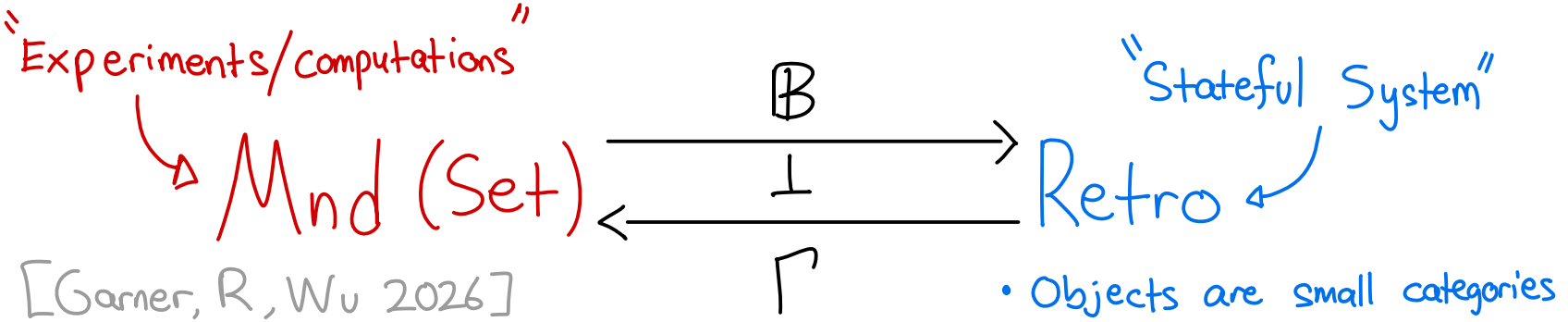
Game Spectrum of a Monad 

Alyssa Renata (She/Her)
(Imperial College London)

Pacific Category Theory (PCT) Seminar 28 August 2026

This talk in one slide

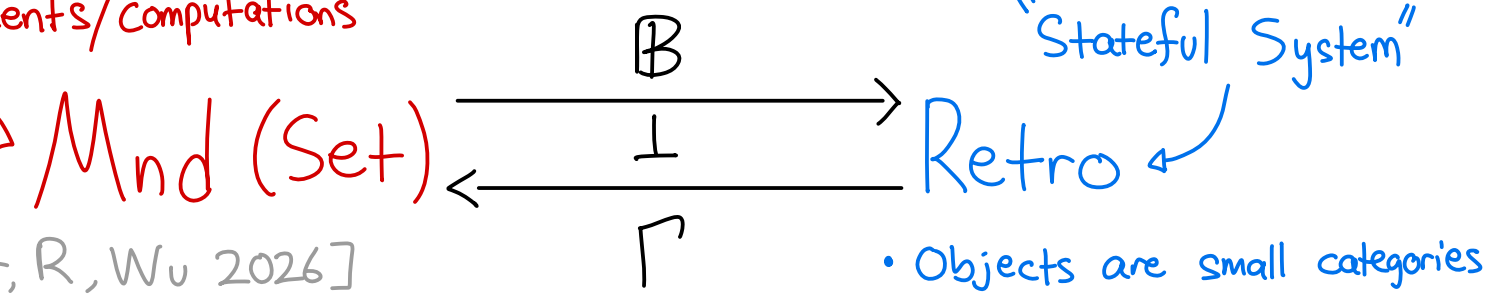


[Garner, R, Wu 2026]

- Objects are small categories
- morphisms are retrofunctors (AKA cofunctors)

This talk in one slide

"Experiments/computations"



[Garner, R, Wu 2026]

- Objects are small categories
- morphisms are retrofunctors (AKA cofunctors)

"Measurements/functions"

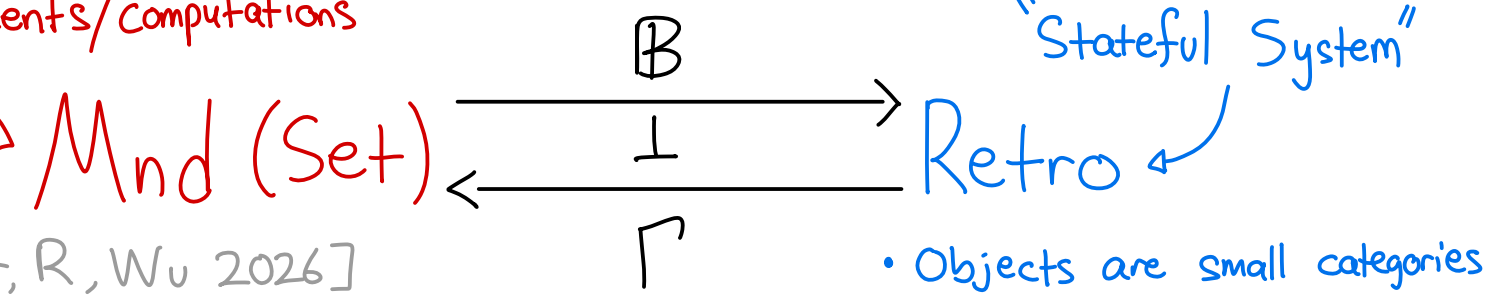
"State space"

- Analogy $\mathbf{CRing} \simeq \mathbf{AffScheme}^{op}$

  or $\mathbf{BA} \simeq \mathbf{Stone}^{op}$

This talk in one slide

"Experiments/computations"



[Garner, R, Wu 2026]

- Objects are small categories
- morphisms are retrofunctors (AKA cofunctors)

"Measurements/functions"

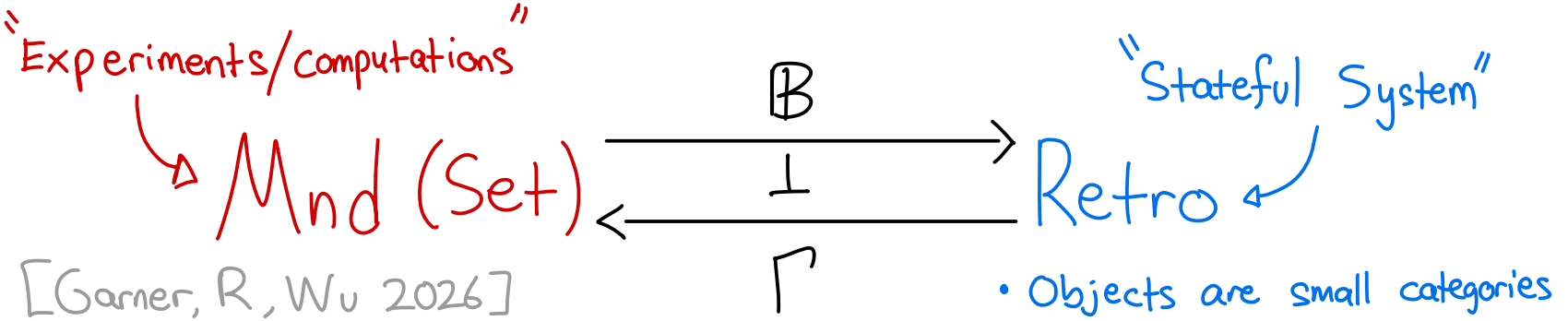
"State space"

- Analogy $\mathbf{CRing} \simeq \mathbf{AffScheme}^{op}$

  or $\mathbf{BA} \simeq \mathbf{Stone}^{op}$

- Problem $\mathbb{B}\mathbf{T}$ is often trivial...

This talk in one slide



"Measurements/functions" "State space"

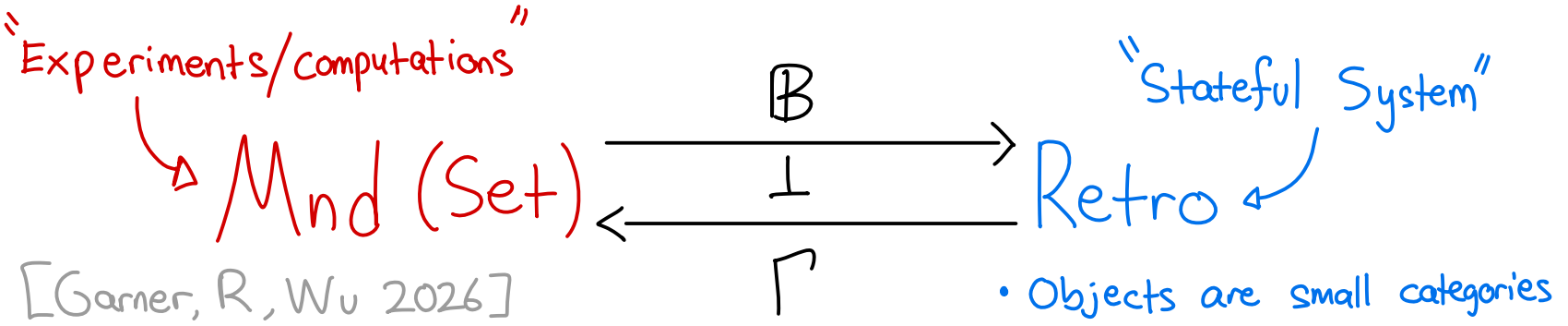
- Analogy $\mathbf{CRing} \simeq \mathbf{AffScheme}^{op}$

  or $\mathbf{BA} \simeq \mathbf{Stone}^{op}$

- Problem $\mathbb{B}\mathbf{T}$ is often trivial...

- Idea Replace Retro with Template Games [Mellies 2019, 2021].

This talk in one slide



[Garner, R, Wu 2026]

- Objects are small categories
- morphisms are retrofunctors (AKA cofunctors)

"Measurements/functions" "state space"

- Analogy CRing $\simeq$ $\mathrm{AffScheme}^{\mathrm{op}}$

  or BA $\simeq$ $\mathrm{Stone}^{\mathrm{op}}$

- Problem $\mathbb{B}\mathrm{T}$ is often trivial...

"games with 2-cells" can deal with equations?

this solves the problem for signatures/free theories

- Idea Replace Retro with Template Games [Mellies 2019, 2021].

This talk in one slide

"Experiments/computations"

Mnd (Set)

[Garner, R, Wu 2026]

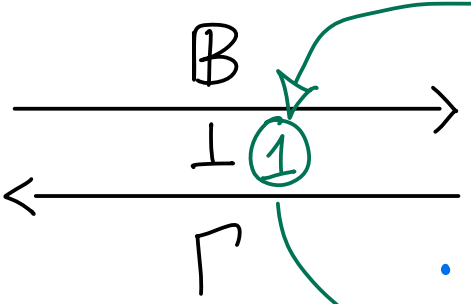
"Measurements/functions"

- Analogy CRing $\simeq$ AffScheme^{op}

  or BA $\simeq$ Stone^{op}

- Problem $\mathbb{BT}$ is often trivial...

- Idea Replace Retro with Template Games [Mellies 2019, 2021].

