



HAL
open science

Games, mobile processes, Dfunctions

Guilhem Jaber, Davide Sangiorgi

► **To cite this version:**

Guilhem Jaber, Davide Sangiorgi. Games, mobile processes, Dfunctions. CSL 2022 - 30th EACSL Annual Conference on Computer Science Logic, Feb 2022, Göttingen, Germany. pp.1-35. hal-03407123

HAL Id: hal-03407123

<https://hal.science/hal-03407123v1>

Submitted on 28 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

1 Games, mobile processes, and functions

2 **Guilhem Jaber**

3 Université de Nantes, France

4 **Davide Sangiorgi**

5 Università di Bologna, Italy

6 — Abstract —

7 We establish a tight connection between two models of the λ -calculus, namely Milner’s encoding
8 into the π -calculus (precisely, the Internal π -calculus), and operational game semantics (OGS). We
9 first investigate the operational correspondence between the behaviours of the encoding provided
10 by π and OGS. We do so for various LTSs: the standard LTS for π and a new ‘concurrent’ LTS
11 for OGS; an ‘output-prioritised’ LTS for π and the standard alternating LTS for OGS. We then
12 show that the equivalences induced on λ -terms by all these LTSs (for π and OGS) coincide. These
13 connections allow us to transfer results and techniques between π and OGS. In particular we import
14 up-to techniques from π onto OGS and we derive congruence and compositionality results for OGS
15 from those of π . The study is illustrated for call-by-value; similar results hold for call-by-name.

16

17

18 **1** Introduction

19 The topic of the paper is the comparison between *Operational Game semantics* (OGS) and
20 the *π -calculus*, as generic models or frameworks for the semantics of higher-order languages.

21 Game semantics [4, 20] provides intentional models of higher-order languages, where the
22 denotation of a program brings up its possible interactions with the surrounding context.
23 Distinct points of game semantics are the rich categorical structure and the emphasis on
24 compositionality. Game semantics provides a modular characterization of higher-order
25 languages with computational effects like control operators [24], mutable store [3, 5] or
26 concurrency [15, 26]. This gives rise to the “Semantic Cube” [2], a characterization of the
27 absence of such computational effects in terms of appropriate restrictions on the interactions,
28 with conditions like *alternation*, *well-bracketing*, *visibility* or *innocence*. For instance, well-
29 bracketing corresponds to the absence of control operators like call/cc.

30 Game semantics has spurred Operational Game Semantics (OGS) [16, 22, 23, 27, 29], as a
31 way to describe the interactions of a program with its environment by embedding programs
32 into appropriate configurations and then defining rules that turn such configurations into
33 an LTS. Besides minor differences on the representation of causality between actions, the
34 main distinction with “standard” game semantics is in the way in which the denotation
35 of programs is obtained: via an LTS, rather than, compositionally, by induction on the
36 structure of the programs (or their types). It is nonetheless possible to establish a formal
37 correspondence between these two representations [29].

38 OGS is particularly effective on higher-order programs. To avoid being too intensional,
39 functional values exchanged between the program and its environment are represented as
40 atoms, seen as free variables. Therefore OGS configurations include open terms. The basic
41 actions in the LTS produced by OGS represent the calls and returns of functions between a
42 program and its environment. The OGS semantics has been shown fully-abstract, that is, to
43 characterize observational equivalence, for a wide class of programming languages, including
44 effectful subsets of ML [21, 27], fragments of Java [23], aspect-oriented programs [22]. The
45 conditions in the above-mentioned Semantic Cube (alternation, well-bracketing, etc.) equally



© Guilhem Jaber and Davide Sangiorgi



Leibniz International Proceedings in Informatics

LIPICIS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany

46 apply to OGS.

47 In this paper, we consider forms of OGS for the pure untyped call-by-value λ -calculus,
 48 which enforce some of such conditions. Specifically we consider: an *Alternating* OGS, where
 49 only one term can be run at a time, and the control on the interactions alternates between
 50 the term and the environment; and a *Concurrent* OGS, where multiple terms can be run in
 51 parallel.

52 The π -calculus is the paradigmatical name-passing calculus, that is, a calculus where
 53 names (a synonym for ‘channels’) may be passed around. In the literature about the π -
 54 calculus, and more generally in Programming Language theory, Milner’s work on functions
 55 as processes [32], which shows how the evaluation strategies of *call-by-name* λ -calculus and
 56 *call-by-value* λ -calculus [1, 35] can be faithfully mimicked, is generally considered a landmark.
 57 The work promotes the π -calculus to be a model for higher-order programs, and provides
 58 the means to study λ -terms in contexts other than the purely sequential ones and with the
 59 instruments available to reason about processes. In the paper, π -calculus is actually meant to
 60 be the *Internal* π -calculus ($I\pi$), a subset of the original π -calculus in which only fresh names
 61 may be exchanged among processes [41]. The use of $I\pi$ avoids a few shortcomings of Milner’s
 62 encodings, notably for call-by-value; e.g., the failure of the β_v rule (i.e., the encodings of
 63 $(\lambda x. M)V$ and $M\{V/x\}$ may be behaviourally distinguishable in π).

64 Further investigations into Milner’s encodings [11, 42] have revealed what is the equivalence
 65 induced on λ -terms by the encodings, whereby two λ -terms are equal if their encodings are
 66 behaviourally equivalent (i.e., bisimilar) $I\pi$ terms. In call-by-value, this equivalence is *eager*
 67 *normal-form bisimilarity* [28], a tree structure proposed by Lassen (and indeed sometimes
 68 referred to as ‘Lassen’s trees’) as the call-by-value counterpart of Böhm Trees (or Lévy-Longo
 69 Trees).

70 In a nutshell, when used to give semantics to a language, major strengths of the π -calculus
 71 are its algebraic structure and the related algebraic properties and proof techniques; major
 72 strengths of OGS are its proximity to the source language — the configurations of OGS are
 73 built directly from the terms of the source language, as opposed to an encoding as in the
 74 π -calculus — and its flexibility — the semantics can be tuned to account for specific features
 75 of the source language like control operators or references.

76 The general goal of this paper is to show that there is a tight and precise correspond-
 77 ence between OGS and π -calculus as models of programming languages, and that such a
 78 correspondence may be profitably used to take advantage of the strengths of the two models.
 79 We carry out the above program in the specific case of (untyped) call-by-value λ -calculus,
 80 Λ_V , which is richer and (as partly suggested above) with some more subtle aspects than
 81 call-by-name. However similar results also hold for call-by-name; see Appendix I for the
 82 technical details for more comments on it. Analogies and similarities between game semantics
 83 and π -calculus have been pointed out in various papers in the literature (e.g., [6, 19]; see
 84 Section 9), and used to, e.g., explain game semantics using π -like processes, and enhance type
 85 systems for π -terms. In this paper, in contrast, we carry out a direct comparison between
 86 the two models, on their interpretation of functions.

87 We take the (arguably) canonical representations of Λ_V into $I\pi$ and OGS. The latter
 88 representation is Milner’s encoding, rewritten in $I\pi$. We consider two variant behaviours for
 89 the $I\pi$ terms, respectively produced by the ordinary LTS of $I\pi$, and by an ‘output-prioritised’
 90 LTS, opLTS, in which input actions may be observed only in absence of outputs and internal
 91 actions. Intuitively, the opLTS is intended to respect sequentiality constraints in the $I\pi$
 92 terms: an output action stands for an ongoing computation (for instance, returning the
 93 result of a previous request) whereas an input action starts a new computation (for instance,

a request of a certain service); therefore, in a sequential system, an output action should have priority over input actions. For OGS, the Λ_V representation is the straightforward adaptation of the OGS representations of typed λ -calculi in the literature, e.g., [27].

We then develop a thorough comparison between the behaviours of the OGS and $I\pi$ representations. For this we define a mapping from OGS configurations to $I\pi$ processes. We also exploit the fact that, syntactically, the actions in the OGS and $I\pi$ LTSs are the same. We derive a tight correspondence between the two models, which allows us to transfer techniques and to switch freely between the two models in the analysis of the OGS and $I\pi$ representations of Λ_V , so to establish new results or obtain new proofs. On these aspects, our main results are the following:

1. We show that the representation of Λ_V in the Alternating OGS is behaviourally the same as the representation in $I\pi$ assuming the opLTS. Thus the semantics on λ -terms induced by the OGS and $I\pi$ representations coincide. The same results are obtained between the Concurrent OGS and $I\pi$ under its ordinary LTS.
2. We transfer ‘bisimulation up-to techniques’ for $I\pi$, notably a form of ‘up-to context’, onto (Concurrent) OGS. The result is a powerful technique, called ‘up-to composition’, that allows us to split an OGS configuration into more elementary configurations during the bisimulation game.
3. We show that the semantics induced on Λ_V by the Alternating and by the Concurrent OGS are the same, both when the equality in OGS is based on traces and when it is based on bisimulation. In other words, all the OGS views of Λ_V (Alternating or Concurrent, traced-based or bisimulation-based) coincide. Moreover, we show that such induced semantics is the equality of Lassen’s trees. We derive the result in two ways: one in which we directly import it from $I\pi$; the other in which we lift eager normal-form bisimulations into OGS bisimulations via the up-to-composition technique.
4. We derive congruence and compositionality properties for the OGS semantics, as well as a notion of tensor product over configurations that computes interleavings of traces.

The results about OGS in (2-4) are obtained exploiting the mapping into $I\pi$ and its algebraic properties and proof techniques, as well as the up-to-composition technique for OGS imported from $I\pi$.

Structure of the paper. Sections 2 to 5 contain background material: general notations, $I\pi$, Λ_V , the representations of Λ_V in the Alternating OGS (A-OGS) and in $I\pi$. The following sections contain the new material. In Section 6 we study the relationship between the two Λ_V representations, in $I\pi$ using the output-prioritised LTS. In Section 7 we establish a similar relationship between a new Concurrent OGS (C-OGS) and $I\pi$ using its ordinary LTS. We also transport up-to techniques onto OGS, and prove that all the semantics of Λ_V examined (OGS, $I\pi$, traces, bisimulations) coincide. We import compositionality results for OGS from $I\pi$ in Section 8.

2 Notations

In the paper we use various LTSs and behavioural relations for them, both for OGS and for the π -calculus. In this section we introduce or summarise common notations.

We use a tilde, like in \tilde{a} , for (possibly empty) tuples of objects (usually names). Let $K \xrightarrow{\mu}_g K'$ be a generic LTS (for OGS or $I\pi$; the grammar for actions in the LTSs for OGS and $I\pi$ will be the same). Actions, ranged over by μ , can be of the form $a(\tilde{b})$, $\bar{a}(\tilde{b})$, τ , and (\tilde{a}) , where τ , called *silent* or (*invisible*) action, represents an internal step in K , that is, an action that does not require interaction with the outside, and (\tilde{a}) is a special action performed by

XX:4 Games, mobile processes, and functions

141 abstractions in $\mathbb{I}\pi$ and initial configurations in OGS. If $\mu \neq \tau$ then μ is a *visible* action; we
 142 use ℓ to range over them. We sometimes abbreviate $\xrightarrow{\tau}_g$ as \rightarrow_g . We write \Longrightarrow_g for the
 143 reflexive and transitive closure of $\xrightarrow{\tau}_g$. We also write $K \xRightarrow{\mu}_g K'$ if $K \Longrightarrow_g \xrightarrow{\mu}_g \Longrightarrow_g K'$
 144 (the composition of the three relations). Then $\xRightarrow{\mu}_g$ is $\xrightarrow{\mu}_g$ if $\mu \neq \tau$, and \Longrightarrow_g if $\mu = \tau$.

145 *Traces*, ranged over by s , are finite (and possibly empty) sequences of visible actions.
 146 If $s = \ell_1, \dots, \ell_n$ ($n \geq 0$), then $K \xRightarrow{s}_g K'$ holds if there are K_0, \dots, K_n with $K_0 = K$,
 147 $K_n = K'$, and $K_i \xrightarrow{\ell_{i+1}}_g K_{i+1}$ for $0 \leq i < n$; and $K \xRightarrow{s}_g K'$ if there is K' with $K \xrightarrow{s}_g K'$.

148 Two states K_1, K_2 of the LTS are *trace equivalent*, written $K_1 \simeq_g K_2$, if $(K_1 \xRightarrow{s}_g K_2)$ iff
 149 $(K_2 \xRightarrow{s}_g K_1)$, for all s .

150 Similarly, *bisimilarity*, written \approx_g , is the largest symmetric relation on the state of the
 151 LTS such that whenever $K_1 \approx_g K_2$ then $K_1 \xrightarrow{\mu}_g K'_1$ implies there is K'_2 with $K_2 \xrightarrow{\mu}_g K'_2$
 152 and $K'_1 \approx_g K'_2$. For instance, in the $\mathbb{I}\pi$ LTS $\xrightarrow{\mu}_\pi$ of Section 3.1, $P \simeq_\pi Q$ means that the $\mathbb{I}\pi$
 153 processes P and Q are trace equivalent, and $P \approx_\pi Q$ means that they are bisimilar.

154 **► Remark 1 (bound names).** In an action $a(\tilde{b})$ or $\bar{a}(\tilde{b})$ or (\tilde{b}) , name a is *free* whereas \tilde{b} are
 155 *bound*; the free and bound names of a trace are defined accordingly. Throughout the paper,
 156 in any statement (concerning OGS or $\mathbb{I}\pi$), the bound names of an action or of a trace that
 157 appears in the statement are supposed to be all *fresh*; i.e., all distinct from each other and
 158 from the free names of the objects in the statement. ◀

3 Background

3.1 The Internal π -calculus

161 The Internal π -calculus, $\mathbb{I}\pi$, is, intuitively, a subset of the π -calculus in which all outputs
 162 are bound. This is syntactically enforced by having outputs written as $\bar{a}(\tilde{b})$ (which in the
 163 π -calculus would be an abbreviation for $\nu \tilde{b} \bar{a}(\tilde{b})$). All tuples of names in $\mathbb{I}\pi$ are made of
 164 pairwise distinct components. *Abstractions* are used to write name-parametrised processes,
 165 for instance, when writing recursive process definitions. The instantiation of the parameters
 166 of an abstraction B is done via the *application* construct $B\langle\tilde{a}\rangle$. Processes and abstractions
 167 form the set of *agents*, ranged over by T . Lowercase letters a, b, \dots, x, y, \dots range over the
 168 infinite set of names. The grammar of $\mathbb{I}\pi$ is thus:

$$\begin{array}{ll}
 P \triangleq & 0 \mid a(\tilde{b}).P \mid \bar{a}(\tilde{b}).P \mid \nu a P \mid P_1 \mid P_2 \mid !a(\tilde{b}).P \mid B\langle\tilde{a}\rangle \quad (\text{processes}) \\
 B \triangleq & (\tilde{a}) P \mid K \quad (\text{abstractions})
 \end{array}$$

170 The operators have the usual meaning; we omit the standard definition of free names, bound
 171 names, and names of an agent, respectively indicated with $\text{fn}(-)$, $\text{bn}(-)$, and $\text{n}(-)$. In the
 172 grammar, K is a constant, used to write recursive definitions. Each constant K has a defining
 173 equation of the form $K \triangleq (\tilde{x}) P$, where $(\tilde{x}) P$ is name-closed (that is, without free names);
 174 \tilde{x} are the formal parameters of the constant. Replication could be avoided in the syntax
 175 since it can be encoded with recursion. However its semantics is simple, and it is useful in
 176 encodings.

177 An application redex $((\tilde{x})P)\langle\tilde{a}\rangle$ can be normalised as $P\{\tilde{a}/\tilde{x}\}$. An agent is *normalised* if
 178 all such application redexes have been contracted. In the remainder of the paper *we identify*
 179 *an agent with its normalised expression*.

180 Since the calculus is polyadic, we assume a *sorting system* [31] to avoid disagreements in
 181 the arities of the tuples of names. Being not essential, it will not be presented here.

182 Operational semantics and behavioural relations

183 In the LTS for $\mathcal{I}\pi$, recalled in Appendix A, transitions are of the form $T \xrightarrow{\mu}_{\pi} T'$, where the
184 bound names of μ are fresh, i.e., they do not appear free in T .

185 Trace equivalence (\simeq_{π}) and bisimilarity (\approx_{π}) have been defined in Section 2. We refer to
186 Appendix A for the standard definition of *expansion*, written \lesssim_{π} . (The expansion relation
187 \lesssim_{π} is an asymmetric variant of \approx_{π} in which, intuitively, $P \lesssim_{\pi} Q$ holds if $P \approx_{\pi} Q$ but also Q
188 has at least as many τ -moves as P .) All behavioural relations are extended to abstractions by
189 requiring ground instantiation of the parameters; this is expressed by means of a transition;
190 e.g., the action $(\tilde{x}) P \xrightarrow{\tilde{x}}_{\pi} P$ (see rule **abs** in Appendix A).

191 The “up-to” techniques

192 The “up-to” techniques allow us to reduce the size of a relation \mathcal{R} to exhibit for proving
193 bisimilarities. Our main up-to technique will be *up-to context and expansion* [40], which
194 admits the use of contexts and of behavioural equivalences such as expansion to achieve the
195 closure of a relation in the bisimulation game. So the bisimulation clause becomes:

196 ■ if $P \mathcal{R} Q$ and $P \xrightarrow{\mu} P''$ then there are a static context C_{ctx} and processes P' and Q' s.t.
197 $P'' \pi \gtrsim C_{\text{ctx}}[P']$, $Q \xrightarrow{\mu} \pi \gtrsim C_{\text{ctx}}[Q']$ and $P' \mathcal{R} Q'$ (*)
198 where a *static context* is a context of the form $\nu \tilde{c} (R \mid \cdot)$.

199 We will also employ: *bisimulation up-to* \approx_{π} [30], whereby bisimilarity itself is employed
200 to achieve the closure of the candidate relation during the bisimulation game; a variant
201 of bisimulation up-to context and expansion, called *bisimulation up-to context and up-to*
202 $(\pi \gtrsim, \approx_{\pi})$, in which, in (*), when μ is a visible action, expansion is replaced by the coarser
203 bisimilarity, at the price of imposing that the static context C_{ctx} cannot interact with the
204 processes P or Q . (This technique, as far as we know, does not appear in the literature.)
205 Details on these techniques may be found in Appendix A.

