The anatomy of innocence revisited

Russ Harmer & Olivier Laurent

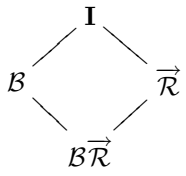
PPS, CNRS & Université Paris 7

Abstract. We refine previous analyses of Hyland-Ong game semantics and its relation to λ- and $\lambda\mu$-calculi and present improved factorization results for bracketing and rigidity that can be combined in any order.

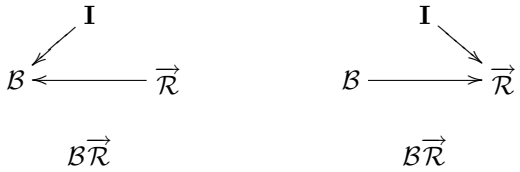
1 Introduction

Innocent strategies [2, 4, 7] provide models of (idealized programming languages based on) the λ- and $\lambda\mu$-calculi, the difference between these two calculi being expressed by the *bracketing condition*: whenever a strategy plays an *answer*, this must respond to the "pending" *question*, *i.e.* the most recent, as yet untreated, request. In [5], Laird analysed this situation and showed that an arbitrary innocent strategy σ can be decomposed as a well-bracketed innocent strategy $\mathcal{B}(\sigma)$ with access to an innocent but non-well-bracketed oracle `call/cc`. This semantic *factorization* mirrors the well-known result from proof theory that classical deductions can be rewritten as intuitionistic deductions with a few copies of Peirce's law as additional hypotheses. In [1], Danos & Harmer introduced a new constraint of *rigidity*, in a certain sense dual to bracketing, which restricts the use of `case` much as bracketing restricts the use of `call/cc` and showed that σ can be decomposed as a rigid innocent strategy $\overrightarrow{\mathcal{R}}(\sigma)$ with access to a non-rigid oracle `case`.

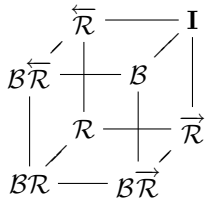
So we have a decomposition of the CCC $\mathbf{I}$ of innocent strategies into a "diamond" of subCCCs:



In $\mathcal{B}$, we can model `case` but not `call/cc` whereas in $\overrightarrow{\mathcal{R}}$ we can model `call/cc` but not `case`. However, neither factorization *preserves* the other constraint, *i.e.* eliminating `call/cc` reintroduces `case` and eliminating `case` reintroduces `call/cc`:

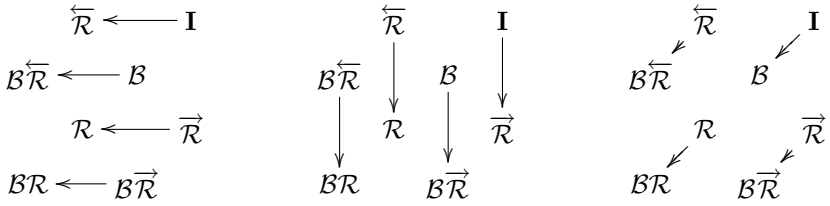


In this paper, we continue this analysis of innocent strategies with the aim of better understanding the role of answers in game semantics. To this end, we introduce an additional constraint, *backward rigidity* or *B-rigidity*, which extends the above decomposition to a "cube" of subCCCs (§3.3):



This constraint can be seen as a dual to rigidity—which, henceforth, we rename as *forward rigidity*, reserving the term *rigidity* ($\mathcal{R}$) for the conjunction of the two—in that forward rigidity restricts the *elimination* (in the sense of natural deduction) of base type constants whereas backward rigidity restricts their *introduction*. The cube provides us with a taxonomy of logics and programming languages based on λ- and $\lambda\mu$-calculi: each node corresponds, via definability and full completeness theorems, to a fragment of μPCF (§3.4).

We then present (§4) a factorization for B-rigidity and modified factorizations for F-rigidity and bracketing, each of which preserves the other two constraints:



This allows us to "navigate" (from $\mathbf{I}$) in the cube, applying factorizations in whichever order we like, and still being sure to end up in $\mathcal{BR}$. From a syntactic point of view, this explains how we can move from one language of the cube to another: a "smaller" language plus an appropriate oracle equals a "larger" language (*e.g.* λ-calculus plus `call/cc` equals $\lambda\mu$-calculus).

We conclude (§4.4) by examining the unary case where the factorizations can be simplified and the connection to logic becomes especially clear: $\mathcal{BR} = \lambda$-calculus and $\mathcal{R} = \lambda\mu$-calculus.

2 Innocent game semantics

This section briefly presents the definitions necessary to construct the category $\mathbf{I}$ of innocent strategies. A more detailed development can be found in [2].

