An overview of game semantics for polymorphism

Guilhem Jaber
ENS Lyon

Nikos Tzevelekos
Queen Mary Uni. of London

GALOP, Thessaloniki, April 2018

what this talk is about

Polymorphism: a powerful abstraction mechanism

$$\Lambda X.\lambda x^X.x : \forall X.\, X \to X$$

it allows programs that “do the same thing” at every type

Modelling polymorphism: meaning of $\forall / \Lambda$?

We look at game semantics answers and different models proposed (including ours)

Setting: System F

$$\frac{}{\Gamma, x{:}\vartheta \vdash x{:}\vartheta} \qquad \frac{\Gamma,\ x:\vartheta \vdash M : \vartheta'}{\Gamma \vdash \lambda x^A.M : \vartheta \to \vartheta'} \qquad \frac{\Gamma \vdash M : \vartheta \to \vartheta' \quad \Gamma \vdash N : \vartheta}{\Gamma \vdash M\,N : \vartheta'}$$

$$\frac{\Gamma \vdash M : \vartheta}{\Gamma \vdash \Lambda X.M : \forall X.\vartheta} \qquad \frac{\Gamma \vdash M : \forall X.\vartheta}{\Gamma \vdash M\vartheta' : \vartheta\{\vartheta'/X\}}$$

$$(\lambda x^A.M)M' \cong M\{M'/x\} \qquad (\Lambda X.M)\vartheta \cong M\{\vartheta/X\}$$

$$\lambda x^A.Mx \cong M \qquad \Lambda X.MX \cong M$$

$$\frac{M \cong M'}{M\,N \cong M'\,N} \qquad \frac{M \cong M'}{M\,\vartheta \cong M'\,\vartheta} \qquad \ldots$$

Examples from System F

$$Nil = \forall X.X$$

$$Unit = \forall X.X\to X \qquad \Lambda X.\lambda x^X.x$$

$$Bool = \forall X.X\to X\to X \qquad \Lambda X.\lambda x^X.\lambda y^X.x \qquad \Lambda X.\lambda x^X.\lambda y^X.y$$

$$Nat = \forall X.(X\to X)\to X\to X \qquad \Lambda X.\lambda f^{X\to X}.\lambda x^X.f\ldots(f x)$$

$$(\Lambda X.\lambda x^X.\,x)\vartheta M \;\to\; (\lambda x^\vartheta.\,x)M \;\to\; M$$

$$\lambda f^{\forall X.X\to X}.\,f : \; Unit\to Unit \qquad \lambda f^{\forall X.X\to X}.\,f\vartheta : \; Unit\to\vartheta\to\vartheta$$

$$(\lambda f^{\forall X.X\to X}.\,f\vartheta)(\Lambda X.\lambda x^X.\,x) \to (\Lambda X.\lambda x^X.\,x)\vartheta \;\to \lambda x^\vartheta.\,x$$

$$(\lambda b^{\,Bool}.\; b\; false\;\; true)\; true \to \; true\; false\; true \to \; false$$

Semantics for System F

How to assign denotations to programs

$$[\![\_]\!] : \text{Syntax} \longrightarrow \mathcal{M}$$

such that (ideally):

$$M \cong M' \Rightarrow [\![M]\!] = [\![M']\!]$$

$[\![M]\!]$ includes 'the essence' of M

$\mathcal{M}$ only contains behaviours from System F

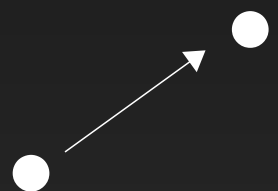
Game semantics

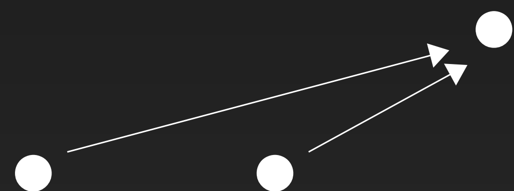
- Computation seen as a 2-player game between:
  - *Opponent* (the environment), aka O
  - *Proponent* (the program), aka P
- Programs = *strategies* for P
- Composition = interleaving & hiding moves
- *Categories* of games

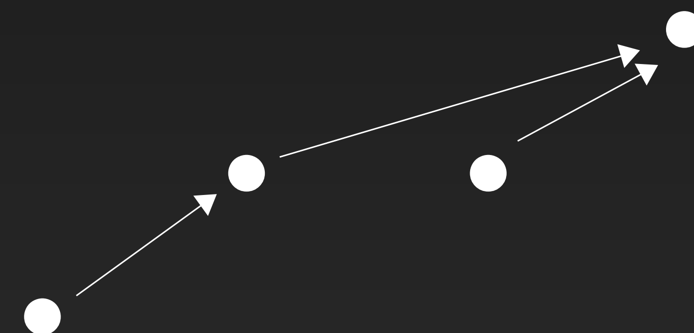
The arena-play-strategy infrastructure

Games have *moves*, taken from *arenas:*

$X \to X$ $\quad$ $X \to X \to X$ $\quad$ $(X \to X) \to X \to X$

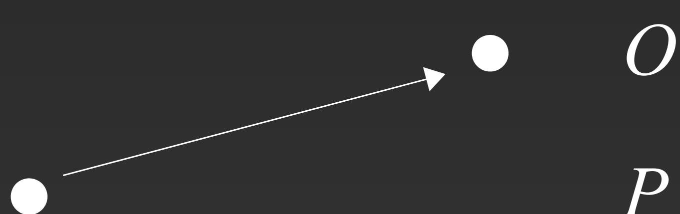


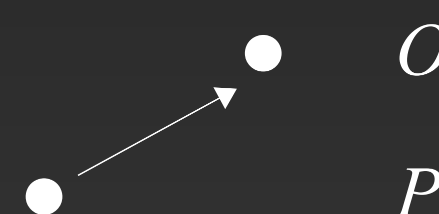




Sequences of moves form *plays*:

$X \to X \to X$ $\quad$ $X \to X \to X$

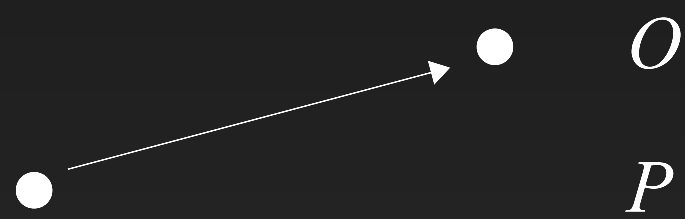




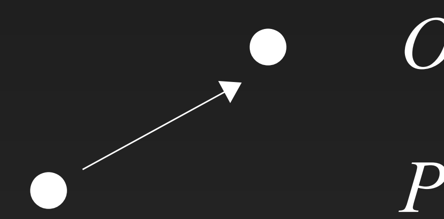
The arena-play-strategy infrastructure

Sets of plays form *strategies:*

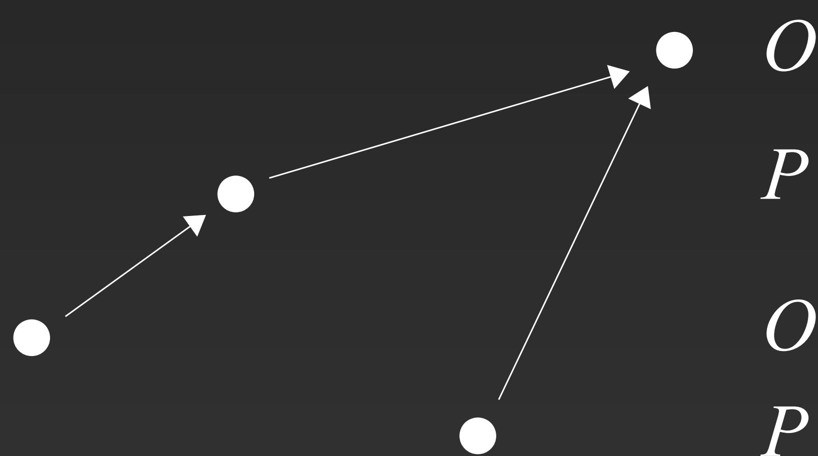
$$X \to X \to X$$



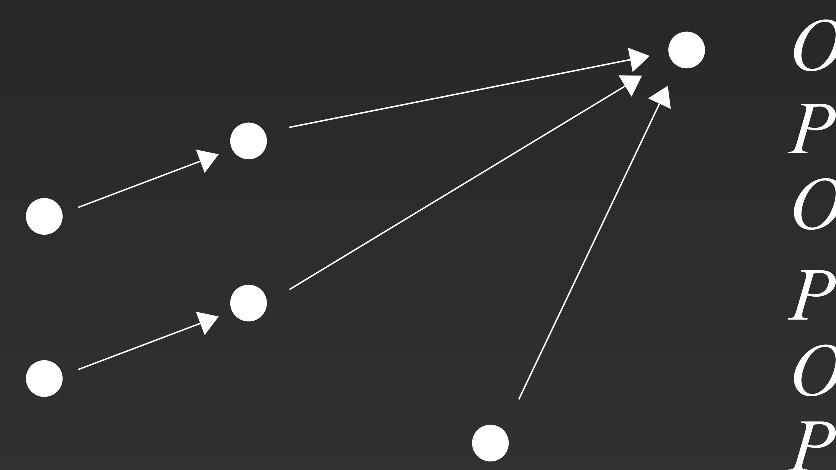
$$X \to X \to X$$



$$(X \to X) \to X \to X$$



$$(X \to X) \to X \to X$$



Game models for System F

An (incomplete) timeline:

- Hughes ‘97, ’00 : *Hypergames*
- Murawski & Ong ‘01 : *Evolving games*
- Abramsky & Jagadeesan ‘03 : *Variable games*
- Laird ‘10,’13 : *Context games*
- J. & Tz. ‘18 : *Nominal traces*

[Hughes LICS'97, PhD'10]

Hypergames

Two notions of move:

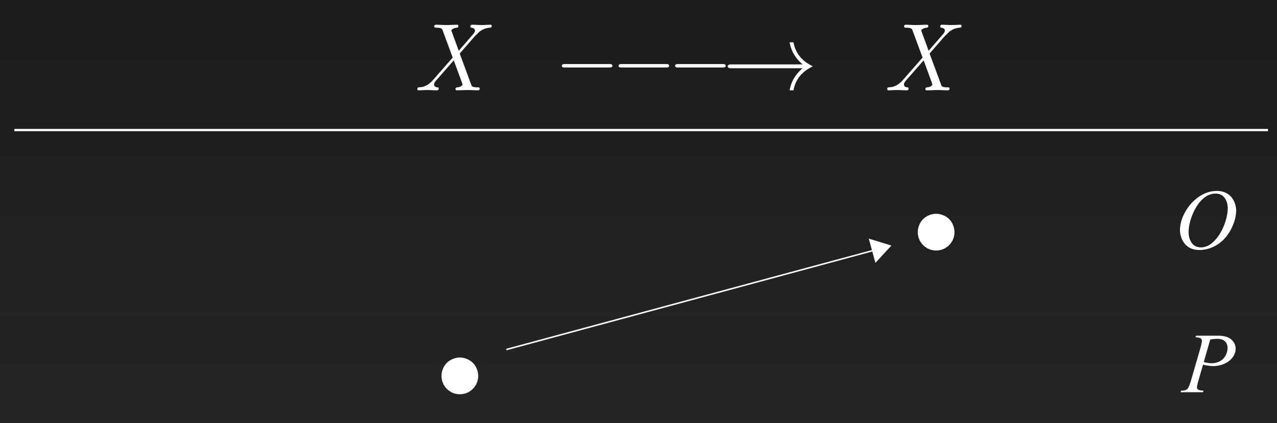
- first-order moves (e.g. for playing X)
- second-order moves (for playing new arenas)

Basic ingredients

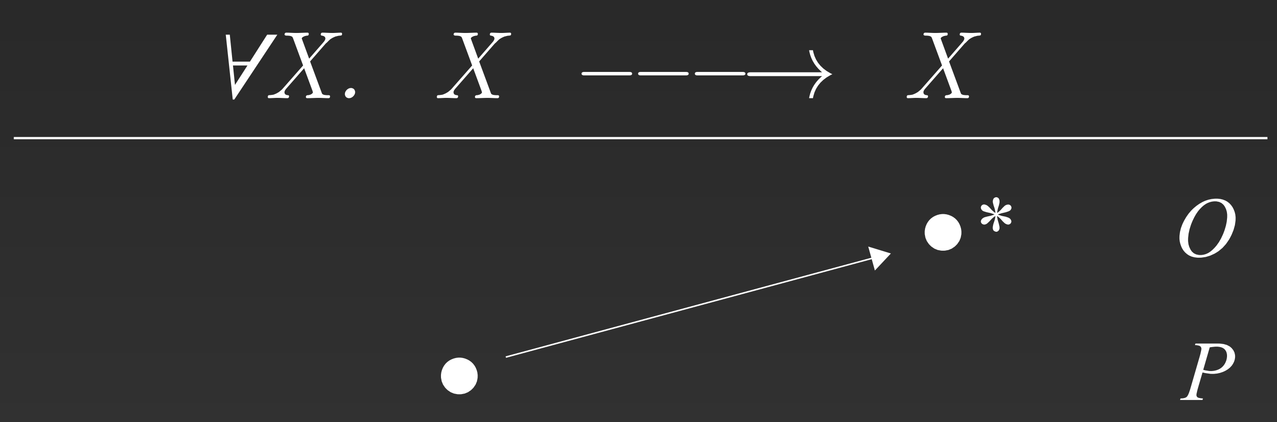
- arenas also contain *holes* – places for 2nd moves
- *copycat rule* – an X should be followed by an X
- *uniformity* – the arenas played by *O* are hidden
- there are only questions and no answers