206 3.2 The Call-By-Value λ -calculus

207 The grammar of the untyped call-by-value λ -calculus, Λ_V , has values V , terms M , evaluation
208 contexts E , and general contexts C :

$$209 \begin{array}{ll} \text{Vals} & V \triangleq x \mid \lambda x. M \\ \text{Terms} & M, N \triangleq V \mid MN \end{array} \qquad \begin{array}{ll} \text{ECtxs} & E \triangleq [\cdot] \mid VE \mid EM \\ \text{Ctxs} & C \triangleq [\cdot] \mid \lambda x. C \mid MC \mid CM \end{array}$$

where $[\cdot]$ stands for the hole of a context. The call-by-value reduction \rightarrow_v has two rules:

$$\frac{}{(\lambda x. M)V \rightarrow_v M\{V/x\}} \qquad \frac{M \rightarrow_v N}{E[M] \rightarrow_v E[N]}$$

210 In the following, we write $M \Downarrow M'$ to indicate that $M \Rightarrow_v M'$ with M' an eager normal
211 form, that is, either a value or a stucked call $E[xV]$.

212 4 Operational Game Semantics

213 We introduce the representation of Λ_V in OGS. The LTS produced by the representation of
214 a term intends to capture the possible interactions between the term and its environment.
215 Values exchanged between the term and the environment are represented by names, akin to
216 free variables, called *variable names* and ranged over by x, y, z . Continuations (i.e., evaluation

$(P\tau)$	$\langle M, p, \gamma, \phi \rangle$	$\xrightarrow{\tau}_a$	$\langle N, p, \gamma, \phi \rangle$	when $M \rightarrow_V N$
(PA)	$\langle V, p, \gamma, \phi \rangle$	$\xrightarrow{\bar{p}(x)}_a$	$\langle \gamma \cdot [x \mapsto V], \phi \uplus \{x\} \rangle$	
(PQ)	$\langle E[xV], p, \gamma, \phi \rangle$	$\xrightarrow{\bar{x}(y,q)}_a$	$\langle \gamma \cdot [y \mapsto V] \cdot [q \mapsto (E, p)], \phi \uplus \{y, q\} \rangle$	
(OA)	$\langle \gamma \cdot [q \mapsto (E, p)], \phi \rangle$	$\xrightarrow{q(x)}_a$	$\langle E[x], p, \gamma, \phi \uplus \{x\} \rangle$	
(OQ)	$\langle \gamma, \phi \rangle$	$\xrightarrow{x(y,p)}_a$	$\langle Vy, p, \gamma, \phi \uplus \{y, p\} \rangle$	when $\gamma(x) = V$
(IOQ)	$\langle [? \mapsto M], \phi \rangle$	$\xrightarrow{(p)}_a$	$\langle M, p, \varepsilon, \phi \uplus \{p\} \rangle$	

■ **Figure 1** The LTS for the Alternating OGS (A-OGS)

217 contexts) are also represented by names, called *continuation names* and ranged over by p, q, r .
 218 Actions μ have been introduced in Section 2. In OGS, we have five kinds of (visible) actions:
 219 ■ *Player Answers* (PA), $\bar{p}(x)$, and *Opponent Answers* (OA), $p(x)$, that exchange a variable
 220 x through a continuation name p ;
 221 ■ *Player Questions* (PQ), $\bar{x}(y, p)$, and *Opponent Questions* (OQ), $x(y, p)$, that exchange a
 222 variable y and a continuation name p through a variable x ;
 223 ■ *Initial Opponent Questions* (IOQ), (p) , that introduce the initial continuation name p .

224 ► **Remark 2.** The denotation of terms is usually represented in game semantics using the
 225 notion of *pointer structure* rather than traces. A pointer structure is defined as a sequence of
 226 *moves*, together with a pointer from each move (but the initial one) to a previous move that
 227 ‘justifies’ it. Taking a trace s , one can reconstruct this pointer structure in the following way:
 228 an action μ is *justified* by an action μ' if the free name of μ is bound by μ' in s (here we are
 229 taking advantage of the ‘freshness’ convention on the bound names of traces, Remark 1).

230 *Environments*, ranged over by γ , maintain the association from names to values and
 231 evaluations contexts, and are partial maps. A single mapping is either of the form $[x \mapsto V]$
 232 (the variable x is mapped onto the value V), or $[p \mapsto (E, q)]$ (the continuation name p is
 233 mapped onto the pair of the evaluation context E and the continuation q).

234 There are two main kinds of configurations F : *active configurations* $\langle M, p, \gamma, \phi \rangle$ and
 235 *passive configurations* $\langle \gamma, \phi \rangle$, where M is a term, p a continuation name, γ an environment
 236 and ϕ a set of names called its *name-support*. Names in $\text{dom}(\gamma)$ are called *P-names*, and
 237 those in $\phi \setminus \text{dom}(\gamma)$ are called *O-names*. So we obtain a *polarity function* pol_F associated to
 238 F , defined as the partial maps from ϕ to $\{O, P\}$ mapping names to their polarity. In the
 239 following, we only consider *valid configurations*, for which:

- 240 ■ $\text{dom}(\gamma) \subseteq \phi$
- 241 ■ $\text{fv}(M), p$ are O-names;
- 242 ■ for all $a \in \text{dom}(\gamma)$, the names appearing in $\gamma(a)$ are O-names.

243 The LTS is introduced in Figure 1. It is called *Alternating*, since, forgetting the $P\tau$
 244 transition, it is bipartite between active configurations, that perform Player actions, and
 245 passive configurations, that perform Opponent actions. Accordingly, we call Alternating
 246 the resulting OGS, abbreviated A-OGS. In the OA rule, E is “garbage-collected” from γ , a
 247 behavior corresponding to *linear continuations*. More details on the rules may be found in
 248 Appendix C.

249 To build the denotation of a term M , we introduce an *initial configuration* associated
 250 to it, written $\langle [? \mapsto M], \phi \rangle$, with ϕ the set of free variables we start with. When this set is
 251 taken to be the free variables of M , we simply write it as $\langle M \rangle$. In the initial configuration

$$\begin{aligned} \mathcal{V}[[V]] &\triangleq (p) \bar{p}(y). \mathcal{V}^*[[V]](y) & \mathcal{V}^*[[\lambda x. M]] &\triangleq (y) !y(x, q). \mathcal{V}[[M]](q) & \mathcal{V}^*[[x]] &\triangleq (y) y \triangleright x \\ \mathcal{V}[[MN]] &\triangleq (p) \nu q (\mathcal{V}[[M]](q) \mid q(y). \nu r (\mathcal{V}[[N]](r) \mid r(w). \bar{y}(w', p'). (w' \triangleright w \mid p' \triangleright p))) \end{aligned}$$

■ **Figure 2** The encoding of call-by-value λ -calculus into $I\pi$

252 the choice of the continuation name p is made, by performing an Initial Opponent question
253 (IOQ). (Formally, initial configurations should be considered as passive configurations.)

254 5 The encoding of call-by-value λ -calculus into the π -calculus

We recall here Milner's encoding of call-by-value λ -calculus, transplanted into $I\pi$. The core of any encoding of the λ -calculus into a process calculus is the translation of function application. This becomes a particular form of parallel combination of two processes, the function and its argument; β -reduction is then modelled as a process interaction. As in OGS, so in $I\pi$ the encoding uses *continuation names* p, q, r, \dots , and *variable names* $x, y, v, w \dots$. Figure 2 presents the encoding. Process $a \triangleright b$ represents a *link* (sometimes called forwarder; for readability we have adopted the infix notation $a \triangleright b$ for the constant \triangleright). It transforms all outputs at a into outputs at b ; thus the body of $a \triangleright b$ is replicated, unless a and b are continuation names:

$$\triangleright \triangleq \begin{cases} (p, q). p(x). \bar{q}(y). y \triangleright x & \text{if } p, q \text{ are continuation names} \\ (x, y). !x(z, p). \bar{y}(w, q). (q \triangleright p \mid w \triangleright z) & \text{if } x, y \text{ are variable names} \end{cases}$$

255 The equivalence induced on call-by-value λ -terms by their encoding into $I\pi$ coincides with
256 Lassen's *eager normal-form (enf) bisimilarity* [28]. That is, $\mathcal{V}[[M]] \approx_{\pi} \mathcal{V}[[N]]$ iff M and N
257 are enf-bisimilar [11]. In proofs about the behaviour of the $I\pi$ representation of λ -terms we
258 sometime follow [11] and use an optimisation of Milner's encoding, reported in Appendix D
259 together with its correctness (Lemma 47).

260 6 Relationship between $I\pi$ and A-OGS

261 To compare the A-OGS and $I\pi$ representations of the (call-by-value) λ -calculus, we set a
262 mapping from A-OGS configurations and environments to $I\pi$ processes. The mapping is
263 reported in Figure 3. It is an extension of Milner's encoding of the λ -calculus and is therefore
264 indicated with the same symbol \mathcal{V} . The mapping uses a representation of environments γ as
265 associative lists.

266 ► **Remark 3.** The encoding of a configuration F with name-support ϕ does not depend
267 on ϕ . This name-support ϕ is used in OGS both to enforce freshness of names, and to
268 deduce the polarity of names, as represented by the function pol . And indeed, the process
269 $\mathcal{V}[[F]]$ has its set of free names included in ϕ , and uses P -names in outputs and O -names in
270 inputs. The polarity property could be stated in π -calculus using i/o-sorting [34]. Indeed, a
271 correspondence between arenas of game semantics (used to enforce polarities of moves) and
272 sorting has been explored [18, 19].

273 6.1 Operational correspondence

274 The following theorems establish the operational correspondence between the A-OGS and
275 $I\pi$ representations. In Theorem 4, as well as in following theorems such as Theorems 5, 7,

<p style="text-align: center;">Encoding of environments:</p> $\begin{aligned} \mathcal{V}[[y \mapsto V] \cdot \gamma'] &\triangleq \mathcal{V}^*[[V]\langle y \rangle \mid \mathcal{V}[\gamma']] \\ \mathcal{V}[[q \mapsto (E, p)] \cdot \gamma'] &\triangleq q(x). \mathcal{V}[[E[x]]\langle p \rangle \mid \mathcal{V}[\gamma']] \\ \mathcal{V}[\varepsilon] &\triangleq 0 \end{aligned}$	<p style="text-align: center;">Encoding of configurations:</p> $\begin{aligned} \mathcal{V}[\langle M, p, \gamma, \phi \rangle] &\triangleq \mathcal{V}[[M]\langle p \rangle \mid \mathcal{V}[\gamma]] \\ \mathcal{V}[\langle \gamma, \phi \rangle] &\triangleq \mathcal{V}[\gamma] \\ \mathcal{V}[\langle [? \mapsto M], \phi \rangle] &\triangleq \mathcal{V}[[M]] \end{aligned}$
---	---

■ **Figure 3** From OGS environments and configurations to $I\pi$

276 and 13, the appearance of the expansion relation $\pi \succsim$ (in place of the coarser \approx_π), in the
 277 statement about silent actions, is essential, both to derive the statement in the theorems
 278 about visible actions, and to use the theorems in up-to techniques for $I\pi$ (more generally, in
 279 applications of the theorems in which one reasons about the number of steps performed).

280 ► **Theorem 4.**

- 281 1. If $F \Longrightarrow_a F'$, then $\mathcal{V}[[F]] \Longrightarrow_\pi \pi \succsim \mathcal{V}[[F']]$;
 282 2. If $F \xrightarrow{\ell}_a F'$, then $\mathcal{V}[[F]] \xrightarrow{\ell}_\pi \approx_\pi \mathcal{V}[[F']]$.

283 ► **Theorem 5.**

- 284 1. If $\mathcal{V}[[F]] \Longrightarrow_\pi P$ then there is F' such that $F \Longrightarrow_a F'$ and $P \pi \succsim \mathcal{V}[[F']]$;
 285 2. If $\mathcal{V}[[F]] \xrightarrow{\ell}_\pi P$ and ℓ is an output, then there is F' such that $F \xrightarrow{\ell}_a F'$ and $P \pi \succsim \mathcal{V}[[F']]$;
 286 3. If F is passive and $\mathcal{V}[[F]] \xrightarrow{\ell}_\pi P$, then there is F' such that $F \xrightarrow{\ell}_a F'$ and $P \approx_\pi \mathcal{V}[[F']]$.

287 In Theorem 5, a clause is missing for input actions from $\mathcal{V}[[F]]$ when F active. Indeed such
 288 actions are possible in $I\pi$, stemming from the (encoding of the) environment of F , whereas
 289 they are not possible in A-OGS. This is rectified in Section 6.2, introducing a constrained
 290 LTS for $I\pi$, and in Section 7, considering a concurrent OGS.

291 ► **Corollary 6.** If $F \xrightarrow{s}_a$ then also $\mathcal{V}[[F]] \xrightarrow{s}_\pi$.

292 **6.2 An output-prioritised Transition System**

293 We define an LTS for $I\pi$ in which input actions are visible only if no output can be consumed,
 294 either as a visible action or through an internal action (i.e., syntactically the process has
 295 no unguarded output). The new LTS, called *output-prioritised* and indicated as opLTS, is
 296 defined on the top of the ordinary one by means of the two rules below. A process P is *input*
 297 *reactive* if whenever $P \xrightarrow{\mu}_\pi P'$, for some μ, P' then μ is an input action.

$$298 \quad \frac{P \xrightarrow{\mu}_\pi P' \quad P \text{ input reactive}}{P \xrightarrow{\mu}_{o\pi} P'} \qquad \frac{P \xrightarrow{\mu}_\pi P'}{P \xrightarrow{\mu}_{o\pi} P'} \mu \text{ is an output or } \tau \text{ action}$$

299 The opLTS captures an aspect of *sequentiality* in π -calculi: a free input prefix is to be
 300 thought of as a service offered to the external environment; in a sequential system such a
 301 service is available only if there is no ongoing computations due to previous interrogations
 302 of the server. An ongoing computation is represented by a τ -action, indicating a step of
 303 computation internal to the server, or an output, indicating either an answer to a client
 304 or a request to an external server. The constraint imposed by the new LTS could also be
 305 formalised compositionally, see Appendix A.

306 Under the opLTS, the analogous of Theorem 4 continue to hold: in A-OGS configurations,
 307 input transitions only occur in passive configurations, and the encodings of passive configura-
 308 tions are input-reactive processes. However, now we have the full converse of Theorem 5 and,
 309 as a consequence, we can also establish the converse direction of Corollary 6.

310 ▶ **Theorem 7.**

- 311 1. If $\mathcal{V}[[F]] \xrightarrow{\tau}_{\text{op}\pi} P$ then there is F' such that $F \Longrightarrow_a F'$ and $P \pi \gtrsim \mathcal{V}[[F']]$;
 312 2. If $\mathcal{V}[[F]] \xrightarrow{\ell}_{\text{op}\pi} P$ then there is F' such that $F \xrightarrow{\ell}_a F'$ and $P \approx_\pi \mathcal{V}[[F']]$.

313 ▶ **Corollary 8.** For any configuration F and trace s , we have $F \xrightarrow{s}_a$ iff $\mathcal{V}[[F]] \xrightarrow{s}_{\text{op}\pi}$.

314 ▶ **Remark 9.** We recall that, following Remark 1 on the usage of bound names, in Corollary 8
 315 the bound names in s are fresh; thus they do not appear in F . (Similarly, in Theorems 5
 316 and 7 for the bound names in ℓ).

317 For both results we first establish a correspondence result on strong transitions. See
 318 Appendix D for details.

319 ▶ **Remark 10.** Corollary 8 relies on Theorems 4 and 7. The corollary talks about the opLTS
 320 of $I\pi$; however the theorems make use of the ordinary expansion relation \lesssim_π , that is defined
 321 on the ordinary LTS. Such uses of expansion can however be replaced by expansion on the
 322 opLTS (defined as ordinary expansion, but on the opLTS). For more details on this, see
 323 Appendix B.

324 As a consequence of Corollary 8, trace equivalence is the same, on A-OGS configurations
 325 and on the encoding $I\pi$ terms. Moreover, from Theorem 7 the same result holds under a
 326 bisimulation semantics. Further, since the LTS produced by A-OGS is deterministic, its trace
 327 semantics coincides with its bisimulation semantics. We can thus conclude as in Corollary 11.
 328 We recall that $\simeq_{\text{op}\pi}$ and $\approx_{\text{op}\pi}$ are, respectively, trace equivalence and bisimilarity between $I\pi$
 329 processes in the opLTS; similarly for \simeq_a and \approx_a between A-OGS terms.

330 ▶ **Corollary 11.** For any F, F' we have: $F \simeq_a F'$ iff $\mathcal{V}[[F]] \simeq_{\text{op}\pi} \mathcal{V}[[F']]$ iff $F \approx_a F'$ iff
 331 $\mathcal{V}[[F]] \approx_{\text{op}\pi} \mathcal{V}[[F']]$.

332 Corollary 11 holds in particular when F is the initial configuration for a λ -term. That is,
 333 the equality induced on call-by-value λ -terms by their representation in A-OGS and in $I\pi$
 334 (under the opLTS) is the same, both employing traces and employing bisimulation to handle
 335 the observables for the two models.

336 ▶ **Corollary 12.** For any λ -terms M, N , we have:

337 $\langle M \rangle \simeq_a \langle N \rangle$ iff $\langle M \rangle \approx_a \langle N \rangle$ iff $\mathcal{V}[[M]] \simeq_{\text{op}\pi} \mathcal{V}[[N]]$ iff $\mathcal{V}[[M]] \approx_{\text{op}\pi} \mathcal{V}[[N]]$.

338 From Theorem 5 and Corollary 8, it also follows that F and $\mathcal{V}[[F]]$ are weakly bisimilar,
 339 on the union of the respective LTSs.

340 **7 Concurrent Operational Game Semantics**

341 In this section we explore another way to derive an exact correspondence between OGS and
 342 $I\pi$, by relaxing the Alternating LTS for OGS so to allow multiple terms in configurations to
 343 run concurrently. We refer to the resulting OGS as the *Concurrent OGS*, briefly C-OGS (we
 344 recall that A-OGS refers to the Alternating OGS of Section 4).