2.1 Arenas and plays

An ***arena*** A is a tuple $\langle M_A, \lambda_A, I_A, \vdash_A \rangle$ where

- M_A is a countable set of ***tokens***.
- $\lambda_A : M_A \to \{\mathsf{O}, \mathsf{P}\} \times \{\mathsf{Q}, \mathsf{A}\}$ ***labels*** each $m \in M_A$ as belonging to Opponent or to Player and as a Question or an Answer.
- I_A is a subset of $\lambda_A^{-1}(\mathsf{OQ})$ known as the ***initial moves*** of A.
- $\vdash_A$ is a binary ***enabling*** relation on M_A satisfying
  (e1) if $m \vdash_A n$ then $\lambda_A^{\mathsf{OP}}(m) \neq \lambda_A^{\mathsf{OP}}(n)$ and $n \notin I_A$;
  (e2) if $m \vdash_A n$ where $\lambda_A^{\mathsf{QA}}(n) = \mathsf{A}$ then $\lambda_A^{\mathsf{QA}}(m) = \mathsf{Q}$.

An arena where answers *never* enable other moves is called an A-***terminal*** arena. A ***flat*** arena has a single OQ-move and a set of PA-moves, all enabled by the O-move. For example, **bool** has an OQ, q, and two PAs, $\mathtt{tt}$ and $\mathtt{ff}$, where $\mathsf{q} \vdash_{\mathbf{bool}} \mathtt{tt}$ and $\mathsf{q} \vdash_{\mathbf{bool}} \mathtt{ff}$. We similarly define $\bot$, **com** and **nat** as the flat arenas over $\varnothing$, $\{\mathtt{t}\}$ and $\{0, 1, 2, \ldots\}$ respectively. Note that a flat arena is always A-terminal.

A ***play*** in arena A is a string s over alphabet M_A with pointers between its occurrences such that, if s_i (the ith symbol of s) points to s_j then $j < i$, if s_j points to s_i then $s_i \vdash_A s_j$ and if s_i has no pointer then $s_i \in I_A$. We write $|s|$ for the length of s. A ***legal play*** in arena A is a play in A that also satisfies OP-***alternation***: $\lambda_A^{\mathsf{OP}}(s_i) \neq \lambda_A^{\mathsf{OP}}(s_{i+1})$ for $1 \leq i < |s|$. Each occurrence in a legal play s is an element m of M_A together with its pointer (unless $m \in I_A$); we call m plus its pointer a ***move*** of s. If m points to n in s, we say that n ***justifies*** m in s. We write $\mathcal{L}_A$ for the set of all legal plays in A.

The ***prefix*** ordering on strings extends to $\mathcal{L}_A$ with least element ε, the empty play. For $s, t \in \mathcal{L}_A$, we write $s \sqsubseteq t$ (resp. $s \sqsubseteq^{\mathsf{O}} t$, resp. $s \sqsubseteq^{\mathsf{P}} t$) when s is a (resp. O-ending, resp. P-ending) prefix of t. We fix the convention that $\varepsilon \sqsubseteq^{\mathsf{P}} s$ for any $s \in \mathcal{L}_A$. We write $s \wedge t$ for the ***longest common prefix*** of s and t, $\mathsf{ip}(s)$ or s^- for the ***immediate prefix*** of non-empty s and, provided the ***last move*** of s, written s_ω, has a pointer, $\mathsf{jp}(s)$ for the ***justifying prefix*** of s, *i.e.* that prefix of s ending with the move that justifies s_ω. We define $\mathsf{ie}(s) = \{t \in \mathcal{L}_A \mid \mathsf{ip}(t) = s\}$, the set of ***immediate extensions*** of s and, if $s \in \mathcal{L}_A$ and $m \in M_A$ such that s_ω enables m in A, we write $s \cdot m$ for the legal play obtained by adding m to the end of s, pointing to the last move.

We have the standard [6] constructors on arenas: the ***product*** $A \times B$ (and its infinite version A^ω), the ***par*** $A \parr B$ and the ***lift*** $\downarrow A$ from which we recover the familiar ***arrow*** $A \Rightarrow B$ as $(\downarrow A) \parr B$. If A and B are ***pointed*** arenas [only one initial move] then $A \parr B$ is also pointed and, in this special case, is written $A \oplus B$. All constructors preserve the property of being A-terminal.

2.2 The ambient SMCC

A ***strategy*** σ for an arena A, written $\sigma : A$, is a non-empty set of P-ending legal plays of A which satisfies

– *prefix-closure*: if $s \in \sigma$ and $s' \sqsubseteq^{\mathsf{P}} s$ then $s' \in \sigma$;
– *determinism*: if $s \in \sigma$ and $t \in \sigma$ then $s \wedge t \in \sigma$.

