The hierarchy of analytic semantics of weakly consistent parallelism

Jade Alglave (MSR-Cambridge, UCL, UK)
Patrick Cousot (NYU, Emer. ENS, PSL)

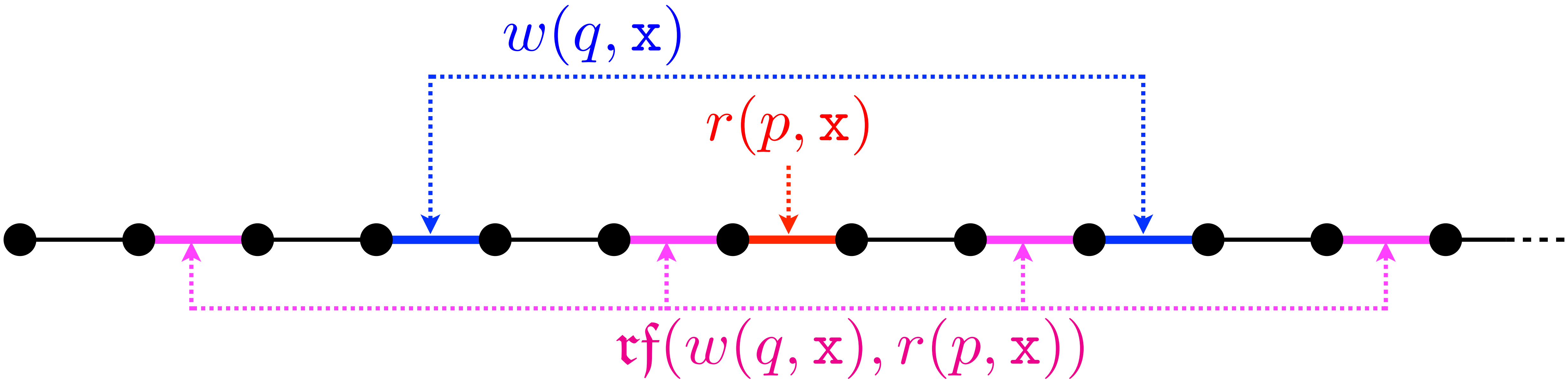
REPS AT SIXTY (co-located with SAS 2016)
Edinburgh, Scotland
September 11, 2016

Analytic semantics

Weak consistency models (WCM)

- Sequential consistency:
  reads $r(p, \mathtt{x})$ are *implicitly coordinated* with writes $w(q, \mathtt{x})$

- WCM:
  No implicit coordination (depends on architecture, program dependencies, and explicit fences)



Analytic semantic specification

- *Analytic* semantics = *Anarchic* semantics ∩ *Communication* semantics
- Anarchic semantics $S^{a}[\![\mathrm{P}]\!]$:

  describes computations of program P, no constraints on communications
- Communication semantics:

  imposes architecture-dependent communication constraints

  e.g.: cat language (Jade Alglave & Luc Maranget)

Jade Alglave, Patrick Cousot, Luc Maranget: **Syntax and semantics of the weak consistency model specification language cat.** CoRR abs/1608.07531 (2016)

Hierarchy of anarchic semantics

- Many different styles to describe the same computations e.g.
  - stateless/stateful
  - eager/lazy communications
  - interleaved versus true parallelism
  - …
- They form a hierarchy of abstractions
- The communication semantics is the same for all semantics in the hierarchy

Example: load buffer (LB)

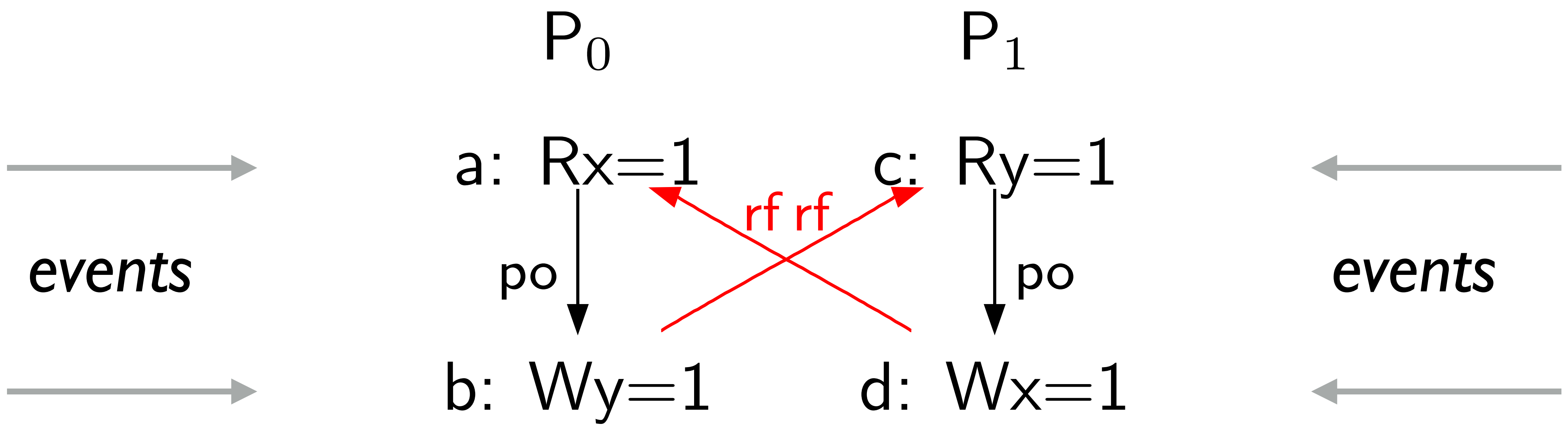
- Program:

```
{ x = 0; y = 0; }
P0          | P1         ;
r[] r1 x    | r[] r2 y   ;
w[] y 1     | w[] x  1   ;
exists(0:r1=1 /\ 1:r2=1)
```

- Example of execution trace $t \in \mathcal{S}^{\perp}[\![\mathtt{P}]\!]$:

$$t = w(\text{start}, \text{x}, 0)\ w(\text{start}, \text{y}, 0)\ r(\texttt{P0}, \text{x}, 1)\ \mathfrak{rf}[w(\texttt{P1}, \text{x}, 1), r(\texttt{P0}, \text{x}, 1)])\ w(\texttt{P0}, \text{y}, 1)\ r(\texttt{P1}, \text{y}, 1)\ w(\texttt{P1}, \text{x}, 1)\ \mathfrak{rf}[w(\texttt{P0}, \text{y}, 1), r(\texttt{P1}, \text{y}, 1)]$$

- Abstraction to cat *candidate execution* $\alpha_{\Xi}(t)$:



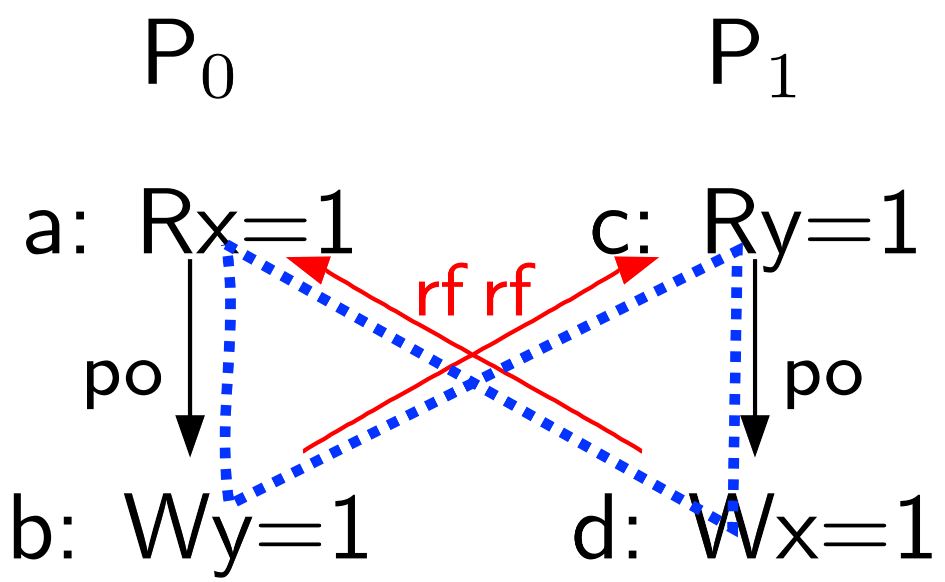
Example: load buffer (LB), cont'd

- cat specification:

```
acyclic (po | rf)+
```

The cat semantics rejects this execution $\alpha_{\Xi}(t)$:

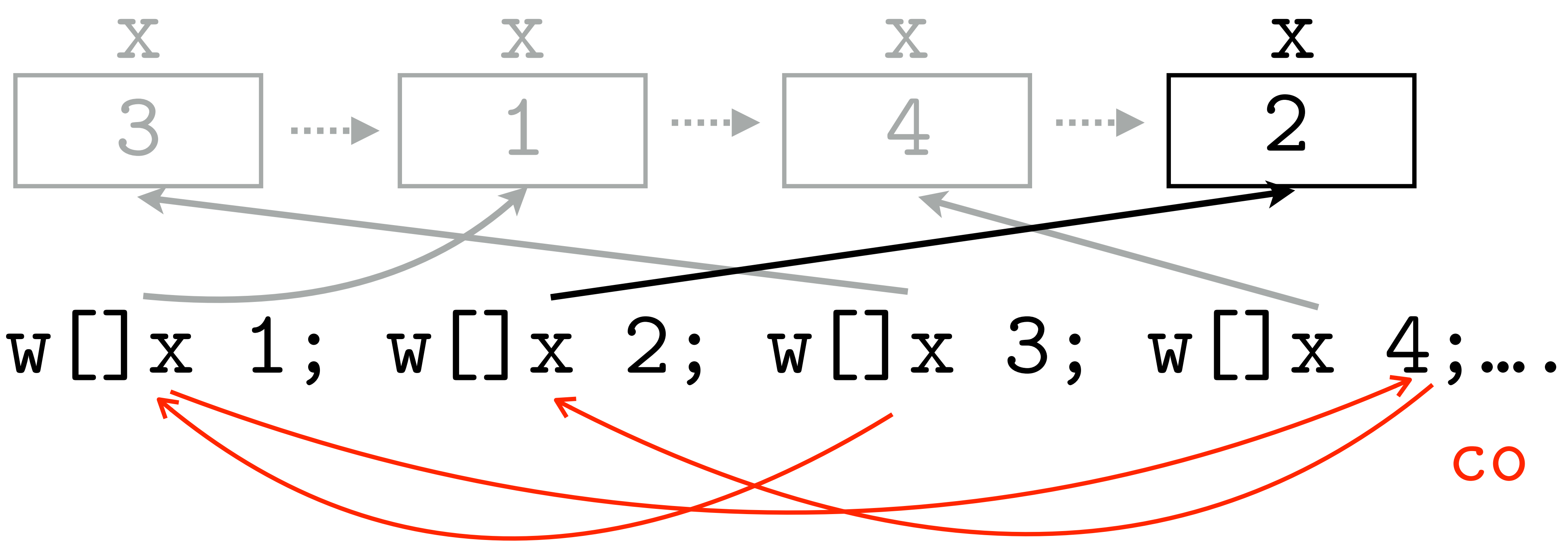
$$🐱[\![\texttt{cat}]\!](\alpha_{\Xi}(t)) = \texttt{forbidden}$$



- The herd7 simulation tool: `virginia.cs.ucl.ac.uk/herd/`

Execution environment

- In general, the semantics of a parallel program depends on hypotheses on the execution environment
- e.g.: coherence order:



x x x x

3 1 4 2

w[]x 1; w[]x 2; w[]x 3; w[]x 4;….

co

- the hypotheses on the execution environment (e.g. co ⊆ po) are part of the communication semantics

The WCM semantics

- Abstraction to a candidate execution:

$$\alpha_{\Xi}(t) \triangleq \langle \alpha_{e}(t),\ \alpha_{po}(t),\ \alpha_{rf}(t),\ \alpha_{iw}(t)\rangle$$

$$\alpha_{\Xi}(\mathsf{S}) \triangleq \{\langle t,\ \alpha_{\Xi}(t)\rangle \mid t \in \mathsf{S}\}$$

  - $\alpha_{e}(t)$ set of all events
  - $\alpha_{po}(t)$ execution order of events on the same process
  - $\alpha_{rf}(t)$ which read events read from which write events
  - $\alpha_{iw}(t)$ initial write events (initialization before starting the parallel execution)

The WCM semantics

- The cat communication semantics

$$\text{🐱}[\![\mathtt{cat}]\!](\varXi)$$

returns:

  - Relations $\varGamma$ on events representing hypotheses on the execution environment (e.g. co)
  - $\mathtt{allowed/forbidden}$ depending on whether the candidate execution $\langle\varXi, \varGamma\rangle$ is consistent or not

$$\alpha_{\text{🐱}}[\![\mathtt{cat}]\!](C) \triangleq \{t, \varGamma \mid \langle t,\ \varXi\rangle \in C \wedge \langle \mathtt{allowed},\ \varGamma\rangle \in \text{🐱}[\![\mathtt{cat}]\!](\varXi)\}$$

The WCM semantics

- The WCM semantics:

$$\mathcal{S}[\![\mathtt{P}]\!] \triangleq \alpha_{\text{🐱}}[\![\mathtt{cat}]\!] \circ \alpha_{\Xi}(\mathcal{S}^{\mathsf{a}}[\![\mathtt{P}]\!])$$

where:

- abstraction to a candidate execution:

$$\alpha_{\Xi}(\mathcal{S}) \triangleq \{\langle t,\ \alpha_{\Xi}(t)\rangle \mid t \in \mathcal{S}\}$$

- the cat communication semantics:

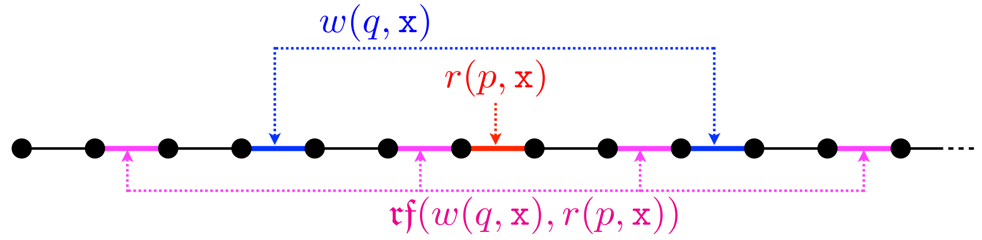
$$\alpha_{\text{🐱}}[\![\mathtt{cat}]\!](\mathcal{C}) \triangleq \{t, \Gamma \mid \langle t,\ \Xi\rangle \in \mathcal{C} \wedge \langle \mathtt{allowed},\ \Gamma\rangle \in \text{🐱}[\![\mathtt{cat}]\!](\Xi)\}$$

- The composition of Galois connections.