Identities in hypergames

$$\lambda x^X.\, x : X \to X$$



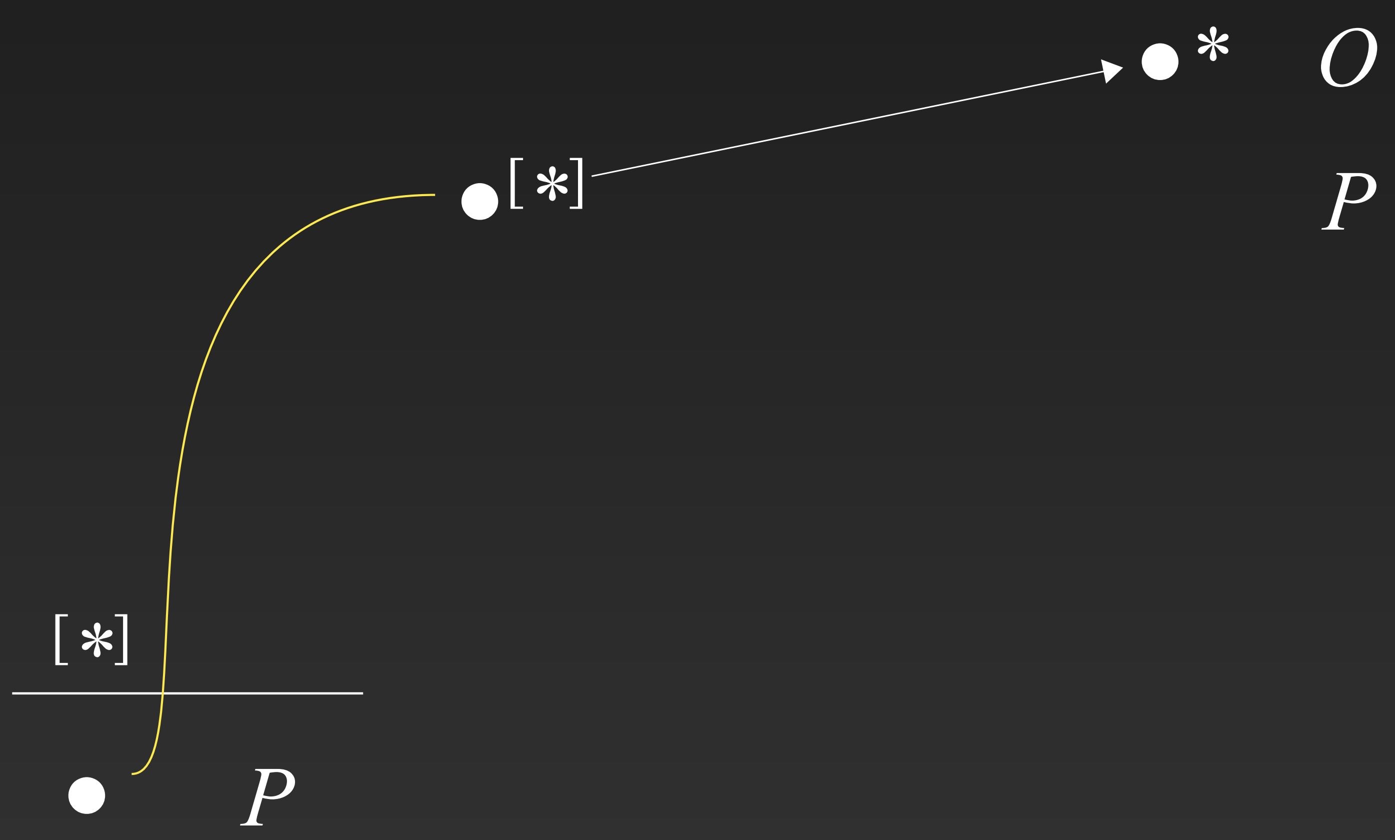
$$\Lambda X.\lambda x^X.\, x : \forall X.X \to X$$



Identities in hypergames

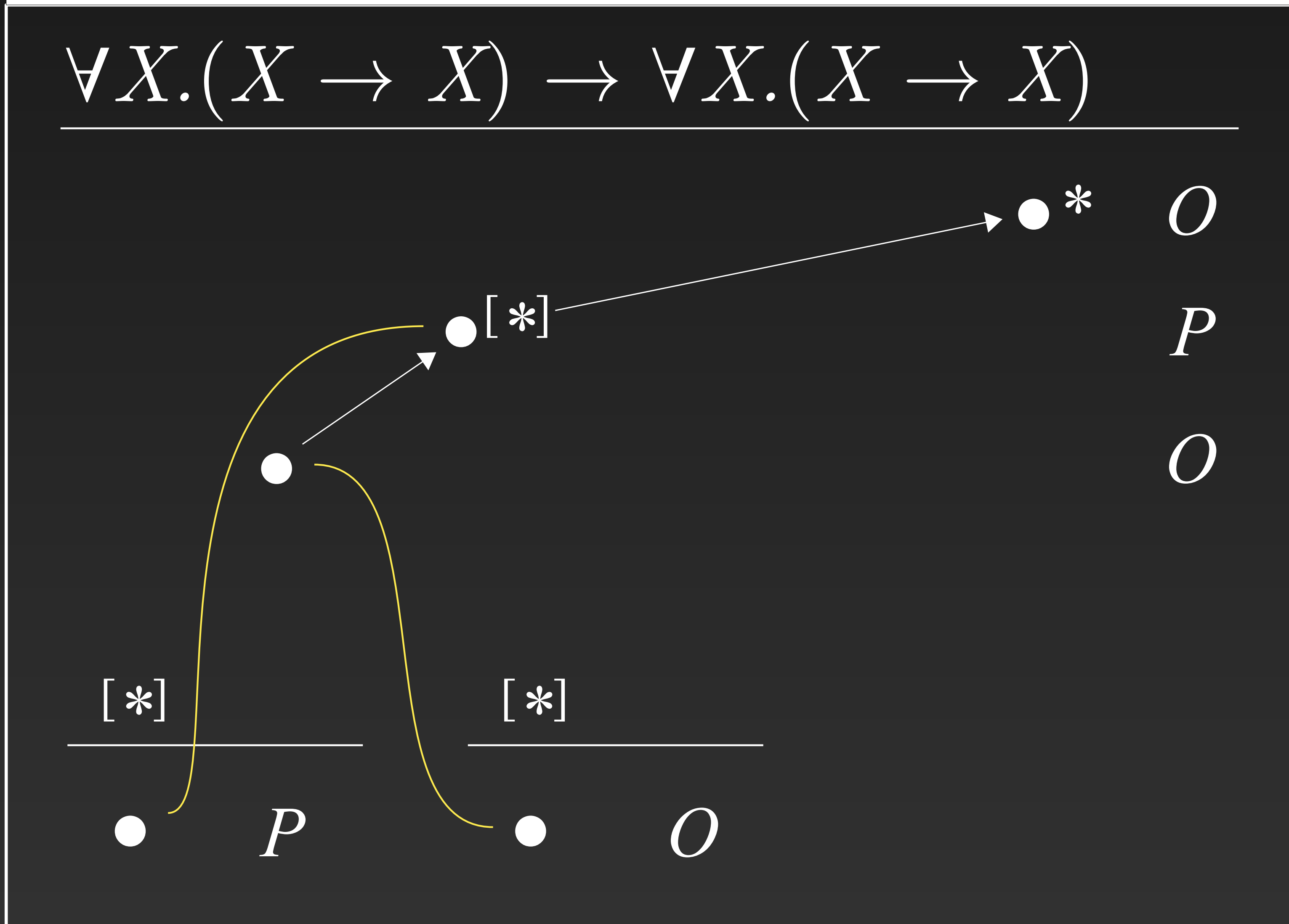
$$\lambda f^{\forall X.X\to X}.\, f : \forall X.(X\to X)\to\forall X.(X\to X)$$

