct access to exception names. To model the other type of handlers, we identify a compositional subclass of strategies that must propagate any exceptions unless they were revealed to the program by the environment. In both cases, we obtain an explicit characterisation of contextual equivalence through the induced sets of *complete* plays, ones in which all questions are answered. In the setting where handling of private exceptions is not available, the latter set needs to be appropriately trimmed, so as to reflect the handling restrictions on environments.

2 A language with local exceptions and ground references

We introduce the language ExnML, which is a fragment of ML with full ground references[4] augmented with nominal exceptions. Its types θ are generated according to the following grammar.

$$\theta \ ::= \ \beta \mid \theta \to \theta \qquad \beta \ ::= \ \mathsf{unit} \mid \mathsf{int} \mid \mathsf{exn} \mid \mathsf{ref}\,\beta$$

Note that reference types are available for each type of the shape β, including the exception type (full ground storage). We assume disjoint denumerable sets $\mathcal{L}$ and $\mathcal{E}$ of *locations* and *exceptions* respectively, such that $\mathcal{L} = \biguplus_\beta \mathcal{L}_\beta$. We range over location names by l, l' and over exception names by e, e'. Terms are typed in contexts (u, Γ), where u a finite subset of $\mathcal{L} \cup \mathcal{E}$ and Γ is a variable context. Moreover, we assume a fixed set of binary integer operators ranged over by $\oplus$. The terms of the language are given by the following grammar (all $i \in \mathbb{Z}$), while its typing rules are in Figure 2.

$$\begin{aligned} M \ ::= \ & () \mid \Omega \mid i \mid l \mid e \mid x \mid \lambda x^{\theta}.M \mid MM \mid \mathsf{if0}\, M \,\mathsf{then}\, M \,\mathsf{else}\, M \mid M \oplus M \\ & \mid M = M \mid \mathsf{ref}_\beta(M) \mid !M \mid M := M \mid \mathsf{exn}() \mid \mathsf{raise}\, M \mid M \,\mathsf{handle}\, x \,\mathsf{=>}\, M \end{aligned}$$

We shall write $\Gamma \vdash M : \theta$ iff $\emptyset, \Gamma \vdash M : \theta$ can be derived using the rules of Figure 2. Similarly, $\vdash M : \theta$ is shorthand for $\emptyset, \emptyset \vdash M : \theta$. In what follows, we write $M; N$

[4] Elements of all ground types are storable.We omit higher-order references in order not to complicate the exposition. The game model of higher-order references from [10] can be extended to exceptions by following Section 3.

$$\Sigma, (\lambda x.M)v \longrightarrow \Sigma, M[v/x] \qquad \Sigma, \mathsf{if0}\ 0\ \mathsf{then}\ N_0\ \mathsf{else}\ N_1 \longrightarrow \Sigma, N_0$$

$$\Sigma, i_1 \oplus i_2 \longrightarrow \Sigma, (i_1 \oplus i_2) \qquad \Sigma, \mathsf{if0}\ i\ \mathsf{then}\ N_0\ \mathsf{else}\ N_1 \longrightarrow \Sigma, N_1 \ \ (i \neq 0)$$

$$\Sigma, !l \longrightarrow \Sigma, s(l) \qquad \Sigma, \mathsf{ref}_\gamma(v) \longrightarrow \Sigma[l \mapsto v], l \ \ (l \notin \mathsf{dom}(\Sigma_1))$$

$$\Sigma, l := v \longrightarrow \Sigma[l \mapsto v], () \qquad \Sigma, \mathsf{exn}() \longrightarrow \Sigma \cup \{e\}, e \ \ (e \notin \Sigma_2)$$

$$\Sigma, e = e \longrightarrow \Sigma, 1 \qquad \Sigma, e = e' \longrightarrow \Sigma, 0 \ \ (e \neq e')$$

$$\Sigma, l = l \longrightarrow \Sigma, 1 \qquad \Sigma, l = l' \longrightarrow \Sigma, 0 \ \ (l \neq l')$$

$$\Sigma, v\ \mathsf{handle}\ x\ \mathsf{=>}\ N \longrightarrow \Sigma, v \qquad \Sigma, (\mathsf{raise}\ e)\ \mathsf{handle}\ x\ \mathsf{=>}\ N \longrightarrow \Sigma, N[e/x]$$

$$\Sigma, E_{\neg H}[\mathsf{raise}\ e] \longrightarrow \Sigma, \mathsf{raise}\ e \qquad \frac{\Sigma, M \longrightarrow \Sigma', M'}{\Sigma, E[M] \longrightarrow \Sigma', E[M']}$$

Fig. 3. Small-step operational semantics of ExnML.

for the term $(\lambda z^\theta.N)M$, where z does not occur in N and θ matches the type of M. let $x = M$ in N will stand for $(\lambda x^\theta.N)M$ in general. *Value terms* v, are given by:

$$v \ ::= \ () \mid i \mid e \mid l \mid \lambda x^\theta.M$$

The operational semantics of the language utilises finite *stores*, which record generated exceptions and assign to locations atomic values of compatible type:

$$Sto = \{s : \mathcal{L} \rightharpoonup_{\text{fin}} Val \ \mid \ l \in \mathsf{dom}(s) \cap \mathcal{L}_\beta \implies s(l) \in Val_\beta\} \times \mathcal{P}_{\text{fin}}(\mathcal{E}),$$

where $Val = Val_{\mathsf{unit}} \cup Val_{\mathsf{int}} \cup Val_{\mathsf{exn}} \cup \biguplus_\beta Val_{\mathsf{ref}\beta}$, $Val_{\mathsf{unit}} = \{*\}$, $Val_{\mathsf{int}} = \mathbb{Z}$, $Val_{\mathsf{exn}} = \mathcal{E}$, $Val_{\mathsf{ref}\beta} = \mathcal{L}_\beta$. We range over Sto by Σ, T (and variants). Given $\Sigma \in \mathsf{Sto}$ we write Σ_1, Σ_2 to refer to its respective components. Stores must be *closed* in the following sense: for all $\Sigma \in \mathsf{Sto}$ and $l \in \mathsf{dom}(\Sigma_1)$,

$$(\Sigma_1(l) \in \mathcal{L} \implies \Sigma_1(l) \in \mathsf{dom}(\Sigma_1)) \wedge (\Sigma_1(l) \in \mathcal{E} \implies \Sigma_1(l) \in \Sigma_2).$$

Finally, we let *evaluation contexts* be given by the syntax:

$$\begin{aligned} E ::= {} & [\_] \mid E\,N \mid (\lambda x.M)E \mid \mathsf{if0}\ E\ \mathsf{then}\ N_0\ \mathsf{else}\ N_1 \mid E \oplus N \mid i \oplus E \mid \mathsf{ref}_\gamma(E) \mid E := N \\ & \mid l := E \mid !E \mid E = N \mid e = E \mid l = E \mid E\ \mathsf{handle}\ x\ \mathsf{=>}\ N \mid \mathsf{raise}\ E. \end{aligned}$$

We write $E_{\neg H}$ for contexts E derived from the above grammar applying any of the rules apart from $E\ \mathsf{handle}\ x\ \mathsf{=>}\ N$. In Figure 3 we give a small-step reduction relation for terms in contexts from Sto. Given $\vdash M : \mathsf{unit}$ we write $M \Downarrow$ iff $(\emptyset, \emptyset), M \longrightarrow\!\!\!\!\!\rightarrow \Sigma, ()$ for some Σ.

Definition 1. We say that the term-in-context $\mathsf{u}, \Gamma \vdash M_1 : \theta$ ***approximates*** $\mathsf{u}, \Gamma \vdash M_2 : \theta$ (written $\mathsf{u}, \Gamma \vdash M_1 \sqsubseteq\!\!\!\!\sim\ M_2$) if $C[M_1] \Downarrow$ implies $C[M_2] \Downarrow$ for any context $C[-]$ such that $u, \emptyset \vdash C[M_1], C[M_2] : \mathsf{unit}$. Two terms-in-context are ***equivalent*** if one approximates the other (written $\mathsf{u}, \Gamma \vdash M_1 \cong M_2$).