Definition of the anarchic semantics

Axiomatic parameterized definition of the anarchic semantics

- The semantics $\mathcal{S}^{\perp}[\![\mathrm{P}]\!]$ is a finite/infinite sequence of *interleaved* events of processes satisfying well-formedness conditions.
- Events:
  - local computations and tests on registers, fences, rmw
  - start writing a shared variable $w(q, \mathtt{x})$
  - start reading of shared variable $r(p, \mathtt{x})$
  - communication event $\mathfrak{rf}(w(q, \mathtt{x}), r(p, \mathtt{x}))$



Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics *S*:
  - uniqueness of events

$$\forall t \in S \,.\, \forall t_1, t_2 \in \mathfrak{E}^*, t_3 \in \mathfrak{E}^{*\infty} \,.\, \forall e, e' \in \mathfrak{E} \,.\, (t = t_1\, e\, t_2\, e'\, t_3) \Longrightarrow (e \neq e') \,. \qquad (\mathsf{Wf}_1(S))$$

  - traces start with an initialization of the shared variables $(\mathsf{Wf}_2(S))$

$$t \;=\; w(\mathsf{start}, \mathtt{x}, 0)\; w(\mathsf{start}, \mathtt{y}, 0)\; r(\mathtt{P0}, \mathtt{x}, 1)\; \mathfrak{rf}[w(\mathtt{P1}, \mathtt{x}, 1), r(\mathtt{P0}, \mathtt{x}, 1)])\; w(\mathtt{P0}, \mathtt{y}, 1)\; r(\mathtt{P1}, \mathtt{y}, 1)\; w(\mathtt{P1}, \mathtt{x}, 1)\; \mathfrak{rf}[w(\mathtt{P0}, \mathtt{y}, 1), r(\mathtt{P1}, \mathtt{y}, 1)] \,.$$

Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics S:
  - finite traces are maximal

$$\forall t \in \mathcal{S} \cap \mathfrak{E}^{+} \,.\, \nexists t' \in \mathfrak{E}^{+\infty} \,.\, t\, t' \in \mathcal{S} \,. \qquad (\mathsf{Wf}_3(\mathcal{S}))$$

Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics *S*:
  - read events must be satisfied by a unique communication event

$$\forall t \in S \,.\, \forall t_1 \in \mathfrak{E}^*, t_2 \in \mathfrak{E}^{*\infty} \,.\, (t = t_1\, r(p, \mathtt{x})\, t_2) \Longrightarrow \qquad (\mathsf{Wf}_4(S))$$
$$(\exists t_3 \in \mathfrak{E}^*, t_4 \in \mathfrak{E}^{*\infty} \,.\, t = t_3\, \mathfrak{rf}[w(q, \mathtt{x}), r(p, \mathtt{x})]\, t_4) \,.$$

$$\forall t \in S \,.\, \forall t_1, t_2 \in \mathfrak{E}^*, t_3 \in \mathfrak{E}^{*\infty} \,. \qquad (\mathsf{Wf}_5(S))$$
$$(t \neq t_1\, \mathfrak{rf}[w(q, \mathtt{x}), r(p, \mathtt{x})]\, t_2\, \mathfrak{rf}[w'(q', \mathtt{x}), r(p, \mathtt{x})]\, t_3) \,.$$

Axiomatic parameterized definition of the anarchic semantics

- Examples of language independent well-formedness conditions of a semantics *S*:
  - communications cannot be spontaneous (must be originated by a read *and* a write)

$$\forall t \in \mathsf{S} \;.\; \forall t_1 \in \mathfrak{E}^*, t_2 \in \mathfrak{E}^{*\infty} \;.\; (t = t_1 \, \mathfrak{rf}[w(q, \mathtt{x}), r(p, \mathtt{x})] \; t_2) \Longrightarrow \qquad (\mathsf{Wf}_6(\mathsf{S}))$$
$$(\exists t_3 \in \mathfrak{E}^*, t_4 \in \mathfrak{E}^{*\infty} \;.\; t = t_3 w(q, \mathtt{x}) t_4 \wedge \exists t_5 \in \mathfrak{E}^*, t_6 \in \mathfrak{E}^{*\infty} \;.\; t = t_5 r(p, \mathtt{x}) t_6) \;.$$

Axiomatic parameterized definition of the anarchic semantics

- The language :

  process

  - Programs : *initialisation* $[\![\mathrm{P}_1 \| \ldots \| \mathrm{P}_n]\!]$
  - Actions (labelled $\ell \in \mathbb{L}(p)$) :

$$
\begin{array}{lllll}
a & ::= & m & \text{imperative actions} & \text{marker} \\
 & \mid & \mathtt{r := } e & & \text{assignment} \\
 & \mid & \mathtt{r := x} & & \text{read of shared variable } \mathtt{x} \\
 & \mid & \mathtt{x := } e & & \text{write of shared variable } \mathtt{x} \\
 & \mid & b \mid \neg b & \text{conditional actions} & \text{test}
\end{array}
$$

  - Next action : $\mathsf{next}(p, \ell)$ $\mathsf{nextt}(p, \ell)$ $\mathsf{nextf}(p, \ell)$

    for tests

- Jade Alglave, Patrick Cousot: **Syntax and analytic semantics of LISA.** CoRR abs/1608.06583 (2016)

Axiomatic parameterized definition of the anarchic semantics

- Example of language-dependent well-formedness condition: computation (*markers*: skip, fence, begin/end of rmw)

Any process p *Any point* k *in trace* *Any label* ℓ *of* p

marker event by process p *in trace* τ

$$\begin{aligned} &\forall p \in \mathbb{P}\mathbb{i} \,.\, \forall k \in [1, 1+|\tau|[\,.\, \forall \ell \in \mathbb{L}(p) \,. \\ &\quad (\exists \theta \in \mathfrak{P}(p) \,.\, \overline{\tau}_k = \mathfrak{m}(\langle p,\ \ell,\ m,\ \theta\rangle)) \\ &\qquad \Longrightarrow (\ell \in \mathsf{N}^p(\tau, k) \wedge \mathsf{action}(p, \ell) = m) \,. \end{aligned} \qquad (\mathsf{Wf}_{21}(\tau))$$