The second condition amounts to asking for $s \wedge t$ to end with a P-move; so only Opponent can branch nondeterministically. We write $\mathsf{dom}(\sigma)$ for the ***domain*** of σ defined as $\bigcup_{s \in \sigma} \mathsf{ie}(s)$, all the O-ending plays of A *accessible* to σ.

We compose strategies $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ by parallel composition plus hiding, *i.e.* σ and τ synchronize on B and hide this from "the outside world", yielding $\sigma ; \tau : A \Rightarrow C$. It can be shown that, by taking arenas as objects and strategies for $A \Rightarrow B$ as arrows between A and B, this notion of composition gives rise to an SMCC $\mathbf{G}$ [2].

2.3 The CCC of innocent strategies

We define the P-***view***, noted $\ulcorner s \urcorner$, of a non-empty legal play $s \in \mathcal{L}_A$ in two stages. First we extract a subsequence of s with pointers defined only for O-moves:

– $\ulcorner s \urcorner = s_\omega$, if s_ω is an initial move;
– $\ulcorner s \urcorner = \ulcorner \mathsf{jp}(s) \urcorner \cdot s_\omega$, if s_ω is a non-initial O-move;
– $\ulcorner s \urcorner = \ulcorner \mathsf{ip}(s) \urcorner \, s_\omega$, if s_ω is a P-move.

In words, we trace back from the end of s, following pointers from O-moves, excising all moves under such pointers, and "stepping over" P-moves, until we reach an initial move. In general, a P-move can "lose its pointer" (if it points to a move that gets erased in this way). The second stage of the definition specifies that, in such a case, the P-move has *no justifier* in the P-view (and so $\ulcorner s \urcorner \notin \mathcal{L}_A$); otherwise it keeps the same justifier as in s.

We say that a legal play $s \in \mathcal{L}_A$ satisfies P-***visibility*** iff $\ulcorner s \urcorner \in \mathcal{L}_A$. In words, no P-move of $\ulcorner s \urcorner$ loses its pointer. Note that this doesn't prevent a P-move of $\ulcorner t \urcorner$ losing its pointer, for t some proper prefix of s. We lift the definition of P-visibility to strategies in the obvious way: σ satisfies P-***visibility*** iff all $s \in \sigma$ do. Note that, for s in P-vis σ as opposed to arbitrary P-vis s, all $t \sqsubseteq^{\mathsf{P}} s$ do in fact satisfy P-visibility—since σ is closed under P-ending prefixes—so $\ulcorner t \urcorner \in \mathcal{L}_A$ for all the P-prefixes t of s.

If $s, t \in \mathcal{L}_A$ where s ends with a P-move, satisfies P-vis and $\ulcorner \mathsf{ip}(s) \urcorner = \ulcorner t \urcorner$ then we denote by $\mathsf{match}(s, t)$ the unique extension of t satisfying $\ulcorner s \urcorner = \ulcorner \mathsf{match}(s, t) \urcorner$, *i.e.* add the last move of s to t using the "same" pointer as in s. We can do this since, by assumption, the last move of s points in $\ulcorner \mathsf{ip}(s) \urcorner = \ulcorner t \urcorner$.

We now say that a *deterministic* P-vis strategy σ is ***innocent*** iff

$$s \in \sigma \wedge t \in \mathsf{dom}(\sigma) \wedge \ulcorner \mathsf{ip}(s) \urcorner = \ulcorner t \urcorner \Rightarrow \mathsf{match}(s, t) \in \sigma.$$

So an innocent strategy is completely determined by its ***view function*** $\ulcorner \sigma \urcorner$ defined to be $\{\ulcorner s \urcorner \mid s \in \sigma\}$. It can be shown that innocent strategies are closed under composition and form a subcategory $\mathbf{I}$ of $\mathbf{G}$ where the monoidal structure becomes Cartesian, *i.e.* $\mathbf{I}$ is a CCC. In the rest of this paper, we restrict to the full subCCC of $\mathbf{I}$ consisting of A-terminal arenas only.

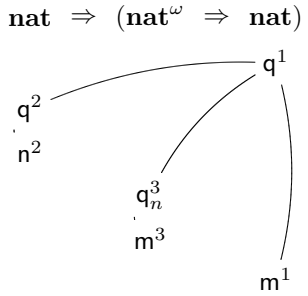
3 Bracketing and rigidity

3.1 Backward rigidity

An innocent strategy satisfies ***backward*** (or ***B-***)***rigidity*** iff every time it plays an answer, the *preceding* O-move was also an answer. This rules out strategies like $\mathsf{skip} = \{\varepsilon, \mathsf{q} \cdot \mathsf{t}\} : \mathbf{com}$ where Player produces an answer "from thin air".