$$\forall X.(X \to X) \to \forall X.(X \to X)$$



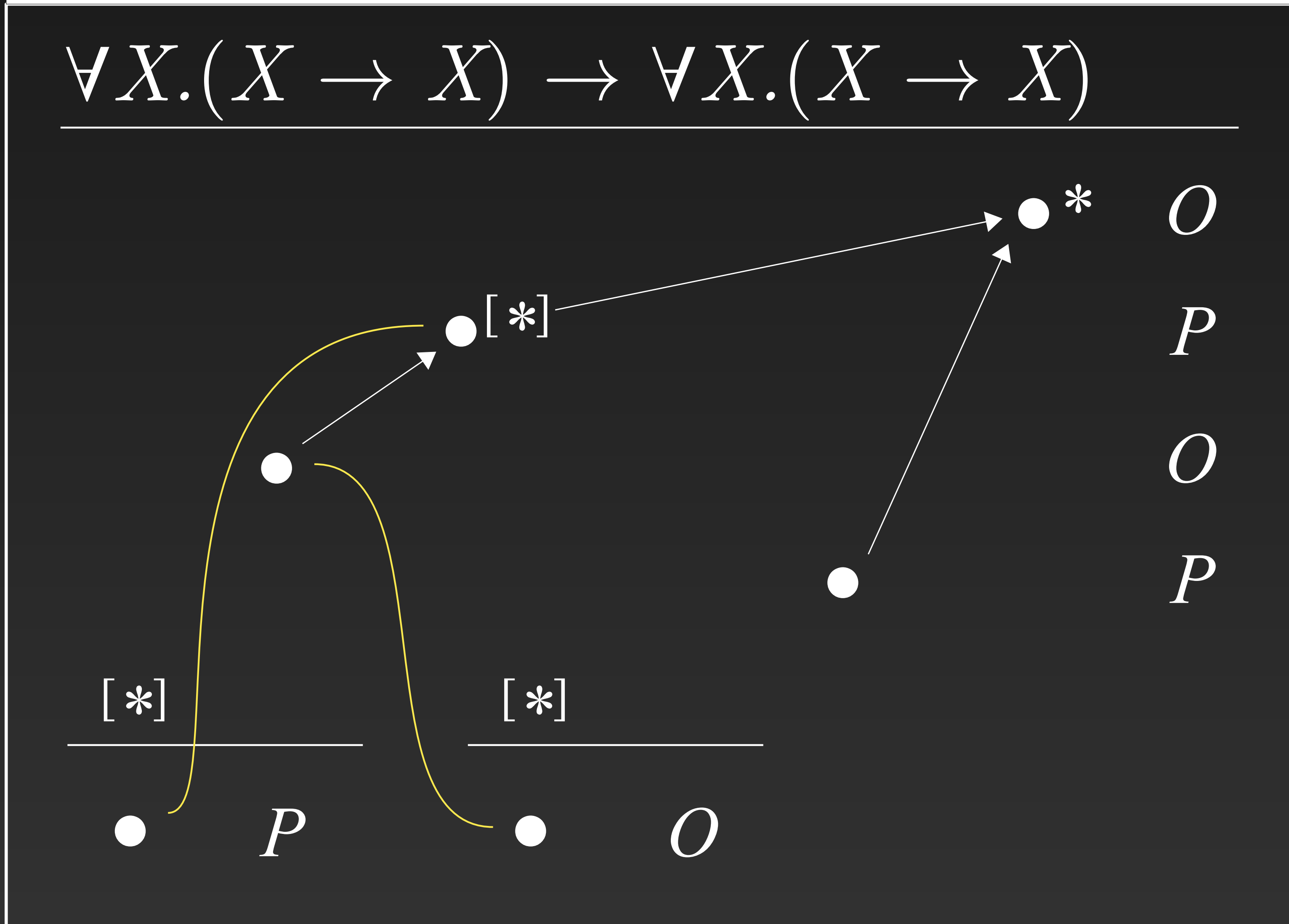
Identities in hypergames

$$\lambda f^{\forall X.X\to X}.\,f : \forall X.(X\to X)\to\forall X.(X\to X)$$



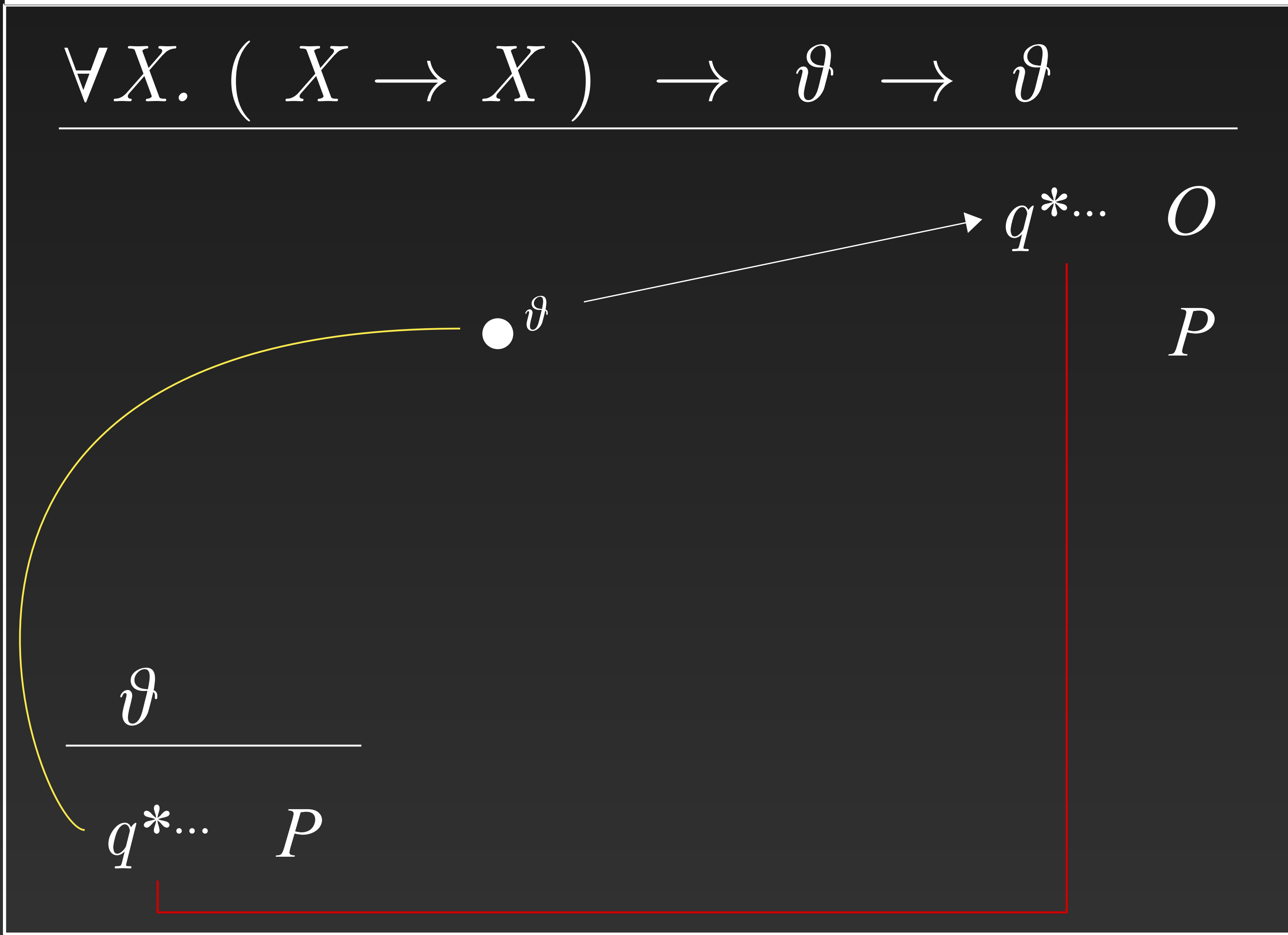
Identities in hypergames

$$\lambda f^{\forall X.X\to X}.\, f : \forall X.(X\to X)\to\forall X.(X\to X)$$



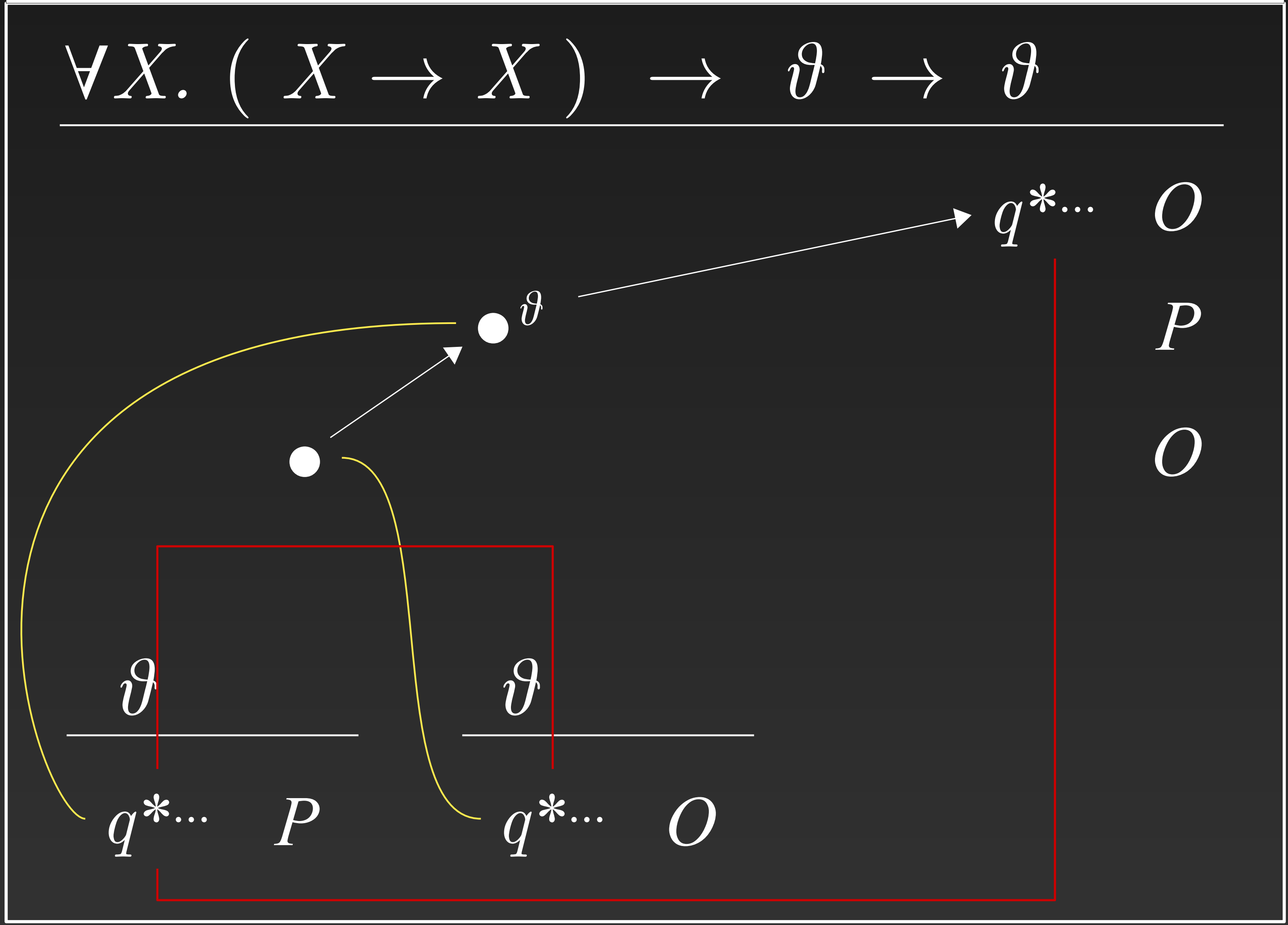
Instantiation in hypergames

$$\lambda f^{\forall X.X\to X}.\, f\,\vartheta : \forall X.(X\to X) \to \vartheta \to \vartheta$$



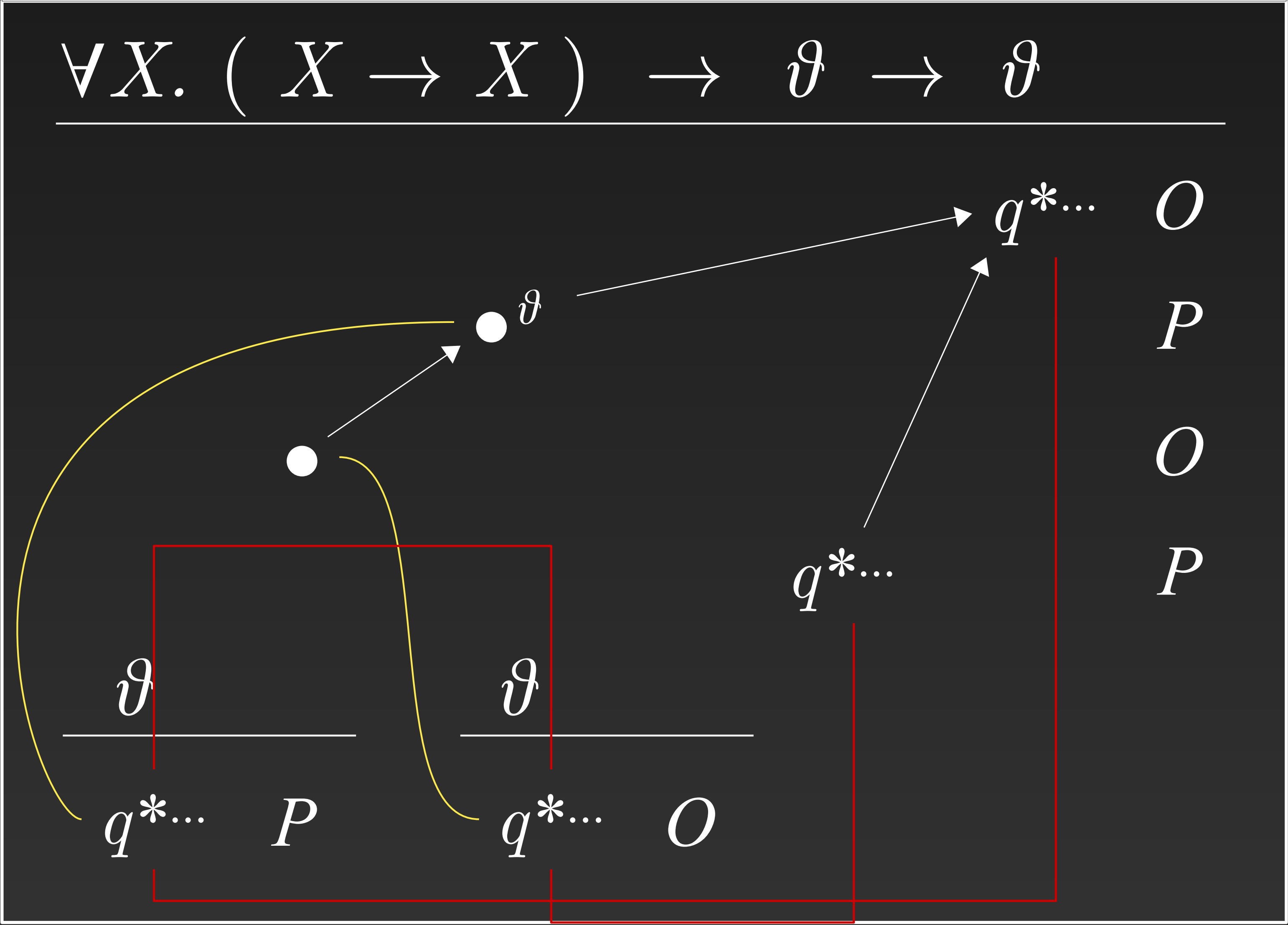
Instantiation in hypergames

$$\lambda f^{\forall X.X\to X}.\, f\,\vartheta : \forall X.(X\to X) \to \vartheta \to \vartheta$$



Instantiation in hypergames

$$\lambda f^{\forall X.X\to X}.\,f\,\vartheta : \forall X.(X\to X) \to \vartheta \to \vartheta$$



Legacy of hypergames

Arguably the prototypical game model of System F:

- solve the problem: fully complete model (i.e. sound and without extra behaviours)
- introduce notions carried over to all models (e.g. CC rule, uniformity)

Complicated:

- (hyper)moves played in several arenas
- composition quite elaborate

Game models for System F

An (incomplete) timeline:

- Hughes ‘97, ’00 : *Hypergames*
- Murawski & Ong ‘01 : *Evolving games*
- Abramsky & Jagadeesan ‘03 : *Variable games*
- Laird ‘10,’13 : *Context games*
- J. & Tz. ‘18 : *Nominal traces*

[Murawski & Ong TLCA'01]

Evolving games

Interactions in hypergames can be elaborate:

- a play proceeds in several arenas
- the arenas of *O* are hidden

Instead, the games can be streamlined by evolution:

- each second-order move expands the arena
- *O* and *P* both have visible arenas

  (but *P* has extra conditions for CC and uniformity)

Similar ideas are used by de Lataillade [APAL’08]

Game models for System F

An (incomplete) timeline:

- Hughes ‘97, ’00 : *Hypergames*
- Murawski & Ong ‘01 : *Evolving games*
- Abramsky & Jagadeesan ‘03 : *Variable games*
- Laird ‘10,’13 : *Context games*
- J. & Tz. ‘18 : *Nominal traces*

[Abramsky & Jagadeesan FOSSACS'03]

Variable games

The evolution of arenas on the fly is still elaborate.