(unique) event stamp θ

control of process p *is at label* ℓ

action of process p *is at label* ℓ *is the marker action* m

Axiomatic parameterized definition of the anarchic semantics

- Example of language-dependent well-formedness condition: computation (local variable assignment)

register assignment event by process p in trace τ

(unique) event stamp θ

$$\forall p \in \mathbb{Pi} \;.\; \forall k \in \left]1, 1+|\tau|\right[\;.\; \forall \ell \in \mathbb{L}(p) \;.\; \forall v \in \mathcal{D} \;. \qquad (\mathsf{Wf}_{22}(\tau))$$
$$(\exists \theta \in \mathfrak{P}(p) \;.\; \overline{\tau}_k = \mathfrak{a}(\langle p,\ \ell,\ \mathtt{r} \mathrel{:=} e,\ \theta\rangle, v))$$
$$\Longrightarrow (\ell \in \mathsf{N}^p(\tau, k) \land \mathsf{action}(p, \ell) = \mathtt{r} \mathrel{:=} e \land v = \mathsf{E}^p[\![e]\!](\tau, k-1)) \;.$$

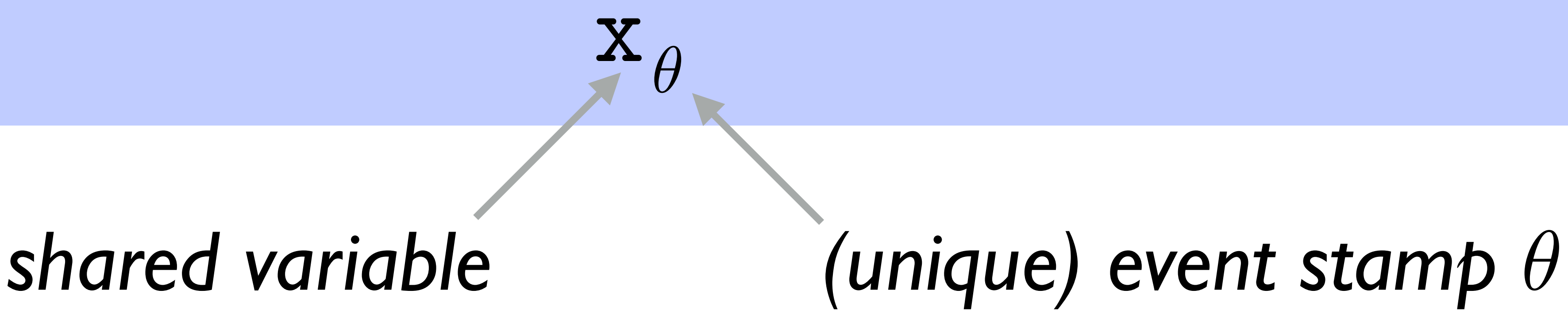
control of process p is at label ℓ

action of process p is at label ℓ is a register assignment

value v of e is evaluated by past-travel

Media variables

- With WCM there is no notion of *“the current value of shared variable* `x`*”*
- At a given time each process may read a *different value* of the shared variable `x` (maybe guessed or unknown since a read may read from a future write)
- We use *pythia variables* (to record the values communicated between a write and read, whether the two accesses are on the same process or not)



$\mathtt{x}_\theta$

shared variable

(unique) event stamp θ

Axiomatic parameterized definition of the anarchic semantics

- Example: communication
  - a read event is initiated by a read action:

read event by process p in trace τ

unique pythia variable

$$\begin{array}{l}\forall p \in \mathbb{Pi} \,.\, \forall k \in \,]1, 1 + |\tau|[\,.\, \forall \ell \in \mathbb{L}(p) \,. \\ \quad (\exists \theta \in \mathfrak{P}(p) \,.\, (\overline{\tau}_k = \mathfrak{r}(\langle p,\ \ell,\ \mathtt{r := x},\ \theta\rangle, \mathtt{x}_\theta))) \\ \qquad \Longrightarrow (\ell \in N^p(\tau, k) \wedge \mathsf{action}(p, \ell) = \mathtt{r := x}) \,.\end{array} \qquad (\mathsf{Wf}_{23}(\tau))$$

  - a read must read-from ($\mathfrak{rf}$) a write (weak fairness):

$$\begin{array}{l}\forall p \in \mathbb{Pi} \,.\, \forall i \in \,]1, 1 + |\tau|[\,.\, \forall r \in \mathfrak{Rf}(p) \,. \\ \quad (\overline{\tau}_i = r) \Longrightarrow (\exists j \in \,]1, 1 + |\tau|[\,.\, \exists w \in \mathfrak{Wi} \,.\, \overline{\tau}_j = \mathfrak{rf}[w, r]) \,.\end{array} \qquad (\mathsf{Wf}_{26}(\tau))$$

communication (read-from) event

Axiomatic parameterized definition of the anarchic semantics

- Predictive evaluation of pythia variables:

$$\begin{aligned} V^{p}_{(32)}[\![\mathtt{x}_\theta]\!](\tau, k) \ \triangleq\ v\ \textbf{where}\ & \exists! i \in [1, 1 + |\tau|[\ .\ (\overline{\tau}_i = \mathfrak{r}(\langle p,\ \ell,\ \mathtt{r}\ \mathtt{:=}\ \mathtt{x},\ \theta\rangle, \mathtt{x}_\theta)) \wedge \\ & \exists! j \in [1, 1 + |\tau|[\ .\ (\overline{\tau}_j = \mathfrak{rf}[\mathfrak{w}(\langle p',\ \ell',\ \mathtt{x}\ \mathtt{:=}\ e',\ \theta'\rangle, v), \overline{\tau}_i]) \end{aligned}$$

- Local past-travel evaluation of an expression:

$$\begin{aligned} E^{p}_{(30)}[\![\mathtt{r}]\!](\tau, k) \ &\triangleq\ v \quad \textbf{if}\ k > 1 \wedge \big((\overline{\tau}_k = \mathfrak{a}(\langle p,\ \ell,\ \mathtt{r}\ \mathtt{:=}\ e,\ \theta\rangle, v)) \vee \\ &\qquad\qquad (\overline{\tau}_k = \mathfrak{r}(\langle p,\ \ell,\ \mathtt{r}\ \mathtt{:=}\ \mathtt{x},\ \theta\rangle, \mathtt{x}_\theta) \wedge V^{p}[\![\mathtt{x}_\theta]\!](\tau, k) = v)\big) \\ E^{p}_{(30)}[\![\mathtt{r}]\!](\tau, 1) \ &\triangleq\ I[\![0]\!] \qquad\qquad i.e.\ \overline{\tau}_1 = \epsilon_{\mathsf{start}}\ \text{by}\ \mathsf{Wf}_{15}(\tau) \\ E^{p}_{(30)}[\![\mathtt{r}]\!](\tau, k) \ &\triangleq\ E^{p}_{(30)}[\![\mathtt{r}]\!](\tau, k - 1) \qquad\qquad \text{otherwise.} \end{aligned}$$