3.2 Forward rigidity

An obvious "dual" to B-rigidity applies the same condition to questions: every time the strategy plays a question, the preceding move must itself have been a question. We call such strategies ***forward*** (or ***F-***)***rigid***. In the setting of A-terminal arenas, this is equivalent to the notion of *rigid* strategy defined in [1]. This condition typically rules out case:



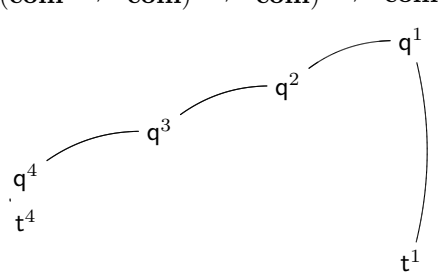
3.3 The bracketing condition

The P-view of an O-ending legal play s has generic form

$$\mathsf{OQ}((\mathsf{PQ} \curvearrowleft \mathsf{OQ})^*(\mathsf{PQ} \curvearrowleft \mathsf{OA})^*)^*$$

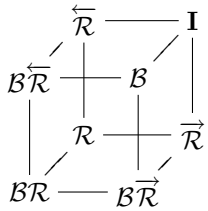
(where we've omitted Player's pointers for clarity). The rightmost OQ of $\ulcorner s \urcorner$ is called the ***pending question*** of s. An innocent strategy satisfies the ***bracketing*** condition iff, every time the strategy plays an answer, that answer is *justified* by the pending question.

This rules out strategies like call/cc à la Peirce:



3.4 The cube of subcategories

Each of the above constraints is preserved by composition, independently of the others. For this reason, we say that the constraints are *orthogonal*. As an immediate consequence of this, we can "unfold" the CCC of innocent strategies $\mathbf{I}$ into a cube of subcategories:



As shown in [1,3], any innocent strategy with finite view function for (the arena interpreting) a simple type over a collection of flat arenas is denoted by a term in the following "Böhm tree" syntax (with appropriate typing rules) where Ω is a divergent term (or constant) of base type and $\mathtt{k}$ ranges over the (other) constants of base type:

$$E ::= \Omega \mid [\alpha]\mathtt{k} \mid (\mathtt{case}\ (x)\vec{F}\ \ \overrightarrow{\mathtt{k} \mapsto E})$$
$$F ::= \lambda\vec{x}\mu\alpha(E)$$

We can "unfold" this rather compact syntax into the following grammar:

$$V ::= \Omega \mid [\alpha]\mathtt{k}$$
$$C ::= \Omega \mid (\mathtt{case}\ (x)\vec{F}\ \vec{M})$$
$$E ::= V \mid C$$
$$M ::= \mathtt{k} \mapsto E$$
$$F ::= \lambda\vec{x}\mu\alpha(E)$$

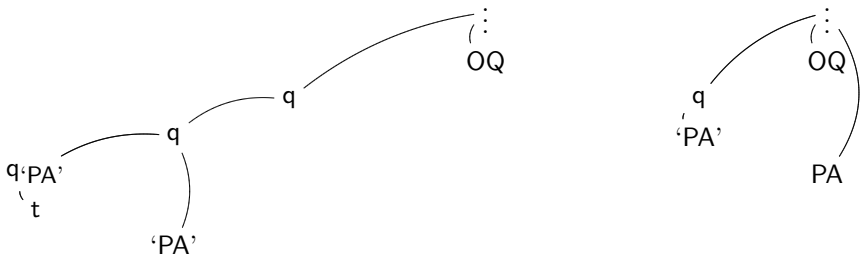
This more accurately reflects the game semantics in that each syntactic class corresponds to a certain kind of move—V for Player answers, C for Player questions, M for Opponent answers and F for Opponent questions—and allows us to easily identify the fragments corresponding to our three semantic constraints: to impose the bracketing condition, we simply erase all $\mu\alpha$s and $[\alpha]$s; to impose F-rigidity, we restrict M by $M' ::= \mathtt{k} \mapsto V$ and to impose B-rigidity, we restrict F by $F' ::= \lambda\vec{x}\mu\alpha(C)$.

4 Factorizations

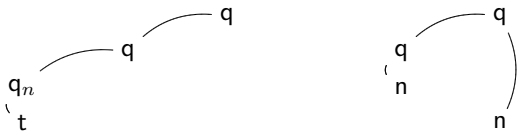
We now present factorizations, one for each of our constraints, each of which forces an innocent strategy to satisfy its constraint whilst preserving the other two. We fix, once and for all, an encoding of the answers (in a given arena) as natural numbers $\mathsf{A} \mapsto$ 'A' and a second encoding of answer-natural number pairs as natural numbers $\mathsf{A}, i \mapsto$ 'A_i'.