345 We introduce *running terms*, ranged over by A, B , as finite mappings from continuation
 346 names to λ -terms. A *concurrent configuration* is a triple $\langle A, \gamma, \phi \rangle$ of a running term A , an
 347 environment γ , and a set of names ϕ . Moreover, the domains of A and γ must be disjoint.
 348 We extend the definition of the polarity function, considering names in the domain of both
 349 A and γ as Player names.

350 Passive and active configurations can be seen as special case of C-OGS configurations
 351 with zero and one running term, respectively. For this reason we still use F, G to range

$(P\tau)$	$\langle A \cdot [p \mapsto M], \gamma, \phi \rangle$	$\xrightarrow{\tau}_c$	$\langle A \cdot [p \mapsto N], \gamma, \phi \rangle$	when $M \rightarrow_V N$
(PA)	$\langle A \cdot [p \mapsto V], \gamma, \phi \rangle$	$\xrightarrow{\bar{p}(x)}_c$	$\langle A, \gamma \cdot [x \mapsto V], \phi \uplus \{x\} \rangle$	
(PQ)	$\langle A \cdot [p \mapsto E[xV]], \gamma, \phi \rangle$	$\xrightarrow{\bar{x}(y,q)}_c$	$\langle A, \gamma \cdot [y \mapsto V] \cdot [q \mapsto (E, p)], \phi \uplus \{y, q\} \rangle$	
(OA)	$\langle A, \gamma \cdot [p \mapsto (E, q)], \phi \rangle$	$\xrightarrow{p(x)}_c$	$\langle A \cdot [q \mapsto E[x]], \gamma, \phi \uplus \{x\} \rangle$	
(OQ)	$\langle A, \gamma, \phi \rangle$	$\xrightarrow{x(y,p)}_c$	$\langle A \cdot [p \mapsto Vy], \gamma, \phi \uplus \{y, p\} \rangle$	when $\gamma(x) = V$
(IOQ)	$\langle [? \mapsto M], \phi \rangle$	$\xrightarrow{(p)}_c$	$\langle [p \mapsto M], \varepsilon, \phi \uplus \{p\} \rangle$	

■ **Figure 4** The LTS for the Concurrent OGS

352 over C-OGS configurations. Moreover we freely take A-OGS configurations to be C-OGS
 353 configurations, and conversely for C-OGS configurations with zero and one running term,
 354 omitting the obvious syntactic coercions. Both the running term and the environment may
 355 be empty.

356 We present the rules of C-OGS in Figure 4. Since there is no more distinction between
 357 passive and active configurations, a given configuration can perform both Player and Opponent
 358 actions. Notice that only Opponent can add a new term to the running term A . A *singleton*
 359 is a configuration F whose P-support has only one element (that is, in C-OGS, F is either of
 360 the form $\langle [p \mapsto M], \varepsilon, \phi \rangle$, or $\langle \varepsilon, [x \mapsto V], \phi \rangle$, or $\langle \varepsilon, [p \mapsto (E, q)], \phi \rangle$).

361 In this and in the following section F, G ranges over C-OGS configurations, as reminded
 362 by the index ‘c’ in the symbols for LTS and behavioural equivalence with which F, G appear
 363 (e.g., \simeq_c).

364 7.1 Comparison between C-OGS and $I\pi$

365 The encoding of C-OGS into $I\pi$ is a simple adaptation of that for A-OGS. We only have
 366 to consider the new or modified syntactic elements of C-OGS, namely running terms and
 367 configurations; the encoding remains otherwise the same. The encoding of running term is:

$$368 \quad \mathcal{V}[[p \mapsto M] \cdot A] \stackrel{\text{def}}{=} \mathcal{V}[M]\langle p \rangle \mid \mathcal{V}[A] \qquad \mathcal{V}[\varepsilon] \stackrel{\text{def}}{=} 0$$

369 The encoding of configurations is then defined as: $\mathcal{V}[\langle A, \gamma, \phi \rangle] \stackrel{\text{def}}{=} \mathcal{V}[A] \mid \mathcal{V}[\gamma]$.

370 The results about operational correspondence between C-OGS and $I\pi$ are as those between
 371 A-OGS and $I\pi$ under the opLTS.

372 ► **Theorem 13.**

- 373 1. If $F \Longrightarrow_c F'$ then $\mathcal{V}[F] \Longrightarrow_{\pi} \pi \gtrsim \mathcal{V}[F']$;
- 374 2. if $F \xrightarrow{\ell}_c F'$ then $\mathcal{V}[F] \xrightarrow{\ell} \pi \approx_{\pi} \mathcal{V}[F']$;
- 375 3. the converse of (1), i.e. if $\mathcal{V}[F] \Longrightarrow_{\pi} P$ then there is F' such that $F \Longrightarrow_c F'$ and
 376 $P \pi \gtrsim \mathcal{V}[F']$.
- 377 4. the converse of (2), i.e. if $\mathcal{V}[F] \xrightarrow{\ell} \pi P$ then there is F' such that $F \xrightarrow{\ell}_c F'$ and
 378 $P \approx_{\pi} \mathcal{V}[F']$.

379 ► **Corollary 14.** For any C-OGS configuration F and trace s , we have $F \xrightarrow{s}_c$ iff $\mathcal{V}[F] \xrightarrow{s}_{\pi}$.

380 From Corollary 14 and Theorem 13, we derive:

381 ► **Lemma 15.** For any F, F' we have:

- 382 1. $F_1 \simeq_c F_2$ iff $\mathcal{V}[[F_1]] \simeq_\pi \mathcal{V}[[F_2]]$;
 383 2. $F_1 \approx_c F_2$ iff $\mathcal{V}[[F_1]] \approx_\pi \mathcal{V}[[F_2]]$.

384 To derive the full analogous of Corollary 12, we now show that, on the $I\pi$ representation
 385 of λ -terms, trace equivalence is the same as bisimilarity. This result needs a little care: it is
 386 known that on deterministic LTSs bisimilarity coincides with trace equivalence. However,
 387 the behaviour of the $I\pi$ representation of a C-OGS configuration need not be deterministic,
 388 because there could be multiple silent transitions as well as multiple output transitions (for
 389 instance, in C-OGS rule OQ may be applicable to different terms).

390 ► **Lemma 16.** *For any M, N we have: $\mathcal{V}[[M]] \simeq_\pi \mathcal{V}[[N]]$ iff $\mathcal{V}[[M]] \approx_\pi \mathcal{V}[[N]]$.*

391 The proof uses the ‘bisimulation up-to context and up-to $(\pi\gtrsim, \approx_\pi)$ ’ technique. We can
 392 finally combine Lemmas 16 and 15 to derive that the C-OGS and $I\pi$ semantics of λ -calculus
 393 coincide, both for traces and for bisimilarity.

394 ► **Corollary 17.** *For all M, N we have: $\langle M \rangle \simeq_c \langle N \rangle$ iff $\langle M \rangle \approx_c \langle N \rangle$ iff $\mathcal{V}[[M]] \simeq_\pi \mathcal{V}[[N]]$ iff
 395 $\mathcal{V}[[M]] \approx_\pi \mathcal{V}[[N]]$.*

396 More details on proofs may be found in Appendix E.

397 7.2 Tensor Product

398 We now introduce a way of combining configurations, which corresponds to the notion of
 399 tensor product of arenas and strategies in (denotational) game semantics.

400 ► **Definition 18.** *Two concurrent configurations F, G are said to be compatible if their
 401 polarity functions $\text{pol}_F, \text{pol}_G$ are compatible — that is, for all $a \in \text{dom}(\text{pol}_F) \cap \text{dom}(\text{pol}_G)$,
 402 we have $\text{pol}_F(a) = \text{pol}_G(a)$.*

403 ► **Definition 19.** *For compatible configurations $F = \langle A, \gamma, \phi \rangle$ and $G = \langle B, \delta, \phi' \rangle$, the tensor
 404 product $F \otimes G$ is defined as $F \otimes G \triangleq \langle A \cdot B, \gamma \cdot \delta, \phi \cup \phi' \rangle$*

405 The polarity function of $F \otimes G$ is then equal to $\text{pol}_F \cup \text{pol}_G$, and $\mathcal{V}[[F_1 \otimes F_2]] \equiv \mathcal{V}[[F_1]] \mid \mathcal{V}[[F_2]]$,
 406 where \equiv is the standard structural congruence of π -calculi. In the following, we write
 407 $\text{inter}(s_1, s_2)$ for the set of traces obtained from an interleaving of the elements in the
 408 sequences s_1 and s_2 .

409 ► **Lemma 20.** *Suppose F_1, F_2 are compatible concurrent configurations. The set of traces
 410 generated by $F_1 \otimes F_2$ is the union of the sets of interleaving $\text{inter}(s_1, s_2)$, for $F_1 \xrightarrow{s_1}_c$ and
 411 $F_2 \xrightarrow{s_2}_c$.*

412 The tensor product of A-OGS configurations is defined similarly, with the additional
 413 hypothesis that at most one of the two configurations is active, in order for their tensor
 414 product to be a valid A-OGS configuration. The details can be found in Appendix H.2.

415 7.3 Up-to techniques for games

416 We introduce up-to techniques for C-OGS, which allow, in bisimulation proofs, to split two
 417 C-OGS configurations into separate components and then to reason separately on these.
 418 These up-to techniques are directly imported from $I\pi$. Abstract settings for up-to techniques
 419 have been developed, see [36, 37]; we cannot however derive the OGS techniques from them
 420 because these setting are specific to first-order LTS (i.e., CCS-like, without binders within
 421 actions).

422 The new techniques are then used to prove that C-OGS and A-OGS yield the same
 423 semantics on λ -terms; a further application is in Section 7.5, discussing eager normal-form
 424 bisimilarity.

425 A relation \mathcal{R} on configuration is *well-formed* if it relates configurations with the same
 426 polarity function. Below, all relations on configurations are meant to be well formed. Given
 427 a well-formed relation \mathcal{R} we write:

- 428 ■ \mathcal{R}^\downarrow for the relation $\{(F_1, F_2) : \exists G \text{ s.t. } F_i = F'_i \otimes G (i = 1, 2) \text{ and } F'_1 \mathcal{R} F'_2\}$.
- 429 ■ $\mathcal{R}^{\downarrow*}$ for the reflexive and transitive closure of \mathcal{R}^\downarrow . Thus from $F_1 \mathcal{R} G_1$ and $F_2 \mathcal{R} G_2$ we
 430 obtain $(F_1 \otimes F_2) \mathcal{R}^{\downarrow*} (G_1 \otimes F_2) \mathcal{R}^{\downarrow*} G_1 \otimes G_2$.
- 431 ■ $\Rightarrow_c \mathcal{R}^{\downarrow*} \Leftarrow_c$ for the closure of $\mathcal{R}^{\downarrow*}$ under reductions. That is, $F_1 \Rightarrow_c \mathcal{R}^{\downarrow*} \Leftarrow_c F_2$ holds if
 432 there are $F'_i, i = 1, 2$ with $F_i \Rightarrow_c F'_i$ and $F'_1 \mathcal{R}^{\downarrow*} F'_2$. (As \Rightarrow_c is reflexive, we may have
 433 $F_i = F'_i$.)

434 ► **Definition 21.** *A relation \mathcal{R} on configurations is a bisimulation up-to reduction and*
 435 *composition if whenever $F_1 \mathcal{R} F_2$:*

- 436 1. *if $F_1 \xrightarrow{\mu}_c F'_1$ then there is F'_2 such that $F_2 \xrightarrow{\hat{\mu}}_c F'_2$ and $F'_1 \Rightarrow_c \mathcal{R}^{\downarrow*} \Leftarrow_c F'_2$;*
- 437 2. *the converse, on the transitions from F_2 .*

438 A variant of the technique in Definition 21, where the bisimulation game is played only
 439 on visible actions at the price of being defined on singleton configurations, is presented in
 440 Appendix F and used in Section 7.5 to lift any eager normal form bisimulation to an OGS
 441 ‘bisimulation up-to’.

442 ► **Theorem 22.** *If \mathcal{R} is bisimulation up-to reduction and composition then $\mathcal{R} \subseteq \approx_c$.*

443 The theorem is proved by showing that the $I\pi$ image of \mathcal{R} is a bisimulation up-to context
 444 and up-to $(\pi \succ, \approx_\pi)$, and appealing to Lemma 15(2). See Appendix F for details.

► **Remark 23.** Results such as Corollary 17 and Theorem 22 might suggest that the equality
 between two configurations implies the equality of all their singleton components. That is, if
 $F \simeq_c G$, with $[p \mapsto M]$ part of F and $[p \mapsto N]$ part of G , then also $\langle [p \mapsto M] \rangle \simeq_c \langle [p \mapsto N] \rangle$.
 A counterexample is given by the configurations

$$\begin{aligned} F_1 &\stackrel{\text{def}}{=} \langle [p_1 \mapsto M] \cdot [p_2 \mapsto \Omega] \rangle & \text{where } M &\stackrel{\text{def}}{=} (\lambda z. \Omega)(x \lambda y. \Omega) \\ F_2 &\stackrel{\text{def}}{=} \langle [p_1 \mapsto \Omega] \cdot [p_2 \mapsto M] \rangle \end{aligned}$$

445 Intuitively the reason why $F_1 \simeq_c F_2$ is that M can produce an output (along the variable x),
 446 but an observer will never obtain access to the name at which M is located (p_1 or p_2). That
 447 is, the term M can interrogate x , but it will never answer, neither at p_1 nor at p_2 .

448 7.4 Relationship between Concurrent and Alternating OGS

449 In Section 6 we have proved that the trace-based and bisimulation-based semantics produced
 450 by A-OGS and by $I\pi$ under the opLTS coincide. In Section 7.1 we have obtained the same
 451 result for C-OGS and $I\pi$ under the ordinary LTS. In this section we develop these results
 452 so to conclude that all such equivalences for λ -terms actually coincide. In other words, the
 453 equivalence induced on λ -terms by their representations in OGS and $I\pi$ is the same, regardless
 454 of whether we adopt the alternating or concurrent flavour for OGS, the opLTS or the ordinary
 455 LTS in $I\pi$, a trace or a bisimulation semantics. For this, in one direction we show that the
 456 trace semantics induced by C-OGS implies that induced by A-OGS (Lemma 63). In the
 457 opposite direction, we lift a bisimulation over the alternating LTS on singleton configurations
 458 into a bisimulation up-to composition over the concurrent LTS.

459 ▶ **Lemma 24.** *If F_1, F_2 are A-OGS singleton configurations and $F_1 \approx_a F_2$, then also*
 460 $F_1 \approx_c F_2$.

461 Details may be found in Appendix G. From Corollaries 12, 17, and Lemmas 63, 24 we
 462 can thus conclude.

463 ▶ **Corollary 25.** *For any λ -terms M, N , the following statements are the same: $\langle M \rangle \simeq_a \langle N \rangle$;*
 464 $\langle M \rangle \approx_a \langle N \rangle$; $\langle M \rangle \simeq_c \langle N \rangle$; $\langle M \rangle \approx_c \langle N \rangle$; $\mathcal{V}[[M]] \simeq_{\text{op}} \mathcal{V}[[N]]$; $\mathcal{V}[[M]] \approx_{\text{op}} \mathcal{V}[[N]]$; $\mathcal{V}[[M]] \simeq_{\pi}$
 465 $\mathcal{V}[[N]]$; $\mathcal{V}[[M]] \approx_{\pi} \mathcal{V}[[N]]$.

466 7.5 Eager Normal Form Bisimulations

467 We recall Lassen's *eager normal-form (enf) bisimilarity* [28].

468 ▶ **Definition 26.** *An enf-bisimulation is a triple of relation on terms $\mathcal{R}_{\mathcal{M}}$, values $\mathcal{R}_{\mathcal{V}}$, and*
 469 *evaluation contexts $\mathcal{R}_{\mathcal{K}}$ that satisfies:*

- 470 ■ $M_1 \mathcal{R}_{\mathcal{M}} M_2$ if either:
 - 471 ■ both M_1, M_2 diverge;
 - 472 ■ $M_1 \Downarrow E_1[xV_1]$ and $M_2 \Downarrow E_2[xV_2]$ for some x , values V_1, V_2 , and evaluation contexts
 473 E_1, E_2 with $V_1 \mathcal{R}_{\mathcal{V}} V_2$ and $K_1 \mathcal{R}_{\mathcal{K}} K_2$;
 - 474 ■ $M_1 \Downarrow V_1$ and $M_2 \Downarrow V_2$ for some values V_1, V_2 with $V_1 \mathcal{R}_{\mathcal{V}} V_2$.
- 475 ■ $V_1 \mathcal{R}_{\mathcal{V}} V_2$ if $V_1 x \mathcal{R}_{\mathcal{M}} V_2 x$ for some fresh x ;
- 476 ■ $K_1 \mathcal{R}_{\mathcal{V}} K_2$ if $K_1[x] \mathcal{R}_{\mathcal{M}} K_2[x]$ for some fresh x .

477 *The largest enf-bisimulation is called enf-bisimilarity.*

478 From Corollary 25 and existing results in the π -calculus [11] we can immediately conclude
 479 that the semantics on λ -terms induced by OGS (Alternating or Concurrent) coincides with
 480 enf-bisimilarity (i.e., Lassen's trees).

481 In this section we show a direct proof of the result, for C-OGS bisimilarity, as an example
 482 of application of the up-to composition technique for C-OGS.

483 Terms, Values, and Evaluations contexts can be directly lifted to singleton concurrent
 484 configurations, meaning that we can transform a relation on terms, values, and contexts \mathcal{R}
 485 into a relation $\widehat{\mathcal{R}}$ on singleton concurrent configurations in the following way:

- 486 ■ If $(M_1, M_2) \in \mathcal{R}$ then $(\langle [p \mapsto M_1], \varepsilon, \phi \rangle, \langle [p \mapsto M_2], \varepsilon, \phi \rangle) \in \widehat{\mathcal{R}}$, with $\phi = \text{fv}(M_1, M_2) \uplus \{p\}$
- 487 ■ If $(V_1, V_2) \in \mathcal{R}$ then $(\langle [x \mapsto V_1], \phi \rangle, \langle [x \mapsto V_2], \phi \rangle) \in \widehat{\mathcal{R}}$, with $\phi = \text{fv}(V_1, V_2) \uplus \{x\}$;
- 488 ■ If $(E_1, E_2) \in \mathcal{R}$ then $(\langle [p \mapsto (E_1, q)], \phi \rangle, \langle [p \mapsto (E_2, q)], \phi \rangle) \in \widehat{\mathcal{R}}$, with $\phi = \text{fv}(E_1, E_2) \uplus$
 489 $\{p, q\}$.