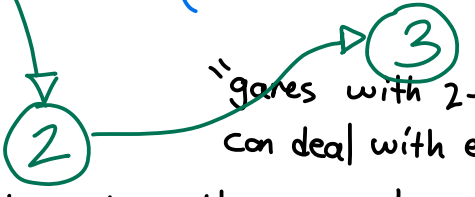


"Stateful System"

Retro

- Objects are small categories
- morphisms are retrofunctors (AKA cofunctors)

"state space"

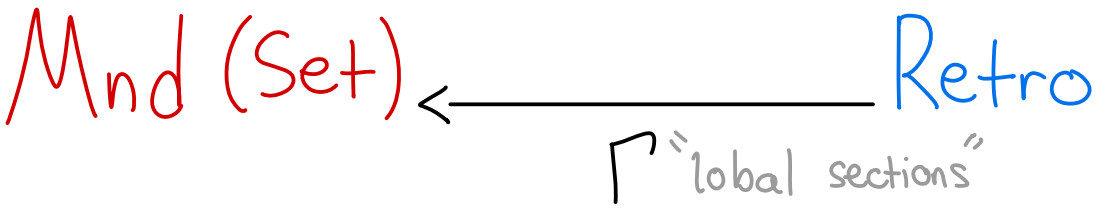


"games with 2-cells" can deal with equations?

this solves the problem for signatures/free theories

Plan for the Talk

① The Behaviour Category of a Monad



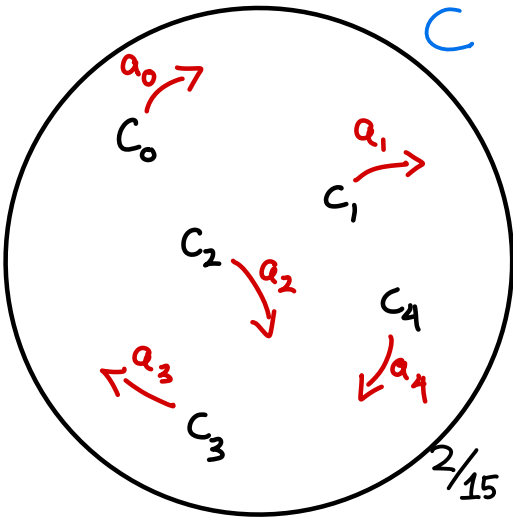
- Objects are small categories
- morphisms are retrofunctors

$\mathbf{Mnd}(\mathbf{Set}) \longleftarrow \mathbf{Retro}$

$\ulcorner$ "lobal sections"

- Objects are small categories
- morphisms are retrofunctors

$$\ulcorner C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$



$\mathbf{Mnd}(\mathbf{Set}) \longleftarrow \mathbf{Retro}$

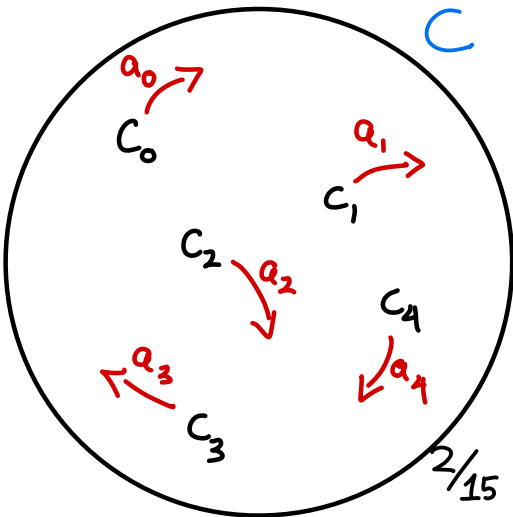
- Objects are small categories
- morphisms are retrofunctors

Γ "global sections"

$$\Gamma\mathbb{C}(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

Special case Let $W \in \mathbf{Set}$.

$$\mathbb{C}\mathrm{odisc}(W) = \quad W \xleftarrow{\sigma} W \times W \xrightarrow{\tau} W$$



$\mathbf{Mnd}(\mathbf{Set}) \longleftarrow \mathbf{Retro}$

- Objects are small categories
- morphisms are retrofunctors

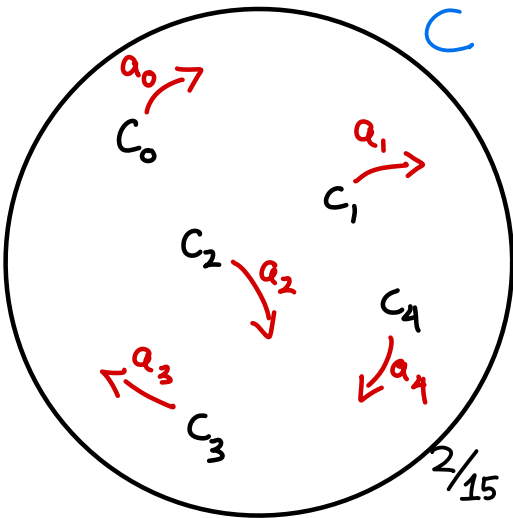
Γ "global sections"

$$\Gamma C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

Special case Let $W \in \mathbf{Set}$.

$$\mathbf{Codisc}(W) = \quad W \times W \;\overset{\sigma}{\swarrow}\; W \qquad W \times W \;\overset{\tau}{\searrow}\; W$$

$$\Gamma\,\mathbf{Codisc}(W)(A) = \left\{ W \xrightarrow{s} W \times W \times A \;\middle|\; \begin{array}{ccc} W & \longrightarrow & W \times W \times A \\ & \searrow & \downarrow \pi_1 \\ & & W \end{array} \right\}$$



Mnd(Set) ← Retro

- Objects are small categories
- morphisms are retrofunctors

"global sections"

$$\Gamma C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

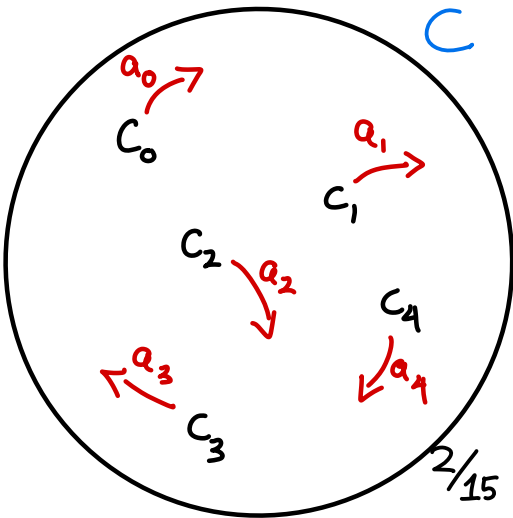
Special case Let $W \in \mathrm{Set}$.

$$\mathrm{Codisc}(W) = \begin{array}{ccccc} & & W \times W & & \\ & \sigma \swarrow & & \searrow \tau & \\ W & & & & W \end{array}$$

$$\Gamma \mathrm{Codisc}(W)(A) = \left\{ W \xrightarrow{s} W \times W \times A \;\middle|\; \begin{array}{ccc} W & \longrightarrow & W \times W \times A \\ & = & \downarrow \pi_1 \\ & & W \end{array} \right\}$$

$$\cong (W \times A)^W$$

"State Monad"



$\mathbf{Mnd}(\mathbf{Set}) \longleftarrow \mathbf{Retro}$

- Objects are small categories
- morphisms are retrofunctors

$F : C \rightsquigarrow D$

$C_0 \longleftarrow D_0 : F_0$

Γ "global sections"

$$\Gamma C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

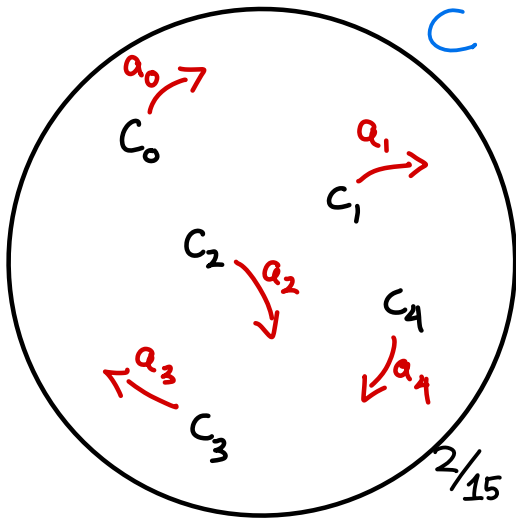
Special case Let $W \in \mathbf{Set}$.

$$\mathbf{Codisc}(W) = \quad W \xleftarrow{\sigma} W \times W \xrightarrow{\tau} W$$

$$\Gamma \mathbf{Codisc}(W)(A) = \left\{ W \xrightarrow{s} W \times W \times A \;\middle|\; \begin{array}{ccc} W & = & W \\ & & \downarrow \pi_1 \\ & & W \end{array} \right\}$$

$$\cong (W \times A)^W$$

"State Monad"



$\mathbf{Mnd}(\mathbf{Set}) \longleftarrow \mathbf{Retro}$

- Objects are small categories
- morphisms are retrofunctors

$F : C \rightsquigarrow D$

$C_0 \longleftarrow D_0 : F_0$

$F_1 : c' \longmapsto F_0 d \longleftarrow d$

"global sections"

$$\Gamma C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

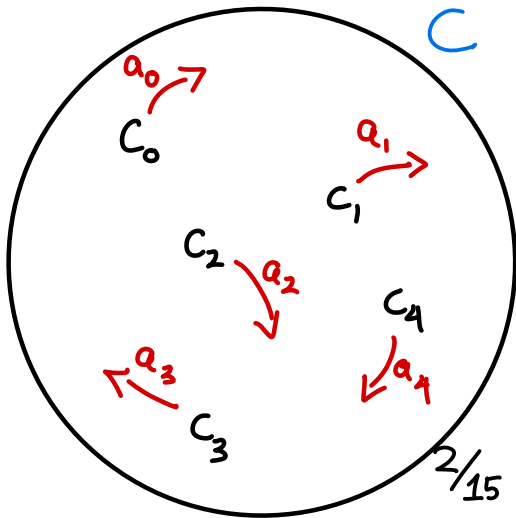
Special case Let $W \in \mathbf{Set}$.

$$\mathbf{Codisc}(W) = W \xleftarrow{\sigma} W \times W \xrightarrow{\tau} W$$

$$\Gamma \mathbf{Codisc}(W)(A) = \left\{ W \xrightarrow{s} W \times W \times A \;\middle|\; \begin{array}{ccc} W & \longrightarrow & W \times W \times A \\ & \searrow & \downarrow \pi_1 \\ & & W \end{array} \right\}$$

$$\cong (W \times A)^W$$

"State Monad"



Mnd(Set) ← Retro

- Objects are small categories
- morphisms are retrofunctors

"global sections"

$$\Gamma C(A) = \left\{ C_0 \xrightarrow{s} C_1 \times A \;\middle|\; \begin{array}{ccc} C_0 & \xrightarrow{s} & C_1 \times A \\ \| & & \downarrow \pi \\ C_0 & \xleftarrow[src]{} & C_1 \end{array} \right\}$$

$F : C \rightsquigarrow D$

$C_0 \leftarrow D_0 : F_0$

$F_1 : c' \leftarrow d' \mapsto d \leftarrow F_0 d$

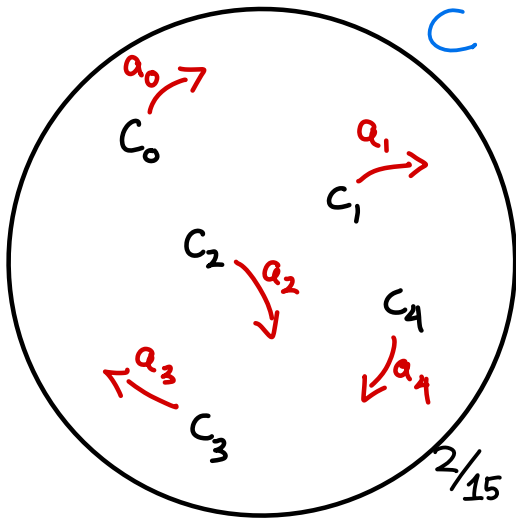
Special case Let $W \in Set$.

$$Codisc(W) = W \xleftarrow{\sigma} W \times W \xrightarrow{\tau} W$$

$$\Gamma Codisc(W)(A) = \left\{ W \xrightarrow{s} W \times W \times A \;\middle|\; \begin{array}{ccc} W & \longrightarrow & W \times W \times A \\ & \searrow & \downarrow \pi_1 \\ & & W \end{array} \right\}$$

$$\cong (W \times A)^W$$

"State Monad"



$$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$$

Bad example 1

Suppose $z \in T\emptyset$.

$$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$$

Bad example 1

Suppose $z \in T\emptyset$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(z) : (W \to W \times \emptyset) \cong \emptyset$, unless $W = \emptyset$.

$$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$$

Bad example 1

Suppose $z \in T\emptyset$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(z) : (W \to W \times \emptyset) \cong \emptyset$, unless $W = \emptyset$.

Bad example 2 Suppose $b \in T2$ s.t. $b(x,y) = b(y,x)$.