4.1 Backward rigidity

To eliminate a violation of B-rigidity—a Player answer preceded by an Opponent question—we transform all $\mathsf{OQ}\,\mathsf{PA}$-ending P-views of σ into two new P-views:



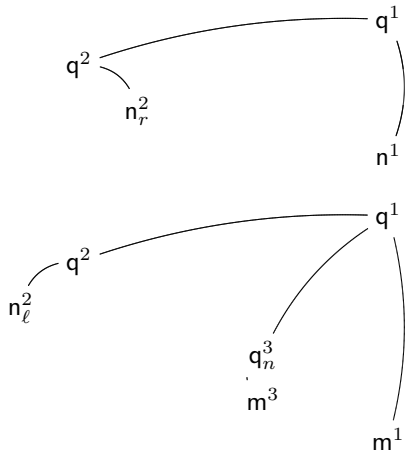
All other P-views remain unchanged. This determines a B-rigid innocent strategy $\overleftarrow{\mathcal{R}}(\sigma) : ((\mathbf{com}^\omega \Rightarrow \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow A$—which is well-bracketed and/or F-rigid if σ is—where we can recover σ by composing with $\mathsf{const} : (\mathbf{com}^\omega \Rightarrow \mathbf{nat}) \Rightarrow \mathbf{nat}$, our *oracle* strategy, defined by the following view function:



4.2 Forward rigidity

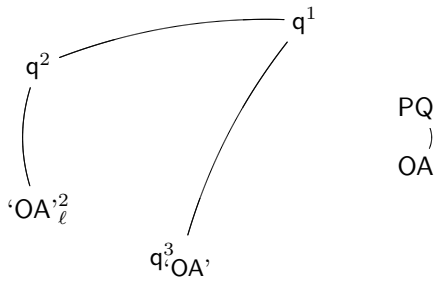
A violation of F-rigidity consists of an Opponent answer followed by a Player question. We would therefore like to transform the view function of $\sigma : A$ by "disguising" OAs as OQs so that σ can play all of its questions in an F-rigid manner. We can do this using $\mathsf{case}_\oplus$, our oracle for F-rigidity, with view function:

$$(\mathbf{nat} \;\oplus\; \mathbf{nat}) \;\Rightarrow\; (\mathbf{nat}^\omega \;\Rightarrow\; \mathbf{nat})$$

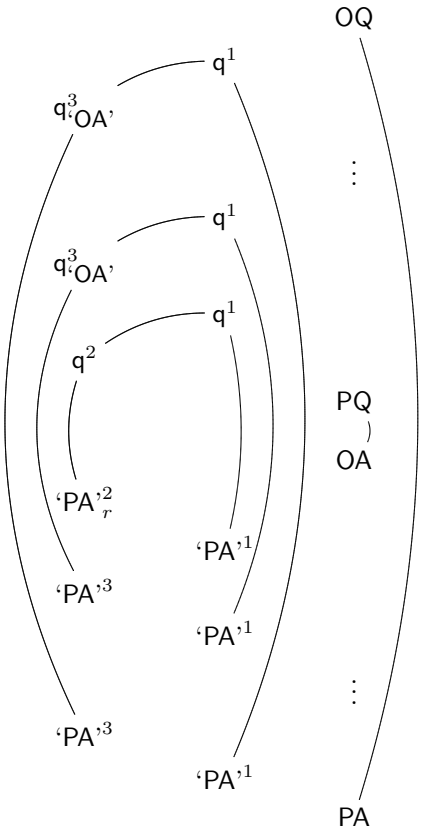


This strategy is deterministic, total, well-bracketed and B-rigid and we have an evident finite version $\mathsf{case}^{k,l}_{\oplus} : (\mathcal{F}_k \oplus \mathcal{F}_\ell) \Rightarrow (\mathcal{F}_\ell^k \Rightarrow \mathcal{F}_\ell)$ where $\mathcal{F}_k$ denotes a base type with k distinct values.

The factorization transforms $\mathsf{PQ} \curvearrowleft \mathsf{OA}$ arches of PQ-ending P-views of σ into $\mathsf{PQ} \curvearrowleft \mathsf{OQ}$ arches on $((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow (\mathbf{nat}^\omega \Rightarrow \mathbf{nat})) \Rightarrow A$:



For *well-bracketed* PA-ending P-views of σ, we must pop all the $\mathsf{q}_{\text{`}\mathsf{OA}\text{'}}$s introduced by the factorization, up to the pending question, so as to preserve bracketing:



The factorized strategy initiates popping by playing $\text{`}\mathsf{PA}\text{'}^2_r$. The oracle propagates this directly to $\mathsf{q}^1$. If the new pending question still belongs to the oracle, the strategy plays $\text{`}\mathsf{PA}\text{'}^3_r$ and the oracle again propagates. This continues until we reach the pending question in A, whence PA is played. Note that this doesn't depend on σ at all: *all* factorized strategies will share the following essentially ***history free*** component, where Player always points to the pending question:

$$\begin{aligned}
&\text{`}\mathsf{PA}\text{'}^1 \mapsto \mathsf{PA}, \text{ if the pending question is in } A\\
&\text{`}\mathsf{PA}\text{'}^1 \mapsto \text{`}\mathsf{PA}\text{'}^3, \text{ otherwise}
\end{aligned}$$