490 ▶ **Theorem 27.** *Taking $(\mathcal{R}_{\mathcal{M}}, \mathcal{R}_{\mathcal{V}}, \mathcal{R}_{\mathcal{K}})$ an enf-bisimulation, then $(\widehat{\mathcal{R}}_{\mathcal{M}} \cup \widehat{\mathcal{R}}_{\mathcal{V}} \cup \widehat{\mathcal{R}}_{\mathcal{K}})$ is a*
 491 *singleton bisimulation up-to composition in C-OGS (as defined in Appendix F).*

492 **Proof.** Taking F_1, F_2 two singleton configurations s.t. $(F_1, F_2) \in \widehat{\mathcal{R}}_{\mathcal{M}}$, then we can write
 493 F_i as $\langle [p \mapsto M_i], \phi \rangle$ for $i = 1, 2$, such that $(M_1, M_2) \in \mathcal{R}$. suppose that $F_1 \xrightarrow{c} \xrightarrow{\ell} F'_1$,
 494 with ℓ a Player action. We only present the Player Question case, which is where 'up-to
 495 composition' is useful. So writing ℓ as $\bar{x}(y, q)$, F'_1 can be written as $\langle [y \mapsto V_1] \cdot [q \mapsto (E_1, p)] \rangle$,
 496 so that $M_1 \rightarrow_v^* E_1[xV_1]$.

497 As $(M_1, M_2) \in \mathcal{R}$, there are E_2, V_2 s.t. $M_2 \rightarrow_v^* E_2[xV_2]$, $(V_1, V_2) \in \mathcal{R}$ and $(E_1, E_2) \in \mathcal{R}$.
 498 Hence $F_2 \xrightarrow{c} \xrightarrow{\ell} F'_2$ with $F'_2 = \langle [y \mapsto V_2] \cdot [q \mapsto (E_2, p)] \rangle$.

499 Finally, $(\langle [y \mapsto V_1] \rangle, \langle [y \mapsto V_2] \rangle) \in \widehat{\mathcal{R}}$ and $(\langle [q \mapsto (E_1, p)] \rangle, \langle [q \mapsto (E_2, p)] \rangle) \in \widehat{\mathcal{R}}$, so that
 500 $(F'_1, F'_2) \in \widehat{\mathcal{R}}^*$.

501 The case of passive singleton configurations is proved in a similar way. ◀

8 Compositionality of OGS via $I\pi$

We now present some compositionality results about C-OGS and A-OGS, that can be proved via the correspondence between OGS and $I\pi$.

Compositionality of OGS amounts to compute the set of traces generated by $\langle M\{V/x\} \rangle$ from the set of traces generated by $\langle M \rangle$ and $\langle [x \mapsto V] \rangle$. This is the cornerstone of (denotational) game semantics, where the combination of $\langle M \rangle$ and $\langle [x \mapsto V] \rangle$ is represented via the so-called ‘parallel composition plus hiding’. This notion of parallel composition of two processes P, Q plus hiding over a name x is directly expressible in $I\pi$ as the process $\nu x(P \mid Q)$. Precisely, suppose F, G are two configurations that agree on their polarity functions, but on a name x ; then we write

$$\nu x(F \mid G) \quad (*)$$

for the $I\pi$ process $\nu x(\mathcal{V}[[F]] \mid \mathcal{V}[[G]])$, the *parallel composition plus hiding over x* . To define this operation directly at the level of OGS, we would have to generalize its LTS, allowing internal interactions over a name x used both in input and in output.

As the translation \mathcal{V} from OGS configurations into $I\pi$ validates the β_v rule, we can prove that the behaviour of $\langle M\{V/x\} \rangle$ (e.g., its set of traces) is the same as that of the parallel composition plus hiding over x of $\langle M \rangle$ and $\langle [x \mapsto V] \rangle$. We use the notation $(*)$ above to express the following two results.

► **Theorem 28.** For $\bowtie \in \{\approx_{o\pi}, \approx_\pi, \simeq_{o\pi}, \simeq_\pi\}$, we have

$$\mathcal{V}[[\langle [p \mapsto M\{V/x\}] \rangle, \gamma, \phi \cup \phi']] \bowtie \nu x(\langle [p \mapsto M] \rangle, \varepsilon, \phi \uplus \{x\} \mid \langle \gamma \cdot [x \mapsto V] \rangle, \phi' \uplus \{x\})$$

► **Corollary 29.** For any trace s :

1. $\langle M\{V/x\} \rangle, p, \gamma, \phi \cup \phi' \xrightarrow{s}_a$ iff $\nu x(\langle [p \mapsto M] \rangle, \varepsilon, \phi \uplus \{x\} \mid \langle \gamma \cdot [x \mapsto V] \rangle, \phi' \uplus \{x\}) \xrightarrow{s}_{o\pi}$
2. $\langle [p \mapsto M\{V/x\}] \rangle, \gamma, \phi \cup \phi' \xrightarrow{s}_c$ iff $\nu x(\langle [p \mapsto M] \rangle, \varepsilon, \phi \uplus \{x\} \mid \langle \gamma \cdot [x \mapsto V] \rangle, \phi' \uplus \{x\}) \xrightarrow{s}_\pi$

Other important properties that we can import in OGS from $I\pi$ are the congruence properties for the A-OGS and C-OGS semantics. We report the result for \simeq_a ; the same result holds for $\approx_a, \simeq_c, \approx_c$.

► **Theorem 30.** If $\langle M \rangle \simeq_a \langle N \rangle$ then for any Λ_V context C , $\langle C[M] \rangle \simeq_a \langle C[N] \rangle$.

The result is obtained from the congruence properties of $I\pi$, Corollary 25, and the compositionality of the encoding \mathcal{V} .

9 Related and Future Work

Analogies between game semantics and π -calculus, as semantic frameworks in which *names* are central, have been pointed out from the very beginning of game semantics. In the pioneering work [19], the authors obtain a translation of PCF terms into the π -calculus from a game model of PCF by representing *strategies* (the denotation of PCF terms in the game model) as processes of the π -calculus. The encoding bears similarities with Milner’s, though they are syntactically rather different (“it is clear that the two are conceptually quite unrelated”, [19]). The connection has been developed in various papers, e.g., [8, 14, 17, 18, 48]. Milner’s encodings into the π -calculus have sometimes been a source of inspiration in the definition of the game semantics models (e.g., transporting the work [19], in call-by-name, onto call-by-value [18]). In [48], a typed variant of π -calculus, influenced by differential linear logic [13], is introduced as a metalanguage to represent game models. In [9], games are defined using algebraic operations on sets of traces, and used to prove type soundness of a

544 simply-typed call-by-value λ -calculus with effects. Although the calculus of traces employed
545 is not a π -calculus (e.g., being defined from operators and relations over trace sets rather
546 than from syntactic process constructs), there are similarities, which would be interesting to
547 investigate.

548 Usually in the above papers the source language is a form of λ -calculus, that is interpreted
549 into game semantics, and the π -calculus (or dialects of it) is used to represent the resulting
550 strategies and games. Another goal has been to shed light into typing disciplines for π -calculus
551 processes, by transplanting conditions on strategies such as well-bracketing and innocence
552 into appropriate typings for the π -calculus (see, e.g., [6, 47]).

553 In contrast with the above works, where analogies between game semantics and π -calculus
554 are used to better understand one of the two models (i.e., explaining game semantics in
555 terms process interactions, or enhancing type systems for processes following structures in
556 game semantics), in the present paper we have carried out a direct comparison between the
557 two models (precisely OGS and $I\pi$). For this we have started from the (arguably natural)
558 representations of the λ -calculus into OGS and $I\pi$ (the latter being Milner's encodings). Our
559 goal was understanding the relation between the behaviours of the terms in the two models,
560 and transferring techniques and results between them.

561 Technically, our work builds on [10, 11], where a detailed analysis of the behaviour of
562 Milner's call-by-value encoding is carried out using proof techniques for π -calculus based
563 on unique-solution of equations. Various results in [10, 11] are essential to our own (the
564 observation that Milner's encodings should be interpreted in $I\pi$ rather than the full π -calculus
565 is also from [10, 11]).

566 Bisimulations over OGS terms, and tensor products of configurations, were introduced
567 in [29], in order to provide a framework to study compositionality properties of OGS. In
568 our case the compositionality result of OGS is derived from the correspondence with the
569 π -calculus. In [39], a correspondence between an i/o typed asynchronous π -calculus and
570 a computational λ -calculus with channel communication is established, using a common
571 categorical model (a compact closed Freyd category). It would be interesting to see if our
572 concurrent operational game model could be equipped with this categorical semantics.

573 *Normal form* (or *open*) bisimulations [28, 44], as game semantics, manipulate open terms,
574 and sometimes make use of environments or stacks of evaluation contexts (see e.g., the
575 recent work [7], where a fully abstract normal-form bisimulation for a λ -calculus with store
576 is obtained).

577 There are also works that build game models directly for the π -calculus, i.e., [12, 25, 26, 38]
578 A correspondence between a synchronous π -calculus with session types and concurrent game
579 semantics [46] is given in [8], relating games (represented as arenas) to session types, and
580 strategies (defined as coincident event structures) to processes.

581 We have exploited the full abstraction results between OGS and $I\pi$ to transport a few
582 up-to techniques for bisimulation from $I\pi$ onto OGS. However, in $I\pi$ there are various other
583 such techniques, even a theory of bisimulation enhancements. We would like to see which
584 other techniques could be useful in OGS, possibly transporting the theory of enhancements
585 itself.

586 — References

- 587 1 Samson Abramsky. The lazy lambda calculus. In D. Turner, editor, *Research Topics in*
588 *Functional Programming*, pages 65–116. Addison-Wesley, 1989.
- 589 2 Samson Abramsky. Game semantics for programming languages (abstract). In *Proceedings*
590 *of the 22nd International Symposium on Mathematical Foundations of Computer Science*,
591 MFCS '97, page 3–4, Berlin, Heidelberg, 1997. Springer-Verlag.
- 592 3 Samson Abramsky, Kohei Honda, and Guy McCusker. A fully abstract game semantics for
593 general references. In *Proceedings of the 13th Annual IEEE Symposium on Logic in Computer*
594 *Science*, LICS '98, page 334, USA, 1998. IEEE Computer Society.
- 595 4 Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for pcf. *Inf.*
596 *Comput.*, 163(2):409–470, December 2000. doi:10.1006/inco.2000.2930.
- 597 5 Samson Abramsky and Guy McCusker. Linearity, sharing and state: a fully abstract game
598 semantics for Idealized Algol with active expressions. In *Algol-like languages*, pages 297–329.
599 Springer, 1997.
- 600 6 Martin Berger, Kohei Honda, and Nobuko Yoshida. Sequentiality and the π -calculus. In
601 *International Conference on Typed Lambda Calculi and Applications*, pages 29–45. Springer,
602 2001.
- 603 7 Dariusz Biernacki, Sergueï Lenglet, and Piotr Polesiuk. A complete normal-form bisimilarity
604 for state. In *International Conference on Foundations of Software Science and Computation*
605 *Structures*, pages 98–114. Springer, 2019.
- 606 8 Simon Castellán and Nobuko Yoshida. Two sides of the same coin: Session types and game
607 semantics: A synchronous side and an asynchronous side. *Proc. ACM Program. Lang.*, (POPL),
608 January 2019. doi:10.1145/3290340.
- 609 9 Tim Disney and Cormac Flanagan. Game semantics for type soundness. In *Proceedings of the*
610 *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, LICS '15,
611 page 104–114, USA, 2015. IEEE Computer Society. doi:10.1109/LICS.2015.20.
- 612 10 Adrien Durier. *Unique solution techniques for processes and functions*. PhD thesis, University
613 of Lyon, France, 2020.
- 614 11 Adrien Durier, Daniel Hirschhoff, and Davide Sangiorgi. Eager functions as processes. In *33rd*
615 *Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018*. IEEE Computer
616 Society, 2018.
- 617 12 Clovis Eberhart, Tom Hirschowitz, and Thomas Seiller. An intensionally fully-abstract sheaf
618 model for pi. In *6th Conference on Algebra and Coalgebra in Computer Science (CALCO*
619 *2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.
- 620 13 Thomas Ehrhard. An introduction to differential linear logic: proof-nets, models and antide-
621 rivatives. *Mathematical Structures in Computer Science*, 28(7):995–1060, 2018.
- 622 14 Marcelo Fiore and Kohei Honda. Recursive types in games: Axiomatics and process represent-
623 ation. In *Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science*,
624 LICS '98, page 345, USA, 1998. IEEE Computer Society.
- 625 15 Dan R. Ghica and Andrzej S. Murawski. Angelic semantics of fine-grained concurrency. In
626 *International Conference on Foundations of Software Science and Computation Structures*,
627 pages 211–225. Springer, 2004.
- 628 16 Dan R. Ghica and Nikos Tzevelekos. A system-level game semantics. *Electron. Notes Theor.*
629 *Comput. Sci.*, 286:191–211, September 2012. doi:10.1016/j.entcs.2012.08.013.
- 630 17 Kohei Honda. Processes and games. *Electron. Notes Theor. Comput. Sci.*, 71:40–69, 2002.
631 doi:10.1016/S1571-0661(05)82528-9.
- 632 18 Kohei Honda and Nobuko Yoshida. Game-theoretic analysis of call-by-value computation.
633 *Theor. Comput. Sci.*, 221(1–2):393–456, June 1999. doi:10.1016/S0304-3975(99)00039-0.
- 634 19 J. M. E. Hyland and C.-H. L. Ong. Pi-calculus, dialogue games and full abstraction PCF. In
635 *Proceedings of the Seventh International Conference on Functional Programming Languages*
636 *and Computer Architecture*, FPCA '95, page 96–107, New York, NY, USA, 1995. Association
637 for Computing Machinery. doi:10.1145/224164.224189.

- 638 20 J.M.E. Hyland and C.-H. L. Ong. On full abstraction for PCF. *Inf. Comput.*, 163(2):285–408,
639 December 2000. doi:10.1006/inco.2000.2917.
- 640 21 Guilhem Jaber and Nikos Tzevelekos. Trace semantics for polymorphic references. In
641 *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*,
642 LICS '16, page 585–594, New York, NY, USA, 2016. Association for Computing Machinery.
643 doi:10.1145/2933575.2934509.
- 644 22 Radha Jagadeesan, Corin Pitcher, and James Riely. Open bisimulation for aspects. In
645 *Transactions on Aspect-Oriented Software Development V*, pages 72–132. Springer, 2009.
- 646 23 Alan Jeffrey and Julian Rathke. Java jr: Fully abstract trace semantics for a core java language.
647 In *European symposium on programming*, pages 423–438. Springer, 2005.
- 648 24 James Laird. Full abstraction for functional languages with control. In *Proceedings of the 12th*
649 *Annual IEEE Symposium on Logic in Computer Science*, LICS '97, page 58, USA, 1997. IEEE
650 Computer Society.
- 651 25 James Laird. A game semantics of the asynchronous π -calculus. In *International Conference*
652 *on Concurrency Theory*, pages 51–65. Springer, 2005.
- 653 26 James Laird. Game semantics for higher-order concurrency. In *Proceedings of the 26th*
654 *International Conference on Foundations of Software Technology and Theoretical Computer*
655 *Science*, FSTTCS'06, page 417–428, Berlin, Heidelberg, 2006. Springer-Verlag. doi:10.1007/
656 11944836.38.
- 657 27 James Laird. A fully abstract trace semantics for general references. In *Proceedings of the*
658 *34th International Conference on Automata, Languages and Programming*, ICALP'07, page
659 667–679, Berlin, Heidelberg, 2007. Springer-Verlag.
- 660 28 Søren B. Lassen. Eager normal form bisimulation. In *20th IEEE Symposium on Logic in*
661 *Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, pages 345–
662 354, 2005. URL: <http://dx.doi.org/10.1109/LICS.2005.15>, doi:10.1109/LICS.2005.15.
- 663 29 Paul Blain Levy and Sam Staton. Transition systems over games. In *Proceedings of the*
664 *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic*
665 *(CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science*
666 *(LICS)*, CSL-LICS '14, New York, NY, USA, 2014. Association for Computing Machinery.
667 doi:10.1145/2603088.2603150.
- 668 30 R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
- 669 31 R. Milner. The polyadic π -calculus: a tutorial. Technical Report ECS-LFCS-91-180, LFCS,
670 1991. Also in *Logic and Algebra of Specification*, ed. F.L. Bauer, W. Brauer and H. Schwichten-
671 berg, Springer Verlag, 1993.
- 672 32 R. Milner. Functions as processes. *MSCS*, 2(2):119–141, 1992.
- 673 33 C.-H. L. Ong and Pietro Di Gianantonio. Games characterizing levy-longo trees. *Theoretical*
674 *computer science*, 312(1):121–142, 2004.
- 675 34 B. Pierce and D. Sangiorgi. Typing and subtyping for mobile processes. *MSCS*, 6(5):409–454,
676 1996.
- 677 35 G.D. Plotkin. Call by name, call by value and the λ -calculus. *TCS*, 1:125–159, 1975.
- 678 36 Damien Pous and Davide Sangiorgi. Enhancements of the bisimulation proof method. In
679 Davide Sangiorgi and Jan Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*.
680 Cambridge University Press, 2012.
- 681 37 Damien Pous and Davide Sangiorgi. Bisimulation and coinduction enhancements: A historical
682 perspective. *Formal Asp. Comput.*, 31(6):733–749, 2019. doi:10.1007/s00165-019-00497-w.
- 683 38 Ken Sakayori and Takeshi Tsukada. A truly concurrent game model of the asynchronous
684 π -calculus. In *International Conference on Foundations of Software Science and Computation*
685 *Structures*, pages 389–406. Springer, 2017.
- 686 39 Ken Sakayori and Takeshi Tsukada. A categorical model of an i/o-typed π -calculus. In
687 *European Symposium on Programming*, pages 640–667. Springer, 2019.
- 688 40 D. Sangiorgi. Locality and non-interleaving semantics in calculi for mobile processes. *TCS*,
689 155:39–83, 1996.