$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$

Bad example 1

Suppose $z \in T\emptyset$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(z) : (W \to W \times \emptyset) \cong \emptyset$, unless $W = \emptyset$.

Bad example 2

Suppose $b \in T2$ s.t. $b(x,y) = b(y,x)$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(b) : W \to W \times 2$ s.t. $W \xrightarrow{\rho(b)} W \times 2$

$\rho(b)$ followed by $W \times \mathrm{flip}$ gives $W \times 2$

so if $\exists \omega \in W$ then $0 = 1$. i.e. $W = \emptyset$.

$$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$$

so the only sensible choice for these is $\mathbb{B}_0 T = \mathbb{B}_1 T = \emptyset$.

Bad example 1

Suppose $z \in T\emptyset$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(z) : (W \to W \times \emptyset) \cong \emptyset$, unless $W = \emptyset$.

Bad example 2

Suppose $b \in T2$ s.t. $b(x,y) = b(y,x)$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(b) : W \to W \times 2$ s.t. $W \xrightarrow{\rho(b)} W \times 2$

$\rho(b)$ followed by $W \times \mathrm{flip}$ gives $W \times 2$

so if $\exists\, w \in W$ then $0 = 1$. i.e. $W = \emptyset$.

$$\mathrm{Mnd}(T, \Gamma C) \cong \mathrm{Retro}(\mathbb{B}T, C)$$

so the only sensible choice for these is $\mathbb{B}_0 T = \mathbb{B}_1 T = \emptyset$.

Bad example 1

Suppose $z \in T\emptyset$. Then

$\rho : T \to (W \times -)^W$ implies $\rho(z) : (W \to W \times \emptyset) \cong \emptyset$, unless $W = \emptyset$.

Bad example 2 Suppose $b \in T2$ s.t. $b(x,y) = b(y,x)$. Then

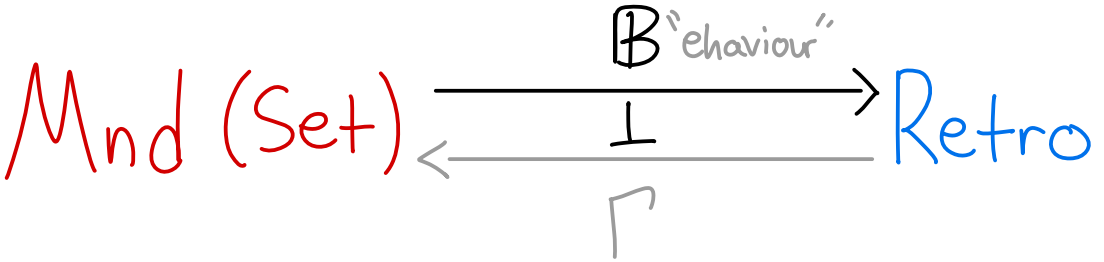
$\rho : T \to (W \times -)^W$ implies $\rho(b) : W \to W \times 2$ s.t. $W \xrightarrow{\rho(b)} W \times 2$

$\rho(b)$ followed by $W \times \mathrm{flip}$ gives $W \times 2$

so if $\exists w \in W$ then $0 = 1$. i.e. $W = \emptyset$.

defn [Power-Shkaravska 2004] A T-comodel is a pair $(W, T \xrightarrow{\rho} (W \times -)^W)$.

(morphisms are $W \to W'$ commuting with the ρ)

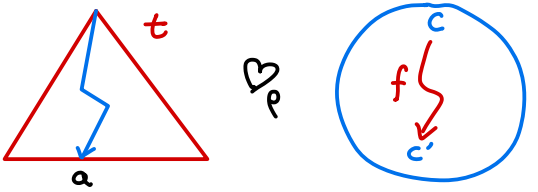

B "behaviour"
Mnd (Set)
⊥
Retro
⌜

$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ $\quad \dashv$

Given $T \xrightarrow{\rho} \Gamma C$,

From T's perspective, $c \in C_0$ tells it how to <u>run</u> a T-term $t \in TA$



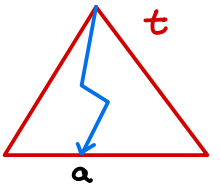
where $\rho(t)(c) = (f : c \to c', a)$.

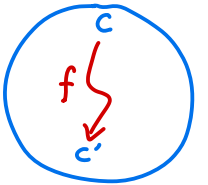
$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ ($\dashv$)

Given $T \xrightarrow{\rho} \Gamma C$,

From T's perspective, $c \in C_0$ tells it how to <u>run</u> a T-term $t \in TA$

so $\mathbb{B}_0 T = \{ \beta : T \to \mathrm{Id} \mid$ 

$\} \heartsuit_\rho$ 

where $\rho(t)(c) = (f : c \to c', a)$.

$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ $\quad \dashv$

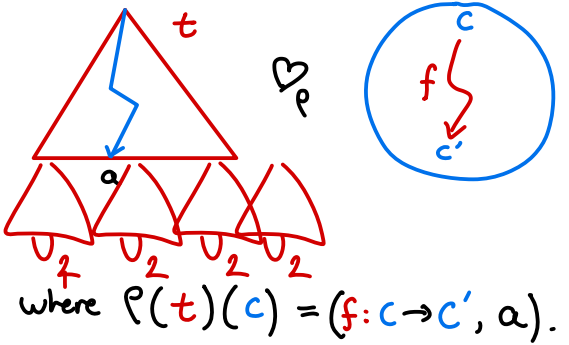
Given $T \xrightarrow{\rho} \Gamma C$,

From T's perspective, $c \in C_o$ tells it how to run a T-term $t \in TA$

so $\mathbb{B}_o T = \{ \beta : T \to Id \mid \beta(t ; u) = \beta(t \gg u(\beta(t))) \}$

$1 \xrightarrow{t} A \xrightarrow{u} B$ Kleisli composition

$1 \xrightarrow{t} A \xrightarrow{!} 1 \xrightarrow{u(\beta(t))} B$



where $\rho(t)(c) = (f : c \to c', a)$.

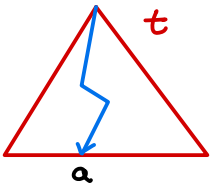
$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ ($\dashv$)

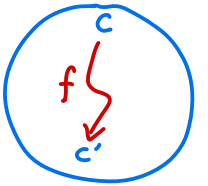
Given $T \xrightarrow{\rho} \Gamma C$,

From T's perspective, $c \in C_0$ tells it how to <u>run</u> a T-term $t \in TA$

so $\mathbb{B}_0 T = \{ \beta : T \to Id \mid \beta(t ; u) = \beta(t \gg u(\beta(t))) \}$



$\heartsuit_\rho$



$1 \xrightarrow{t} A \xrightarrow{u} B$ Kleisli composition

$1 \xrightarrow{t} A \xrightarrow{!} 1 \xrightarrow{u(\beta(t))} B$

where $\rho(t)(c) = (f : c \to c', a)$.

From c's perspective, t and t' are indistinguishable as long as they induce the same f.

so $\mathbb{B}_1 T = \coprod_{\beta \in \mathbb{B}_0 T} T1 / \sim_\beta$

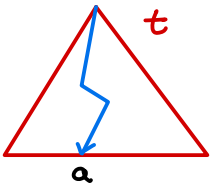
$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ ($\dashv$)

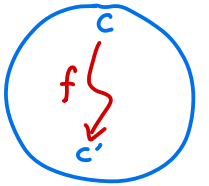
Given $T \xrightarrow{\rho} \Gamma C$,

From T's perspective, $c \in C_0$ tells it how to <u>run</u> a T-term $t \in TA$

so $\mathbb{B}_0 T = \{ \beta : T \to \mathrm{Id} \mid \beta(t ; u) = \beta(t \gg u(\beta(t))) \}$



$\heartsuit_\rho$



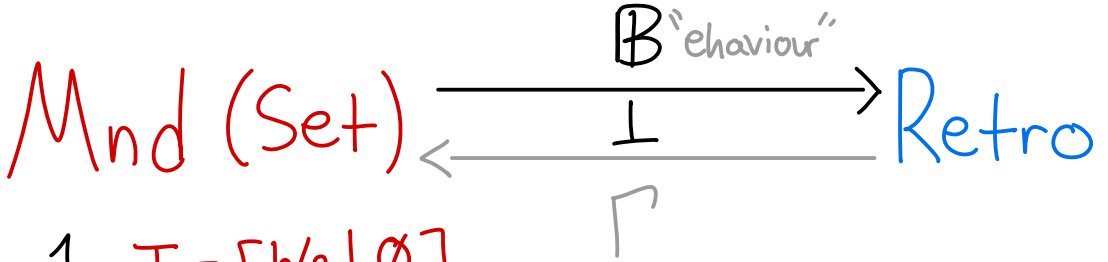
$1 \xrightarrow{t} A \xrightarrow{u} B$ — Kleisli composition

$1 \xrightarrow{t} A \xrightarrow{!} 1 \xrightarrow{u(\beta(t))} B$

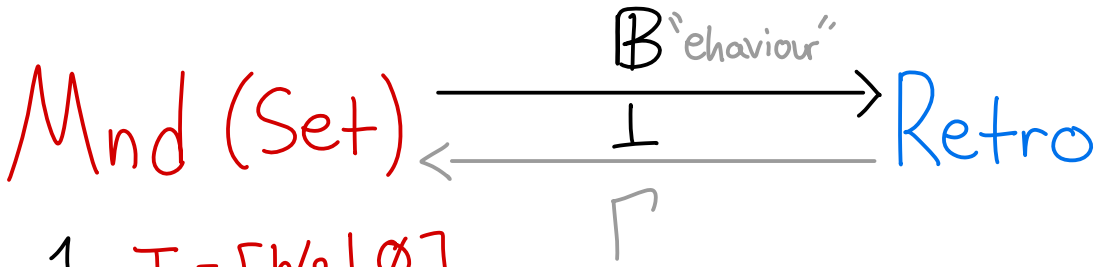
where $\rho(t)(c) = (f : c \to c', a)$.

From c's perspective, t and t' are indistinguishable as long as they induce the same f.

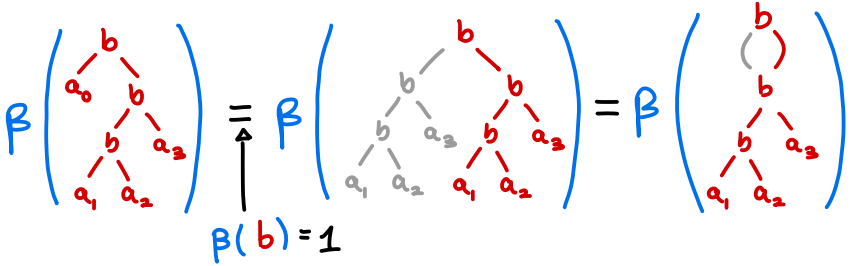
so $\mathbb{B}_1 T = \coprod_{\beta \in \mathbb{B}_0 T} T1 / \sim_\beta$ $\ni$ $[t]_\beta : \beta \longrightarrow \underbrace{\beta(t \gg -)}_{\partial_t \beta}$