To formally describe the σ-dependent component, we define, for $s \in \ulcorner\sigma\urcorner$, its (empty or singleton) ***principal*** P-view $\overline{s}$ and its set of ***auxiliary*** P-views $\mathcal{A}_s$:

$$\begin{aligned}
\varepsilon &\mapsto (\varepsilon, \varnothing)\\
s \cdot \mathsf{OQ}\,\mathsf{PQ} &\mapsto (\overline{s} \cdot \mathsf{OQ}\,\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{PQ}, \mathcal{A}_s)\\
s \cdot \mathsf{OA}\,\mathsf{PQ} &\mapsto (\overline{s}^{--} \cdot \mathsf{q}_{\text{`}\mathsf{OA}\text{'}}\,\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{PQ}, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\text{`}\mathsf{OA}\text{'}^2_\ell\})\\
s \cdot \mathsf{OQ}\,\mathsf{PA} &\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OQ}\,\mathsf{PA}\})\\
s \cdot \mathsf{OA}\,\mathsf{PA} &\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\mathsf{PA}\}), \text{ if } s \cdot \mathsf{OA}\,\mathsf{PA} \text{ violates bracketing}\\
&\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\text{`}\mathsf{PA}\text{'}^2_r\}), \text{ otherwise}
\end{aligned}$$

Lemma 1. *If σ is an innocent strategy for A then*

$$\overline{\sigma} = \bigcup_{s \in \ulcorner\sigma\urcorner} \overline{s} \cup \mathcal{A}_s$$

is a view function for $((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow (\mathbf{nat}^\omega \Rightarrow \mathbf{nat})) \Rightarrow A$.

We can now formally define $\overrightarrow{\mathcal{R}}(\sigma)$ as the innocent strategy determined by $\overline{\sigma}$ and the history free component. This meshes perfectly with $\mathsf{case}_\oplus$ to implement our factorization:

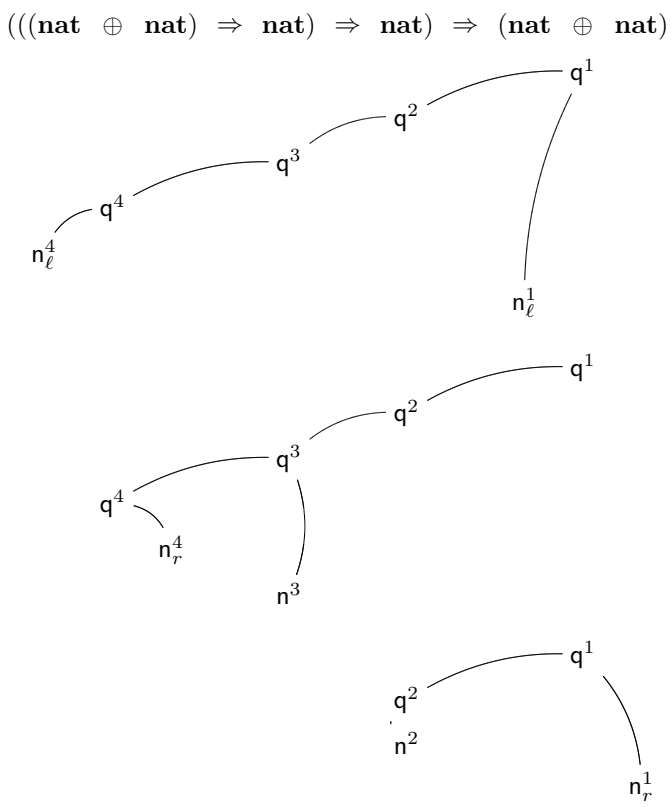
Theorem 1. *If σ is an innocent strategy for A then $\overrightarrow{\mathcal{R}}(\sigma)$ is an F-rigid innocent strategy for $((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow (\mathbf{nat}^\omega \Rightarrow \mathbf{nat})) \Rightarrow A$ satisfying*

$$\mathsf{case}_\oplus \,;\, \overrightarrow{\mathcal{R}}(\sigma) = \sigma.$$

If σ is well-bracketed and/or B-rigid then so is $\overrightarrow{\mathcal{R}}(\sigma)$.