Example 2. Take the terms $\vdash M_1, M_2 : \mathsf{unit} \to \mathsf{unit}$ to be respectively

$$M_1 \equiv \mathsf{let}\, y = \mathsf{exn}() \,\mathsf{in}\, \lambda x^{\mathsf{unit}}.\mathsf{raise}\, y \qquad \text{and} \qquad M_2 \equiv \lambda x^{\mathsf{unit}}.\mathsf{raise}\,(\mathsf{exn}()).$$

Their game semantics will contain the following plays respectively

$$q_0^{\Sigma_0} \star_0^{\Sigma_0} q^{\Sigma_0} e_1!^{\Sigma_1} q^{\Sigma_1} e_1!^{\Sigma_1} \cdots q^{\Sigma_1} e_1!^{\Sigma_1} \qquad q_0^{\Sigma_0} \star_0^{\Sigma_0} q^{\Sigma_0} e_1!^{\Sigma_1} \cdots q^{\Sigma_{k-1}} e_k!^{\Sigma_k}$$

where $\Sigma_i = (\emptyset, \{e_1, \cdots, e_i\})$. Handlers of ExnML can extract the name of an exception and remember it for future comparisons. Accordingly, we have $\vdash M_1 \ncong M_2$ (cf. Example 19).

3 Game semantics

We construct a game model for ExnML by extending the fully abstract model of *Ground ML* [11] so as to incorporate nominal exceptional effects. Let $\mathbb{A}$ be a countably infinite collection of *names*, corresponding to reference and exception names:

$$\mathbb{A} = \biguplus_{\beta} \mathbb{A}_\beta \uplus \mathbb{A}_{\mathrm{e}} \;\text{ where }\; \mathbb{A}_\beta = \mathcal{L}_\beta,\; \mathbb{A}_{\mathrm{e}} = \mathcal{E}.$$

We range over names with a, b, etc, and also l, e when we want to be specific about their kind. The model is constructed using mathematical objects (moves, plays, strategies) that will feature names drawn from $\mathbb{A}$. Although names underpin various elements of our model, their precise nature is irrelevant. Hence, all of our definitions preserve name-invariance, i.e. our objects are (strong) *nominal sets* [2, 16]. Note that we do not need the full power of the theory but mainly the basic notion of name-permutation. Here permutations are bijections $\pi : \mathbb{A} \to \mathbb{A}$ with finite support which respects the indexing of name-sets. For an element x belonging to a (nominal) set X, we write $\nu(x)$ for its name-support, i.e. the set of names occurring in x. Moreover, for any $x, y \in X$, we write $x \sim y$ if x and y are the same up to a permutation of $\mathbb{A}$. We let $[x] = \{y \in X \mid x \sim y\}$.

Our model is couched in the Honda-Yoshida style of modelling call-by-value computation [3]. Before we define what it means to play our games, let us introduce the auxiliary concept of an arena.

Definition 3. An ***arena*** $A = \langle M_A, I_A, \lambda_A, \vdash_A, M_{\mathrm{e}} \rangle$ is given by:

- a set M_A of ordinary moves, a set $I_A \subseteq M_A$ of initial moves,
- a labelling function $\lambda_A : M_A \cup M_{\mathrm{e}} \to \{O, P\} \times \{Q, A\}$,
- a justification relation $\vdash_A \subseteq M_A \times (M_A \setminus I_A) \cup M_{\mathrm{e}}$,
- and a fixed set $M_{\mathrm{e}} = \{e!_O \mid e \in \mathbb{A}_{\mathrm{e}}\} \cup \{e!_P \mid e \in \mathbb{A}_{\mathrm{e}}\}$ of exceptional moves;

such that $M_A \cap M_{\mathrm{e}} = \emptyset$ and, for all $m, m' \in M_A$ and $e \in \mathbb{A}_{\mathrm{e}}$:

- $m \in I_A \implies \lambda_A(m) = (P, A)$,
- $m \vdash_A m' \wedge \lambda_A^{QA}(m) = A \implies \lambda_A^{QA}(m') = Q$,
- $m \vdash_A m' \implies \lambda_A^{OP}(m) \neq \lambda_A^{OP}(m')$,
- $\lambda_A(e!_O) = (O, A) \wedge (\lambda_A(m) = (P, Q) \implies m \vdash_A e!_O)$,
- $\lambda_A(e!_P) = (P, A) \wedge (\lambda_A(m) = (O, Q) \implies m \vdash_A e!_P)$.

We write λ_A^{OP} (resp. λ_A^{QA}) for λ_A post-composed with the first (second) projection.

$$M_{A\otimes B} = (I_A \times I_B) \uplus \bar{I}_A \uplus \bar{I}_B \qquad M_{A+B} = M_A \uplus M_B$$
$$I_{A\otimes B} = I_A \times I_B \qquad I_{A+B} = I_A \cup I_B$$
$$\lambda_{A\otimes B} = [(\boldsymbol{i}_A, \boldsymbol{i}_B) \mapsto PA,\ \lambda_A \restriction \bar{I}_A,\ \lambda_B \restriction \bar{I}_B] \qquad \lambda_{A+B} = [\lambda_A, \lambda_B]$$
$$\vdash_{A\otimes B} = \{((\boldsymbol{i}_A, \boldsymbol{i}_B), m) \mid \boldsymbol{i}_A \vdash_A m \vee \boldsymbol{i}_B \vdash_B m\} \cup \bar{\vdash}_A \cup \bar{\vdash}_B \qquad \vdash_{A+B} = \vdash_A \cup \vdash_B$$

$$M_{A\Rightarrow B} = \{\star\} \uplus M_A \uplus M_B \qquad M_{A\to B} = M_A \uplus M_B$$
$$I_{A\Rightarrow B} = \{\star\} \qquad I_{A\to B} = I_A$$
$$\lambda_{A\Rightarrow B} = [\star \mapsto PA,\ \overline{\lambda}_A[\boldsymbol{i}_A \mapsto OQ],\ \lambda_B] \qquad \lambda_{A\to B} = [\overline{\lambda}_A[\boldsymbol{i}_A \mapsto OQ], \lambda_B]$$
$$\vdash_{A\Rightarrow B} = \{(\star, \boldsymbol{i}_A), (\boldsymbol{i}_A, \boldsymbol{i}_B)\} \cup \vdash_A \cup \vdash_B \qquad \vdash_{A\to B} = \{(\boldsymbol{i}_A, \boldsymbol{i}_B)\} \cup \vdash_A \cup \vdash_B$$

Fig. 4. Basic arena and prearena constructions

Note that, as M_{e} is fixed for all arenas A and so are the parts of $\lambda_A, \vdash_A$ concerning moves from it, we will not be specifying them explicitly in definitions, We shall refer to moves from $M_A \cup M_{\mathrm{e}}$ collectively as *moves of* A, we shall use $\boldsymbol{i}$ to range over initial moves (which are necessarily ordinary moves), and we shall range over exceptional moves via $e!$. Let $\overline{\lambda}_A$ be the OP-complement of λ_A. We define the basic (flat) arenas:

$$1 = \langle \{\star\}, \{\star\}, \{(\star, PA)\}, \emptyset \rangle \qquad \mathbb{A}_\beta = \langle \mathbb{A}_\beta, \mathbb{A}_\beta, \{(a, PA) \mid a \in \mathbb{A}_\beta\}, \emptyset \rangle$$
$$\mathbb{Z} = \langle \mathbb{Z}, \mathbb{Z}, \{(i, PA) \mid i \in \mathbb{Z}\}, \emptyset \rangle \qquad \mathbb{A}_{\mathrm{e}} = \langle \mathbb{A}_{\mathrm{e}}, \mathbb{A}_{\mathrm{e}}, \{(a, PA) \mid a \in \mathbb{A}_{\mathrm{e}}\}, \emptyset \rangle$$

Given arenas A, B, the arenas $A \otimes B$ and $A \Rightarrow B$ are constructed as in Figure 4, where $\bar{I}_A = M_A \setminus I_A$, $\bar{\vdash}_A = (\vdash_A \restriction \bar{I}_A \times \bar{I}_A)$ (and similarly for B). For each type θ we can now define the corresponding arena $[\![\theta]\!]$ by setting:

$$[\![\mathsf{unit}]\!] = 1 \qquad [\![\mathsf{ref}\ \beta]\!] = \mathbb{A}_\beta \qquad [\![\mathsf{int}]\!] = \mathbb{Z} \qquad [\![\mathsf{exn}]\!] = \mathbb{A}_{\mathrm{e}} \qquad [\![\theta \to \theta']\!] = [\![\theta]\!] \Rightarrow [\![\theta']\!]$$

Although types are interpreted by arenas, the actual games will be played in *prearenas*, which are defined in the same way as arenas with the exception that initial moves are O-questions. Given arenas A, B we define the prearena $A \to B$ as in Figure 4. The moves will be accompanied by an explicit store component Σ. A ***move-with-store*** on a prearena A is thus a pair m^Σ with $m \in M_A \cup M_{\mathrm{e}}$ and $\Sigma \in \mathsf{Sto}$.