- 690 **41** D. Sangiorgi. π -calculus, internal mobility and agent-passing calculi. *TCS*, 167(2):235–274,
691 1996.
- 692 **42** D. Sangiorgi. Lazy functions and mobile processes. In G. Plotkin, C. Stirling, and M. Tofte,
693 editors, *Proof, Language and Interaction: Essays in Honour of Robin Milner*. MIT Press, 2000.
- 694 **43** D. Sangiorgi and D. Walker. *The π -calculus: a Theory of Mobile Processes*. Cambridge
695 University Press, 2001.
- 696 **44** Kristian Støvring and Søren B. Lassen. A complete, co-inductive syntactic theory of sequential
697 control and state. In *Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on*
698 *Principles of Programming Languages*, POPL '07, page 161–172, New York, NY, USA, 2007.
699 Association for Computing Machinery. doi:10.1145/1190216.1190244.
- 700 **45** Kristian Støvring and Søren B. Lassen. A complete, co-inductive syntactic theory of
701 sequential control and state. In *Semantics and Algebraic Specification, Essays Dedic-*
702 *ated to Peter D. Mosses on the Occasion of His 60th Birthday*, pages 329–375, 2009.
703 doi:10.1007/978-3-642-04164-8_17.
- 704 **46** Glynn Winskel, Silvain Rideau, Pierre Clairambault, and Simon Castellan. Games and
705 strategies as event structures. *Logical Methods in Computer Science*, 13, 2017.
- 706 **47** Nobuko Yoshida, Martin Berger, and Kohei Honda. Strong normalisation in the π -Calculus.
707 In *16th Annual IEEE Symposium on Logic in Computer Science (LICS'01)*, pages 311–322.
708 IEEE Computer Society, 2001.
- 709 **48** Nobuko Yoshida, Simon Castellan, and Léo Stefanescu. Game semantics: Easy as pi. *arXiv*
710 *preprint arXiv:2011.05248*, 2020.

711 **A** Additional material on $I\pi$

712 **A.1 LTSs**

713 In Figures 5 and 6 we report, respectively:

714 ■ the standard LTS of $I\pi$;

715 ■ a compositional definition of the output-prioritised LTS.

716 The symbol \equiv_α is used to indicate α -equivalence between $I\pi$ agents. If $\mu = a(\tilde{b})$ then $\bar{\mu}$
 717 is $\bar{a}(\tilde{b})$, and conversely. Rule **abs** is used to formalise, by means of an action, the ground
 718 instantiation of the parameters of an abstraction. In both tables we omit the symmetric of
parL, called **parR**.

$$\begin{array}{l}
 \text{alpha: } \frac{P \equiv_\alpha P' \quad P' \xrightarrow{\mu} P''}{P \xrightarrow{\mu} P''} \quad \text{pre: } \frac{}{\mu.P \xrightarrow{\mu} P} \quad \text{parL: } \frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q} \\
 \text{res: } \frac{P \xrightarrow{\mu} P' \quad x \notin \mathbf{n}(\mu)}{\nu x P \xrightarrow{\mu} \nu x P'} \quad \text{com: } \frac{P \xrightarrow{\mu} P' \quad Q \xrightarrow{\bar{\mu}} Q' \quad \mu \neq \tau, \tilde{x} = \mathbf{bn}(\mu)}{P \mid Q \xrightarrow{\tau} \nu \tilde{x} (P' \mid Q')} \\
 \text{rep: } \frac{\mu.P \xrightarrow{\mu} P'}{!\mu.P \xrightarrow{\mu} P' \mid !\mu.P} \quad \text{con: } \frac{P \xrightarrow{\mu} P' \quad K \stackrel{\text{def}}{=} (\tilde{y}).Q \quad (\tilde{y}).Q \equiv_\alpha (\tilde{x}).P}{K(\tilde{x}) \xrightarrow{\mu} P'} \\
 \text{abs: } \frac{(B = (\tilde{y}).Q \text{ or } (B = K \text{ and } K \stackrel{\text{def}}{=} (\tilde{y}).Q)) \quad (\tilde{y}).Q \equiv_\alpha (\tilde{x}).P}{B \xrightarrow{\tilde{x}} P}
 \end{array}$$

719 ■ **Figure 5** The standard LTS for $I\pi$

719

$$\begin{array}{l}
 \text{alpha: } \frac{P \equiv_\alpha P' \quad P' \xrightarrow{\mu} P''}{P \xrightarrow{\mu} P''} \quad \text{pre: } \frac{\mathbf{bn}(\mu) = \tilde{x} \quad \tilde{x} \cap \phi = \emptyset}{\mu.P \xrightarrow{\mu} P} \\
 \text{parL: } \frac{P \xrightarrow{\mu} P' \quad (Q \text{ input reactive}) \text{ or } (\mu \text{ is an output or } \tau)}{P \mid Q \xrightarrow{\mu} P' \mid Q} \\
 \text{com: } \frac{P \xrightarrow{\mu} P' \quad Q \xrightarrow{\bar{\mu}} Q' \quad \mu \neq \tau, \tilde{x} = \mathbf{bn}(\mu)}{P \mid Q \xrightarrow{\tau} \nu \tilde{x} (P' \mid Q')} \\
 \text{res: } \frac{P \xrightarrow{\mu} P' \quad x \notin \mathbf{n}(\mu)}{\nu x P \xrightarrow{\mu} \nu x P'} \quad \text{rep: } \frac{\mu.P \xrightarrow{\mu} P'}{!\mu.P \xrightarrow{\mu} P' \mid !\mu.P} \\
 \text{con: } \frac{P \xrightarrow{\mu} P' \quad K \stackrel{\text{def}}{=} (\tilde{y}).Q \quad (\tilde{y}).Q \equiv_\alpha (\tilde{x}).P}{K(\tilde{x}) \xrightarrow{\mu} P'}
 \end{array}$$

■ **Figure 6** A compositional presentation of the output-prioritised LTS for $I\pi$

720 **A.2 The expansion relation and “up-to” techniques**

721 For the proof of the results in this paper, we appeal to the expansion relation and to a few
722 “up-to” techniques, that are discussed in Section 3.1 and in the following two subsections.

723 **The expansion relation** The expansion relation, \lesssim_π [43], is an asymmetric variant of
724 \approx_π which allows us to count the number of τ -actions performed by the processes. Thus,
725 $P \lesssim_\pi Q$ holds if $P \approx_\pi Q$ but also Q has at least as many τ -moves as P . We recall that
726 $P \xrightarrow{\hat{\mu}} P'$ holds if $P \xrightarrow{\mu} P'$ or ($\mu = \tau$ and $P = P'$).

727 **► Definition 31** (expansion). *A relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is an expansion if $P \mathcal{R} Q$ implies:*

- 728 1. *Whenever $P \xrightarrow{\mu} P'$, there exists Q' s.t. $Q \xrightarrow{\hat{\mu}}_\pi Q'$ and $P' \mathcal{R} Q'$;*
- 729 2. *whenever $Q \xrightarrow{\mu} Q'$, there exists P' s.t. $P \xrightarrow{\hat{\mu}} P'$ and $P' \mathcal{R} Q'$.*

730 *We say that Q expands P , written $P \lesssim_\pi Q$, if $P \mathcal{R} Q$, for some expansion \mathcal{R} .*

731 Relation \lesssim_π is a precongruence, and is strictly included in \approx_π .

732 **Bisimulation up-to \approx_π**

733 **► Definition 32** (bisimulation up-to \approx_π). *A symmetric relation \mathcal{R} on $\mathcal{P} \times \mathcal{P}$ is a bisimulation
734 up-to \approx_π if $P \mathcal{R} Q$ and $P \xrightarrow{\mu}_\pi P'$ imply that there exists Q' s.t. $Q \xrightarrow{\hat{\mu}} Q'$ and
735 $P' \approx_\pi \mathcal{R} \approx_\pi Q'$.*

736 **► Theorem 33.** *Suppose \mathcal{R} is a bisimulation up-to \approx_π . Then $\mathcal{R} \subseteq \approx_\pi$.*

737 **Bisimulations up-to contexts**

738 **► Definition 34** (bisimulation up-to context and up-to $\pi \gtrsim$). *A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$
739 is a bisimulation up-to context and up-to $\pi \gtrsim$ if $P \mathcal{R} Q$ and $P \xrightarrow{\mu} P''$ imply that there are
740 a static context C_{ctx} and processes P' and Q' s.t. $P'' \pi \gtrsim C_{\text{ctx}}[P']$, $Q \xrightarrow{\hat{\mu}} \pi \gtrsim C_{\text{ctx}}[Q']$ and
741 $P' \mathcal{R} Q'$.*

742 The soundness of the above technique relies on the following lemma.

743 **► Lemma 35.** *Suppose \mathcal{R} is a bisimulation up-to context and up-to $\pi \gtrsim$, $(P, Q) \in \mathcal{R}$ holds
744 and C_{ctx} is a static context. If $C_{\text{ctx}}[P] \xrightarrow{\mu_1} \dots \xrightarrow{\mu_n} P_1$, $n \geq 0$, then there are a static context
745 C' and processes P' and Q' s.t. $P_1 \pi \gtrsim C'[P']$, $C_{\text{ctx}}[Q] \xrightarrow{\hat{\mu}_1} \dots \xrightarrow{\hat{\mu}_n} \pi \gtrsim C'[Q']$ and $P' \mathcal{R} Q'$.*

746 **► Theorem 36.** *If \mathcal{R} is a bisimulation up-to context and up-to $\pi \gtrsim$, then $\mathcal{R} \subseteq \approx_\pi$.*

747 **Proof.** By showing that the relation \mathcal{S} containing all pairs (P, Q) such that

748
$$\begin{aligned} & \text{for some static context } C_{\text{ctx}} \text{ and proceses } P', Q' \\ & \text{it holds that } P \pi \gtrsim C_{\text{ctx}}[P'], Q \pi \gtrsim C_{\text{ctx}}[Q'] \text{ and } P' \mathcal{R} Q' \end{aligned}$$

749 is a bisimulation. See [40] for details (adapting the proof to $\mathcal{I}\pi$ is straightforward). ◀

750 We now consider a refinement of the above proof technique, namely bisimulation up-to
751 context and up-to $(\pi \gtrsim, \approx_\pi)$.

752 **► Definition 37.** *A static context C_{ctx} cannot interact with a process P if there is no name
753 that appears free both in the context and in the process and with opposite polarities (that is,
754 either in input position in the context and output position in the process, or the converse).*

755 If C_{ctx} cannot interact with P , whenever $C_{\text{ctx}}[P] \longrightarrow Q$ then the interaction is either internal
 756 to C_{ctx} or internal to P . Moreover, the property is invariant for reduction; i.e., if $Q = C'[P']$
 757 where C' is the derivative of C_{ctx} and P' the derivative of P , then also C' cannot interact
 758 with P' .

759 ► **Definition 38.** [bisimulation up-to context and up-to $(\pi \gtrsim, \approx_\pi)$] A symmetric relation
 760 $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a bisimulation up-to context and up-to $(\pi \gtrsim, \approx_\pi)$ if $P \mathcal{R} Q$ implies:

- 761 1. if $P \xrightarrow{\ell} P''$ then that there are a static context C_{ctx} and processes P' and Q' s.t.
 762 $P'' \approx_\pi C_{\text{ctx}}[P']$, $Q \xrightarrow{\hat{\mu}} \approx_\pi C_{\text{ctx}}[Q']$ and $P' \mathcal{R} Q'$, and, moreover, C_{ctx} cannot interact
 763 with P or Q ;
- 764 2. if $P \longrightarrow P''$ then there are a static context C_{ctx} and processes P' and Q' s.t. $P'' \pi \gtrsim$
 765 $C_{\text{ctx}}[P']$, $Q \xrightarrow{\hat{\mu}} \pi \gtrsim C_{\text{ctx}}[Q']$ and $P' \mathcal{R} Q'$.

766 ► **Lemma 39.** Suppose \mathcal{R} is a bisimulation up-to context and up-to $(\pi \gtrsim, \approx_\pi)$ and C_{ctx} is a
 767 static context that does not interact with processes P and Q .

- 768 1. If $C_{\text{ctx}}[P] \Longrightarrow P_1$, then there are a static context C' and processes P' and Q' s.t.
 769 $P_1 \pi \gtrsim C'[P']$, $C_{\text{ctx}}[Q] \Longrightarrow \pi \gtrsim C'[Q']$ and $P' \mathcal{R} Q'$.
- 770 2. If $C_{\text{ctx}}[P] \xrightarrow{\ell} P_1$, then there are a static context C' and processes P' and Q' s.t.
 771 $P_1 \approx_\pi C'[P']$, $C_{\text{ctx}}[Q] \xrightarrow{\ell} \approx_\pi C'[Q']$ and $P' \mathcal{R} Q'$.

772 ► **Theorem 40.** If \mathcal{R} is a bisimulation up-to context and up-to $(\pi \gtrsim, \approx_\pi)$, then $\mathcal{R} \subseteq \approx_\pi$.

773 **Proof.** By showing that the relation \mathcal{R} containing all pairs (P, Q) such that

774 for some static context C_{ctx} and processes P', Q'
 where C_{ctx} does not interact with P', Q'
 it holds that $P \approx_\pi C_{\text{ctx}}[P']$, $Q \approx_\pi C_{\text{ctx}}[Q']$ and $P' \mathcal{R} Q'$

775 is a bisimulation. In a challenge transition $P \xrightarrow{\mu} P_1$ one distinguishes the case when μ is a
 776 silent or visible action. In both cases, one exploits Lemma 39. ◀

777 **B Behavioural relations in the ordinary LTS and in the opLTS for $\mathbb{I}\pi$**

778 A behavioural relation on the ordinary LTS ‘almost always’ implies the corresponding relation
 779 on the opLTS. The only exceptions are produced by processes that are related although they
 780 exhibit different divergence behaviours (by divergence we mean the possibility of performing
 781 an infinite number of τ actions) and at the same time may perform input actions. For
 782 instance, using τ^ω to indicate a purely divergent process (i.e., a process whose only transition
 783 is $\tau^\omega \xrightarrow{\tau} \tau^\omega$), the processes $a.0 \mid \tau^\omega$ and $a.0$ are bisimilar (and trace) equivalent in the
 784 ordinary LTS, but they are not so in the opLTS, since the divergence in the former process
 785 prevents observation of the input at a .

786 Indeed, a ‘bisimilarity respecting divergence’ would imply bisimilarity on the opLTS.
 787 Such ‘bisimilarity respecting divergence’, written \approx_π^\uparrow , is defined as \approx_π with the additional
 788 requirement that $P \approx_\pi Q$ implies

$$789 \quad \blacksquare \quad P \uparrow \text{ iff } Q \uparrow \quad (*)$$

790 where the predicate \uparrow holds for an agent R if it has a divergence (an infinite path consisting
 791 of only τ actions).

792 ► **Lemma 41.** Relation \approx_π^\uparrow implies $\approx_{\text{op}\pi}$.

793 The same result applies to other relations; e.g., the ‘expansion respecting divergence’ (defined
 794 by adding the clause $(*)$ above to those of the expansion relation \lesssim_π) on the ordinary LTS
 795 implies expansion on the opLTS.

796 The results of operational correspondence between $\mathbb{I}\pi$ and A-OGS rely on Lemmas 47-50.
 797 In the proofs of all these lemmas [10], the uses of expansion (\lesssim_π) come from application of
 798 some simple algebraic laws that respect divergence (i.e., the laws do not add or remove any
 799 divergence). Indeed the laws either are valid for strong bisimilarity, or they are laws such as

$$800 \quad \nu a (a(\tilde{x}). P \mid \bar{a}(\tilde{x}). Q) \pi \succ \nu a (P \mid Q)$$

801 As a consequence, all occurrences of \lesssim_π in Lemmas 47-50 can be replaced by the expansion
 802 relation defined on the opLTS ($\lesssim_{\text{op}\pi}$). This in turn implies that the same could be done for
 803 all occurrences of \lesssim_π in Theorems 4-7 (that rely on the lemmas).

804 A final remark concerns the encoding of λ -terms. First we recall that the equivalence
 805 induced on call-by-value λ -terms by their encoding into $\mathbb{I}\pi$ coincides with Lassen’s *eager*
 806 *normal-form (enf) bisimilarity* [28, 45] (also recalled in Definition 26).

807 ► **Theorem 42** ([11]). $\mathcal{V}[M] \approx_\pi \mathcal{V}[N]$ iff M and N are enf-bisimilar.

808 On the encoding of λ -terms, the ordinary bisimilarity \approx_π implies the ‘bisimilarity respecting
 809 divergence’ \approx_π^\uparrow . This because, intuitively, a term $\mathcal{V}[M]\langle p \rangle$ is divergent iff M is so in the
 810 call-by-value λ -calculus. Formally, the result is proved by rephrasing the result about the
 811 correspondence between \approx_π and Lassen’s trees (Theorem 42) in terms of \approx_π^\uparrow in place of \approx_π .
 812 Again, this property boils down to the fact that the uses of \lesssim_π in Lemmas 47-50 (on which
 813 also the characterisation in terms of Lassen’s trees is built) respect divergence. Thus, using
 814 also Lemma 41, we derive the following result.

815 ► **Theorem 43.** For $M, N \in \Lambda$, we have: $\mathcal{V}[M]\langle p \rangle \approx_\pi \mathcal{V}[N]\langle p \rangle$ iff $\mathcal{V}[M]\langle p \rangle \approx_{\text{op}\pi} \mathcal{V}[N]\langle p \rangle$.

816 Below we report an alternative proof of the theorem above, however always relying on
 817 the variant of Lemmas 47-50 with $\lesssim_{\text{op}\pi}$ in place of \lesssim_π .

818 ► **Lemma 44.** If F is a singleton configuration, then $\mathcal{V}[F]$ is of three possible forms:

- 819 1. $\mathcal{V}[M]\langle p \rangle$, for some M, p ;
- 820 2. $q(x). \mathcal{V}[E[x]]\langle p \rangle$, for some E, x, q, p ;
- 821 3. $\mathcal{V}^*[V]\langle y \rangle$, for some V, y (that is, either $!y(x, q). \mathcal{V}[M]\langle q \rangle$ if $V = \lambda x. M$, or $y \triangleright x$ if $V = x$).