An alternative approach is for arenas to be fully ‘evolved’ from the beginning:

- there is a common universe of moves
- each arena hole is saturated with moves
- as the game proceeds, *O* reveals the real arena

Strategies on $A(X)$ include plays for all instantiations of X, and are *generic* on them:

$$\sigma : A \quad = \quad \Lambda X.\sigma : \forall X.A$$

Game models for System F

An (incomplete) timeline:

- Hughes ‘97, ’00 : *Hypergames*
- Murawski & Ong ‘01 : *Evolving games*
- Abramsky & Jagadeesan ‘03 : *Variable games*
- Laird ‘10,’13 : *Context games*
- J. & Tz. ‘18 : *Nominal traces*

[Laird LICS'10, JACM'13]

Context games

The complexities of game models lie in interpreting

$$\forall X.A$$

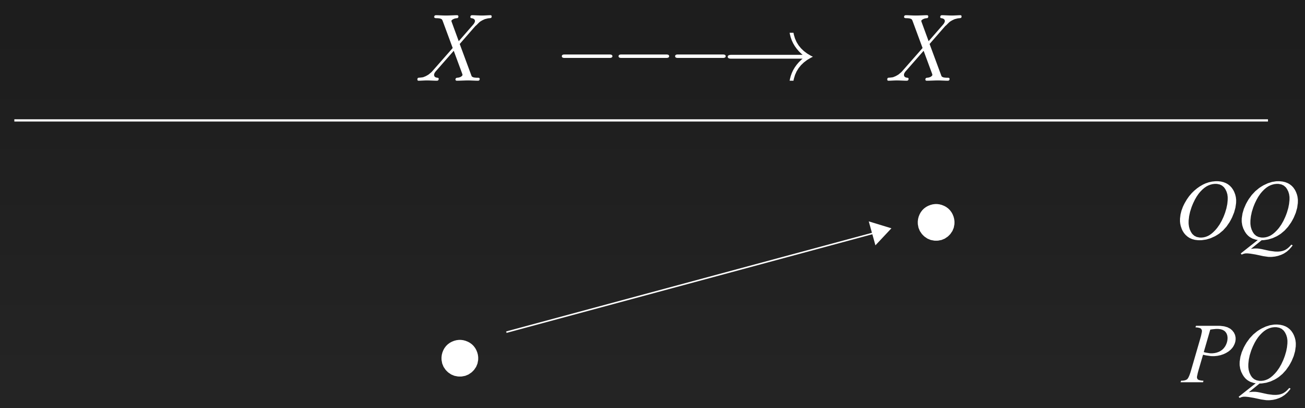
which, in some way, incorporates every $A\{\vartheta/X\}$.

An alternative is to completely separate the two:

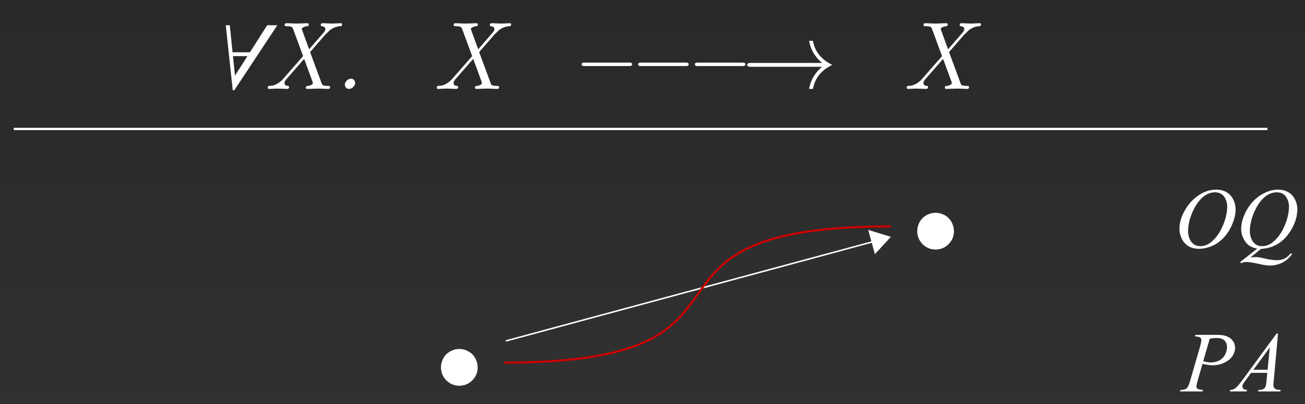
- in $\forall X.A$ the occurrences of X are dummy moves
- they cannot be opened/expanded with arenas
- they record the generic behaviour needed in $A\{\vartheta/X\}$

Identities in context games

$$\lambda x^X.\, x : X \to X$$



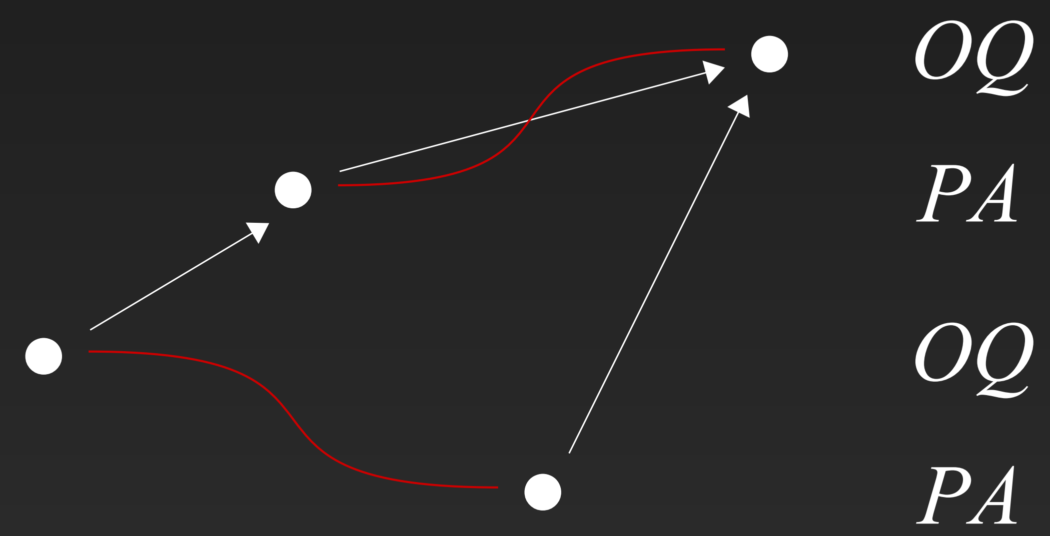
$$\Lambda X. \lambda x^X.\, x : \forall X. X \to X$$



More examples

$$\Lambda X.\lambda f^{X\to X}.\lambda x^{X}.\,fx : \forall X.(X\to X)\to X\to X$$