Definition 4. A ***justified sequence*** on a prearena A is a sequence of moves-with-store s on A such that, apart from the first move, which must be of the form $\boldsymbol{i}^\Sigma$ with $\boldsymbol{i} \in I_A$, every move $n^{\Sigma'}$ in s is equipped with a pointer to an earlier move m^Σ such that $m \vdash_A n$. We then call m the *justifier* of n and, if $\lambda_A^{QA}(n) = A$, we also say that n *answers* m.

Remark 5. Note that, by definition, any exceptional move $e!$ can answer any question move in a play, as long as the latter has not already be answered. Thus, exceptional moves will model situations when evaluation of some term leads to raising an exception.

We shall write $s \sqsubseteq s'$ to mean that s is a prefix of s'. For each $S \subseteq \mathbb{A}$ and Σ we define $\Sigma^0(S) = S$ and $\Sigma^{i+1}(S) = \Sigma_1(\Sigma^i(S)) \cap \mathbb{A}$ ($i \geq 0$). Let $\Sigma^*(S) = \bigcup_i \Sigma^i(S)$. The set of *available names* of a justified sequence is defined inductively by $\mathsf{Av}(\epsilon) = \emptyset$ and $\mathsf{Av}(sm^\Sigma) = \Sigma^*(\mathsf{Av}(s) \cup \nu(m))$. The *view* of a justified sequence s is defined as follows: $view(\epsilon) = \epsilon$, $view(m^\Sigma) = m^\Sigma$ and $view(s\, \overbrace{m^\Sigma\, t\, n^{\Sigma'}}) = view(s)\, m^\Sigma n^{\Sigma'}$.

Definition 6. Let A be a prearena. A justified sequence s on A is called a ***play***, if it satisfies the conditions below.

- No adjacent moves belong to the same player (*Alternation*).
- The justifier of each answer is the most recent unanswered question (*Bracketing*).
- For any $s'm^{\Sigma} \sqsubseteq s$ with non-empty s', the justifier of m occurs in $view(s')$ (*Visibility*).
- For any $s'm^{\Sigma} \sqsubseteq s$, $\nu(\Sigma) = \mathsf{Av}(s'm^{\Sigma})$ (*Frugality*).

We write P_A for the set of plays on A.

We say that a name a is a *P-name* of a play s if there is $s'm^{\Sigma} \sqsubseteq s$ such that $a \in \nu(m^{\Sigma}) \setminus \nu(s')$ and m is a P-move. We write $P(s)$ for the set of all P-names of s. The set $O(s)$ is defined dually. We moreover define a partial function on alternating justified sequences which imposes the frugality condition by restricting the stores in moves to available names. More precisely, we define $\gamma(s)$ inductively by $\gamma(\epsilon) = \epsilon$ and:

$$\begin{array}{ll} \gamma(s\, m^{\Sigma}) = \gamma(s)\, m^{\Sigma \restriction \mathsf{Av}(sm^{\Sigma})} & \text{if } m \text{ an O-move} \\ \gamma(s\, m^{\Sigma}) = \gamma(s)\, m^{\Sigma \restriction \mathsf{Av}(sm^{\Sigma})} & \text{if } m \text{ a P-move, } \mathsf{Av}(s\, m^{\Sigma}) \cap \nu(s) \subseteq \mathsf{Av}(s) \\ & \text{and } \forall a \in \mathsf{dom}(\Sigma) \setminus \mathsf{Av}(s\, m^{\Sigma}).\ \Sigma(a) = \Sigma'(a) \end{array}$$

where, in the last clause above, the last move of s has store Σ' and, for each store Σ and set $S \subseteq \mathbb{A}$, $\Sigma \restriction S = (\{(a, v) \in \Sigma_1 \mid a \in S\}, \Sigma_2 \cap S)$. Note that partiality arises from sequences breaking the conditions of the last clause.

Definition 7. A ***strategy*** σ on a prearena A is a set of even-length plays of A satisfying:

- If $so^{\Sigma}p^{\Sigma'} \in \sigma$ then $s \in \sigma$ (*Even-prefix closure*).
- If $s \in \sigma$ and $s \sim t$ then $t \in \sigma$ (*Equivariance*).
- If $s_1 p_1^{\Sigma_1}, s_2 p_2^{\Sigma_2} \in \sigma$ and $s_1 \sim s_2$ then $s_1 p_1^{\Sigma_1} \sim s_2 p_2^{\Sigma_2}$ (*Nominal determinacy*).

We write $\sigma : A$ for σ being a strategy on A.

Example 8. For each arena A, the strategy $\mathsf{id}_A : A \to A$, is defined by

$$\mathsf{id}_A = \{\, s \in P^{even}_{A \to A} \mid \forall s' \sqsubseteq^{even} s.\ s' \restriction A_l = s' \restriction A_r \,\},$$

where the indices l, r distinguish the two copies of A, and $s' \restriction A_x$ is the subsequence of s' containing only moves from the x-copy, along with any exceptional moves justified from them. For each arena A, let us write TA for the arena $1 \Rightarrow A$, i.e. $M_{TA} = \{\star_1, \star_2\} \uplus M_A$. Next we define the following exception-related strategies:

- $\mathsf{raiz}_A : \mathbb{A}_{\mathrm{e}} \to A = \{e^{\{e\}} e!^{\{e\}} \mid e \in \mathbb{A}_{\mathrm{e}}\}$
- $\mathsf{trap}_A : TA \to (A + \mathbb{A}_{\mathrm{e}}) = \{\star_1\, \star_2\, s \mid s \in \mathsf{id}_A\} \cup \{\star_1\, \star_2\, e!^{\{e\}} e^{\{e\}} \mid e \in \mathbb{A}_{\mathrm{e}}\}$
- $\mathsf{new}_{\mathrm{e}} : 1 \to \mathbb{A}_{\mathrm{e}} = \{\star\, e^{(\emptyset, \{e\})} \mid e \in \mathbb{A}_{\mathrm{e}}\}$

Note that, in definitions like the above, we implicitly assume that we close the constructed set of plays under even-prefix closure. Thus, in raiz_A the play starts with O providing an exception name e, to which P answers by raising the exception $e!$ (thus, $e^{\{e\}}$ justifies $e!^{\{e\}}$); note that the play never opens in arena A. On the other hand, in trap_A, O starts the play by opening the initial move $\star_1$ under TA, to which P responds

with a question $\star_2$. At this point, O is given two choices: (a) to answer with an initial move of A, so the play will transform into a copycat between the A components of TA and $A + \mathbb{A}_{\mathrm{e}}$; (b) to answer with an exceptional move $e!$, in which case P will 'trap' the name e and return it in the $\mathbb{A}_{\mathrm{e}}$ component of $A + \mathbb{A}_{\mathrm{e}}$. Finally, in $\mathsf{new}_{\mathrm{e}}$ P answers the initial move by playing a fresh exception name and adding it to the store. We will see below that the above mechanisms give us all the structure we need for modelling exceptional behaviours.