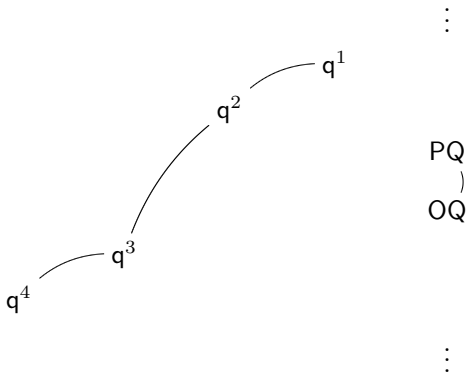
4.3 Bracketing

A violation of bracketing consists of a Player answer pointing beyond the pending question. We thus need to transform P-views in such a way that each $\mathsf{P} \curvearrowleft \mathsf{O}$ arch can, if necessary, be "popped" at a later point so as to recover a well-bracketed strategy. To do this, we use $\mathsf{call/cc}_\oplus$, a variant of $\mathsf{call/cc}$, as oracle:

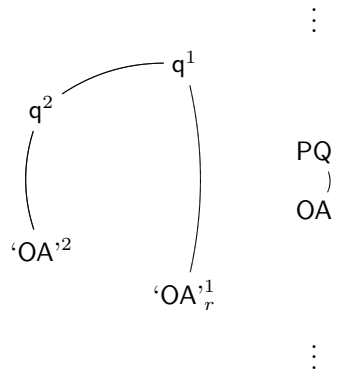


This strategy is deterministic, total and (B- and F-)rigid. If we restrict to finite enumerated types, we have $\mathsf{call/cc}^{k,\ell}_{\oplus} : (((\mathcal{F}_k \oplus \mathcal{F}_\ell) \Rightarrow \mathcal{F}_\ell) \Rightarrow \mathcal{F}_m) \Rightarrow (\mathcal{F}_k \oplus \mathcal{F}_m)$.

Our factorization sandwiches each $\mathsf{PQ} \curvearrowleft \mathsf{OQ}$ arch of a P-view with $\mathsf{q}^1\,\mathsf{q}^2$ and $\mathsf{q}^3\,\mathsf{q}^4$. So, if we subsequently play 'PA_j'$^4_\ell$, $\mathsf{call/cc}_{\oplus}$ replies with 'PA_j'$^1_\ell$, "popping" the arch.



The factorization also transforms $\mathsf{PQ} \curvearrowleft \mathsf{OA}$ arches:



As for F-rigidity, the popping phase is mainly implemented by a history free component: the factorized strategy has only to initiate this process by providing an offset j—the number of OQs of σ between the answer we wish to play and its justifier. Formally, we map each P-view of $\ulcorner\sigma\urcorner$ on A to a principal P-view and set of auxiliary P-views on $((((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow (\mathbf{nat} \oplus \mathbf{nat})) \Rightarrow A$:

$$
\begin{aligned}
\varepsilon &\mapsto (\varepsilon, \varnothing)\\
\mathsf{OQ}\,\mathsf{PQ} &\mapsto (\mathsf{OQ}\,\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{PQ}, \varnothing)\\
s \cdot \mathsf{OQ}\,\mathsf{PQ} &\mapsto (\overline{s} \cdot \mathsf{OQ}\,\mathsf{q}^3 \cdot \mathsf{q}^4\,\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{PQ}, \mathcal{A}_s)\\
s \cdot \mathsf{OA}\,\mathsf{PQ} &\mapsto (\overline{s}^{--} \cdot \text{‘}\mathsf{OA}\text{’}^1_r\,\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{PQ}, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\text{‘}\mathsf{OA}\text{’}^2\})\\
\mathsf{OQ}\,\mathsf{PA} &\mapsto (\mathsf{OQ}\,\mathsf{PA}, \varnothing)\\
s \cdot \mathsf{OA}\,\mathsf{PA} &\mapsto (\overline{s} \cdot \mathsf{OA}\,\mathsf{PA}, \mathcal{A}_s), \text{ if } s \text{ contains no non-initial } \mathsf{OQ}\text{s}\\
s \cdot \mathsf{OQ}\,\mathsf{PA} &\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OQ}\,\mathsf{q}^3 \cdot \mathsf{q}^4\,\text{‘}\mathsf{PA}\text{’}^4_r\}), \text{ if } j = 0\\
&\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OQ}\,\mathsf{q}^3 \cdot \mathsf{q}^4\ \text{‘}\mathsf{PA}_{j-1}\text{’}^4_\ell\}), \text{ otherwise}\\
s \cdot \mathsf{OA}\,\mathsf{PA} &\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\text{‘}\mathsf{OA}\text{’}^2, \overline{s}^{--} \cdot \text{‘}\mathsf{OA}\text{’}^1_r\,\text{‘}\mathsf{PA}\text{’}^4_r\}), \text{ if } j = 0\\
&\mapsto (\varnothing, \mathcal{A}_s \cup \{\overline{s} \cdot \mathsf{OA}\,\text{‘}\mathsf{OA}\text{’}^2, \overline{s}^{--} \cdot \text{‘}\mathsf{OA}\text{’}^1_r\,\text{‘}\mathsf{PA}_{j-1}\text{’}^4_\ell\}), \text{ otherwise}
\end{aligned}
$$

Lemma 2. *If σ is an innocent strategy for A then*

$$\overline{\sigma} = \bigcup_{s \in \ulcorner\sigma\urcorner} \overline{s} \cup \mathcal{A}_s$$

is a view function for $((((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow (\mathbf{nat} \oplus \mathbf{nat})) \Rightarrow A$.