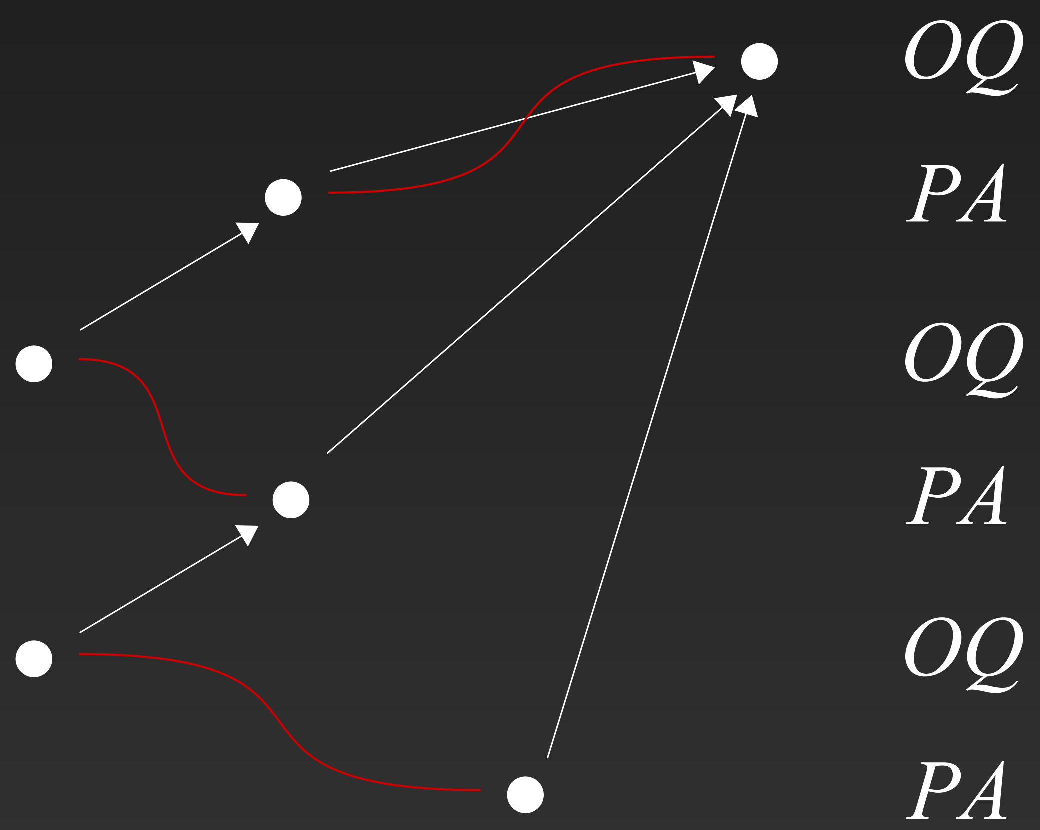
$$\forall X.\ (X \to X) \to X \to X$$



More examples

$$\Lambda X.\lambda f^{X\to X}.\lambda x^{X}.\,f(fx)\ :\ \forall X.(X\to X)\to X\to X$$

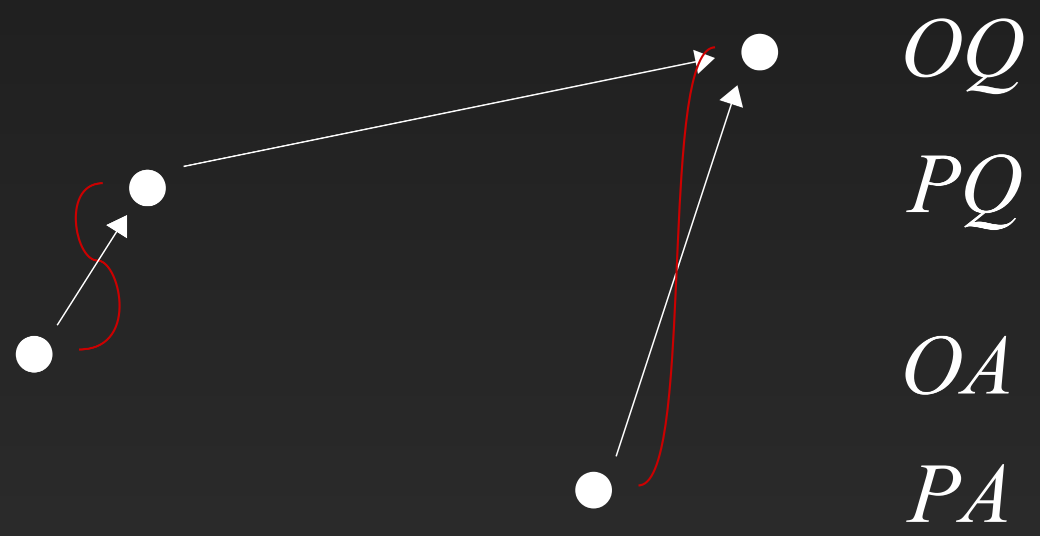
$$\forall X.\ (X \to X) \to X \to X$$



Instantiation

$$\lambda f^{\forall X.X\to X}.\,f : \forall X.(X\to X)\to\forall X.(X\to X)$$

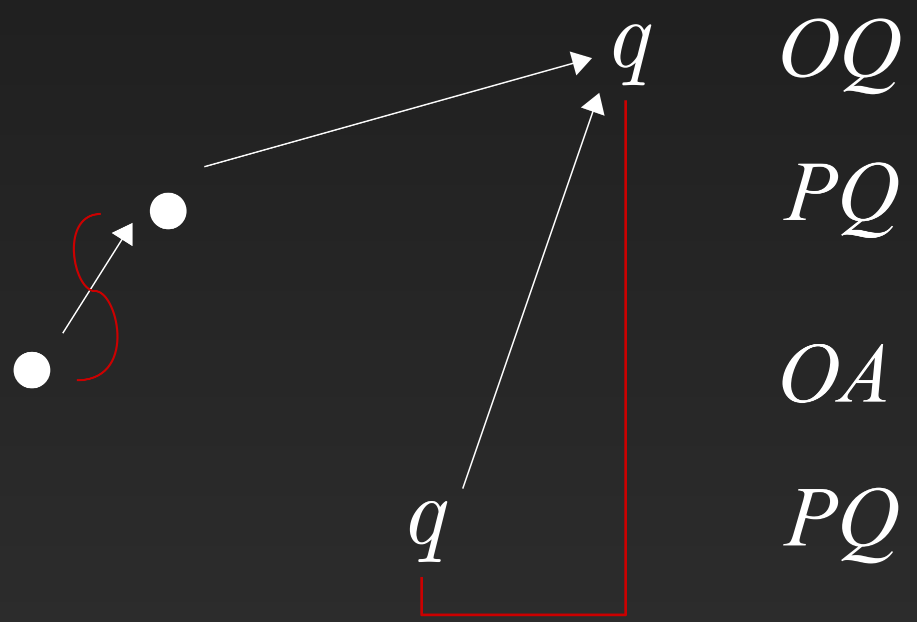
$$\forall X.(X\to X) \to \forall X.(X\to X)$$



Instantiation

$$\lambda f^{\forall X.X\to X}.\,f\,\vartheta : \forall X.(X\to X)\to\vartheta\to\vartheta$$

$$\forall X.(X\to X) \to \vartheta \to \vartheta$$



Context games

Simpler than previous models:

- dummy moves ~ hidden arenas of hypergames (but both for *O* and *P*)
- we go from $\forall X.A$ to $A\{\vartheta/X\}$ by expanding X
- we go from M to $M\vartheta$ by doing copycat between matched *O*/*P* holes

The model captures the more general language F_{ref}

- fewer conditions
- seems adaptable to System F (full completeness?)

Game models for System F

An (incomplete) timeline:

- Hughes ‘97, ’00 : *Hypergames*
- Murawski & Ong ‘01 : *Evolving games*
- Abramsky & Jagadeesan ‘03 : *Variable games*
- Laird ‘10,’13 : *Context games*
- J. & Tz. ‘18 : *Nominal traces*

Nominal traces

We can reason about System F operationally:

- we derive traces by executing open terms
- ~ operational analogue of games

E.g. $\Lambda X.\lambda x^X.\, x : \forall X.X \to X$ is expecting two arguments:

- a type (for X), which we represent by a *type name* α
- a term (for x), represented by a *computation name* c

Thus, the trace starts with a call on two arguments:

$$\diamond \xrightarrow{\;call\ (\alpha,c)\;} (\Lambda X.\lambda x^X.\, x)\alpha c \longrightarrow c \xrightarrow{\;call\ c()\;} [\,] \xrightarrow{\;ok\ ok\;} \diamond$$