We proceed to strategy composition. Given arenas A, B, C, we define the prearena $A \to B \to C$ by setting $M_{A\to B\to C} = M_{A\to B} \uplus M_C$, $I_{A\to B\to C} = I_A$ and:

$$\lambda_{A\to B\to C} = [\lambda_{A\to B}[\boldsymbol{i}_B \mapsto PQ], \overline{\lambda}_C] \qquad \vdash_{A\to B\to C} = \vdash_{A\to B} \cup \{(\boldsymbol{i}_B, \boldsymbol{i}_C)\} \cup \vdash_C$$

Let u be a justified sequence on $A \to B \to C$. We define $u \upharpoonright BC$ to be u in which all A-moves and all exceptional moves justified by A-moves are suppressed. $u \upharpoonright AB$ is defined analogously, only that we also remove any exceptional move justified by an initial move of B. $u \upharpoonright AC$ is defined similarly with the caveat that, if there was a pointer from an initial C-move (resp. an exceptional move) to an initial B-move, which in turn had a pointer to an A-move, we add a pointer from the C-move (the exceptional move) to the A-move. Let us write $u \upharpoonright_\gamma X$ for $\gamma(u \upharpoonright X)$ with $X \in \{AB, BC, AC\}$. Below we shall often say that a move is an O- or a P-move in X meaning ownership in the associated prearena ($A \to B$, $B \to C$ or $A \to C$).

Definition 9. A justified sequence u on $A \to B \to C$ is an ***interaction sequence*** on A, B, C if it satisfies bracketing and frugality and, for all $X \in \{AB, BC, AC\}$, we have $(u \upharpoonright_\gamma X) \in P_X$ and the following conditions hold.

- $P(u \upharpoonright_\gamma AB) \cap P(u \upharpoonright_\gamma BC) = \emptyset$;
- $O(u \upharpoonright_\gamma AC) \cap (P(u \upharpoonright_\gamma AB) \cup P(u \upharpoonright_\gamma BC)) = \emptyset$;
- For each $u' \sqsubseteq u$ ending in $m^{\Sigma} m'^{\Sigma'}$ and $a \in \mathsf{dom}(\Sigma')$ if
  - m' is a P-move in AB and $a \notin \mathsf{Av}(u' \upharpoonright AB)$,
  - or m' is a P-move in BC and $a \notin \mathsf{Av}(u' \upharpoonright BC)$,
  - or m' is an O-move in AC and $a \notin \mathsf{Av}(u' \upharpoonright AC)$,

  then $\Sigma(a) = \Sigma'(a)$.

We write $Int(A, B, C)$ for the set of interaction sequences on A, B, C, and $\sigma \| \tau$ for the set of interactions between strategies $\sigma : A \to B$ and $\tau : B \to C$:

$$\sigma \| \tau = \{\, u \in Int(A, B, C) \mid (u \upharpoonright_\gamma AB) \in \sigma \wedge (u \upharpoonright_\gamma BC) \in \tau \,\}.$$

and let $\sigma; \tau : A \to C = \{u \upharpoonright_\gamma AC \mid u \in \sigma \| \tau\}$.

The following result is deduced by translating our strategies into [7, 10].

Lemma 10. *Strategy composition is associative and identity strategies are neutral elements. Thus, arenas and strategies yield a category of games $\mathcal{G}$.*

A first property of $\mathcal{G}$ is that it has coproducts, given by $+$ and copairings $[\sigma, \tau] : (A + B) \to C = \sigma \cup \tau$ (for $A \xrightarrow{\sigma} C \xleftarrow{\tau} B$). Richer structure is highlighted below.

Remark 11. $\mathcal{G}$ can be shown to host a lluf subcategory $\mathcal{G}'$, consisting of a variant of *single-threaded* strategies [7], where $(1, \otimes)$ yield finite products. Moreover, the operation T on arenas extends to a strong monad in $\mathcal{G}'$ with T-exponentials, i.e. for all A, B, C there is a bijection $\Lambda^T : \mathcal{G}'(A \otimes B, TC) \cong \mathcal{G}'(A, B \Rightarrow C)$ natural in A, C. Then one can show that there exists a bijection $\Phi : \mathcal{G}(A, B) \cong \mathcal{G}'(A, TB)$, which establishes equivalence of $\mathcal{G}$ and the Kleisli category on $\mathcal{G}'$ determined by T ($\mathcal{G}'_T$). We write $\langle \_, \_ \rangle$ for the left-pairing obtained in $\mathcal{G}$ from pairing in $\mathcal{G}'_T$, and $\Lambda(\_)$ for the weak exponential structure.

The above provides a canonical interpretation of application and λ-abstraction in $\mathcal{G}$. To interpret the remaining constructs of ExnML in $\mathcal{G}$, we need to define special morphisms for reference manipulation (cf. [14]) while for exceptions we shall use the morphisms from Example 8.

$$\begin{aligned}
\mathsf{get}_\beta &: \mathbb{A}_\beta \to [\![\beta]\!] = \{ l^\Sigma \Sigma(l)^\Sigma \in P_{\mathbb{A}_\beta \to [\![\beta]\!]} \} \\
\mathsf{set}_\beta &: \mathbb{A}_\beta \otimes [\![\beta]\!] \to 1 = \{ (l, v)^\Sigma \star^{\Sigma[l \mapsto v]} \in P_{\mathbb{A}_\beta \otimes [\![\beta]\!] \to 1} \} \\
\mathsf{new}_\beta &: [\![\beta]\!] \to \mathbb{A}_\beta = \{ v^\Sigma \, l^{\Sigma[l \mapsto v]} \in P_{[\![\beta]\!] \to \mathbb{A}_\beta} \mid l \notin \mathsf{dom}(\Sigma) \}
\end{aligned}$$

We interpret any term-in-context $\mathsf{u}, \Gamma \vdash M : \theta$ with a strategy $[\![\mathsf{u}, \Gamma \vdash M : \theta]\!] : [\![\mathsf{u}, \Gamma \vdash \theta]\!]$, denoted also as $[\![M]\!] : [\![\mathsf{u}, \Gamma \vdash \theta]\!]$. The interpretation is given explicitly below. Suppose that $\mathsf{u} = \{a_1, \cdots, a_n\}$ and $\Gamma = \{x_1 : \theta_1, \cdots, x_k : \theta_k\}$. We write $[\![\mathsf{u}, \Gamma]\!]$ for the arena $[\![\mathsf{u}]\!] \otimes [\![\theta_1]\!] \otimes \cdots \otimes [\![\theta_k]\!]$, where $[\![\mathsf{u}]\!]$ is the flat arena $\langle M_\mathsf{u}, I_\mathsf{u}, \lambda_\mathsf{u}, \vdash_\mathsf{u} \rangle$ with $M_\mathsf{u} = I_u = [(a_1, \cdots, a_n)]$.