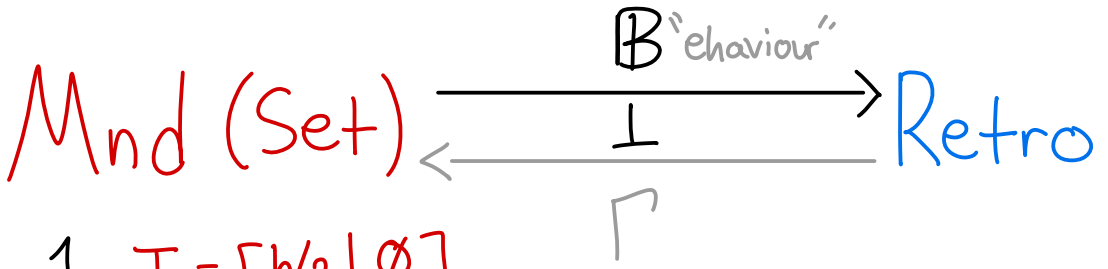
Example 1 $T = [b/2 \mid \emptyset]$



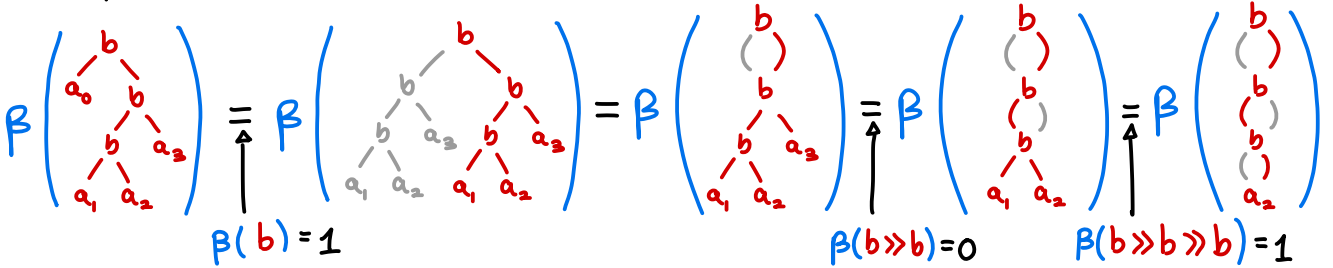
Example 1 $T = [b/2 \mid \emptyset]$

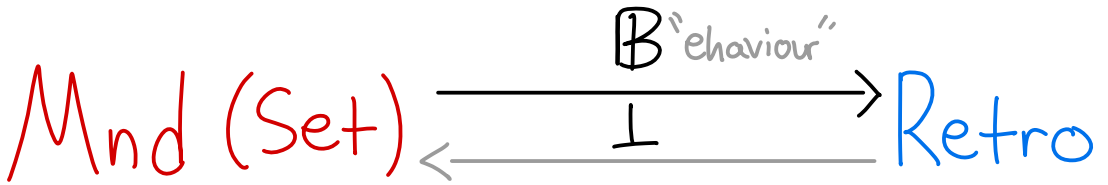


$\beta(b) = 1$

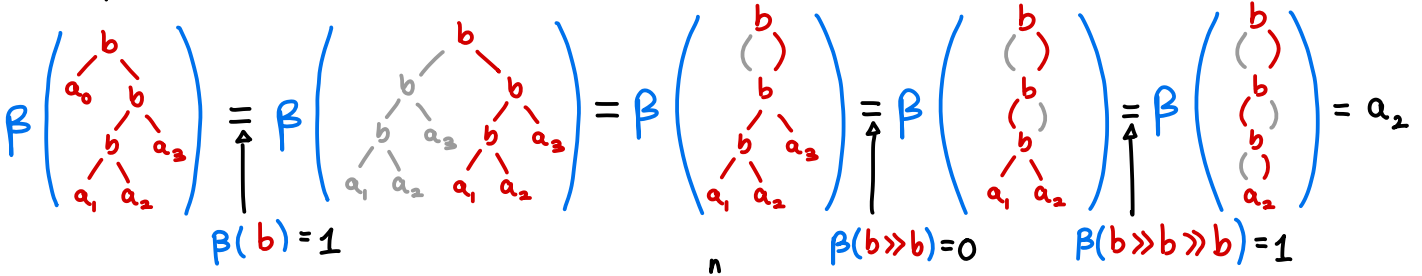


Example 1 $T = [b/2 \mid \emptyset]$

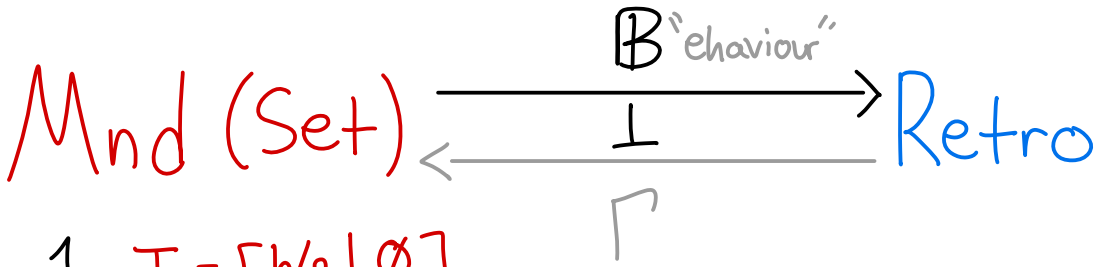




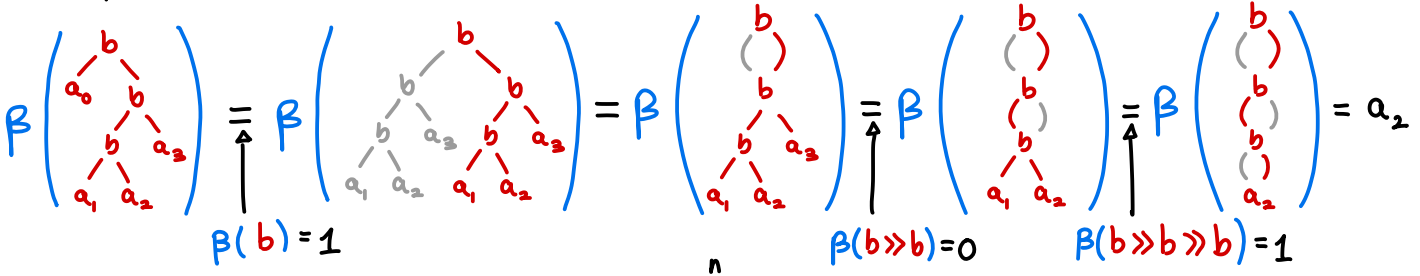
Example 1 $T = [b/2 \mid \emptyset]$



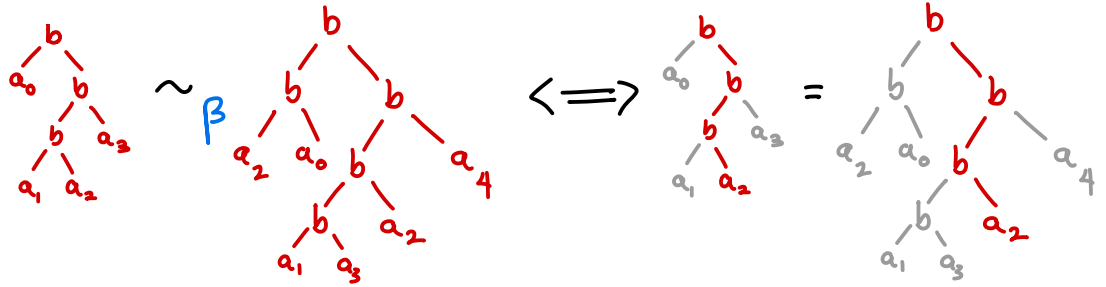
So β determined by $\forall n \in \omega.\ \beta(\overbrace{b \gg \cdots \gg b}^{n})$ i.e. $\mathcal{B}_0 T \cong 2^\omega$.



Example 1 $T = [b/2 \mid \emptyset]$



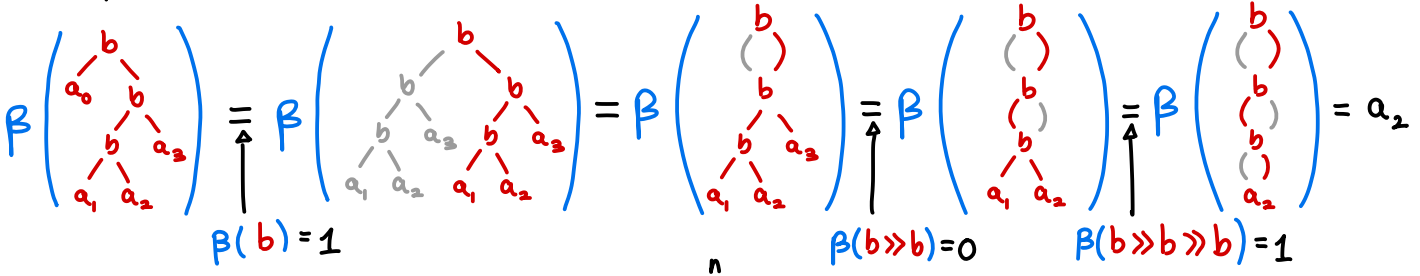
So β determined by $\forall n \in \omega.\ \beta(\overbrace{b \gg \cdots \gg b}^{n})$ i.e. $\mathcal{B}_0 T \cong 2^\omega$.



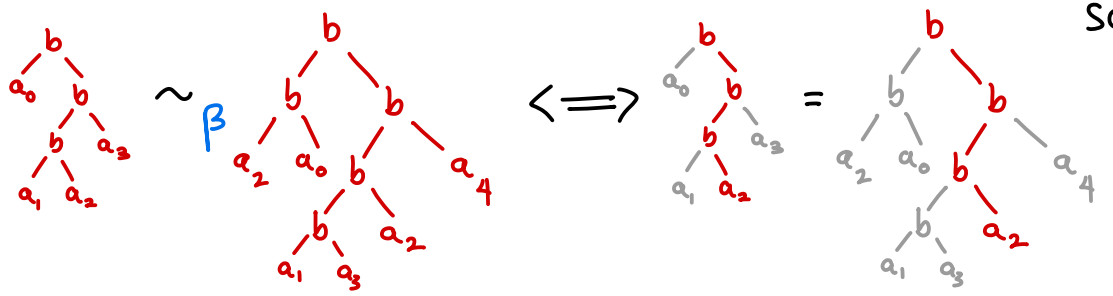
$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ (⊣)

Example 1 $T = [b/2 \mid \emptyset]$



So β determined by $\forall n \in \omega.\ \beta(\overbrace{b \gg \cdots \gg b}^{n})$ i.e. $\mathbb{B}_0 T \cong 2^\omega$.

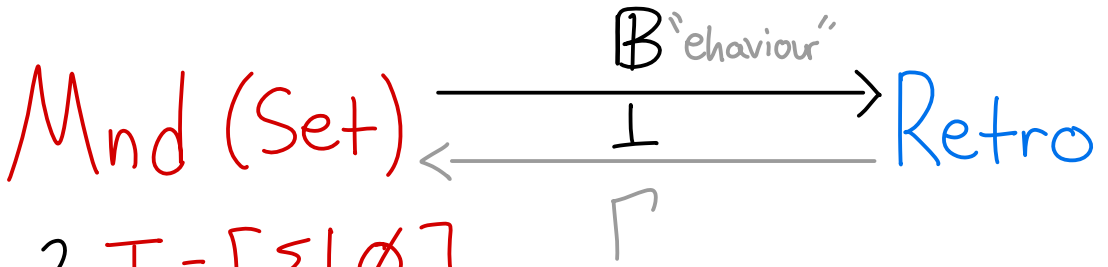


So $\mathbb{B}_1 T = \coprod_{\beta \in \mathbb{B}_0 T} T1/{\sim_\beta}$

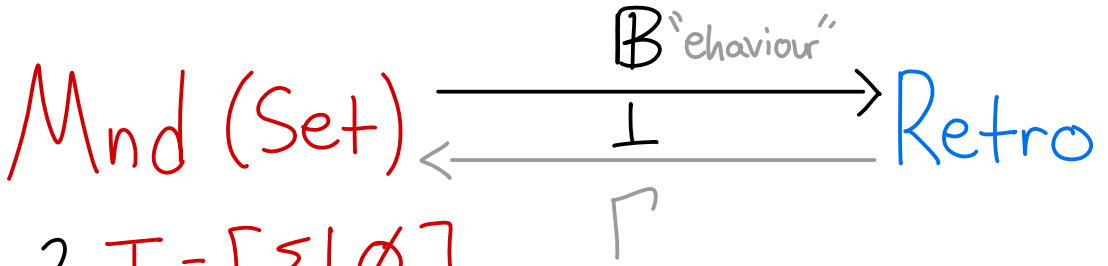
$\cong \coprod_{\beta} \mathbb{N} = \mathbb{B}_0 T \times \mathbb{N}$