More traces

$$\Lambda X.\lambda f^{X\to X}.\lambda x^X.\, fx : \forall X.\ (X\to X) \to X \to X$$

$$\diamond \xrightarrow{call\ (\alpha, c_f, c_x)} (\Lambda X.\lambda f.\lambda x.\, fx)\alpha c_f c_x : \alpha \longrightarrow c_f c_x : \alpha$$

$$\xrightarrow{call\ c_f(c)} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \diamond$$

$$\xrightarrow{call\ c()} c_x : \alpha \xrightarrow{call\ c_x()} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \diamond$$

$$c \mapsto c_x$$

More traces

$$\Lambda X.\lambda f^{X\to X}.\lambda x^{X}.\,fx : \forall X.\ (X\to X) \to X \to X$$

$$\Diamond \xrightarrow{call\ (\alpha, c_f, c_x)} (\Lambda X.\lambda f.\lambda x.\ fx)\alpha c_f c_x : \alpha \longrightarrow c_f c_x : \alpha$$

$$\xrightarrow{call\ c_f(c)} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \Diamond$$

$$\xrightarrow{call\ c()} c_x : \alpha \xrightarrow{call\ c_x()} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \Diamond$$

$$c \mapsto c_x$$

$$\Lambda X.\lambda f^{X\to X}.\lambda x^{X}.\,f(fx) : \forall X.\ (X\to X) \to X \to X$$

$$\Diamond \xrightarrow{call\ (\alpha, c_f, c_x)} (\Lambda X.\lambda f.\lambda x.\ f(fx))\alpha c_f c_x : \alpha \longrightarrow c_f(c_f c_x) : \alpha$$

$$\xrightarrow{call\ c_f(c)} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \Diamond$$

$$\xrightarrow{call\ c()} c_f c_x : \alpha \xrightarrow{call\ c_f(c')} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \Diamond$$

$$\xrightarrow{call\ c'()} c_x : \alpha \xrightarrow{ok\ ok} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \Diamond$$

$$c \mapsto c_f c_x$$
$$c' \mapsto c_x$$

Instantiation

$$\lambda f^{\forall X.X\to X}.f : \forall X.(X\to X) \to \forall X.(X\to X)$$

$$\diamond \xrightarrow{call\ (c_f,\alpha,c)} (\lambda f.\,f)\,c_f\,\alpha\,c : \alpha \longrightarrow c_f\,\alpha\,c : \alpha$$

$$\xrightarrow{call\ c_f(\alpha',c')} [\,] : \alpha'\to\alpha$$

$$\xrightarrow{call\ c'()} c : \alpha' :: [\,] : \alpha'\to\alpha \xrightarrow{ok\ \ ok} c : \alpha$$

$$\xrightarrow{call\ c()} [\,] : \alpha\to\alpha \xrightarrow{ok\ \ ok} \diamond$$

$$\alpha' \mapsto \alpha$$

$$c' \mapsto c : \alpha'$$

Instantiation

$\lambda f^{\forall X.X\to X}.\,f\,\vartheta : \forall X.(X\to X) \to \vartheta \to \vartheta$

$\diamond \xrightarrow{call\ (c_f, c_\vartheta, \boldsymbol{c})} (\lambda f.\, f\,\vartheta)\, c_f\, c_\vartheta\, \boldsymbol{c} : \alpha \longrightarrow c_f\, \vartheta\, c_\vartheta\, \boldsymbol{c} : \alpha$

$\xrightarrow{call\ c_f(\alpha', c')} [\,]\boldsymbol{c} : \alpha'\to\alpha$

$\xrightarrow{call\ c'()} c_\vartheta : \alpha' :: [\,]\boldsymbol{c} : \alpha'\to\alpha \xrightarrow{ok\ ok} c_\vartheta\, \boldsymbol{c} : \alpha$

$\xrightarrow{call\ c_\vartheta(\boldsymbol{c}')} [\,] : \alpha\to\alpha \xrightarrow{ok\ ok} \diamond$

$\xrightarrow{call\ c_i'(..)} c_i\, .. : \beta \xrightarrow{call\ c_i(..)} [\,] : \beta'\to\beta \longrightarrow \ldots$

$\alpha' \mapsto \vartheta$

$c' \mapsto c_\vartheta : \alpha'$

$c_i' \mapsto c_i$

Traces formally

(INT) $\langle (M,\theta) :: \mathcal{E}, \gamma, \phi\rangle \rightarrow \langle (M',\theta) :: \mathcal{E}, \gamma, \phi\rangle$ when $M \rightarrow^* M'$ with M' a head normal form.

(OQ$_0$) $\langle \Diamond, \gamma, \phi\rangle \xrightarrow{c(a_1,\ldots,a_n)} \langle [(M a_1 \cdots a_n, \alpha)], \gamma, \phi \cdot \phi'\rangle$

with $\gamma(c) = (M,\theta)$, $((a_1,\ldots,a_n),\phi',\alpha) \in [\![\theta]\!]$ and $\alpha \in \mathrm{dom}(\phi\cdot\phi')$.

(OQ) $\langle (E, \alpha \rightsquigarrow \theta') :: \mathcal{E}, \gamma, \phi\rangle \xrightarrow{c(a_1,\ldots,a_n)} \langle (M a_1 \cdots a_n, \alpha') :: (E, \alpha \rightsquigarrow \theta') :: \mathcal{E}, \gamma, \phi\cdot\phi'\rangle$

with $\alpha \in \mathrm{dom}(\gamma)$, $\gamma(c) = (M,\theta)$, $((a_1,\ldots,a_n),\phi',\alpha') \in [\![\theta]\!]$ and $\alpha' \in \mathrm{dom}(\phi\cdot\phi') \cup \{\alpha\}$.

(PQ$_0$) $\langle [(E[c\hat{M}_1 \cdots \hat{M}_n],\theta)], \gamma, \phi\rangle \xrightarrow{\bar{c}(a_1,\ldots,a_n)} \langle [(E, \alpha \rightsquigarrow \theta)], \gamma\cdot\gamma', \phi\rangle$

when θ is a closed with empty support, $\mathrm{ext}(\phi(c)) = (\tau_1,\ldots,\tau_n,\xi)$
and $((a_1,\ldots,a_n),\gamma',\alpha) \in \mathsf{AVal}((\hat{M}_1,\tau_1),\ldots,(\hat{M}_n,\tau_n),\xi)$.

(PQ) $\langle (E[c\hat{M}_1 \cdots \hat{M}_n],\alpha') :: \mathcal{E}, \gamma, \phi\rangle \xrightarrow{\bar{c}(a_1,\ldots,a_n)} \langle (E, \alpha \rightsquigarrow \alpha') :: \mathcal{E}, \gamma\cdot\gamma', \phi\rangle$ when $\alpha' \in \mathrm{dom}(\phi)$,

$\mathrm{ext}(\phi(c)) = (\tau_1,\ldots,\tau_n,\xi)$ and $((a_1,\ldots,a_n),\gamma',\alpha) \in \mathsf{AVal}((\hat{M}_1,\tau_1),\ldots,(\hat{M}_n,\tau_n),\xi)$.

(OA) $\langle (\bullet, \alpha \rightsquigarrow \alpha) :: \mathcal{E}, \gamma, \phi\rangle \xrightarrow{\mathsf{OK}\overline{\mathsf{OK}}} \langle \mathcal{E}, \gamma, \phi\rangle$ when $\alpha \in \mathrm{dom}(\phi)$.

(PA) $\langle (M,\alpha) :: (E, \alpha \rightsquigarrow \theta) :: \mathcal{E}, \gamma, \phi\rangle \xrightarrow{\overline{\mathsf{OK}}\mathsf{OK}} \langle (E[M],\theta) :: \mathcal{E}, \gamma, \phi\rangle$ when $\alpha \in \mathrm{dom}(\gamma)$ and M a hnf.

Properties of the model

Positives of context games:

- simple treatment of quantification
- nominal ideas: polymorphic values are just names

Trace representation:

- model is executable, does not need composition
- allows operational reasoning (e.g. Strachey equivalent → logically related)
- not compositional: composition a theorem, and instantiation is not clear

Further work

Questions on the current model:

- Formal link between context games and trace model
- Notion of equivalence captured

Extensions:

- Model for relational parametricity
- Calculi for dynamic binding (types bog us down)