- $[\![\mathsf{u}, \Gamma \vdash () : \mathsf{unit}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{!} 1$, where $! = \{ (\bar{a}, \boldsymbol{i}_\Gamma)^\Sigma \star^\Sigma \}$.
- $[\![\mathsf{u}, \Gamma \vdash \Omega : \mathsf{unit}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\perp} 1$, where $\perp = \{\epsilon\}$.
- $[\![\mathsf{u}, \Gamma \vdash i : \mathsf{int}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{!} 1 \xrightarrow{\hat{i}} \mathbb{Z}$, where $\hat{i} = \{\star i\}$.
- $[\![\mathsf{u}, \Gamma \vdash x_j : \theta_j]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\pi_{n+j}} [\![\theta_j]\!]$.
- $[\![\mathsf{u}, \Gamma \vdash M_1 \oplus M_2 : \mathsf{int}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\langle [\![M_1]\!], [\![M_2]\!] \rangle} \mathbb{Z} \otimes \mathbb{Z} \xrightarrow{\sigma_\oplus} \mathbb{Z}$, where $\sigma_\oplus = \{(i_1, i_2)\,(i_1 \oplus i_2)\}$.
- $[\![\mathsf{u}, \Gamma \vdash \mathsf{if0}\, M \,\mathsf{then}\, N_0 \,\mathsf{else}\, N_1 : \theta]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\langle [\![M]\!], \mathsf{id} \rangle} \mathbb{Z} \otimes [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\mathsf{if0} \otimes \mathsf{id}} (1+1) \otimes [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\cong} [\![\mathsf{u}, \Gamma]\!] + [\![\mathsf{u}, \Gamma]\!] \xrightarrow{[[\![N_0]\!], [\![N_1]\!]]} [\![\theta]\!]$, where $\mathsf{if0} = \{0 \star_l,\ i \star_r \mid i \neq 0\}$.
- $[\![\mathsf{u}, \Gamma \vdash \mathsf{ref}_\beta(M) : \mathsf{ref}\ \beta]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{[\![M]\!]} [\![\beta]\!] \xrightarrow{\mathsf{new}_\beta} \mathbb{A}_\beta$.
- $[\![\mathsf{u}, \Gamma \vdash !M : \beta]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{[\![M]\!]} \mathbb{A}_\beta \xrightarrow{\mathsf{get}} [\![\beta]\!]$.
- $[\![\mathsf{u}, \Gamma \vdash M := N : \mathsf{unit}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\langle [\![M]\!], [\![N]\!] \rangle} \mathbb{A}_\beta \otimes [\![\beta]\!] \xrightarrow{\mathsf{set}_\beta} 1$.
- $[\![\mathsf{u}, \Gamma \vdash MN : \theta']\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\langle [\![M]\!], [\![N]\!] \rangle} ([\![\theta]\!] \Rightarrow [\![\theta']\!]) \otimes [\![\theta]\!] \xrightarrow{\mathsf{ev}} [\![\theta']\!]$.
- $[\![\mathsf{u}, \Gamma \vdash \lambda x.M : \theta \to \theta']\!] = \Lambda([\![M]\!] : [\![\mathsf{u}, \Gamma]\!] \otimes [\![\theta]\!] \to [\![\theta']\!])$.
- $[\![\mathsf{u}, \Gamma \vdash \mathsf{exn}() : \mathsf{exn}]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\mathsf{t}} 1 \xrightarrow{\mathsf{new}_\mathrm{e}} \mathbb{A}_\mathrm{e}$.
- $[\![\mathsf{u}, \Gamma \vdash \mathsf{raise}\, M : \theta]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{[\![M]\!]} \mathbb{A}_\mathrm{e} \xrightarrow{\mathsf{raiz}_{[\![\theta]\!]}} [\![\theta]\!]$.
- $[\![\mathsf{u}, \Gamma \vdash M \,\mathsf{handle}\, x \Rightarrow N : \theta]\!] = [\![\mathsf{u}, \Gamma]\!] \xrightarrow{\langle \mathsf{id}, \Phi([\![M]\!]) \rangle} [\![\mathsf{u}, \Gamma]\!] \otimes T[\![\theta]\!] \xrightarrow{\mathsf{id} \otimes \mathsf{trap}_{[\![\theta]\!]}} [\![\mathsf{u}, \Gamma]\!] \otimes ([\![\theta]\!] + \mathbb{A}_\mathrm{e}) \xrightarrow{\cong} ([\![\mathsf{u}, \Gamma]\!] \otimes [\![\theta]\!]) + ([\![\mathsf{u}, \Gamma]\!] \otimes \mathbb{A}_\mathrm{e})) \xrightarrow{[\pi_2, [\![N]\!]]} [\![\theta]\!]$.

We can demonstrate that the game model is sound for contextual approximation (Proposition 12) by following the traditional route through Computational Soundness and Adequacy. For the former we show that we work in a modified version of a *$\nu\epsilon\rho$-model* [15, Def. 5.13]. Recall that a play is *complete* if each question occurring in it justifies an answer. Given a set of plays X, let us write $\mathsf{comp}(X)$ for the set of complete plays in X.

Proposition 12. *Let $\Gamma \vdash M_1, M_2 : \theta$ be terms of $\mathsf{ExnML.comp}([\![\Gamma \vdash M_1 : \theta]\!]) \subseteq \mathsf{comp}([\![\Gamma \vdash M_2 : \theta]\!])$ implies $\Gamma \vdash M_1 \sqsubseteq_{\approx} M_2$.*

4 Full abstraction

We prove full abstraction by showing that all finitary behaviours in the model are definable in ExnML. For the latter we use a factorisation argument which decomposes, in three steps, a strategy from $\mathcal{G}$ into an *exception-free* strategy and strategies managing handling, raising and creation of exceptions respectively. Then, for the class of exception-free strategies we show that finitary members can be expressed in the fragment of ExnML corresponding to Ground ML.

We call a strategy σ ***finitary*** if the set $[\sigma] = \{[s] \mid s \in \sigma\}$ is finite (i.e. σ is *orbit-finite* in the nominal sense). For the first factorisation, we restrict strategies in the following manner. First, for each even-length play s, we let $\phi(s)$ be the justified sequence obtained from s by deleting all its O-moves of the form $e!^{\Sigma}$ (any e, Σ), as well as the moves following these. That is, $\phi(\epsilon) = \epsilon$ and

$$\phi(s\, m^{\Sigma} n^{T}) = \begin{cases} \phi(s) & \text{if } m^{\Sigma} = e!^{T} \\ \phi(s)\, m^{\Sigma} n^{T} & \text{otherwise} \end{cases}$$

We say that a play $s \in P_A$ is *exception-propagating* if $\gamma(\phi(s))$ is defined and, for all $s' e!^{\Sigma} m^{T} \sqsubseteq_{\mathrm{even}} s$, $m^{T} = e!^{\Sigma}$. We write P_A^{prop} for the set of exception-propagating plays on A. We say that a strategy $\sigma : A$ is exception-propagating if $\sigma \subseteq P_A^{\mathrm{prop}}$ and, for all $s \in \sigma$,

- for all $s\, e!^{\Sigma} \in P_A$, we have $s\, e!^{\Sigma} e!^{\Sigma} \in \sigma$;
- for all $s' \in P_A^{\mathrm{prop}}$ with $\gamma(\phi(s)) = \gamma(\phi(s'))$, we have $s' \in \sigma$.

The former condition says that P always copycats raised exceptions, and the latter ensures that P cannot register moves that raise exceptions. We say that an exception-propagating strategy σ is *ϕ-finitary* if the set $\{[\gamma(\phi(s))] \mid s \in \sigma\}$ is finite.

Lemma 13. *Let $\sigma : A \to B$ be a strategy in $\mathcal{G}$. There is an exception-propagating strategy $\hat{\sigma} : \mathbb{A}_{\mathrm{e}} \otimes A \otimes ((1 \Rightarrow 1) \Rightarrow \mathbb{A}_{\mathrm{e}}) \to B$ such that*[5]

$$\sigma = \langle (!; [\![\mathsf{exn}()]\!], \mathsf{id} \rangle, !; [\![\lambda f.\, f()\ \mathsf{handle}\, x\ \mathsf{=>}\ x]\!] \rangle; \hat{\sigma}.$$

Moreover, if σ is finitary then $\hat{\sigma}$ is ϕ-finitary.

[5] The role of the leftmost $\mathbb{A}_{\mathrm{e}}$ in $\hat{\sigma}$ is purely technical and fulfils two functions: (a) it supplies the default return value of $f()\ \mathsf{handle}\, x\ \mathsf{=>}\ x$; (b) it provides a default exception name to be used in subsequent factorisations removing reference generation (the name will be used as initial value for external generators of names in $\mathbb{A}_{\mathsf{exn}}$).

Proof. Let $\tau = \langle\langle !; [\![\mathsf{exn}()]\!], \mathsf{id}\rangle, !; [\![\lambda f.\, f()\,\mathsf{handle}\, x \Rightarrow x]\!]\rangle$ and $C = A_{\mathrm{e}} \otimes A \otimes ((1 \Rightarrow 1) \Rightarrow \mathbb{A}_{\mathrm{e}}) \to B$. We construct $\hat{\sigma} : C$ as follows. For each $s \in \sigma$, build $\hat{s}$ in two stages. In the first stage, perform the following move replacements in s, from left to right.