Abstractions of the anarchic semantics

Abstractions

- Anarchic semantics:

$$S^{\perp}[\![\mathrm{P}]\!] \triangleq \boldsymbol{\lambda}\langle\mathcal{B}, \mathsf{sat}, \mathcal{D}, \mathit{I}, \mathfrak{S}, V, E, N\rangle \bullet \{\tau \in \mathfrak{T}[\![\mathrm{P}]\!]|_{\cong} \mid \mathsf{Wf}_1(\tau) \wedge \ldots \wedge \mathsf{Wf}_{29}(\tau)\}$$

parameters of the semantics

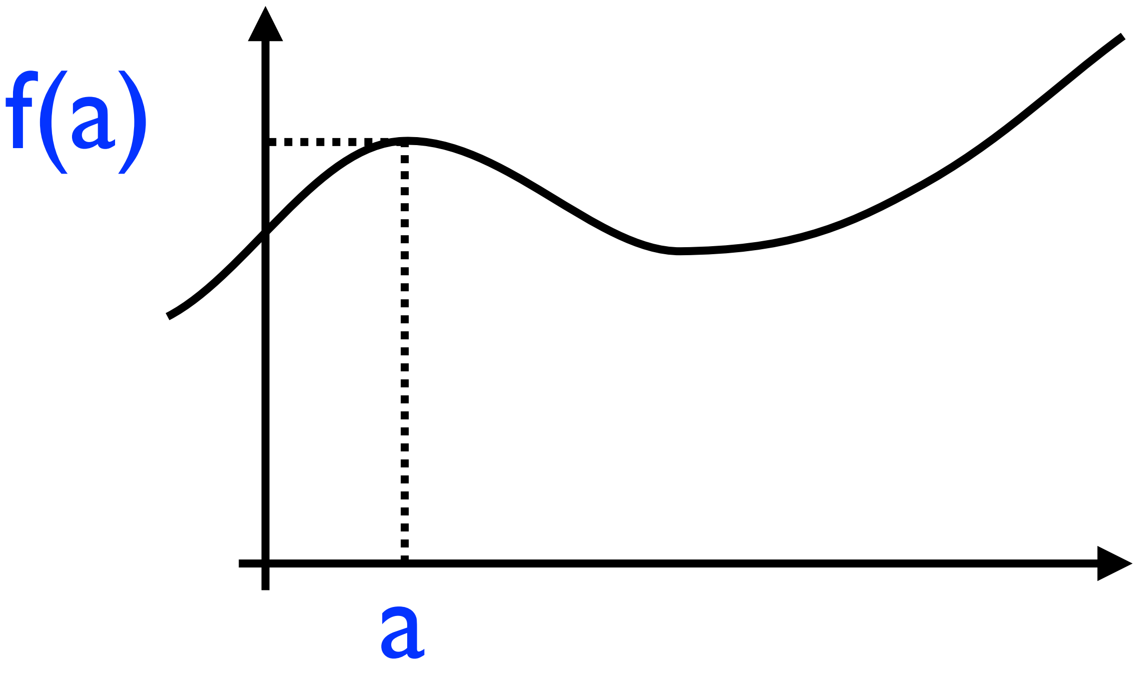
trace well-formedness conditions

- Examples of abstractions:
  - Choose data (e.g. ground values, uninterpreted symbolic expressions, interpreted symbolic expressions i.e. "symbolic guess")
  - Bind parameters (e.g. how expressions are evaluated)
  - …

Binding a parameter of the semantics

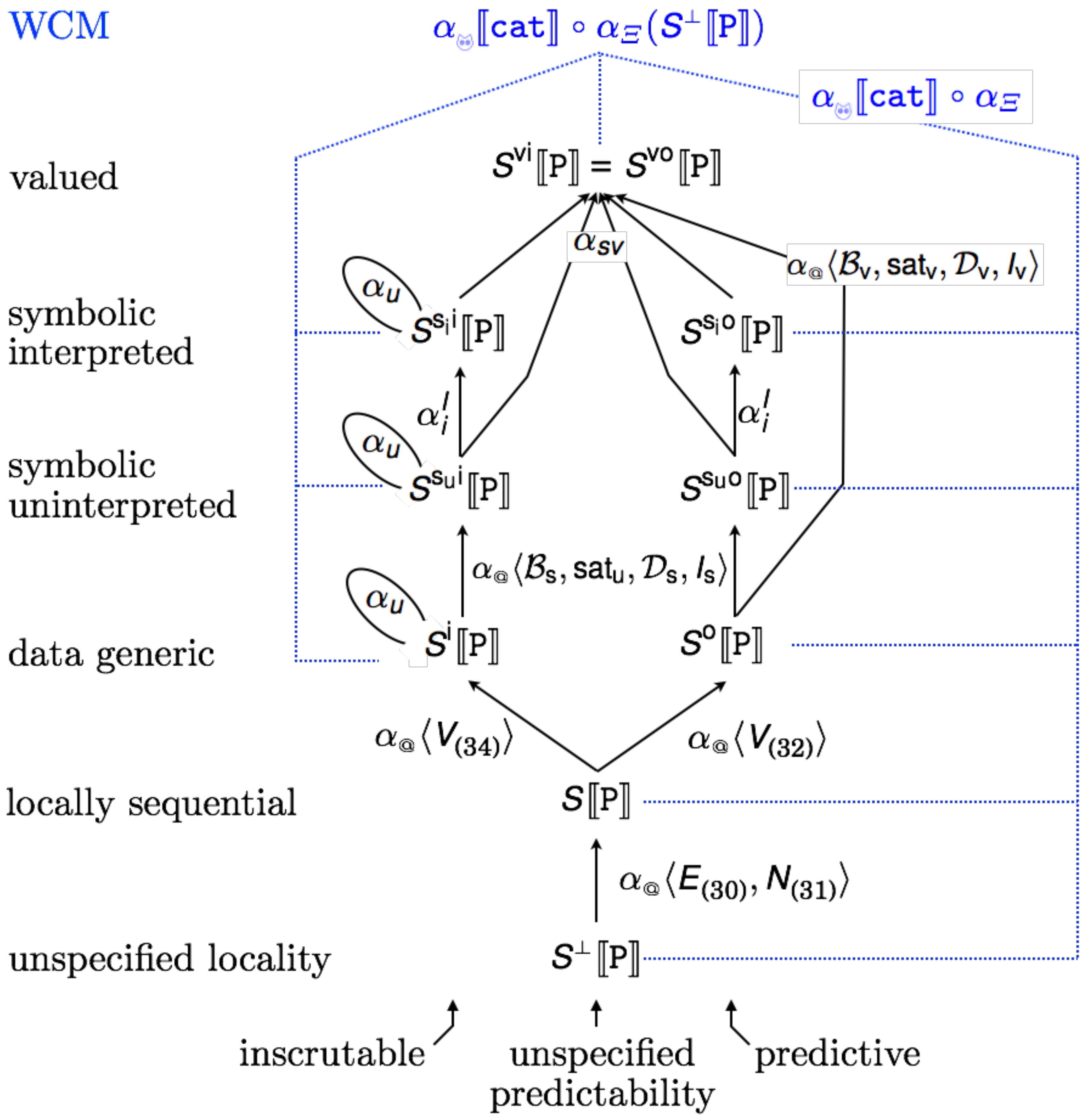
- The abstraction

$$\alpha_a(f) \stackrel{\text{def}}{=} f(a)$$



$$\langle \wp(A,B,\ldots)\longrightarrow\wp(R),\dot{\subseteq}\rangle \underset{\alpha_a}{\overset{\gamma_a}{\leftrightarrows}} \langle \wp(B,\ldots)\longrightarrow\wp(R),\dot{\subseteq}\rangle$$