822 ► **Lemma 45.**

- 823 ■ Suppose $P = P_1 \mid !a(\tilde{b}). P_2$, and $Q = Q_1 \mid !a(\tilde{b}). Q_2$, and $P \approx_\pi Q$. Then also $P_1 \approx_\pi Q_1$.
- 824 ■ Similarly, suppose $P = P_1 \mid a(\tilde{b}). P_2$, and $Q = Q_1 \mid a(\tilde{b}). Q_2$, and $P \approx_\pi Q$. Then also
 825 $P_1 \approx_\pi Q_1$.

826 ► **Theorem 46.** If F and G are singleton configurations, then $\mathcal{V}[F] \approx_\pi \mathcal{V}[G]$ implies
 827 $\mathcal{V}[F] \approx_{\text{op}\pi} \mathcal{V}[G]$.

828 **Proof.** Let \mathcal{R} be the relations with all pairs (P, Q) where P and Q are of the form

$$829 \quad \begin{aligned} P &= P_1 \mid \dots \mid P_n \\ Q &= Q_1 \mid \dots \mid Q_n \end{aligned}$$

830 for some n , where all P_i, Q_i are encodings of singleton configurations and for all i , we have
 831 $P_i \approx_\pi Q_i$. Moreover:

- 832 ■ there is at most one i for which P_i, Q_i are encodings of active configurations;

833 ■ in any P_j (resp. Q_j) that is the encoding of a passive configuration, the name of the
834 initial input does not appear free in any other component $P_{j'}$ (resp. $Q_{j'}$) for $j' \neq j$.

835 A consequence of the two conditions above is that two distinct components P_i and P_j
836 cannot interact. That is, if $P \xrightarrow{\mu} P'$, then there is i such that $P_i \xrightarrow{\mu} P'_i$ and $P' = P''_1 \mid \dots \mid$
837 P''_n where $P''_j = P_j$ if $j \neq i$ and $P''_i = P'_i$. And similarly if $P \xrightarrow{\mu}_{\circ\pi} P'$.

838 We show that \mathcal{R} is a $\approx_{\circ\pi}$ -bisimulation up-to $\lesssim_{\circ\pi}$ (the bisimulation up-to expansion is
839 sound for bisimilarity in any LTS). In the proof below, Lemmas 49 and 50 actually refer to
840 the versions of the lemmas mentioned above, with $\lesssim_{\circ\pi}$ in place of \lesssim_{π} .

841 Suppose $P \xrightarrow{\mu}_{\circ\pi} P'$. The action originates from some P_i alone, say $P_i \xrightarrow{\mu}_{\circ\pi} P'_i$. If μ
842 is a τ action, then also $P_i \xrightarrow{\mu} P'_i$; and since $P_i \approx_{\pi} Q_i$, we have $Q_i \Longrightarrow Q'_i$ with $P'_i \approx_{\pi} Q'_i$.
843 Moreover, also $Q_i \Longrightarrow_{\circ\pi} Q'_i$ — and the corresponding transition from Q .

844 Using Lemma 49 we infer that, up-to expansion, the derivative processes P'_i and Q'_i can
845 be rewritten into processes that fit the definition of \mathcal{R} .

846 For the case when μ is an output, one reasons similarly, possibly also using Lemma 45
847 when μ is of the form $\bar{x}(z, q)$.

848 Finally, if $P \xrightarrow{\mu}_{\circ\pi} P'$ and μ is an input, then P is input reactive. From the conditions in
849 the definition of \mathcal{R} we infer that Q is input reactive too. Then we can conclude, reasoning as
850 in the previous cases, but this time using Lemma 50 with γ a singleton. ◀

851 Theorem 43 is then a corollary of Theorem 46.

852 C Explanation of the A-OGS LTS

853 This appendix contains some additional explanations on the rules of the A-OGS LTS. To
854 help the reader, we first recall the rules (also presented in the main text as Figure 1):

(P τ)	$\langle M, p, \gamma, \phi \rangle$	$\xrightarrow{\tau}_{\mathbf{a}}$	$\langle N, p, \gamma, \phi \rangle$	when $M \rightarrow_{\gamma} N$
(PA)	$\langle V, p, \gamma, \phi \rangle$	$\xrightarrow{\bar{p}(x)}_{\mathbf{a}}$	$\langle \gamma \cdot [x \mapsto V], \phi \uplus \{x\} \rangle$	
(PQ)	$\langle E[xV], p, \gamma, \phi \rangle$	$\xrightarrow{\bar{x}(y, q)}_{\mathbf{a}}$	$\langle \gamma \cdot [y \mapsto V] \cdot [q \mapsto (E, p)], \phi \uplus \{y, q\} \rangle$	
(OA)	$\langle \gamma \cdot [q \mapsto (E, p)], \phi \rangle$	$\xrightarrow{q(x)}_{\mathbf{a}}$	$\langle E[x], p, \gamma, \phi \uplus \{x\} \rangle$	
(OQ)	$\langle \gamma, \phi \rangle$	$\xrightarrow{x(y, p)}_{\mathbf{a}}$	$\langle Vy, p, \gamma, \phi \uplus \{y, p\} \rangle$	when $\gamma(x) = V$
(IOQ)	$\langle [? \mapsto M], \phi \rangle$	$\xrightarrow{(p)}_{\mathbf{a}}$	$\langle M, p, \varepsilon, \phi \uplus \{p\} \rangle$	

856 The term of an active configuration determines the next transition to perform. First
857 the term needs to be reduced, using the rule (P τ). When the term is a value V , a Player
858 Answer (PA) is performed, providing a fresh variable x to Opponent, while V is stored in γ
859 at position x . Freshness is enforced using the disjoint union \uplus . When the term is a callback
860 $E[xV]$, with p the current continuation name, a Player Question (PQ) at x is performed,
861 providing two fresh names y, q to Opponent, while storing V at y and (E, p) at q in γ .

862 On passive configurations, Opponent has the choice to perform different actions. It can
863 perform an Opponent Answer (OA), by interrogating an evaluation context E stored in γ .
864 For this, Opponent provides a fresh variable x that is plugged into the hole of E , while
865 the continuation name q associated to E in γ is restored. Opponent may also perform
866 an Opponent Question (OQ), by interrogating a value V stored in γ . For this, Opponent
867 provides a fresh variable y as an argument to V .

868 **D** Auxiliary results for Section 6

869 We present results that are needed to establish the operational correspondence between OGS
870 and $\mathcal{I}\pi$, studied in Section 6.

871 We begin discussing an optimisation \mathcal{O} of the encoding of call-by-value λ -calculus.
872 Following Durier et al.'s [11], we sometimes use \mathcal{O} to simplify proofs. We report the
873 full definition of \mathcal{O} ; for convenience, we also list the definition of \mathcal{O} on OGS environments
874 and configurations, thought is the same as that for the initial encoding \mathcal{V} — just replacing \mathcal{V}
875 with \mathcal{O} in Figure 3).

876 The encoding \mathcal{O} is obtained from the initial one \mathcal{V} by inlining the encoding and performing
877 a few (deterministic) reductions, at the price of a more complex definition. Precisely, in the
878 encoding of application some of the initial communications are removed, including those with
879 which a term signals to have become a value. To achieve this, the encoding of an application
880 goes by a case analysis on the occurrences of values in the subterms.

881 We begin with a few results from [10,11] that are needed to reason about the optimised
882 encoding \mathcal{O} . The first is about the correctness of the otimisation, established by algebraic
883 reasoning [10,11]; its extension to encodings of configurations is straightforward.

884 ► **Lemma 47** (correctness of the optimisation). *For all M , it holds that $\mathcal{V}\llbracket M \rrbracket \pi \gtrsim \mathcal{O}\llbracket M \rrbracket$.
885 Similarly, for all A-OGS configurations F , it holds that $\mathcal{V}\llbracket F \rrbracket \pi \gtrsim \mathcal{O}\llbracket F \rrbracket$.*

► **Lemma 48.** *We have:*

$$\mathcal{O}\llbracket E[xV] \rrbracket \langle p \rangle \pi \gtrsim \bar{x}(z, q). (\mathcal{O}^*\llbracket V \rrbracket \langle z \rangle \mid q(y). \mathcal{O}\llbracket E[y] \rrbracket \langle p \rangle).$$

886 The proof [10,11] goes by induction on the evaluation context E .

887 The following lemma [10,11] uses Lemma 48 to establish the shape of the possible
888 transitions that a term $\mathcal{O}\llbracket M \rrbracket \langle p \rangle$ can perform.

889 ► **Lemma 49.** *For any $M \in \Lambda$ and p , process $\mathcal{O}\llbracket M \rrbracket \langle p \rangle$ has exactly one immediate transition,
890 and exactly one of the following clauses holds:*

- 891 1. $\mathcal{O}\llbracket M \rrbracket \langle p \rangle \xrightarrow{\bar{p}(y)} P$ and M is a value, with $P = \mathcal{O}^*\llbracket M \rrbracket \langle y \rangle$;
- 892 2. $\mathcal{O}\llbracket M \rrbracket \langle p \rangle \xrightarrow{\bar{x}(z, q)} P$ and M is of the form $E[xV]$, for some E, x, V , with

$$P \pi \gtrsim \mathcal{O}^*\llbracket V \rrbracket \langle z \rangle \mid q(y). \mathcal{O}\llbracket E[y] \rrbracket \langle p \rangle,$$

893 and moreover z is not free in $q(y). \mathcal{O}\llbracket E[y] \rrbracket \langle p \rangle$ whereas q is not free in $\mathcal{O}^*\llbracket V \rrbracket \langle z \rangle$;

- 894 3. $\mathcal{O}\llbracket M \rrbracket \langle p \rangle \xrightarrow{\tau} P$ and there is N with $M \rightarrow N$ and $P \pi \gtrsim \mathcal{O}\llbracket N \rrbracket \langle p \rangle$.

895 We now move to reasoning about the behaviour of the encoding of configurations. Two
896 key lemmas are the following ones; they are derived from Lemmas 48 and 49.

897 ► **Lemma 50.** *If $\mathcal{O}\llbracket \gamma \rrbracket \xrightarrow{\mu} P$ and μ is an input action, then we have three possibilities:*

- 898 1. μ is an input $y(x, q)$ and $\gamma = [y \mapsto \lambda x. M] \cdot \gamma'$, and $P \pi \gtrsim \mathcal{O}\llbracket M \rrbracket \langle q \rangle \mid \mathcal{O}\llbracket \gamma' \rrbracket$;
- 899 2. μ is an input $y(x, q)$ and $\gamma = [y \mapsto z] \cdot \gamma'$, and $P \pi \gtrsim \mathcal{O}\llbracket zx \rrbracket \langle q \rangle \mid \mathcal{O}\llbracket \gamma' \rrbracket$;
- 900 3. μ is an input $q(x)$ and $\gamma = [q \mapsto (E, p)] \cdot \gamma'$, and $P \pi \gtrsim \mathcal{O}\llbracket E[x] \rrbracket \langle p \rangle \mid \mathcal{O}\llbracket \gamma' \rrbracket$.

901 ► **Lemma 51.** $\mathcal{O}^*\llbracket V \rrbracket \langle y \rangle \xrightarrow{y(x, q)} \approx_{\pi} \mathcal{O}\llbracket Vx \rrbracket \langle q \rangle \mid \mathcal{O}^*\llbracket V \rrbracket \langle y \rangle$

902 In Lemma 51, the occurrence of \approx_{π} cannot be replaced by $\pi \gtrsim$; for instance, if $V = \lambda x. M$
then, using Lemma 49, we can infer that the derivative process is in the relation $\pi \gtrsim$ with

Encoding of λ -terms:

$$\begin{aligned}
\mathcal{O}[xV] &\stackrel{\text{def}}{=} (p) \bar{x}(z, q). (\mathcal{O}^*[V]\langle z \rangle \mid q \triangleright p) \\
\mathcal{O}[(\lambda x. M)V] &\stackrel{\text{def}}{=} (p) \nu y, w (\mathcal{O}^*[\lambda x. M]\langle y \rangle \mid \mathcal{O}^*[V]\langle w \rangle \mid \bar{y}(w', r'). (w' \triangleright w \mid r' \triangleright p)) \\
\mathcal{O}[VM] &\stackrel{\text{def}}{=} (p) \nu y (\mathcal{O}^*[V]\langle y \rangle \mid \nu r (\mathcal{O}[M]\langle r \rangle \mid r(w). \bar{y}(w', r'). (w' \triangleright w \mid r' \triangleright p))) \quad (*) \\
\mathcal{O}[MV] &\stackrel{\text{def}}{=} (p) \nu q (\mathcal{O}[M]\langle q \rangle \mid q(y). \nu w (\mathcal{O}^*[V]\langle w \rangle \mid \bar{y}(w', r'). (w' \triangleright w \mid r' \triangleright p))) \quad (*) \\
\mathcal{O}[MN] &\stackrel{\text{def}}{=} (p) \nu q (\mathcal{O}[M]\langle q \rangle \mid q(y). \nu r (\mathcal{O}[N]\langle r \rangle \mid r(w). \bar{y}(w', r'). (w' \triangleright w \mid r' \triangleright p))) \quad (**) \\
\mathcal{O}[V] &\stackrel{\text{def}}{=} (p) \bar{p}(y). \mathcal{O}^*[V]\langle y \rangle
\end{aligned}$$

where in the rules marked (*), M is not a value, and in the rule marked (**) M and N are not values; and where $\mathcal{O}^*[V]$ is thus defined :

$$\begin{aligned}
\mathcal{O}^*[\lambda x. M] &\stackrel{\text{def}}{=} (y) !y(x, q). \mathcal{O}[M]\langle q \rangle \\
\mathcal{O}^*[x] &\stackrel{\text{def}}{=} (y) y \triangleright x
\end{aligned}$$

.....

Encoding of environments:

$$\begin{aligned}
\mathcal{O}[[y \mapsto V] \cdot \gamma'] &\stackrel{\text{def}}{=} \mathcal{O}^*[V]\langle y \rangle \mid \mathcal{O}[\gamma'] \\
\mathcal{O}[[q \mapsto (E, p)] \cdot \gamma'] &\stackrel{\text{def}}{=} q(x). \mathcal{O}[E[x]]\langle p \rangle \mid \mathcal{O}[\gamma'] \\
\mathcal{O}[\varepsilon] &\stackrel{\text{def}}{=} 0
\end{aligned}$$

.....

Encoding of configurations:

$$\begin{aligned}
\mathcal{O}[\langle M, p, \gamma \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{O}[M]\langle p \rangle \mid \mathcal{O}[\gamma], \phi \rangle \\
\mathcal{O}[\langle \gamma \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{O}[\gamma], \phi \rangle \\
\mathcal{O}[\langle M \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{O}[M], \phi \rangle
\end{aligned}$$

■ **Figure 7** The optimised encoding

903 $\mathcal{O}\llbracket M \rrbracket\langle q \rangle$; however it is not in the same relation with $\mathcal{O}\llbracket Vx \rrbracket\langle q \rangle$ (which has extra τ transitions
904 with respect to $\mathcal{O}\llbracket M \rrbracket\langle q \rangle$).

905 We can now establish the operational correspondence between OGS and $I\pi$. (In all
906 the results about operational correspondence, we exploit the convention on freshness of
907 bound names of actions and traces produced by OGS configurations and $I\pi$ processes, as by
908 Remark 1.)

909 ► **Lemma 52** (strong transitions, from A-OGS to $I\pi$).

- 910 1. If $F \longrightarrow_a F'$, then $\mathcal{O}\llbracket F \rrbracket \longrightarrow_{\pi} \pi \gtrsim \mathcal{O}\llbracket F' \rrbracket$;
911 2. If $F \xrightarrow{\ell}_a F'$, then $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\ell}_{\pi} \approx_{\pi} \mathcal{O}\llbracket F' \rrbracket$.

912 **Proof.** The proof goes by a case analysis on the rule used to derive the transition from F ,
913 using the definition of the encoding, using Lemmas 49-51. ◀

914 ► **Lemma 53** (strong transitions, from $I\pi$ to A-OGS).

- 915 1. If $\mathcal{O}\llbracket F \rrbracket \longrightarrow_{\pi} P$ then there is F' such that $F \longrightarrow_a F'$ and $P \pi \gtrsim \mathcal{O}\llbracket F' \rrbracket$;
916 2. If $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\mu}_{\pi} P$ and μ is an output, then there is F' such that $F \xrightarrow{\mu}_a F'$ and $P \pi \gtrsim \mathcal{O}\llbracket F' \rrbracket$
917 ;
918 3. If F is passive and $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\mu}_{\pi} P$, then there is F' such that $F \xrightarrow{\mu}_a F'$ and $P \approx_{\pi} \mathcal{O}\llbracket F' \rrbracket$.

919 **Proof.** The proof goes by a case analysis on the possible transition from $\mathcal{O}\llbracket F \rrbracket$, again
920 exploiting Lemmas 49-51. ◀

921 We can strengthen both Lemma 52 and Theorem 4 to a correspondence between transitions
922 from A-OGS configurations and from their $I\pi$ translation under the (more restrictive) opLTS.
923 Further, as discussed in Remark 10 and Appendix B, we can use the expansion relation and
924 bisimilarity on the opLTS, written $\lesssim_{o\pi}$ and $\approx_{o\pi}$. We write $I\pi^{op}$ to refer to $I\pi$ under the
925 opLTS.

926 ► **Theorem 54** (strong transitions, from A-OGS to $I\pi^{op}$).

- 927 1. If $F \longrightarrow_a F'$, then $\mathcal{O}\llbracket F \rrbracket \longrightarrow_{o\pi} \approx_{o\pi} \mathcal{O}\llbracket F' \rrbracket$;
928 2. If $F \xrightarrow{\ell}_a F'$, then $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\ell}_{o\pi} \approx_{o\pi} \mathcal{O}\llbracket F' \rrbracket$.

929 ► **Theorem 55** (weak transitions, from A-OGS to $I\pi^{op}$).

- 930 1. If $F \Longrightarrow_a F'$, then $\mathcal{V}\llbracket F \rrbracket \Longrightarrow_{o\pi} \approx_{o\pi} \mathcal{V}\llbracket F' \rrbracket$;
931 2. If $F \xrightarrow{\ell}_a F'$, then $\mathcal{V}\llbracket F \rrbracket \xrightarrow{\ell}_{o\pi} \approx_{o\pi} \mathcal{V}\llbracket F' \rrbracket$.