- Replace the initial move i^{Σ} with $(h, \mathit{i}, \star)^{\Sigma}$, for some fresh $h \in \mathbb{A}_{\mathrm{e}}$.
- Replace each P-question q^{Σ} with a sequence $q_1^{\Sigma} q_2^{\Sigma} q^{\Sigma}$, where q_1 a question justified by the (newly added) initial $\star$, and q_2 justified by q_1.
- Replace each exceptional move $e!^{T}$ of O, answering some previous q^{Σ}, with $e!^{T} e!^{T} e^{T}$, where the first (resp. second) $e!$ is justified by q (q_2), and e is justified by q_1. Diagrammatically:

$$\mathit{i} \cdots q^{\Sigma} \cdots e!^{T} \cdots \;\longmapsto\; (h, \mathit{i}, \star) \cdots q_1^{\Sigma}\, q_2^{\Sigma}\, q^{\Sigma} \cdots e!^{T}\, e!^{T}\, e^{T} \cdots$$

- Replace each P-answer m^{T} (to some previous q'^{Σ}) with $h^{T} h^{T} \cdots h^{T} h^{T} m^{T}$, where m is justified by q, and the the h^{T}'s answer all open q_1 and q_2 moves that were added in the second step above and appear after q'. Note that these q_i's are visible at the corresponding h because they are, in each such case, the pending question.

In the second stage, replace each store Σ in the resulting play with $(\Sigma_1, \Sigma_2 \uplus \{h\})$ (h is chosen fresh for s). We take $\hat{\sigma} = \{t \in P_C^{\mathrm{prop}} \mid \exists s \in \sigma.\, \gamma(\phi(t)) = \gamma(\phi(\hat{s}))\}$. Note first that $\hat{\sigma}$ includes the strategy $\sigma' = \{\hat{s} \mid s \in \sigma\}$, as $\hat{s} \in P_C^{\mathrm{prop}}$ for all $s \in \sigma$, and $\tau; \sigma' = \sigma$. Hence, $\sigma = \tau; \hat{\sigma}$. By construction, $\hat{\sigma}$ is exception-propagating, and $[\gamma(\phi(\hat{\sigma}))]$ is finite if $[\sigma]$ is finite. Finally, note that the passage from σ' to $\hat{\sigma}$ does not break determinacy, as the moves deleted by ϕ are pairs of identical O/P moves. □

The next factorisation eliminates from strategies the capability of raising exceptions. We say that an exception-propagating strategy σ is *handle/raise-free* if, for all $s m^{T} e!^{\Sigma} \in \sigma$, we have $m = e!$.

Lemma 14. *Let $\sigma : \mathbb{A}_{\mathrm{e}} \otimes A \to B$ be an exception-propagating strategy. There is a handle/raise-free $\hat{\sigma} : \mathbb{A}_{\mathrm{e}} \otimes A \otimes (\mathbb{A}_e \Rightarrow 1) \to B$ such that $\sigma = \langle \mathsf{id}, !; [\![\vdash \lambda x.\mathsf{raise}\, x]\!]\rangle; \hat{\sigma}$. Moreover, if σ is ϕ-finitary then so is $\hat{\sigma}$.*

Proof. Let $\tau = \langle \mathsf{id}, !; [\![\vdash \lambda x.\mathsf{raise}\, x]\!]\rangle$ and $C = \mathbb{A}_{\mathrm{e}} \otimes A \otimes (\mathbb{A}_e \Rightarrow 1) \to B$. For each $s \in \sigma$ we construct $\hat{s}$ by replacing each initial move $(h, \mathit{i})^{\Sigma}$ with $(h, \mathit{i}, \star)^{\Sigma}$, and each P-move $e!^{T}$ braking handle/raise-freeness with a sequence $e^{T} e!^{T} e!^{T}$. Diagrammatically ($m \neq e!$ and we omit some stores for brevity):

$$(h, \mathit{i}) \cdots q \cdots m^{\Sigma} e!^{T} \cdots \;\longmapsto\; (h, \mathit{i}, \star) \cdots q \cdots m^{\Sigma}\, e^{T}\, e!^{T}\, e!^{T} \cdots$$

We let $\hat{\sigma} = \{t \in P_C^{\mathrm{prop}} \mid \exists s \in \sigma.\, \gamma(\phi(t)) = \gamma(\phi(\hat{s}))\}$. As above, we have that $\tau; \hat{\sigma} = \sigma$. Since σ is exception-propagating, the move m above cannot be of the form $e'!$ (any $e' \in \mathbb{A}_{\mathrm{e}}$), and therefore $\hat{\sigma}$ preserves the exception-propagating conditions. Moreover, by construction, $\hat{\sigma}$ is handle/raise-free, and $[\gamma(\phi(\hat{\sigma}))]$ is finite if $[\gamma(\phi(\sigma))]$ is. □

Our final factorisation concerns removing any exception-name generation capability from our strategies. The technique is similar to the one used in the factorisations above and amounts to delegating all fresh exception-name creation to an external generator. Formally, a handle/raise-free strategy is called *exception-free* if, for all $s \in \sigma$, $P(s) \cap \mathbb{A}_{\mathrm{e}} = \emptyset$.

Lemma 15. *Let $\sigma : \mathbb{A}_e \otimes A \to B$ be a handle/raise-free strategy. There is an exception-free $\hat{\sigma} : \mathbb{A}_e \otimes A \otimes (1 \Rightarrow \mathbb{A}_e) \to B$ such that $\sigma = \langle \mathsf{id}, !; [\![\lambda z.\mathsf{exn}()]\!] \rangle; \hat{\sigma}$. Moreover, if σ is ϕ-finitary then so is $\hat{\sigma}$.*

Let us call $\mathsf{ExnML}_{\neg \mathrm{e}}$ the fragment of ExnML obtained by suppressing the constructors $\mathsf{handle}, \mathsf{raise}$ and $\mathsf{exn}()$. We can show that exception-freeness is captured by $\mathsf{ExnML}_{\neg \mathrm{e}}$ in the following sense.

Lemma 16. *Let $\sigma : \mathbb{A}_\mathrm{e} \otimes A \to B$ a ϕ-finitary exception-free strategy over a denotable prearena. There is an $\mathsf{ExnML}_{\neg \mathrm{e}}$ term $\mathsf{u}, \Gamma \vdash M : \theta$ such that $[\![M]\!] = \sigma$.*

Combining the four previous lemmas we obtain the following.

Proposition 17. *Let $\mathbb{A}_\mathrm{e} \otimes A \to B$ be a denotable prearena and $\sigma : A \to B$ a finitary strategy. There is an ExnML term $\mathsf{u}, \Gamma \vdash M : \theta$ such that $[\![M]\!] = \sigma$.*

Theorem 18. *For all ExnML-terms $\Gamma \vdash M_1, M_2 : \theta$, we have $\mathsf{comp}([\![\Gamma \vdash M_1 : \theta]\!]) \subseteq \mathsf{comp}([\![\Gamma \vdash M_2 : \theta]\!])$ if, and only if, $\Gamma \vdash M_1 \sqsubseteq_{\approx} M_2$.*

5 Idealised exceptions

The design of exception handling in ExnML was guided by common practice. In an idealised world, private exceptions should not be amenable to handling. This can be achieved by the alternative handling construct:

$$\frac{\mathsf{u}, \Gamma \vdash M, N' : \theta \quad \mathsf{u}, \Gamma \vdash N : \mathsf{exn}}{\mathsf{u}, \Gamma \vdash M \,\mathsf{handle}\, N \,\texttt{->}\, N' : \theta}$$

We call $\mathsf{ExnML}_\$$ the language which differs from ExnML in featuring the above construct instead of "$M \,\mathsf{handle}\, x \,\texttt{=>}\, N$". The new language has additional reduction rules:

$$\Sigma, (\mathsf{raise}\, e) \,\mathsf{handle}\, e \,\texttt{->}\, N \longrightarrow \Sigma, N$$
$$\Sigma, E_{\neg e}[\mathsf{raise}\, e] \longrightarrow \Sigma, \mathsf{raise}\, e$$

Evaluation contexts are now given by:

$$E ::= [\_] \mid E\,N \mid (\lambda x.M)E \mid \mathsf{if0}\, E \,\mathsf{then}\, N_0 \,\mathsf{else}\, N_1 \mid E \oplus N \mid i \oplus E \mid \mathsf{ref}_\gamma(E) \mid E := N$$
$$\mid l := E \mid !E \mid E = N \mid e = E \mid M \,\mathsf{handle}\, E \,\texttt{->}\, N \mid E \,\mathsf{handle}\, e \,\texttt{->}\, N \mid \mathsf{raise}\, E$$

and, for each $e \in \mathcal{E}$, we write $E_{\neg e}$ for contexts E derived by the above grammar applying any of the rules apart from $E \,\mathsf{handle}\, e \,\texttt{->}\, N$. Note that the new handler is easily definable in ExnML by:

$$M \,\mathsf{handle}\, N \,\texttt{->}\, N' \;\equiv\; \mathsf{let}\, z = N \,\mathsf{in}\, (M \,\mathsf{handle}\, x \,\texttt{=>}\, (\mathsf{if0}\, x = z \,\mathsf{then}\, \mathsf{raise}\, x \,\mathsf{else}\, N'))$$

Thus, $\mathsf{ExnML}_\$$ is a sublanguage of ExnML in terms of expressivity.