ω

$n_\beta : \beta \to \mathrm{tail}^n \beta$

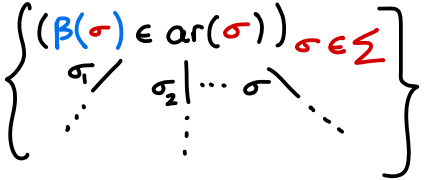


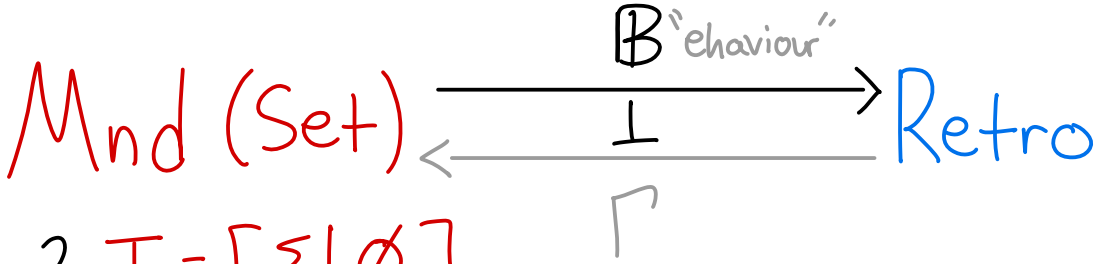
Example 2 $T = [\Sigma \mid \emptyset]$



Example 2 $T = [\Sigma \mid \emptyset]$

$$\mathcal{B}_o T = \nu X. \prod_{\sigma \in \Sigma} ar(\sigma) \times X^{\Sigma} = \left\{ \left(\beta(\sigma) \in ar(\sigma) \right)_{\sigma \in \Sigma} \right\}$$

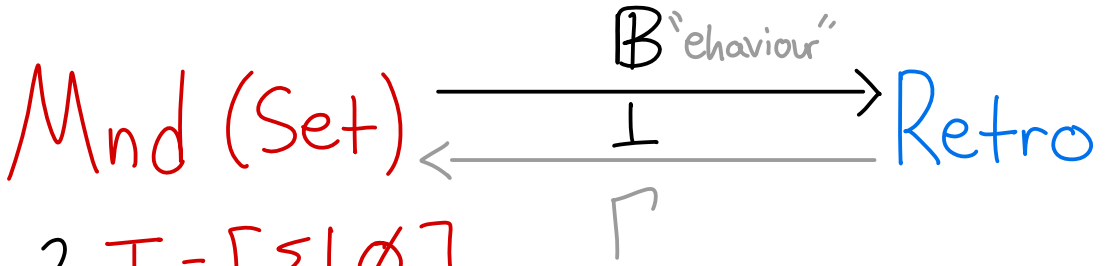




Example 2 $T = [\Sigma \mid \emptyset]$

$$\mathcal{B}_0 T = \nu X. \prod_{\sigma \in \Sigma} ar(\sigma) \times X^{\Sigma} = \left\{ \begin{array}{c} (\beta(\sigma) \in ar(\sigma))_{\sigma \in \Sigma} \\ \sigma_1 \quad \sigma_2 \quad \cdots \quad \sigma \\ \vdots \quad \vdots \quad \quad \ddots \end{array} \right\}$$

$$\mathcal{B}_1 T = \mathcal{B}_0 T \times \Sigma^*$$



Example 2 $T = [\Sigma \mid \emptyset]$

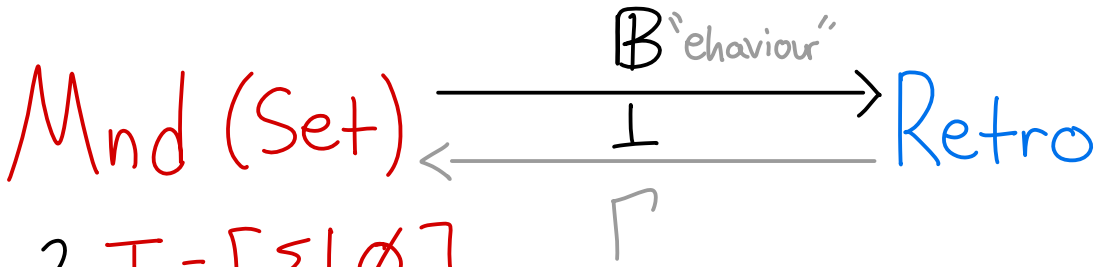
$$\mathbb{B}_0 T = \nu X. \prod_{\sigma \in \Sigma} ar(\sigma) \times X^{\Sigma} = \left\{ \begin{matrix} (\beta(\sigma) \in ar(\sigma))_{\sigma \in \Sigma} \\ \sigma_1 \quad \sigma_2 \quad \cdots \quad \sigma \\ \vdots \quad \vdots \quad \quad \ddots \end{matrix} \right\}$$

$$\mathbb{B}_1 T = \mathbb{B}_0 T \times \Sigma^*$$

Example 3

- $T = (W \times -)^W$

$$\Longrightarrow \quad \mathbb{B} T = \text{codisc}(W)$$



Example 2 $T = [\Sigma \mid \emptyset]$

$$\mathbb{B}_0 T = \nu X. \prod_{\sigma \in \Sigma} ar(\sigma) \times X^{\Sigma} = \left\{ \begin{matrix} (\beta(\sigma) \in ar(\sigma))_{\sigma \in \Sigma} \\ \sigma_1 \quad \sigma_2 \mid \cdots \quad \sigma \\ \vdots \quad\quad \vdots \end{matrix} \right\}$$

$$\mathbb{B}_1 T = \mathbb{B}_0 T \times \Sigma^*$$

Example 3

- $T = (W \times -)^W \implies \mathbb{B}T = \mathrm{codisc}(W)$
- $T = \mathcal{D}_\omega^B$ for Boolean Algebra $B \implies \mathbb{B}T = \mathrm{disc}(\mathrm{Spec}(B))$

$\mathbb{B}$ "behaviour"

$\mathbf{Mnd}(\mathbf{Set}) \rightleftarrows \mathbf{Retro}$ ($\dashv$)

We also have a topological/localic version generalizing Stone Duality!

Example 2 $T = [\Sigma \mid \emptyset]$

$$\mathbb{B}_0 T = \nu X. \prod_{\sigma \in \Sigma} \mathrm{ar}(\sigma) \times X^{\Sigma} = \left\{ \begin{array}{ccccc} (\beta(\sigma) \in \mathrm{ar}(\sigma))_{\sigma \in \Sigma} \\ \sigma_1 & \sigma_2 & \cdots & \sigma & \cdots \\ \vdots & \vdots \end{array} \right\}$$

$$\mathbb{B}_1 T = \mathbb{B}_0 T \times \Sigma^*$$

Example 3

- $T = (W \times -)^W \implies \mathbb{B}T = \mathrm{codisc}(W)$
- $T = \mathcal{D}_\omega^B$ for Boolean Algebra $B \implies \mathbb{B}T = \mathrm{disc}(\mathrm{Spec}(B))$

2 Games from Algebraic Signatures

Our intuition for $\mathbb{B}T$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

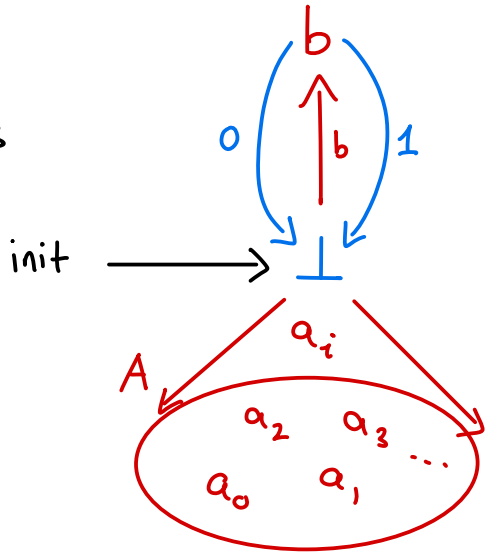
Consider the following alternating 2-player game $\mathcal{G}_A T$:

nodes are positions

edges are moves

= Program

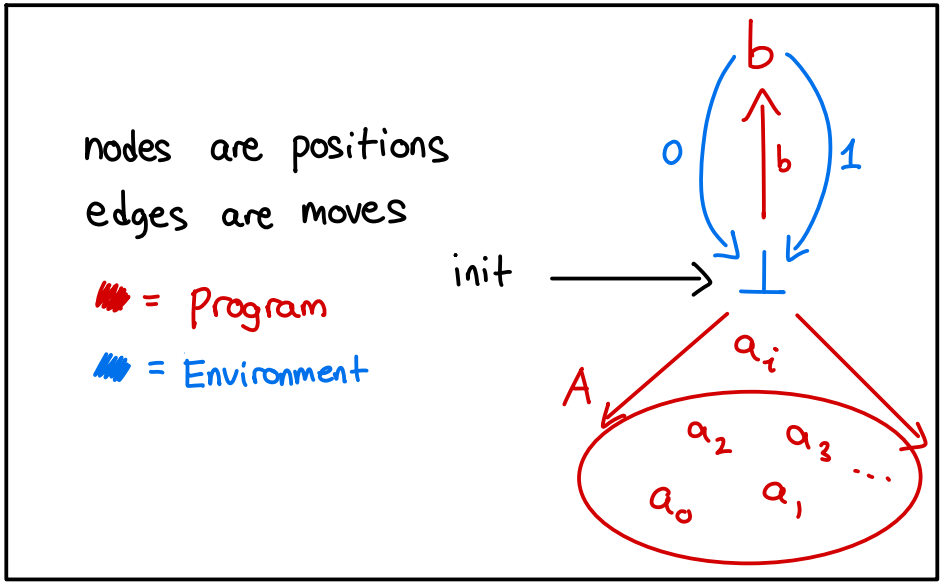
= Environment



Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

Consider the following alternating 2-player game $g_A T$:



winning strategies for program:

$\xrightarrow{b}$

Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

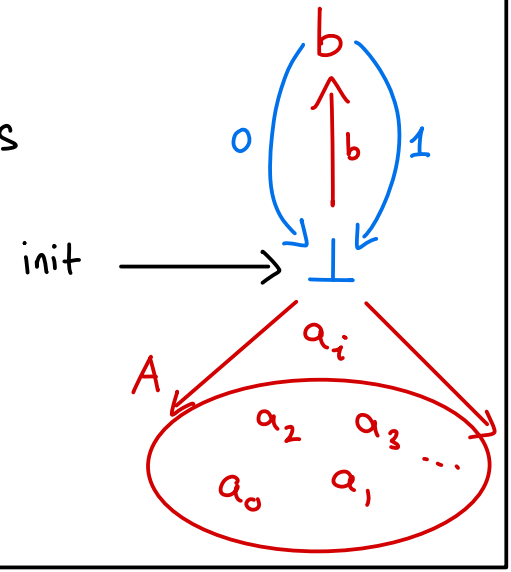
Consider the following alternating 2-player game g_A^T:

nodes are positions

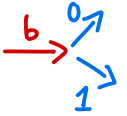
edges are moves

= Program

= Environment



winning strategies for program:



Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

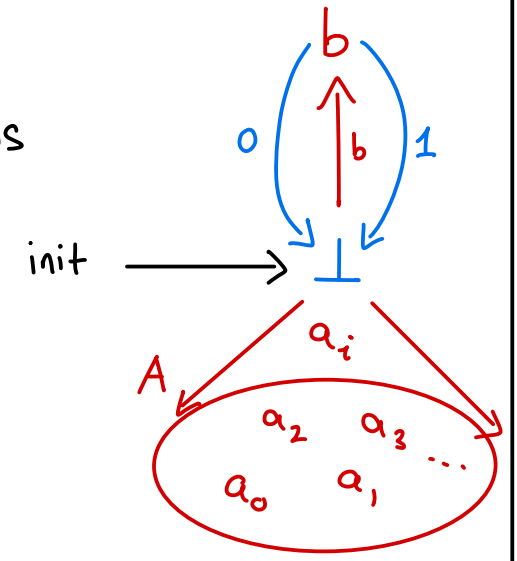
Consider the following alternating 2-player game $\mathcal{G}_A T$:

nodes are positions

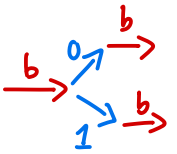
edges are moves

= Program

= Environment



winning strategies for program:



Our intuition for $\mathbb{B}T$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

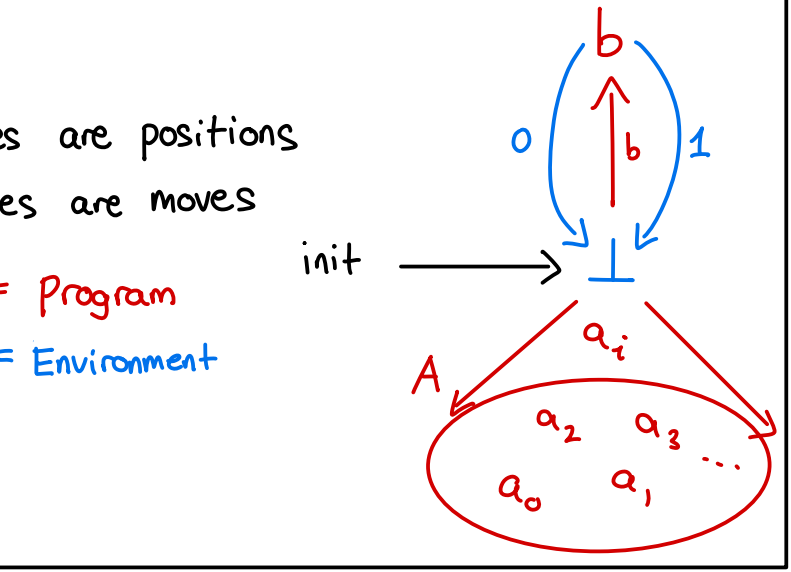
Consider the following alternating 2-player game $g_A T$:

nodes are positions

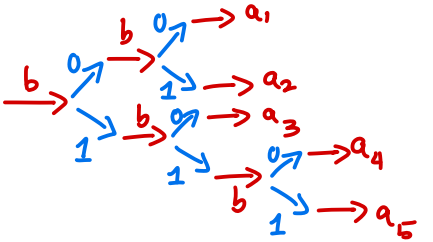
edges are moves

= Program

= Environment



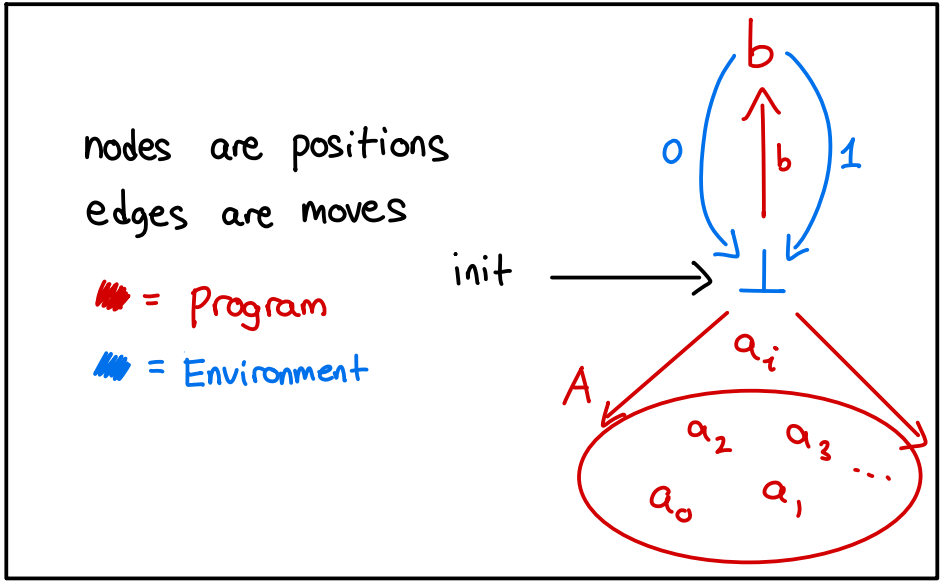
winning strategies for program:



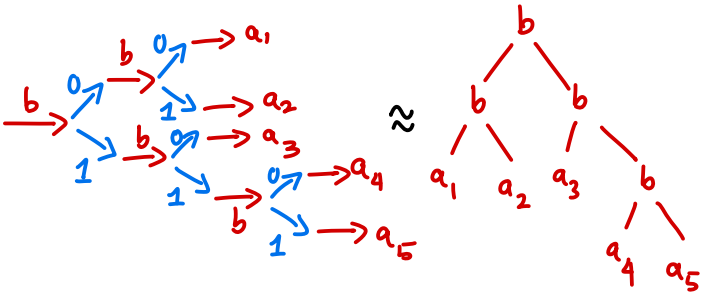
Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

Consider the following alternating 2-player game $g_A T$:



Winning strategies for program:

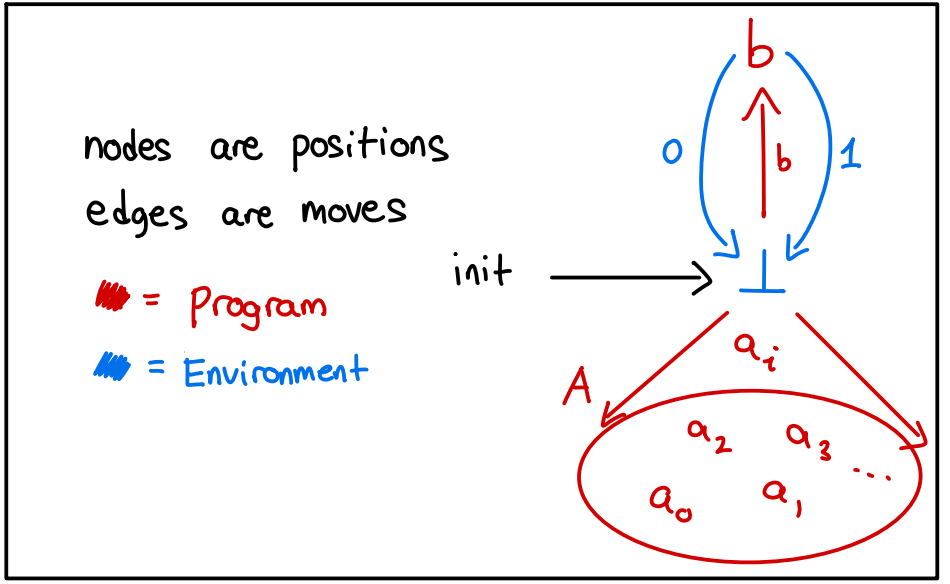


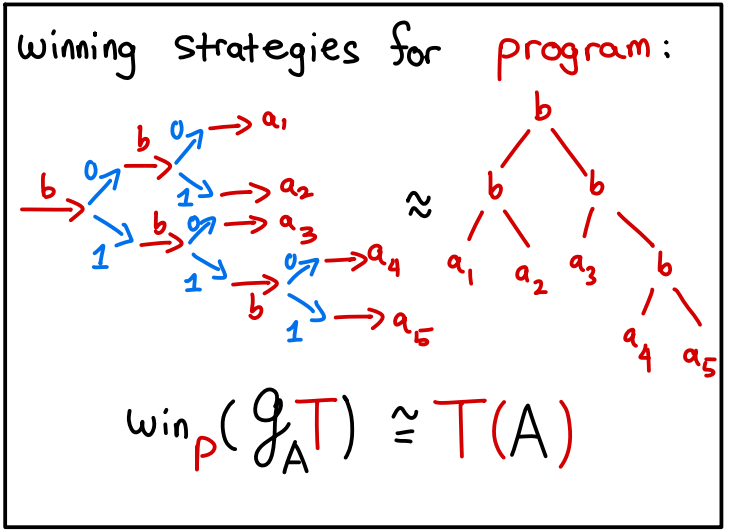
$$\mathrm{win}_P(g_A T) \cong T(A)$$

Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

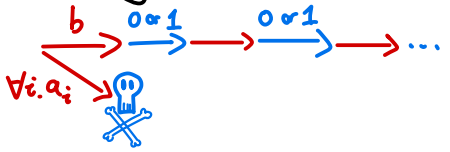
Example Let A be a set, $T = [b/2 \mid \emptyset]$.

Consider the following alternating 2-player game $g_A T$:





"winning" strategies for Environment:



Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

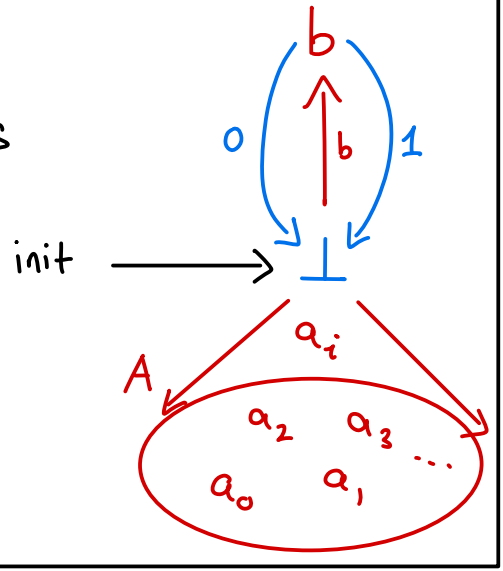
Consider the following alternating 2-player game $\mathcal{G}_A T$:

nodes are positions

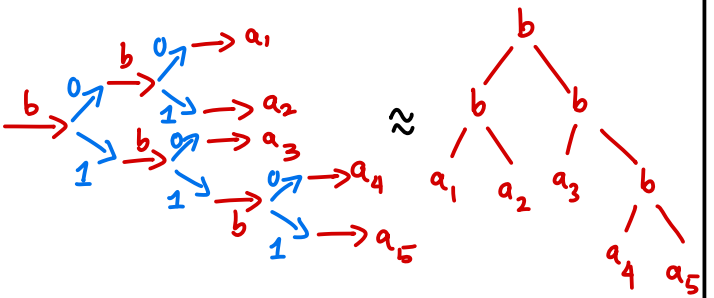
edges are moves

= Program

= Environment

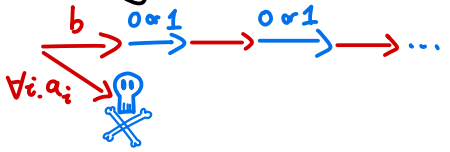


winning strategies for program:



$$\mathrm{win}_P(\mathcal{G}_A T) \cong T(A)$$

"winning" strategies for Environment:



$$\mathrm{win}_E(\mathcal{G}_\emptyset T) \cong 2^\omega$$

Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

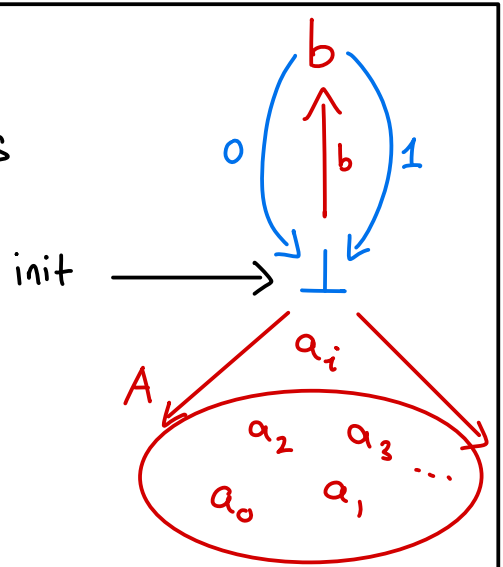
Consider the following alternating 2-player game $\mathcal{G}_A T$:

nodes are positions

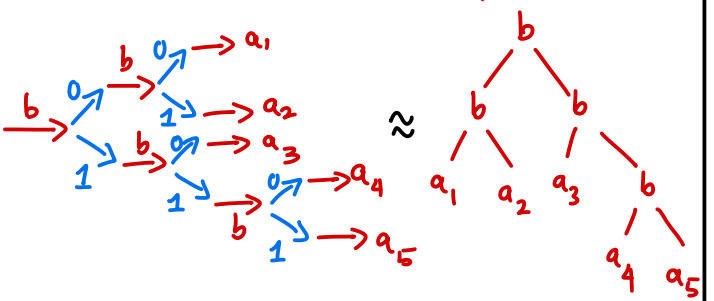
edges are moves

= Program

= Environment

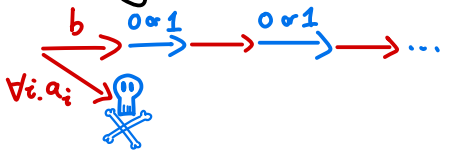


Winning strategies for program:



$$\mathrm{win}_P(\mathcal{G}_A T) \cong T(A)$$

"Winning" strategies for Environment:



$$\mathrm{win}_E(\mathcal{G}_\emptyset T) \cong 2^\omega$$

Let $\beta \in \mathrm{win}_E(\mathcal{G}_\emptyset T)$, $t \in \mathrm{win}_P(\mathcal{G}_A T)$

Our intuition for $\mathbb{BT}$ is very interactive... Can we formalize this?