We write $\mathcal{B}(\sigma)$ for the innocent strategy determined by $\overline{\sigma}$ and the (almost) history free component:

$$
\begin{aligned}
\text{‘}\mathsf{PA}_{j+1}\text{’}^1_\ell &\mapsto \text{‘}\mathsf{PA}_j\text{’}^4_\ell\\
\text{‘}\mathsf{PA}_0\text{’}^1_\ell &\mapsto \mathsf{PA}, \text{ if the pending question is initial}\\
&\mapsto \text{‘}\mathsf{PA}\text{’}^4_r, \text{ otherwise}\\
\text{‘}\mathsf{PA}\text{’}^3 &\mapsto \mathsf{PA}
\end{aligned}
$$

Theorem 2. *If σ is an innocent strategy for A then $\mathcal{B}(\sigma)$ is a well-bracketed innocent strategy for* $((((\mathbf{nat} \oplus \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow \mathbf{nat}) \Rightarrow (\mathbf{nat} \oplus \mathbf{nat})) \Rightarrow A$ *satisfying*

$$\mathsf{call}/\mathsf{cc}_{\oplus} \,;\, \mathcal{B}(\sigma) = \sigma.$$

If σ is F-rigid and/or B-rigid then so is $\mathcal{B}(\sigma)$.

4.4 The unary case

If we restrict ourselves to a single base type $\mathtt{com}$ with constants $\Omega, \mathtt{t} : \mathtt{com}$, we can simplify our factorizations and oracles. For B-rigidity, $\mathsf{skip} : \mathbf{com}$ suffices as oracle; the factorization simply inserts $\mathsf{q} \cdot \mathsf{t}$ just before all violating PAs. For F-rigidity, we use $\mathsf{seq} : \mathbf{com} \Rightarrow \mathbf{com} \Rightarrow \mathbf{com}$, the unary $\mathtt{case}$, as oracle; the factorization simply transforms all $\mathsf{PQ} \curvearrowleft \mathsf{OA}$ arches into $\mathsf{q}^1 \curvearrowleft \mathsf{q}^3$, popping (if necessary) as usual.

For bracketing, Peirce's law $\mathtt{cc} : ((\mathbf{com} \Rightarrow \mathbf{com}) \Rightarrow \mathbf{com}) \Rightarrow \mathbf{com}$ acts as oracle. The factorization transforms every $\mathsf{OQ}\,\mathsf{PQ}$-ending P-view $s \in \ulcorner\sigma\urcorner$ by inserting $\mathsf{q}^1 \cdot \mathsf{q}^2\,\mathsf{q}^3 \cdot \mathsf{q}^4$ between the OQ and the PQ, where q^3 points to the q^2 occurring after the "answering justifier" of s: consider the unique, maximal P-view $t \in \ulcorner\sigma\urcorner$ extending s where all O-moves (after s) are answers; if t ends with an answer, the ***answering justifier*** is the question answered by the last move; otherwise (including the case where *no* such maximal P-view exists), the answering justifier is just the pending question of s. Syntactically, this corresponds to the translation $\mu\alpha(t) \mapsto (\mathtt{cc}\ \lambda\alpha(t))$ and $[\alpha]t \mapsto (\alpha)t$. Instead of popping arches one-by-one, this pops *everything* between a PA and its justifying question "in one go". This exploits the property (of the unary case) that we can statically determine the answering justifier of a P-view. In the general case, the answering justifier can only be known at runtime and so we have to pop arches dynamically.

References

1. V. Danos and R. Harmer. The anatomy of innocence. In *Proceedings, Tenth Annual Conference of the European Association for Computer Science Logic*. Springer Verlag, 2001.
2. R. Harmer. Innocent game semantics. Lecture notes, 2004–2006.
3. H. Herbelin. Games and weak-head reduction for classical PCF. In *Proceedings, Third International Conference on Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science. Springer, 1997.
4. J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. *Information and Computation*, 163:285–408, 2000.
5. J. Laird. Full abstraction for functional languages with control. In *Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, 1997.
6. O. Laurent. Polarized games. *Annals of Pure and Applied Logic*, 130(1–3):79–123, December 2004.
7. H. Nickau. Hereditarily sequential functionals. In *Proceedings, Logical Foundations of Computer Science*, Lecture Notes in Computer Science. Springer-Verlag, 1994.