Example 19. Recall the terms M_1 and M_2 from Example 2. They will turn out equivalent in $\mathsf{ExnML}_\$$, because in either case the private exceptions raised by the terms can only be propagated. Next we shall develop game-semantic constraints that reflect such scenarios.

6 Games propagating private exceptions

We derive the game model of $\mathsf{ExnML}_\$$ by restricting the category $\mathcal{G}$ with an additional condition on strategies. We need to depict semantically that terms in $\mathsf{ExnML}_\$$ are only able to handle exception names that are 'known' to them. In particular, fresh exceptions cannot be handled and will break through any evaluation context. Moreover, such exceptions cannot be remembered and neither can their accompanying stores. We therefore define the following notion of available subplay. For any even-length play s over some prearena A, we define the justified sequence $\$(s)$ inductively by $\$(\epsilon) = \epsilon$ and:

$$\$(s\, m^{\Sigma} n^{T}) = \begin{cases} \$(s) & \text{if } m = e! \text{ and } e \notin \mathsf{Av}(\$(s)) \\ \$(s)\, m^{\Sigma} n^{T} & \text{otherwise} \end{cases}$$

We let $\mathsf{Av}_\$(s) = \mathsf{Av}(\$(s))$. The above definition disregards not only fresh exceptions raised by O, but also the P-moves succeeding them. This is due to the fact that the terms (and strategies) we consider simply propagate such exceptions.

Definition 20. We say that a play $s \in P_A$ is $\$$-***propagating*** if $\gamma(\$(s))$ is defined and, for all $s' e!^{\Sigma} m^{T} \sqsubseteq_{\mathrm{even}} s$ with $e \notin \mathsf{Av}_\$(s)$, $m^{T} = e!^{\Sigma}$.
We say that a strategy $\sigma : A$ is $\$$-*propagating* if $\sigma \subseteq P_A^{\$\mathrm{prop}}$ and, for all $s \in \sigma$,

- for all $s\, e!^{\Sigma} \in P_A$ and $e \notin \mathsf{Av}_\$(s)$, we have $s\, e!^{\Sigma} e!^{\Sigma} \in \sigma$;
- for all $s' \in P_A^{\$\mathrm{prop}}$ with $\gamma(\$(s)) = \gamma(\$(s'))$, we have $s' \in \sigma$.

We write $P_A^{\$\mathrm{prop}}$ for the set of $\$$-propagating plays of A.

Thus, the former condition stipulates that strategies propagate raised exceptions if these feature fresh exception names. The latter ensures that strategies do not depend on these raised exceptions or their stores. We can show that these conditions are compositional. Suppose we compose $\$$-propagating strategies $\sigma : A \to B$ and $\tau : B \to C$. Exceptional moves suppressed by $\$$ are O-moves, carrying O-names. Thus, by the name-ownership conditions of strategy-composition, if a move is suppressed in a composite play in AC, then it is also suppressed in its constituent plays in AB and BC. As a result, suppressed exceptions are propagated in σ and τ, resulting in propagation by $\sigma; \tau$. Similarly, saturation under $\gamma(\$(\_))$ of $\sigma; \tau$ is ensured by componentwise saturation of σ and τ respectively.

Lemma 21. *If $\sigma : A \to B, \tau : B \to C$ are $\$$-propagating then so is $\sigma; \tau$.*

Identity strategies are $\$$-propagating by construction. We therefore obtain a lluf category $\mathcal{G}_\$$ of $\$$-propagating strategies. Terms from $\mathsf{ExnML}_\$$ are given denotations in $\mathcal{G}_\$$ as before, only that now we use the strategies $\mathsf{hdl}\$_A : TA \otimes \mathbb{A}_{\mathrm{e}} \to A + 1 =$

$$\{(\star_1, e)^{\{e\}} \star_2^{\{e\}} s \mid s \in \mathsf{id}_A, e \in \mathbb{A}_{\mathrm{e}}\} \cup \{(\star_1, e)^{\{e\}} \star_2^{\{e\}} e!^{\{e\}} \star^{\{e\}} \mid e \in \mathbb{A}_{\mathrm{e}}\}$$

instead of trap_A, which break $\$$-propagation. With these constructs we obtain a $\nu\epsilon\rho$-model [15] with references restricted to ground types. We thus have soundness.

Lemma 22. *For all $\mathsf{ExnML}_\$$-terms $\Gamma \vdash M_1, M_2 : \theta$, if $[\![\Gamma \vdash M_1 : \theta]\!] \subseteq [\![\Gamma \vdash M_2 : \theta]\!]$ then $\Gamma \vdash M_1 \sqsubseteq\!\!\!\sim M_2$.*

Remark 23. In previous game models of control, the control-manipulating effect of privately-propagating exceptions was captured by relaxing the bracketing condition [4, 8]. The latter was achieved in the expense of adding an additional pointer structure, called *control (or contingency) pointers*, to mark violations of bracketing. Here we took a different approach by exposing the private-exception mechanism that caters for such violations (cf. [15, 8]). As a result, our model consists of plays that still satisfy bracketing, albeit in this extended setting. As in the case of control pointers, to avoid being overly intentional, we need to hide access to private exceptions via the propagation conditions.

While previously term approximation was characterised by inclusion of complete plays, now we have to restrict the set of plays to take into account \$-propagation on the part of the environment. Given a complete play s on a prearena $1 \to A$ with final store Σ, we let $\hat{s} \equiv \star \overleftarrow{s} \star^{\Sigma} \in P_{(1\Rightarrow A)\to 1}$. For each \$-propagating strategy $\sigma : 1 \to A$, we then define $\mathsf{comp}_{\$}(\sigma) = \{\gamma(\$(\hat{s})) \mid s \in \mathsf{comp}(\sigma),\ \hat{s} \in P^{\$\mathrm{prop}}_{(1\Rightarrow A)\to 1}\}$.

Proposition 24. *For all* $\mathsf{ExnML}_{\$}$*-terms* $\vdash M_1, M_2 : \theta$*, if* $\mathsf{comp}_{\$}([\![M_1]\!]) \subseteq \mathsf{comp}_{\$}([\![M_2]\!])$ *then* $\vdash M_1 \sqsubseteq\!\!\!\sim M_2$.