932 The following Theorem 56 is needed in the proof of Theorem 7: it relates strong transitions
933 from $I\pi^{op}$ processes to transitions of A-OGS configurations. Again, in both theorems, the
934 occurrences of \lesssim_{π} and \approx_{π} can be replaced by $\lesssim_{o\pi}$ and $\approx_{o\pi}$.

935 ► **Theorem 56.**

- 936 1. If $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\tau}_{o\pi} P$ then there is F' such that $F \longrightarrow_a F'$ and $P \pi \gtrsim \mathcal{O}\llbracket F' \rrbracket$;
937 2. If $\mathcal{O}\llbracket F \rrbracket \xrightarrow{\ell}_{o\pi} P$ then there is F' such that $F \xrightarrow{\ell}_a F'$ and $P \approx_{\pi} \mathcal{O}\llbracket F' \rrbracket$.

938 We report now more details on the proof of Corollary 8. First, we recall the assertion:

Suppose F has an irredundant name-support.
939 For any trace s , we have $F \xrightarrow{s}_a$ iff $\mathcal{V}\llbracket F \rrbracket \xrightarrow{s}_{o\pi}$.

940 **Proof.** The result is first established for \mathcal{O} , and then extended to \mathcal{V} , exploiting the correctness
 941 of the optimisations in \mathcal{O} (Lemma 47; again, in this lemma, the occurrence of \lesssim_π can be
 942 lifted to $\lesssim_{\circ\pi}$).

943 For \mathcal{O} , both directions are proved proceeding by induction on the length of a trace
 944 ℓ_1, \dots, ℓ_n . In the direction from left to right, we use Theorem 55(2). For the converse
 945 direction, we proceed similarly, this time relying on Theorem 7(2) (again, the version with
 946 $\approx_{\circ\pi}$ in place of \approx_π). ◀

947 **E** Auxiliary results for Section 7.1

948 We report here auxiliary results needed to establish the relationship between Concurrent
 949 OGS (C-OGS) and $\text{I}\pi$.

950 The two lemmas below are needed in the proofs of Theorems 13 and 22. The reasoning
 951 in their proofs is similar to that for Lemmas 52 and 53.

952 ▶ **Lemma 57** (from C-OGS to $\text{I}\pi$, strong transitions).

- 953 1. If $F \rightarrow_c F'$, then $\mathcal{O}[F] \rightarrow_{\pi\pi} \gtrsim \mathcal{O}[F']$;
- 954 2. If $F \xrightarrow{\ell}_c F'$, then $\mathcal{O}[F] \xrightarrow{\ell}_\pi \approx_\pi \mathcal{O}[F']$;

955 ▶ **Lemma 58** (from $\text{I}\pi$ to C-OGS, strong transitions).

- 956 1. If $\mathcal{O}[F] \rightarrow_\pi P$ then there is F' such that $F \rightarrow_c F'$ and $P \pi \gtrsim \mathcal{O}[F']$;
- 957 2. If $\mathcal{O}[F] \xrightarrow{\ell}_\pi P$ then there is F' such that $F \xrightarrow{\ell}_c F'$ and $P \approx_\pi \mathcal{O}[F']$.

958 The results below are used in the proof of Lemma 16.

959 A process P has *deterministic immediate transitions* if, for any μ , whenever $P \xrightarrow{\mu}_\pi P_1$
 960 and $P \xrightarrow{\mu}_\pi P_2$, then $P_1 = P_2$.

961 ▶ **Lemma 59.** Suppose P, Q have deterministic immediate transitions, $P \simeq_\pi Q$, and $P \xrightarrow{\mu}_\pi P_1$.
 962 Then $Q \xrightarrow{\mu}_\pi Q_1$ implies $P_1 \simeq_\pi Q_1$.

963 Lemma below is the analogous to Lemma 45 for traces.

964 ▶ **Lemma 60.**

- 965 ■ Suppose $P = P_1 \mid !a(\tilde{b}).P_2$, and $Q = Q_1 \mid !a(\tilde{b}).Q_2$, and $\langle P, \phi \rangle \simeq_\pi \langle Q, \phi \rangle$. Then also
 966 $\langle P_1, \phi \rangle \simeq_\pi \langle Q_1, \phi \rangle$.
- 967 ■ Similarly, suppose $P = P_1 \mid a(\tilde{b}).P_2$, and $Q = Q_1 \mid a(\tilde{b}).Q_2$, and $\langle P, \phi \rangle \simeq_\pi \langle Q, \phi \rangle$. Then
 968 also $\langle P_1, \phi \rangle \simeq_\pi \langle Q_1, \phi \rangle$.

969 We can now conclude the proof of Lemma 16. We recall the assertion:

970 For any M, N we have:

$$\mathcal{V}[M] \simeq_\pi \mathcal{V}[N] \text{ iff } \mathcal{V}[M] \approx_\pi \mathcal{V}[N].$$

971 **Proof.** We have to prove that trace equivalence implies bisimilarity. We work with the
 972 optimised encoding \mathcal{O} . The relation

973 $(\mathcal{O}[F], \mathcal{O}[G]) : F, G \text{ are singleton with } F \simeq_c G$

974 is a bisimulation up-to context and up-to $(\pi \gtrsim, \approx_\pi)$.

975 F and G , as singleton, are of the forms described in Lemma 44. Moreover, to be in the
 976 relation \simeq_c they must be of the same form, and, using also Lemmas 49 and 50 we deduce
 977 that they have deterministic immediate transitions.

978 Suppose F and G are of the form $\mathcal{V}[M]\langle p \rangle$ and $\mathcal{V}[N]\langle q \rangle$, respectively.

979 Consider the case $\mathcal{V}[[M]]\langle p \rangle \xrightarrow{\tau} P$; then by Lemma 49, $P \pi \gtrsim \mathcal{V}[[M']]\langle p \rangle$, for some M' .
 980 Moreover, we have $\mathcal{V}[[M']]\langle p \rangle \simeq_c \mathcal{V}[[M]]\langle p \rangle \simeq_c \mathcal{V}[[N]]\langle q \rangle$, and we are done.

981 The case when $\mathcal{V}[[M]]\langle p \rangle$ performs an output action $\bar{p}(x)$ is simple, as usual using Lemma 49;
 982 we also deduce that $p = q$.

983 Suppose now $\mathcal{V}[[M]]\langle p \rangle$ performs an output action $\bar{x}(z, p')$. Using Lemma 49, we have

$$984 \quad \mathcal{V}[[M]]\langle p \rangle \xrightarrow{\bar{x}(z, p')} \pi \gtrsim \mathcal{O}^*[[V]]\langle z \mid p'(y) \rangle \cdot \mathcal{O}[[E[y]]]\langle p \rangle$$

985 for some V, y, E . Since $\mathcal{V}[[M]]\langle p \rangle \simeq_c \mathcal{V}[[N]]\langle q \rangle$, and again using Lemma 49, we deduce

$$986 \quad \mathcal{V}[[N]]\langle q \rangle \implies \xrightarrow{\bar{x}(z, p')} \pi \gtrsim \mathcal{O}^*[[W]]\langle z \mid q'(y) \rangle \cdot \mathcal{O}[[E'[y]]]\langle q \rangle$$

987 By Lemma 59,

$$988 \quad \begin{aligned} & \mathcal{O}^*[[V]]\langle z \mid p'(y) \rangle \cdot \mathcal{O}[[E[y]]]\langle p \rangle \simeq_c \\ & \mathcal{O}^*[[W]]\langle z \mid q'(y) \rangle \cdot \mathcal{O}[[E'[y]]]\langle q \rangle \end{aligned}$$

989 Applying Lemma 60 twice, we deduce

$$990 \quad \mathcal{O}^*[[V]]\langle z \rangle \simeq_c \mathcal{O}^*[[W]]\langle z \rangle$$

991 and

$$992 \quad q'(y) \cdot \mathcal{O}[[E'[y]]]\langle q \rangle \simeq_c q'(y) \cdot \mathcal{O}[[E[y]]]\langle q \rangle$$

993 and we are done, up to expansion and context.

994 The cases when F and G are of a different form are similar, this time using Lemma 50. ◀

995 **F** Auxiliary results for Section 7.3

996 In Definition 21 we have introduced the technique of *bisimulation up-to reduction and*
 997 *composition*. Sometimes we do not need the ‘closure under reduction’; that is, referring to
 998 the clauses of Definition 21 we may take $F'_i = F_i$, $i = 1, 2$). In this case we say that \mathcal{R} is a
 999 ‘bisimulation up-to composition’.

1000 We now present an additional up-to technique, which is used in when relating OGS bisim-
 1001 ilarity with eager normal form bisimilarity. When \mathcal{R} is a relation on singleton configuration,
 1002 a straightforward simplification of ‘bisimulation up-to reduction and composition’ allows us to
 1003 play the bisimulation game only on visible actions; the bisimulation clause becomes:

1004 ■ if $F_1 \implies_c \xrightarrow{\ell} F'_1$ then there is F'_2 such that $F_2 \implies_c \xrightarrow{\ell} F'_2$ and $F'_1 \mathcal{R}^! F'_2$.

1005 We call this a *singleton bisimulation up-to composition*; its soundness is a corollary of that
 1006 for bisimulation up-to reduction and composition, derived from the up-to techniques for $\text{I}\pi$
 1007 and the mapping from C-OGS onto $\text{I}\pi$.

1008 ► **Proposition 61.** *If \mathcal{R} is a singleton bisimulation up-to composition, then $\mathcal{R} \subseteq \approx_c$.*

Proof. Take

$$\mathcal{S} \stackrel{\text{def}}{=} \mathcal{R} \cup \{(F_1, F_2) : \begin{array}{l} \text{there are } F'_1, F'_2 \text{ with } F'_1 \mathcal{R} F'_2 \\ \text{and } F'_i \implies_c F_i \text{ } i = 1, 2 \end{array}\}$$

1009 Intuitively, \mathcal{S} is the closure of \mathcal{R} under deterministic reductions. Then \mathcal{S} is a bisimulation
 1010 up-to composition. (Note: we exploit the fact that transitions of singleton configurations are
 1011 deterministic.) ◀

1012 We now report more details on the proof of Theorem 22. We recall its assertion:

1013 (**Theorem 22**) If \mathcal{R} is bisimulation up-to reduction and composition then $\mathcal{R} \subseteq \approx_c$.

1014 We first need the following Lemma 62, which is a consequence of the compositionality of
1015 the mapping from C-OGS to $I\pi$.

1016 ► **Lemma 62.** *For a relation \mathcal{R} on C-OGS configurations, if $(F_1, F_2) \in \mathcal{R}^{l^*}$, then we have*
1017 $(\mathcal{O}[[F_1]], \mathcal{O}[[F_2]]) \in (\mathcal{O}[[\mathcal{R}]])^{\text{ctx}}$.

1018 We are now ready to show the proof of Theorem 22.

1019 **Proof.** We prove the result via $I\pi$, exploiting Lemma 15(2). Here again, it is convenient to
1020 use the optimised encoding \mathcal{O} .

1021 If \mathcal{R} is a relation on C-OGS configurations, then $\mathcal{O}[[\mathcal{R}]]$ is the relation on $I\pi$ processes
1022 obtained by mapping each pair in \mathcal{R} in the expected manner:

$$1023 \quad \mathcal{O}[[\mathcal{R}]] \stackrel{\text{def}}{=} \{(\mathcal{O}[[F_1]], \mathcal{O}[[F_2]]) : F_1 \mathcal{R} F_2\}$$

1024 And, if \mathcal{S} is a relation on $I\pi$ processes, then \mathcal{S}^{ctx} is the closure of \mathcal{S} under polyadic contexts,
1025 i.e., the set of pairs of the form $(C_{\text{ctx}}[P_1, \dots, P_n], C_{\text{ctx}}[Q_1, \dots, Q_n])$ where C_{ctx} is a polyadic
1026 context and for each i , $P_i \mathcal{S} Q_i$.

1027 As a consequence of the compositionality of the mapping from C-OGS to $I\pi$, for a relation
1028 \mathcal{R} on C-OGS configurations, if $(F_1, F_2) \in \mathcal{R}^{l^*}$, then $(\mathcal{O}[[F_1]], \mathcal{O}[[F_2]]) \in (\mathcal{O}[[\mathcal{R}]])^{\text{ctx}}$.

1029 Now, let \mathcal{R} be the relation in the hypothesis of the theorem. We prove the theorem
1030 by showing that $\mathcal{O}[[\mathcal{R}]]$ is a bisimulation up-to context and up-to $(\pi \succ, \approx_\pi)$ in $I\pi$, and then
1031 appealing to Theorem 40.

1032 Suppose $\mathcal{O}[[F_1]] \xrightarrow{\ell} \pi P$. Then there is F'_1 with $F_1 \xrightarrow{\ell} \pi F'_1$ and $P \approx_\pi \mathcal{O}[[F'_1]]$ (this is
1033 derived from Lemma 57). Since \mathcal{R} is bisimulation up-to reduction and composition and
1034 $F_1 \mathcal{R} F_2$, there are F''_1 , and F'_2 , with $F'_1 \Longrightarrow_c F''_1$, $F_2 \xrightarrow{\ell} \pi \Longrightarrow_c F'_2$ and $F''_1 \mathcal{R}^{l^*} F'_2$.

1035 Using Theorem 13(1) from $F'_1 \Longrightarrow_c F''_1$ and the inclusion $\pi \succ \subseteq \approx_\pi$, we derive $\mathcal{O}[[F'_1]] \approx_\pi$
1036 $\mathcal{O}[[F''_1]]$; and by Theorem 13(1), and Lemmas 47 and 57, from $F_2 \xrightarrow{\ell} \pi \Longrightarrow_c F'_2$ we derive
1037 $\mathcal{O}[[F_2]] \xrightarrow{\ell} \pi \approx_\pi \mathcal{O}[[F'_2]]$.

1038 Finally, from $(F''_1, F'_2) \in \mathcal{R}^{l^*}$, we derive $(\mathcal{O}[[F''_1]], \mathcal{O}[[F'_2]]) \in (\mathcal{O}[[\mathcal{R}]])^{\text{ctx}}$. This closes the proof,
1039 up-to polyadic contexts and bisimilarity.

1040 The case when $\mathcal{O}[[F_1]]$ makes a silent move is simpler, using expansion in place of
1041 bisimilarity. ◀

1042 **G** Auxiliary results for Section 7.4

1043 ► **Lemma 63.** *For any term A-OGS configuration, its traces in A-OGS are a subset of the*
1044 *traces in C-OGS, precisely, the traces with an alternation between Player and Opponent*
1045 *actions.*

1046 We now prove Lemma 24.

1047 **Proof.** In this proof, we write $F \subseteq G$ when, as partial maps, G is an extension of F .

1048 Let \mathcal{R} be the relation on configurations with $F \mathcal{R} G$ if:

- 1049 1. F, G are singleton, and
- 1050 2. either
 - 1051 a. there are passive F', G' with $F \subseteq F'$ and $G \subseteq G'$ and $F' \approx_a G'$,

XX:30 Games, mobile processes, and functions

1052 b. or there are p, M, N, γ, δ with $F = \langle [p \mapsto M], \phi \rangle$, $G = \langle [p \mapsto N], \phi \rangle$, and $\langle [p \mapsto$
 1053 $M], \gamma, \phi \rangle \approx_a \langle [p \mapsto N], \delta, \phi \rangle$

1054 We show that \mathcal{R} is a bisimulation up-to composition. We distinguish the cases in which
 1055 (2.a) or (2.b) holds. First suppose $F \mathcal{R} G$ because (2.a) holds, and there are F', G' as in
 1056 (2.a). Suppose

$$1057 \quad F \xrightarrow{\mu}_c F_1 \tag{1}$$

in C-OGS. The rule can be OA or OQ. We assume it is OQ; the case for OA is similar. We thus have, for some p, V, x, y

$$\begin{aligned} F &= \langle \gamma_1, \phi \rangle \text{ with } \gamma_1 = [x \mapsto V] \\ \mu &= x(y, p) \\ F_1 &= \langle A, \gamma_1, \phi \uplus \{y\} \rangle \text{ with } A = [p \mapsto Vy] \end{aligned}$$

1058 with $F \xrightarrow{\mu}_a F_1$. Moreover, we have $G = \langle \delta_1, \phi \rangle$ with $\delta_1 = [x \mapsto V']$. since $F \approx_a G$, there is
 1059 G_1 such that

$$1060 \quad G \xrightarrow{\mu}_a G_1 \approx_a F_1$$

1061 and $G_1 = \langle B, \delta_1, \phi \uplus \{y\} \rangle$ for $B = [p \mapsto Vy]$. We also have

$$1062 \quad G \xrightarrow{\mu}_c G_1$$

1063 This is sufficient to match (1), as $F_1 \mathcal{R}^{\uparrow} G_1$: indeed we have both $\langle \gamma_1, \phi \rangle \mathcal{R} \langle \delta_1, \phi \rangle$ and (using
 1064 clause (2.b)) $\langle A, \phi \uplus \{y\} \rangle \mathcal{R} \langle B, \phi \uplus \{y\} \rangle$.

1065 Now the case (2.b); let p, γ, δ, ϕ as in (2.b). There are 3 possibilities of transitions,
 1066 corresponding to rules P τ , PA, PQ. The case of P τ is straightforward, since bisimilarity
 1067 is preserved by internal actions. We only look at PQ, as PA is simpler. Thus we have
 1068 $F = \langle [p \mapsto E[xV]], \phi \rangle$, for some E, x, V and then

$$1069 \quad F \xrightarrow{\bar{x}(y,q)}_c \langle [y \mapsto V] \cdot [q \mapsto (E, p)], \phi \uplus \{y, q\} \rangle \stackrel{\text{def}}{=} F_1 \tag{2}$$

1070 We also have

$$1071 \quad F \xrightarrow{\bar{x}(y,q)}_a F_1$$

1072 As $F \approx_a G$, there are E', W such that

$$1073 \quad G \Longrightarrow \langle [p \mapsto E'[xW]], \phi \rangle \xrightarrow{\bar{x}(y,q)}_a G_1$$

1074 with $G_1 \stackrel{\text{def}}{=} \langle [y \mapsto W] \cdot [q \mapsto (E', p)], \phi \uplus \{y, q\} \rangle$ and $F_1 \approx_a G_1$. We also have $G \xrightarrow{\bar{x}(y,q)}_c G_1$. From
 1075 $F_1 \approx_a G_1$, appealing to (2.a) we deduce that both $\langle [y \mapsto V], \phi \uplus \{y\} \rangle \mathcal{R} \langle [y \mapsto W], \phi \uplus \{y\} \rangle$
 1076 and $\langle [q \mapsto (E, p)], \phi \uplus \{q\} \rangle \mathcal{R} \langle [q \mapsto (E', p)], \phi \uplus \{q\} \rangle$ hold. Hence $F_1 \mathcal{R}^{\uparrow} G_1$.