The hierarchy of interleaved semantics



WCM

$\alpha_{\text{🐱}}[\![\texttt{cat}]\!] \circ \alpha_{\Xi}(S^{\perp}[\![\mathrm{P}]\!])$

$\alpha_{\text{🐱}}[\![\texttt{cat}]\!] \circ \alpha_{\Xi}$

valued

$S^{\mathsf{vi}}[\![\mathrm{P}]\!] = S^{\mathsf{vo}}[\![\mathrm{P}]\!]$

α_{sv}

$\alpha_{@}\langle \mathcal{B}_{\mathsf{v}}, \mathsf{sat}_{\mathsf{v}}, \mathcal{D}_{\mathsf{v}}, I_{\mathsf{v}} \rangle$

symbolic interpreted

α_u

$S^{\mathsf{s_i i}}[\![\mathrm{P}]\!]$

$S^{\mathsf{s_i o}}[\![\mathrm{P}]\!]$

α_i^I

α_i^I

symbolic uninterpreted

α_u

$S^{\mathsf{s_u i}}[\![\mathrm{P}]\!]$

$S^{\mathsf{s_u o}}[\![\mathrm{P}]\!]$

$\alpha_{@}\langle \mathcal{B}_{\mathsf{s}}, \mathsf{sat}_{\mathsf{u}}, \mathcal{D}_{\mathsf{s}}, I_{\mathsf{s}} \rangle$

data generic

α_u

$S^{\mathsf{i}}[\![\mathrm{P}]\!]$

$S^{\mathsf{o}}[\![\mathrm{P}]\!]$

$\alpha_{@}\langle V_{(34)} \rangle$

$\alpha_{@}\langle V_{(32)} \rangle$

locally sequential

$S[\![\mathrm{P}]\!]$

$\alpha_{@}\langle E_{(30)}, N_{(31)} \rangle$

unspecified locality

$S^{\perp}[\![\mathrm{P}]\!]$

inscrutable

unspecified predictability

predictive

True parallelism with local communications

- Extract from interleaved executions:
  - The subtrace of each process keeping communications in the process that read

⟹ no more global time between processes

⟹ local time between local actions and communications (a read can still tell when it is satisfied by which write)

True parallelism with local communications

- Interleaved execution:

global time →

$t\;=\;w(\mathsf{start},\mathrm{x},0)\;w(\mathsf{start},\mathrm{y},0)\;r(\mathtt{P0},\mathtt{x},1)\;\mathfrak{rf}[w(\mathtt{P1},\mathtt{x},1),r(\mathtt{P0},\mathtt{x},1)])\;w(\mathtt{P0},\mathtt{y},1)\;r(\mathtt{P1},\mathtt{y},1)\;w(\mathtt{P1},\mathtt{x},1)\;\mathfrak{rf}[w(\mathtt{P0},\mathtt{y},1),r(\mathtt{P1},\mathtt{y},1)]$

- Parallel executions with interleaved communications:

t = Initialization: $w(\mathsf{start},\mathrm{x},0)\;w(\mathsf{start},\mathrm{y},0)$

local time →

P0: $r(\mathtt{P0},\mathtt{x},1)\;\mathfrak{rf}[w(\mathtt{P1},\mathtt{x},1),r(\mathtt{P0},\mathtt{x},1)])\;w(\mathtt{P0},\mathtt{y},1)$

local time →

P1: $r(\mathtt{P1},\mathtt{y},1)\;w(\mathtt{P1},\mathtt{x},1)\;\mathfrak{rf}[w(\mathtt{P0},\mathtt{y},1),r(\mathtt{P1},\mathtt{y},1)]$

True parallelism of computations and communications

- Extract from interleaved executions:
  - The subtrace of each process (sequential execution of actions)
  - The rf communication relation (interactions between processes)

⟹ no more global time between processes

⟹ no more global/local time for communications

True parallelism with separate communications

- Parallel executions with interleaved communications:

Initialization: $w(\mathsf{start}, \mathrm{x}, 0)\ w(\mathsf{start}, \mathrm{y}, 0)$

P0: $r(\mathtt{P0}, \mathtt{x}, 1)\ w(\mathtt{P0}, \mathtt{y}, 1)$ *local time* →

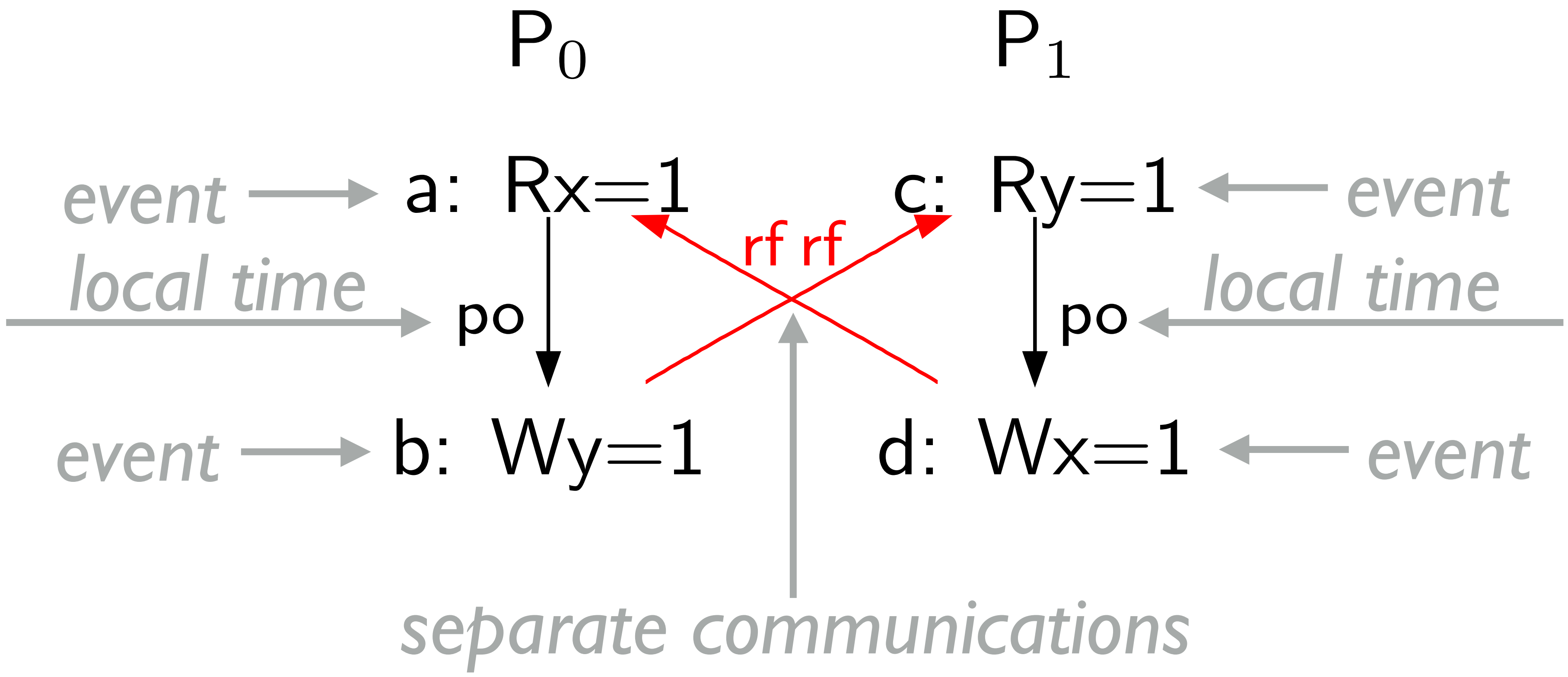
P1: $r(\mathtt{P1}, \mathtt{y}, 1)\ w(\mathtt{P1}, \mathtt{x}, 1)$ *local time* →