Proof. Suppose the inclusion holds and let $C[M_1]\Downarrow$ for some context C. Then, by Lemma 22, $[\![C[M_1]]\!] = \{\star\star\}$, that is, $[\![\lambda z.M_1]\!]; [\![f \vdash C[f()]]\!] = \{\star\star\}$. Let us write M_i' for $\lambda z.M_i$ $(i = 1, 2)$, N for $[\![f \vdash C[f()]]\!]$, $\gamma_{\$}(\_)$ for $\gamma(\$(\_))$ and let $A = [\![\theta]\!]$. Then, $[\![M_1']\!]; [\![N]\!] = \{\star\star\}$, the latter due to composing some complete play $\star \star s \in [\![M_1']\!]$ with some $\star s \star^{\Sigma} \in [\![N]\!]$. Since $M_1' \equiv \lambda\vec{x}.M_1$, s must be an interleaving of complete plays $s_1, \cdots, s_k \in [\![M_1]\!]$. For each i, we have $\hat{s}_i \in P^{\$\mathrm{prop}}_{(1\Rightarrow A)\to 1}$, because $\star s \star^{\Sigma} \in [\![N]\!] \subseteq P^{\$\mathrm{prop}}_{(1\Rightarrow A)\to 1}$, and therefore $\gamma_{\$}(\hat{s}_i) \in \mathsf{comp}_{\$}([\![M_1]\!])$. By hypothesis, $\gamma_{\$}(\hat{s}_i) \in \mathsf{comp}_{\$}([\![M_2]\!])$, and so there is $s_i' \in \mathsf{comp}([\![M_2]\!])$ such that $\gamma_{\$}(\hat{s}_i) = \gamma_{\$}(\hat{s}_i')$. Let $s' \in P_{1\to(1\Rightarrow A)}$ be the interleaving of $s_1', \cdots, s_k'$ obtained by simulating the interleaving pattern of s. Note that, for each i, since $\gamma_{\$}(\hat{s}_i) = \gamma_{\$}(\hat{s}_i')$, s_i and s_i' share the same structure apart from P/O pairs of exceptional moves deleted by \$, which do not affect simulating the interleaving of s (change of thread can only occur in O-moves). We thus obtain some $\star s' \star^{\Sigma'} \in P^{\mathrm{prop}}_{(1\Rightarrow A)\to 1}$ and, by lifting equality under $\gamma_{\$}$ from threads to thread-interleavings, we have $\gamma_{\$}(\star s \star^{\Sigma}) = \gamma_{\$}(\star s' \star^{\Sigma})$. But, since $[\![N]\!]$ is \$-propagating, $\star s \star^{\Sigma} \in [\![N]\!]$ implies $\star s' \star^{\Sigma'} \in [\![N]\!]$ and therefore $[\![M_2']\!]; [\![N]\!] = \{\star\star\} = [\![C[M_2]]\!]$. By Lemma 22, then, $C[M_2]\Downarrow$. □

For completeness, we again work our way through a finitary definability result, established via factorisations. The first factorisation brings us to handle-free strategies, from which the factorisations of the previous section can be applied. We say that an \$-propagating strategy σ is \$-*finitary* if the set $\{[\gamma(\$(s))] \mid s \in \sigma\}$ is finite.

Lemma 25. *Let* $\sigma : A \to B$ *be a* \$*-finitary strategy. There is some* $n \in \omega$ *and a* ϕ*-finitary handle-free strategy* $\hat{\sigma} : \mathbb{A}_e \otimes A \otimes \mathbb{A}^n_{\mathsf{exn}} \otimes ((1 \Rightarrow 1) \Rightarrow \mathbb{A}_{\mathrm{e}}) \to B$ *such that*

$$\sigma = \langle !; [\![\mathsf{exn}()]\!], \mathsf{id}\rangle; \langle \mathsf{id}, \pi_1; \langle \overrightarrow{[\![x : \mathsf{exn} \vdash \mathsf{ref}\,(x)]\!]}, \mathsf{id}\rangle\rangle; \langle \pi_1, \overrightarrow{[\![z : \mathsf{ref}\,\mathsf{exn}}, h : \mathsf{exn} \vdash M]\!]\rangle; \hat{\sigma}$$

with $M \equiv \lambda f.\,(\cdots(f(); h\,\mathsf{handle}\,!z_1 \texttt{->} !z_1)\,\mathsf{handle}\,!z_2 \texttt{->} !z_2 \cdots)\,\mathsf{handle}\,!z_n \texttt{->} !z_n : \mathsf{exn}$.

The above factorisation is identical to the corresponding one in the previous section, only that instead of simply delegating exception handling to the environment, the strategy $\hat{\sigma}$ also stores all exception names encountered, apart from those in \$-removable moves, in the variables z_i.

Proposition 26. *Let $A \to B$ be a denotable arena. For each \$-finitary strategy $\sigma : A \to B$ there is an* $\mathsf{ExnML}_\$$ *term* $\mathsf{u}, \Gamma \vdash M : \theta$ *such that* $[\![M]\!] = \sigma$.

We can now prove completeness, and thus full abstraction.

Theorem 27. *For all* $\mathsf{ExnML}_\$$*-terms* $\vdash M_1, M_2 : \theta$*, we have* $\mathsf{comp}_\$([\![M_1]\!]) \subseteq \mathsf{comp}_\$([\![M_2]\!])$ *if, and only if,* $\vdash M_1 \sqsubseteq\!\!\!\approx M_2$.

Proof. We show completeness (right-to-left). Let us write M_i' for $\lambda z.M_i$ ($i = 1, 2$). Suppose $s \in \mathsf{comp}_\$([\![M_1]\!]) \setminus \mathsf{comp}_\$([\![M_2]\!])$. Let $A = [\![\theta]\!]$ and define the strategy $\rho : (1 \Rightarrow A) \to 1$ by: $\rho = \{t \in P^{\$\mathrm{prop}}_{(1\Rightarrow A)\to 1} \mid \gamma(\$(t)) \sqsubseteq_{\mathrm{even}} s\}$. By construction, ρ is \$-propagating and \$-finitary. Hence, there is a term $f : \mathsf{unit} \to \theta \vdash N : \mathsf{unit}$ such that $[\![N]\!] = \rho$. Moreover, $s \in [\![N]\!]$ and thus, by Lemma 22, $(\lambda f.N)M_1'\Downarrow$. We claim that $\star\star \notin [\![M_2']\!]; \rho$ and therefore $(\lambda f.N)M_2'\not\Downarrow$. For suppose $\star\star \in [\![M_2']\!]; \rho$, because of composing $[\![M_2']\!]$ with some play $t = \star s'\star^{\Sigma} \in \rho$. Then, $s' \in \mathsf{comp}([\![M_2]\!])$ and $\gamma(\$(\star s'\star^{\Sigma})) \sqsubseteq s$. Since $\star^{\Sigma}$ is not deleted by \$, we have in fact $\gamma(\$(\star s'\star^{\Sigma})) = s$, hence $s \in \mathsf{comp}_\$([\![M_2]\!])$, contradicting the hypothesis. □

Example 28. Let us revisit the terms from Examples 2 and 19. We have $\mathsf{comp}_\$([\![M_i]\!]) = \{\star^{\Sigma_0} q_0^{\Sigma_0}\ \star^{\Sigma_0}\ \star^{\Sigma_0}\}$ for $i = 1, 2$, because other plays from $[\![M_i]\!]$ would give rise to non-propagating interactions $\hat{s}$. Thus, in $\mathsf{ExnML}_\$$ we do have $\vdash M_1 \cong M_2$.

References

1. S. Abramsky, D. R. Ghica, A. S. Murawski, C.-H. L. Ong, and I. D. B. Stark. Nominal games and full abstraction for the nu-calculus. In *LICS'04*.
2. M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13:341–363, 2002.
3. K. Honda and N. Yoshida. Game-theoretic analysis of call-by-value computation. *Theoretical Computer Science*, 221(1–2):393–456, 1999.
4. J. Laird. *A semantic analysis of control*. PhD thesis, University of Edinburgh, 1998.
5. J. Laird. A fully abstract games semantics of local exceptions. In *LICS'01*.
6. J. Laird. A game semantics of local names and good variables. In *FOSSACS'04*.
7. J. Laird. A game semantics of names and pointers. *Ann. Pure Appl. Logic*, 151:151-169, 2008.
8. J. Laird. Combining and relating control effects and their semantics. In *COS*, 2013.
9. E. Moggi. Notions of computation and monads. *Inf. and Comput.*, 93:55–92, 1991.
10. A. S. Murawski and N. Tzevelekos. Game semantics for good general references. In *LICS'11*.
11. A. S. Murawski and N. Tzevelekos. Algorithmic games for full ground references. *ICALP'12*.
12. A. S. Murawski and N. Tzevelekos. Full abstraction for Reduced ML. *Ann. Pure Appl. Logic*, 164(11):1118–1143, 2013.
13. J. C. Reynolds. The essence of Algol. In J. W. de Bakker and J.C. van Vliet, editors, *Algorithmic Languages*, pages 345–372. North Holland, 1981.
14. I. D. B. Stark. *Names and Higher-Order Functions*. PhD thesis, University of Cambridge, 1995. Technical Report No. 363.
15. N. Tzevelekos. Nominal game semantics. D.Phil. thesis, Oxford University, 2008.
16. N. Tzevelekos. Full abstraction for nominal general references. *LMCS*, 5(3), 2009.