1077 In summary, as an answer to the challenge (2), we have found G_1 such that $G \xrightarrow{\bar{x}(y,q)}_c G_1$
 1078 and $F_1 \mathcal{R}^{\uparrow} G_1$; this closes the proof. ◀

1079 **H** Tensor Product

1080 In this section, we prove results about the tensor product of compatible configurations for
 1081 the Concurrent, the Alternating and the Well-Bracketed LTS presented in Section 7.2. We
 1082 relate the traces generated by the the tensor product of two configurations to the set of
 1083 interleavings of traces generated by the component configurations themselves. The result is
 1084 proved by going through the π -calculus.

1085 **H.1** C-OGS

1086 Two π -calculus processes P_1, P_2 *cannot interact* if there is no name that appears free in both
 1087 processes and with opposite polarities (that is, in input position in one, and output position
 1088 in the other). In $\mathcal{I}\pi$, this property is invariant for transitions (the invariance is false in the
 1089 full π -calculus).

1090 ► **Lemma 64.** *If P_1, P_2 cannot interact, and $P_1 \mid P_2 \xrightarrow{\mu}_{\pi} P'_1 \mid P'_2$ then also P'_1, P'_2 cannot*
 1091 *interact.*

1092 ► **Lemma 65.** *Suppose P_1, P_2 cannot interact.*

- 1093 1. *If $P_1 \mid P_2 \xrightarrow{s}_{\pi} P'_1 \mid P'_2$, then, for $i = 1, 2$, there is s_i such that $P_i \xrightarrow{s_i}_{\pi} P'_i$, and*
 1094 *$s \in \mathbf{inter}(s_1, s_2)$;*
- 1095 2. *Conversely, if for $i = 1, 2$, we have $P_i \xrightarrow{s_i}_{\pi} P'_i$, and $s \in \mathbf{inter}(s_1, s_2)$, then $P_1 \mid P_2 \xrightarrow{s}_{\pi}$*
 1096 *$P'_1 \mid P'_2$.*

1097 **Proof.** Straightforward. The interesting case is (1). Consider the sequence of one-step
 1098 actions (including silent actions) that are performed to obtain the trace $P_1 \mid P_2 \xrightarrow{s}_{\pi} P'_1 \mid P'_2$.
 1099 Then tag each of these actions with L or R depending on whether in the derivation proof of
 1100 that action, the last rule applied was **parL** or **parR**. Then s_1 is obtained by collecting the
 1101 subsequence of s with the **parL** tag, and similarly for s_2 with **parR**. ◀

1102 ► **Lemma 66.** *Taking two C-OGS configurations F_1, F_2 , if their polarity function are com-*
 1103 *patible then $\mathcal{V}[[F_1]]$ and $\mathcal{V}[[F_2]]$ cannot interact.*

1104 The proof of Lemma 20 then follows from these two lemmas, Corollary 14, and the fact
 1105 that $\mathcal{V}[[F_1 \otimes F_2]] \equiv \mathcal{V}[[F_1]] \mid \mathcal{V}[[F_2]]$.

1106 **H.2** A-OGS

1107 ► **Definition 67.** *Taking F_1, F_2 two A-OGS configurations, with γ_i, ϕ_i their respective enviro-*
 1108 *nement and set of names, $F_1 \otimes F_2$ is defined as:*

- 1109 ■ $\langle M, p, \gamma_1 \cdot \gamma_2, \phi_1 \cup \phi_2 \rangle$ *when one of F_1, F_2 is active, with M, p its toplevel term and*
 1110 *continuation name;*
- 1111 ■ $\langle \gamma_1 \cdot \gamma_2, \phi_1 \cup \phi_2 \rangle$ *when F_1, F_2 are passive;*

1112 We can then state the adaptation of Lemma 20 to A-OGS. We write $\mathbf{inter}_a(s_1, s_2)$ for
 1113 the subset of $\mathbf{inter}(s_1, s_2)$ formed by *alternated* traces.

1114 ► **Lemma 68.** *Suppose F_1, F_2 are compatible A-OGS configurations, and at most one of the*
 1115 *two is active. Then the set of traces generated by $F_1 \otimes F_2$ is equal to the set of interleavings*
 1116 *$\mathbf{inter}_a(s_1, s_2)$, with $F_1 \xrightarrow{s_1}_a$ and $F_2 \xrightarrow{s_2}_a$.*

1117 To prove Lemma 68, we use the corespondance between A-OGS and $\xrightarrow{\cdot}_{o\pi}$ the output-
 1118 prioritised LTS introduced at the end of Section 6.1, via the following Lemma.

1119 ► **Lemma 69.** *Suppose F_1, F_2 are compatible configurations, and at most one of the two is*
 1120 *active.*

- 1121 1. *If $\mathcal{O}[[F_1]] \mid \mathcal{O}[[F_2]] \xrightarrow{s}_{o\pi} P_1 \mid P_2$, then, for $i = 1, 2$, there are traces s_i such that*
 1122 *$\mathcal{O}[[F_i]] \xrightarrow{s_i}_{o\pi} P_i$, and $s \in \mathbf{inter}_a(s_1, s_2)$.*
- 1123 2. *Conversely, if for $i = 1, 2$, we have $\mathcal{O}[[F_i]] \xrightarrow{s_i}_{o\pi} P_i$, $s \in \mathbf{inter}_a(s_1, s_2)$, then also*
 1124 *$\mathcal{O}[[F_1]] \mid \mathcal{O}[[F_2]] \xrightarrow{s}_{o\pi} P_1 \mid P_2$.*

E	$:= EM \mid [\cdot]$	evaluation contexts
V	$:= \lambda x. M \mid v$	values
γ	$:= [v \mapsto V] \cdot \gamma' \mid [q \mapsto (E, p)] \cdot \gamma' \mid [x \mapsto M] \cdot \gamma' \mid \varepsilon$	environments
F	$:= \langle L, p, \gamma, \phi \rangle \mid \langle \gamma, \phi \rangle \mid \langle [? \mapsto \phi], M \rangle$	configurations
L	$:= M \mid E[vM] \mid v$	extended λ -terms

■ **Figure 8** Grammars for call-by-name A-OGS configurations

1125 **Proof.** Proof of (1). We know

$$1126 \quad \mathcal{O}[[F_1]] \mid \mathcal{O}[[F_2]] \xRightarrow{s}_{o\pi} P_1 \mid P_2 \tag{3}$$

Hence, by the definition of the LTS \mapsto , we also have

$$\mathcal{O}[[F_1]] \mid \mathcal{O}[[F_2]] \xRightarrow{s}_{\pi} P_1 \mid P_2.$$

1127 We now apply Lemma 65 to infer $\mathcal{O}[[F_i]] \xRightarrow{s_i}_{\pi} P_i$. It also holds $\mathcal{O}[[F_i]] \xRightarrow{s_i}_{o\pi} P_i$ (the
 1128 observations in s_i were possible in (3), with an extra parallel component, which may only
 1129 reduce the possibility of transitions in the LTS \mapsto).

1130 For (2): $s \in \mathbf{inter}_a(s_1, s_2)$ implies that each s_i is alternating, at most one of them is
 1131 P -starting, s is alternating, and active iff one of the composing traces is P -starting. This
 1132 means that we can assume that when an input transition of s is performed the source process
 1133 is input reactive. ◀

1134 Then Lemma 68 follows from the previous lemma and the operational correspondence for
 1135 the output-prioritised LTS $\Longrightarrow_{o\pi}$ (Corollary 8).

1136 I The call-by-name λ -calculus

1137 The relationship between OGS and π -calculus representations of the λ -terms examined in
 1138 Sections 6 and 7 is not specific to call-by-value. A similar relationship can be established
 1139 for call-by-name. We summarise here the main aspects and the main results. We maintain
 1140 terminologies and notations in the call-by-value sections. Call-by-name language is given in
 1141 Figure 8. Its reduction \rightarrow_n is defined by the following two rules:

$$\frac{}{(\lambda x. M)N \rightarrow_n M\{N/x\}} \quad \frac{M \rightarrow_n N}{E[M] \rightarrow_n E[N]}$$

1142 The OGS for call-by-name uses three kinds of names: *continuation names* (ranged over
 1143 by p, q), *variable names* (ranged over by x, y), and *value-variables* (ranged over by v, w).
 1144 Continuation names are used as in call-by-value; variable names acts as pointers to λ -terms
 1145 and correspond to the variables of the λ -calculus; value names are pointers to values.

1146 The OGS uses the syntactic categories in Figure 8, where L range over extended λ -terms,
 1147 which are needed to accommodate value names. We write Λ^+ for the set of extended λ -terms.

1148 In a call-by-name setting, Player Questions are of two possible shapes:

- 1150 ■ *Player Value-Question* (PVQ) $\bar{v}(x, p)$, receiving a variable x and a continuation name p
 1151 through a value name v ;

(τ)	$\langle M, p, \gamma, \phi \rangle$	$\xrightarrow{\tau}_a$	$\langle N, p, \gamma, \phi \rangle$	when $M \rightarrow_n N$
(PA)	$\langle V, p, \gamma, \phi \rangle$	$\xrightarrow{\bar{p}(v)}_a$	$\langle \gamma \cdot [v \mapsto V], \phi \uplus \{v\} \rangle$	
(PVQ)	$\langle E[vM], p, \gamma, \phi \rangle$	$\xrightarrow{\bar{v}(y,q)}_a$	$\langle \gamma \cdot [y \mapsto M] \cdot [q \mapsto (E, p)], \phi \uplus \{y, q\} \rangle$	
(PTQ)	$\langle E[x], p, \gamma, \phi \rangle$	$\xrightarrow{\bar{x}(q)}_a$	$\langle \gamma \cdot [q \mapsto (E, p)], \phi \uplus \{q\} \rangle$	
(OA)	$\langle \gamma \cdot [p \mapsto (E, q)], \phi \rangle$	$\xrightarrow{p(v)}_a$	$\langle E[v], q, \gamma, \phi \uplus \{v\} \rangle$	
(OVQ)	$\langle \gamma \cdot [v \mapsto V], \phi \rangle$	$\xrightarrow{v(y,p)}_a$	$\langle Vy, p, \gamma, \phi \uplus \{p, y\} \rangle$	when $\gamma(v) = V$
(OTQ)	$\langle \gamma, \phi \rangle$	$\xrightarrow{x(p)}_a$	$\langle M, p, \gamma, \phi \uplus \{p\} \rangle$	when $\gamma(x) = M$
(IOQ)	$\langle [? \mapsto M], \phi \rangle$	$\xrightarrow{(p)}_a$	$\langle M, p, \varepsilon, \phi \uplus \{p\} \rangle$	

■ **Figure 9** CBN OGS LTS

- 1152 ■ *Player Term-Question* (PTQ) $\bar{x}(p)$, receiving a continuation name p through a variable x .
 1153 Opponent Questions are of two different shapes too:
 1154 ■ *Opponent Value-Question* (OVQ) $v(x, p)$, sending a variable x and a continuation name
 1155 p through a value name v ;
 1156 ■ *Opponent Term-Question* (OTQ) $x(p)$ sending a continuation name p through a variable
 1157 x .

1158 The encodings of call-by-name λ -calculus, OGS environments and configurations, and
 1159 associated syntactic categories, are reported in Figure 10. The $\text{I}\pi$ representation uses the
 1160 same 3 types of names as the OGS representation.

1161 The results of operational correspondence between OGS and $\text{I}\pi$ under the opLTS are the
 1162 same as for call-by-value. We only report the analogous of Corollary 11.

1163 ► **Corollary 70.** *For any F, F' we have: $F \simeq_a F'$ iff $\mathcal{N}[F] \simeq_{o\pi} \mathcal{N}[F']$ iff $F \approx_a F'$ iff*
 1164 $\mathcal{N}[F] \approx_{o\pi} \mathcal{N}[F']$.

1165 The modification to obtain the concurrent version of OGS (C-OGS) for call-by-name are
 1166 similar to those for call-by-value in Section 7.

1167 We report the results of operational correspondence and full abstraction.

1168 ► **Lemma 71** (C-OGS and $\text{I}\pi$, weak transitions). *Suppose $F \in \text{C-OGS}$.*

- 1169 1. *If $F \xrightarrow{\mu} F'$ then $\mathcal{N}[F] \xrightarrow{\mu} \approx_{\pi} \mathcal{N}[F']$;*
 1170 2. *conversely, if $\mathcal{N}[F] \xrightarrow{\mu} P$ then there is F' such that $F \xrightarrow{\hat{\mu}} F'$ and $P \approx_{\pi} \mathcal{N}[F']$.*

1171 As for call-by-value, so for call-by-name the 'up-to composition' technique for bisimulation
 1172 (Definition 21) can be imported from the π -calculus, and then used to establish that the
 1173 equivalence induced on λ -terms by the sequential and concurrent versions of OGS coincide.
 1174 Specifically, the technique is needed to prove the call-by-name analogous of Lemma 24

1175 ► **Corollary 72.** *For any λ -terms M, N , the following statement are the same: $\langle M \rangle \simeq_a \langle N \rangle$;*
 1176 $\langle M \rangle \approx_a \langle N \rangle$; $\langle M \rangle \simeq_c \langle N \rangle$; $\langle M \rangle \approx_c \langle N \rangle$; $\mathcal{N}[M] \simeq_{o\pi} \mathcal{N}[N]$; $\mathcal{N}[M] \approx_{o\pi} \mathcal{N}[N]$; $\mathcal{N}[M] \simeq_{\pi}$
 1177 $\mathcal{N}[N]$; $\mathcal{N}[M] \approx_{\pi} \mathcal{N}[N]$.

1178 The equivalence induced by the $\text{I}\pi$ (or π -calculus) encoding of the call-by-name λ -calculus
 1179 is known to coincide with that of the Lévy-Longo Trees [42]. In the light of Corollary 72, the
 1180 result can thus be transported to OGS, both in its Alternating and its Concurrent variants,
 1181 and both for traces and for bisimilarity.

Encoding of call-by-name Λ into $\text{I}\pi$ processes:

$$\begin{aligned} \mathcal{N}[\lambda x. M] &\stackrel{\text{def}}{=} (p) \bar{p}(v). !v(x, q). \mathcal{N}[M]\langle q \rangle \\ \mathcal{N}[x] &\stackrel{\text{def}}{=} (p) \bar{x}(r). r \triangleright p \\ \mathcal{N}[MN] &\stackrel{\text{def}}{=} (p) \nu q \left(\mathcal{N}[M]\langle q \rangle \mid !q(v). \bar{v}(x, p'). (p' \triangleright p \mid !x(r). \mathcal{N}[N]\langle r \rangle) \right) \end{aligned}$$

.....
Extension to Λ^+ :

$$\begin{aligned} \mathcal{N}[E[vM]] &\stackrel{\text{def}}{=} (p) \bar{v}(x, r). (\mathcal{N}[[r \mapsto (E, p)]] \mid !x(q). \mathcal{N}[M]q) \\ \mathcal{N}[v] &\stackrel{\text{def}}{=} (p) \bar{p}(w). w \triangleright v \end{aligned}$$

.....
Encoding of OGS environments to $\text{I}\pi$ processes:

$$\begin{aligned} \mathcal{N}[[v \mapsto V] \cdot \gamma] &\stackrel{\text{def}}{=} \begin{cases} v(x, q). \mathcal{N}[M]\langle q \rangle \mid \mathcal{N}[\gamma] & \text{if } V = \lambda x. M \\ v \triangleright w \mid \mathcal{N}[\gamma] & \text{if } V = w \end{cases} \\ \mathcal{N}[[x \mapsto M] \cdot \gamma] &\stackrel{\text{def}}{=} !x(q). \mathcal{N}[M]\langle q \rangle \mid \mathcal{N}[\gamma] \\ \mathcal{N}[[q \mapsto (E, p)] \cdot \gamma] &\stackrel{\text{def}}{=} \begin{cases} q(v). \bar{v}(x, r). (\mathcal{N}[[r \mapsto E'], p]] \mid !x(q). \mathcal{N}[M]\langle q \rangle) \mid \mathcal{N}[\gamma] & \text{if } E = E'M \\ q \triangleright p \mid \mathcal{N}[\gamma] & \text{if } E = [\cdot] \end{cases} \\ \mathcal{N}[\emptyset] &\stackrel{\text{def}}{=} 0 \end{aligned}$$

.....
Encoding of OGS configurations to $\text{I}\pi$ configurations:

$$\begin{aligned} \mathcal{N}[\langle M, p, \gamma, \phi \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{N}[M]\langle p \rangle \mid \mathcal{N}[\gamma], \phi \rangle \\ \mathcal{N}[\langle v, p, \gamma, \phi \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{N}[v]\langle p \rangle \mid \mathcal{N}[\gamma], \phi \rangle \\ \mathcal{N}[\langle E[vM], p, \gamma, \phi \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{N}[E[vM]]\langle p \rangle \mid \mathcal{N}[\gamma], \phi \rangle \\ \mathcal{N}[\langle \gamma, \phi \rangle] &\stackrel{\text{def}}{=} \langle \mathcal{N}[\gamma], \phi \rangle \end{aligned}$$

■ **Figure 10** The encoding of call-by-name λ into $\text{I}\pi$

1182 Connections between game semantics and tree representation of λ -terms have been
1183 previously explored, e.g., in [33], where a game model of untyped call-by-name λ -calculus
1184 is shown to correspond to Lévy-Longo trees. The connections that we have derived above,
1185 however, go beyond Lévy-Longo Trees: in particular, the bisimilarities that have been
1186 established between the observables (i.e., the dynamics) in the two models set a tight
1187 relationship between them while allowing one to transfer results and techniques along the
1188 lines of what we have shown for call-by-value.