Communications:

$$\{\ \mathfrak{rf}[w(\mathtt{P1}, \mathtt{x}, 1), r(\mathtt{P0}, \mathtt{x}, 1)]\mathrm{)},\ \mathfrak{rf}[w(\mathtt{P0}, \mathtt{y}, 1), r(\mathtt{P1}, \mathtt{y}, 1)]\ \}$$

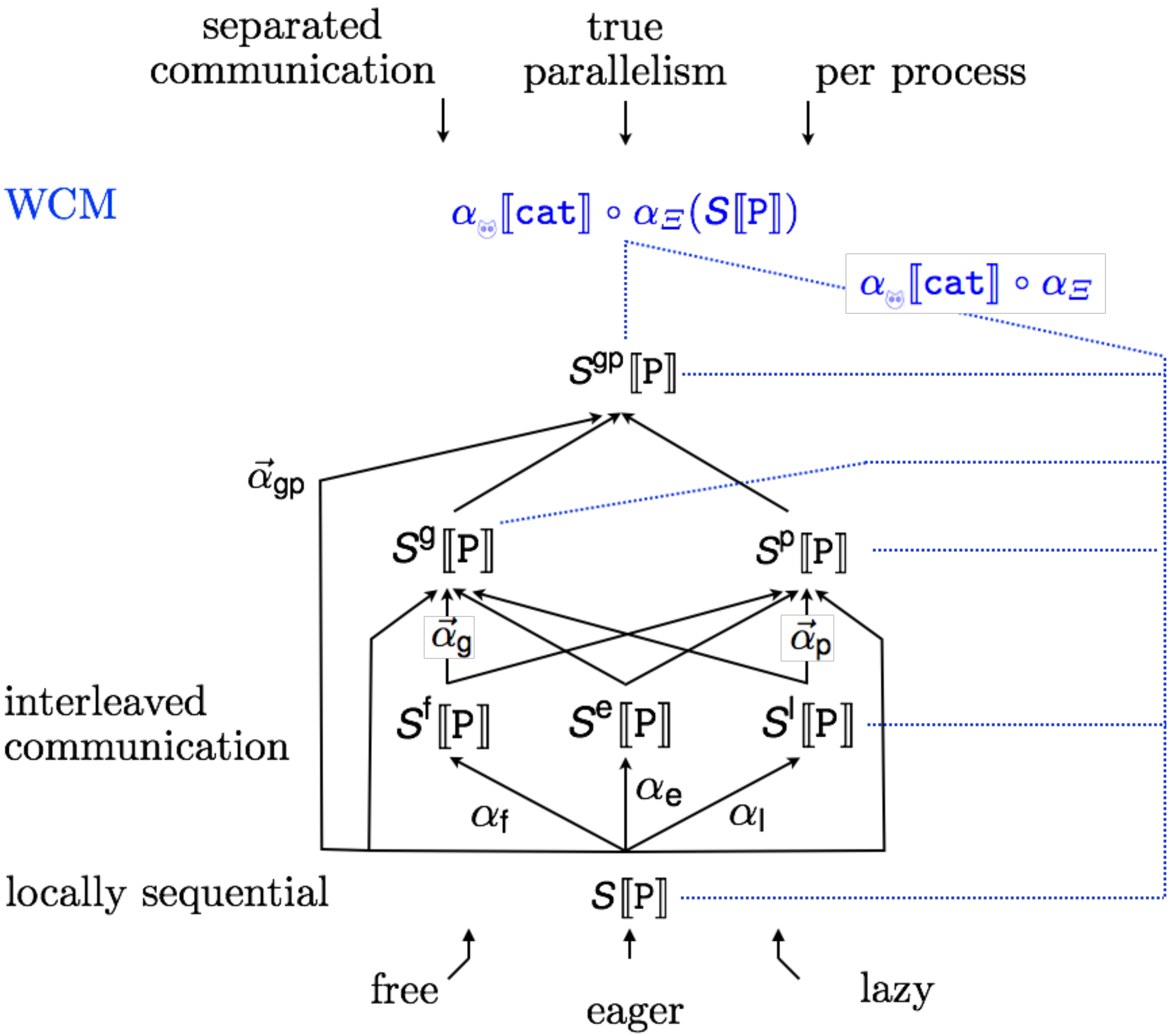
True parallelism with separate communications

- This is the semantics used by the herd7 tool:



+ interpreted symbolic expressions i.e. "symbolic guess"

The true parallelism hierarchy


separated communication
true parallelism
per process
WCM
$\alpha[\![\mathtt{cat}]\!] \circ \alpha_{\Xi}(S[\![\mathrm{P}]\!])$
$\alpha[\![\mathtt{cat}]\!] \circ \alpha_{\Xi}$
$S^{\mathsf{gp}}[\![\mathrm{P}]\!]$
$\vec{\alpha}_{\mathsf{gp}}$
$S^{\mathsf{g}}[\![\mathrm{P}]\!]$
$S^{\mathsf{p}}[\![\mathrm{P}]\!]$
$\vec{\alpha}_{\mathsf{g}}$
$\vec{\alpha}_{\mathsf{p}}$
interleaved communication
$S^{\mathsf{f}}[\![\mathrm{P}]\!]$
$S^{\mathsf{e}}[\![\mathrm{P}]\!]$
$S^{\mathsf{l}}[\![\mathrm{P}]\!]$
α_{f}
α_{e}
α_{l}
locally sequential
$S[\![\mathrm{P}]\!]$
free
eager
lazy

States

- At each point in a trace, the state abstracts the past computation history up to that point
- Example: classical environment (assigning values to register at each point k of the trace):

$$\rho^p(\tau, k) \triangleq \boldsymbol{\lambda}\, \mathtt{r} \in \mathbb{R}(p) \bullet E^p[\![\mathtt{r}]\!](\tau, k)$$

$$\nu^p(\tau, k) \triangleq \boldsymbol{\lambda}\, \mathtt{x}_\theta \bullet V^p_{(32)}[\![\mathtt{x}_\theta]\!](\tau, k)$$

Prefixes, transitions, . . .