Original Idea found in [Koenig 2021] but also Richard + Soichiro?

Example Let A be a set, $T = [b/2 \mid \emptyset]$.

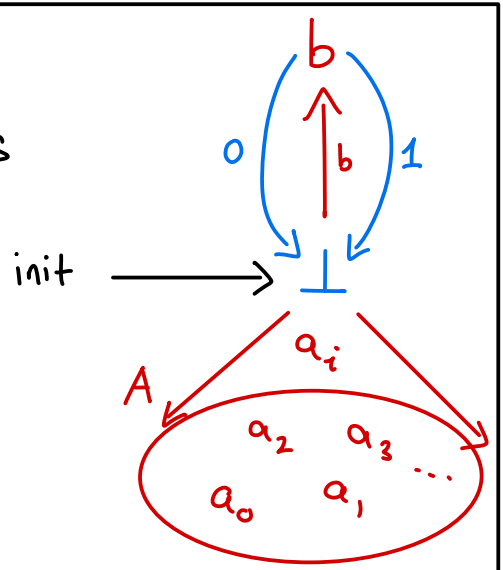
Consider the following alternating 2-player game $\mathcal{G}_A T$:

nodes are positions

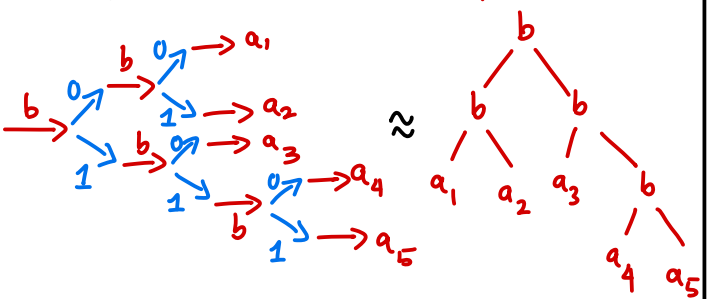
edges are moves

= Program

= Environment

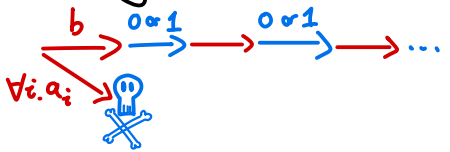


Winning strategies for program:



$$\mathrm{win}_P(\mathcal{G}_A T) \cong T(A)$$

"winning" strategies for Environment:



$$\mathrm{win}_E(\mathcal{G}_\emptyset T) \cong 2^\omega$$

Let $\beta \in \mathrm{win}_E(\mathcal{G}_\emptyset T)$, $t \in \mathrm{win}_P(\mathcal{G}_A T)$

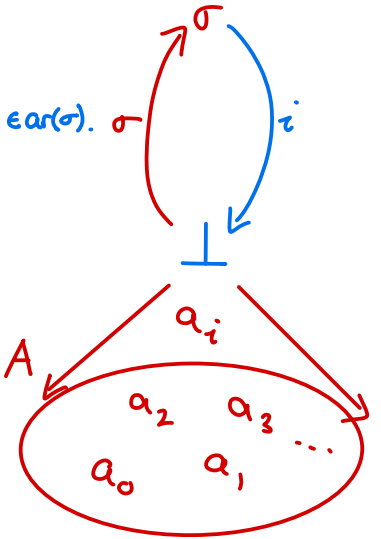
then $\langle \beta \mid t \rangle \approx ([t]_\beta, \beta(t))$

$\mathrm{Hom}_{\mathbb{BT}}(\beta, -) \cong \{\langle \beta \mid t \rangle \mid t \in T1\}$

In general, for $T = [\Sigma \mid \emptyset]$:

definition

$$q_A \Sigma = \forall \sigma . \forall i \in ar(\sigma) .\ \sigma\ i$$



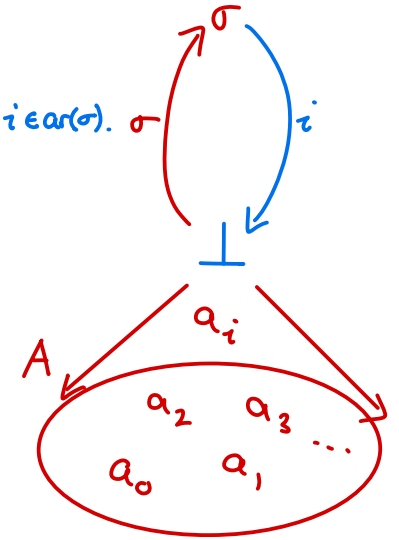
= Program

= Environment

In general, for $T = [\Sigma \mid \emptyset]$:

definition

$$\mathcal{G}_A \Sigma = \quad \forall \sigma . \forall i \in ar(\sigma) .$$



= Program

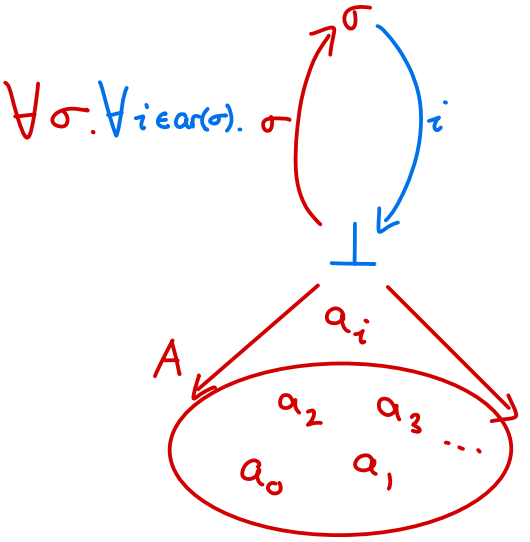
= Environment

remark if $z/0 \in \Sigma$ then $win_E(\mathcal{G}_\emptyset \Sigma) \cong \emptyset$.
but $\mathcal{G}_\emptyset \Sigma$ is not trivial!

In general, for $T = [\Sigma \mid \emptyset]$:

definition

$$\mathcal{G}_A\Sigma =$$



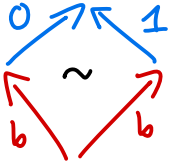
= Program

= Environment

remark if $z/0 \in \Sigma$ then $\mathrm{win}_E(\mathcal{G}_\emptyset\Sigma) \cong \emptyset$.
but $\mathcal{G}_\emptyset\Sigma$ is not trivial!

question What about $[b/2 \mid b(x,y) = b(y,x)]$?

Intuitively, want to add 2-cell:

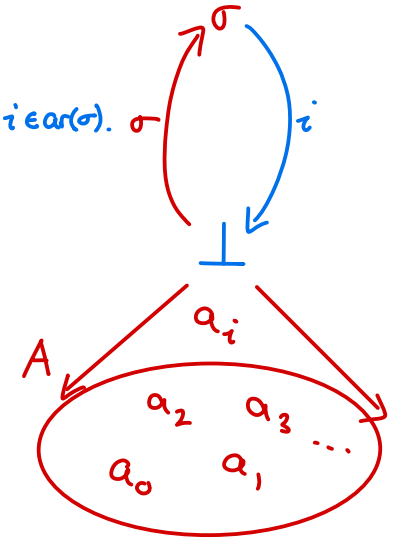


$$\{\beta \in \mathrm{win}_E(\mathcal{G}_\emptyset b) \mid \beta \sim \beta\} = \emptyset$$

In general, for $T = [\Sigma \mid \emptyset]$:

definition

$$g_A\Sigma = \quad \forall \sigma.\forall_{i \in ar(\sigma)}.$$



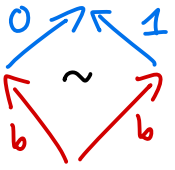
= Program

= Environment

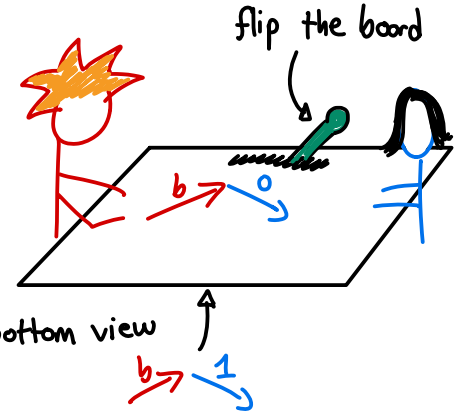
remark if $z/0 \in \Sigma$ then $win_E(g_\emptyset\Sigma) \cong \emptyset$.
but $g_\emptyset\Sigma$ is not trivial!

question What about $[b/2 \mid b(x,y) = b(y,x)]$?

Intuitively, want to add 2-cell:



$$\{\beta \in win_E(g_\emptyset{}^b) \mid \beta \sim \beta\} = \emptyset$$



The most "useless" definition of games

"template"

defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{t} \in Cat(\mathcal{C})$.

The most "useless" definition of games

"template"

defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{T} \in Cat(\mathcal{C})$.

The double category of **template games** is the double slice

$$\$ames(\mathbb{T}) = \$pan(\mathcal{C})/_{\mathbb{T}}$$

viewing $\mathbb{T}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{T}} \$pan(\mathcal{C})$.

The most "useless" definition of games

"template"

defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{T} \in Cat(\mathcal{C})$.

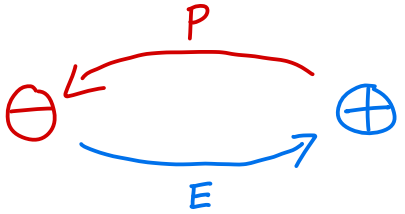
The double category of template games is the double slice

$$\mathbb{G}ames(\mathbb{T}) = \mathbb{S}pan(\mathcal{C})/_{\mathbb{T}}$$

viewing $\mathbb{T}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{T}} \mathbb{S}pan(\mathcal{C})$.

Of interest to us: $\mathcal{C} = Cat$,

$\mathbb{T}_\circ$ (freely generated by)



The most "useless" definition of games

"template"

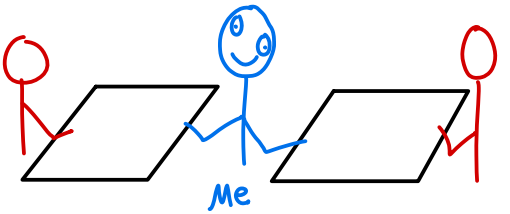
defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{T} \in \mathrm{Cat}(\mathcal{C})$.