- Abstract traces by their prefixes:

$$\overleftarrow{\alpha}(\mathsf{S}) \triangleq \bigcup\{\overleftarrow{\alpha}(\tau) \mid \tau \in \mathsf{S}\}$$

$$\overleftarrow{\alpha}(\tau) \triangleq \{\tau\langle\!\!| j \langle\!\!| \mid j \in [1, 1 + |\tau|[\}$$

$$\tau\langle\!\!| j \langle\!\!| \triangleq \langle \xrightarrow{\overline{\tau}_i} \underline{\tau}_i \mid i \in [1, 1 + j[\rangle$$

- and transitions: extract transitions from traces

  ⟹ communication fairness is lost, inexact abstraction,
  ⟹ add fairness condition
  ⟹ impossible to implement with a scheduler (≠ process fairness)

Effect of the cat specification on the hierarchy

Exactness and cat preservation

$$\alpha_{🐱}[\![\texttt{cat}]\!] \circ \alpha_{\Xi}(\alpha(S^{\perp}[\![\mathrm{P}]\!]))$$
$$\alpha(\alpha_{🐱}[\![\texttt{cat}]\!] \circ \alpha_{\Xi}(S^{\perp}[\![\mathrm{P}]\!])) =$$

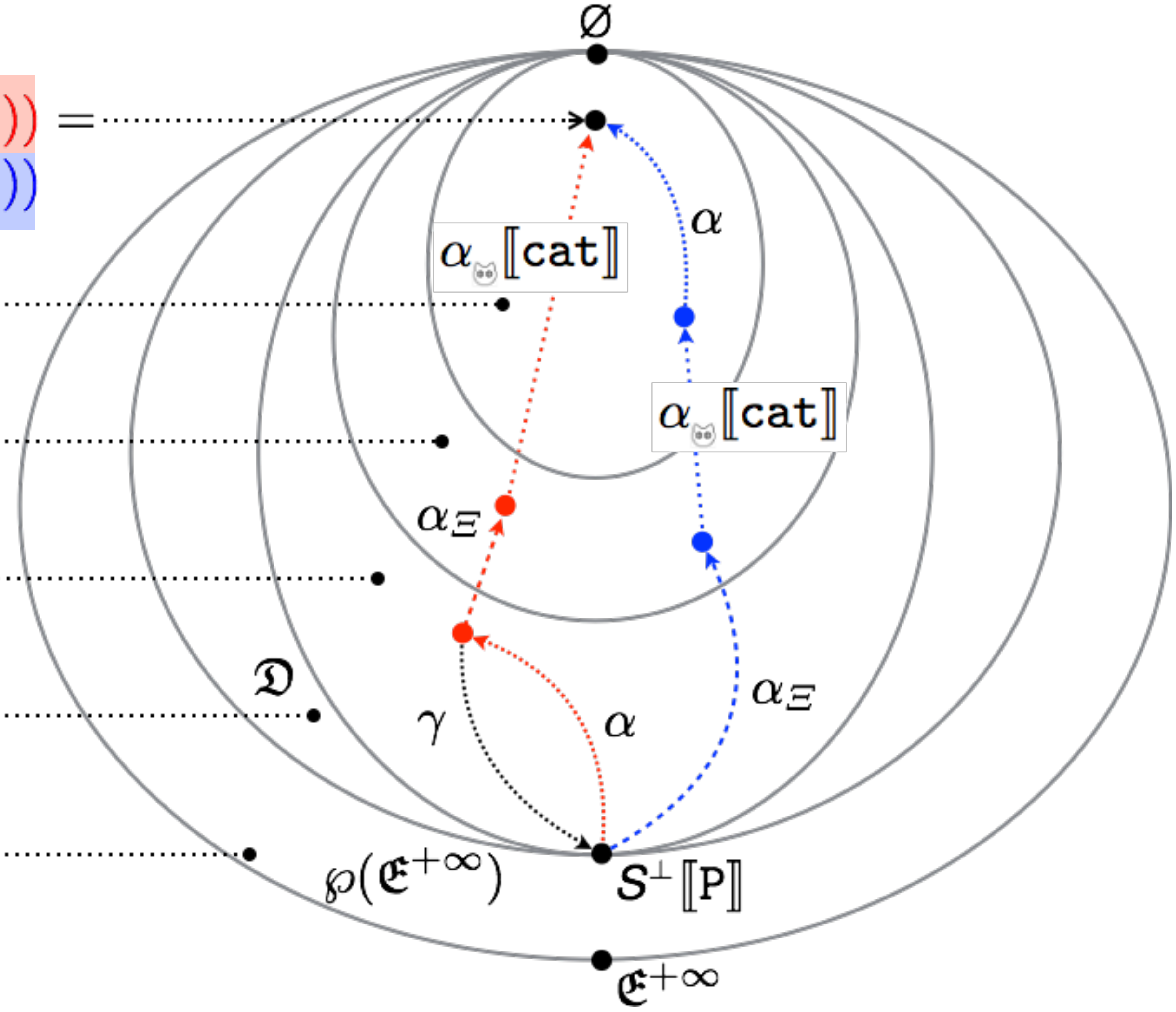
WCM semantics

Concurrent execution semantics

Exact and cat-preserving semantics

Anarchic semantics

Semantics


$\varnothing$
$\alpha_{🐱}[\![\texttt{cat}]\!]$
α
$\alpha_{🐱}[\![\texttt{cat}]\!]$
α_{Ξ}
$\mathfrak{D}$
γ
α
α_{Ξ}
$\wp(\mathfrak{E}^{+\infty})$
$S^{\perp}[\![\mathrm{P}]\!]$
$\mathfrak{E}^{+\infty}$

The cat abstraction

- The same cat specification $\alpha_{\text{🐱}}[\![\texttt{cat}]\!]$ applies equally to any concurrent execution abstraction α_{Ξ} of any interleaved/truly parallel semantics in the hierarchy
- The appropriate level of abstraction to specify WCM:
  - No states, only `marker (e.g. fence)`, `r`, `w`, `rf(w,r)` events
  - No values in events
  - No global time (only `po` order of events per process)
  - Time of communications forgotten (only `rf` of who communicates with whom)
  - Hypotheses on the execution environment independent of computed and communicated values

Conclusion

Conclusion

- Analytic semantics: a new style of semantics
- The hierarchy of anarchic semantics describes the same computations and potential communications in very different styles
- The cat semantics restricts communications to a machine/network architecture in the same way for all semantics in the hierarchy
- This idea of parameterized semantics at various levels of abstraction is useful for
  - Verification
  - Static analysis

The End