The double category of template games is the double slice

$$\mathbb{G}\mathrm{ames}(\mathbb{T}) = \mathbb{S}\mathrm{pan}(\mathcal{C})/_{\mathbb{T}}$$

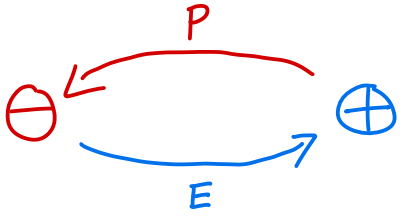
viewing $\mathbb{T}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{T}} \mathbb{S}\mathrm{pan}(\mathcal{C})$.

Of interest to us: $\mathcal{C} = \mathrm{Cat}$,

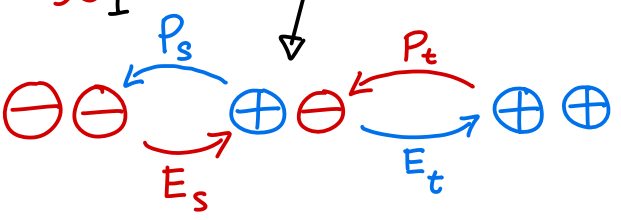


"war on two fronts"

$\mathbb{T}_0$ (freely generated by)



$\mathbb{T}_1$



The most "useless" definition of games

"template"

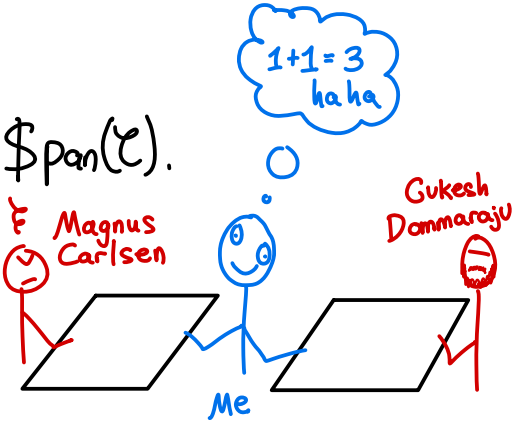
defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{t} \in \mathrm{Cat}(\mathcal{C})$.

The double category of *template games* is the double slice

$$\mathbb{G}\mathrm{ames}(\mathbb{t}) = \mathbb{S}\mathrm{pan}(\mathcal{C})/\mathbb{t}$$

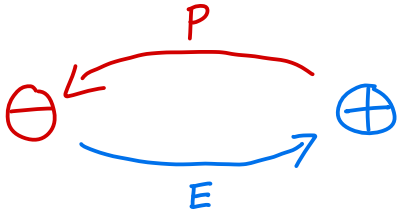
viewing $\mathbb{t}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{t}} \mathbb{S}\mathrm{pan}(\mathcal{C})$.

Of interest to us: $\mathcal{C} = \mathrm{Cat}$,

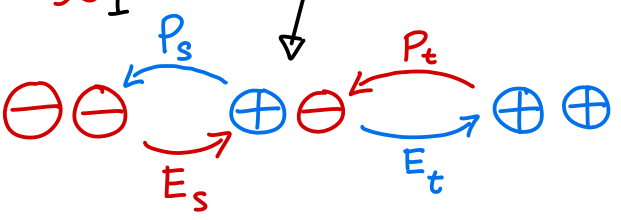


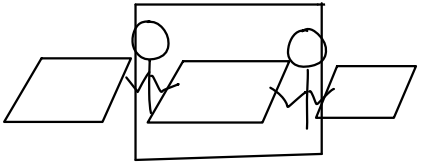
"war on two fronts"

$\mathbb{t}_0$ (freely generated by)



$\mathbb{t}_1$





Parallel Composition + hiding

The most "useless" definition of games

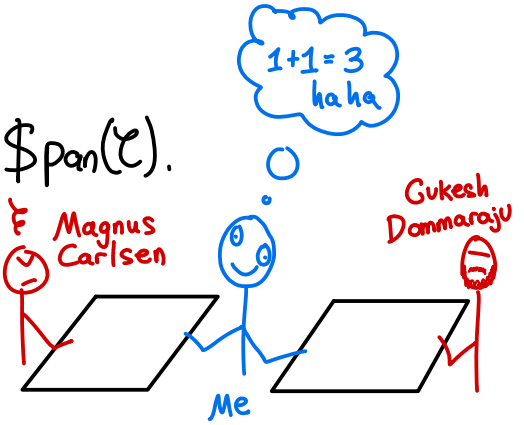
"template"

defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{t} \in \mathrm{Cat}(\mathcal{C})$.

The double category of *template games* is the double slice

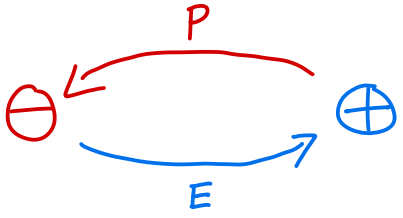
$$\mathbb{G}\mathrm{ames}(\mathbb{t}) = \mathbb{S}\mathrm{pan}(\mathcal{C})/\mathbb{t}$$

viewing $\mathbb{t}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{t}} \mathbb{S}\mathrm{pan}(\mathcal{C})$.

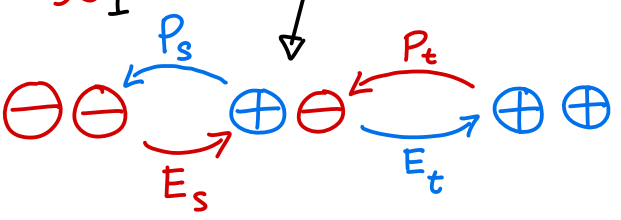


Of interest to us: $\mathcal{C} = \mathrm{Cat}$, "war on two fronts"

$\mathbb{t}_0$ (freely generated by)



$\mathbb{t}_1$



The most "useless" definition of games

"template"

defn [Mellies 2019] Let $\mathcal{C}$ category with finite limits, and $\mathbb{t} \in \mathrm{Cat}(\mathcal{C})$.

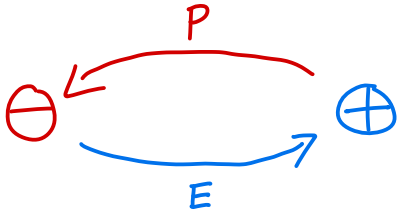
The double category of *template games* is the double slice

$$\mathbb{G}\mathrm{ames}(\mathbb{t}) = \mathbb{S}\mathrm{pan}(\mathcal{C})/\mathbb{t}$$

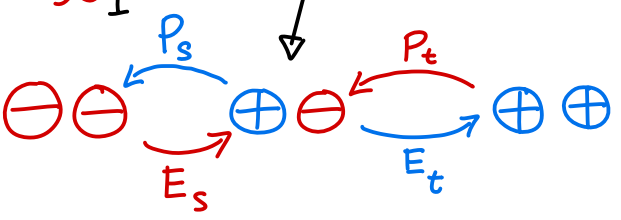
viewing $\mathbb{t}$ as a monad, i.e. lax double functor $1 \xrightarrow{\mathbb{t}} \mathbb{S}\mathrm{pan}(\mathcal{C})$.

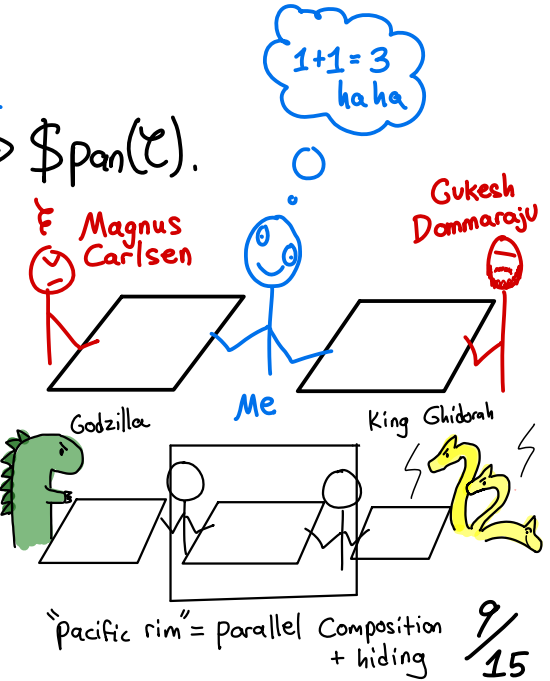
Of interest to us: $\mathcal{C} = \mathrm{Cat}$, "war on two fronts"

$\mathbb{t}_0$ (freely generated by)



$\mathbb{t}_1$





Less "useless" definition

- objects are games $\left(G \in \mathrm{Cat}, G \xrightarrow{\lambda_G} \not{\pi}_0\right)$

Less "useless" definition

- objects are games $\left(G \in \mathrm{Cat}, G \xrightarrow{\lambda_G} \mathcal{X}_0\right)$
- tight maps are reductions $G \xrightarrow{f} G'$

Less "useless" definition

- objects are games $\left(G \in \mathrm{Cat},\ G \xrightarrow{\lambda_G} \mathbb{t}_0\right)$
- tight maps are reductions $G \xrightarrow{f} G'$ (commuting over $\mathbb{t}_0$)
- loose maps are strategies $G_1 \xrightarrow{(S,\lambda,s,t)} G_2$

$$\begin{array}{ccccc} G_1 & \xleftarrow{s} & S & \xrightarrow{t} & G_2 \\ \downarrow & & \downarrow & & \downarrow \\ \mathbb{t}_0 & \longleftarrow & \mathbb{t}_1 & \longrightarrow & \mathbb{t}_0 \end{array}$$

Less "useless" definition

- objects are games $\left(G \in \mathrm{Cat}, G \xrightarrow{\lambda_G} \mathcal{X}_0\right)$
- tight maps are reductions $G \xrightarrow{f} G'$ over $\mathcal{X}_0$
- loose maps are strategies $G_1 \xrightarrow{(S, \lambda, s, t)} G_2$

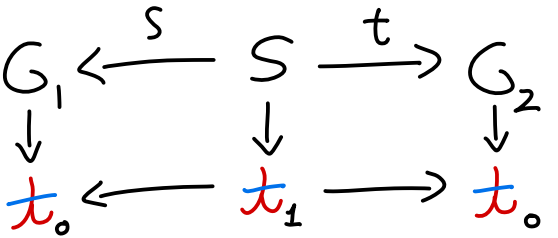
$$\begin{array}{ccccc} G_1 & \xleftarrow{s} & S & \xrightarrow{t} & G_2 \\ \downarrow & & \downarrow & & \downarrow \\ \mathcal{X}_0 & \longleftarrow & \mathcal{X}_1 & \longrightarrow & \mathcal{X}_0 \end{array}$$

- 2-cells are $S \to S'$ over $\mathcal{X}_1$ making bla bla bla commute.

Less "useless" definition

- objects are games $(G \in \mathrm{Cat}, G \xrightarrow{\lambda_G} \mathcal{A}_0)$
- tight maps are reductions $G \xrightarrow{f} G'$ over $\mathcal{A}_0$
- loose maps are strategies $G_1 \xrightarrow{(S, \lambda, s, t)} G_2$
- 2-cells are $S \to S'$ over $\mathcal{A}_1$ making bla bla bla commute.

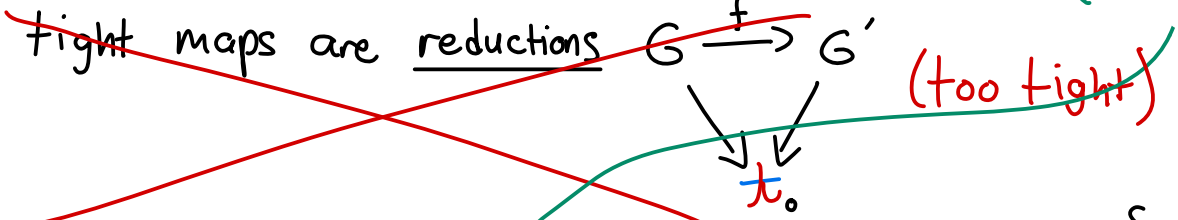
Note that this is very abstract and doesn't demand the usual constraints (winning, deterministic, responsive)



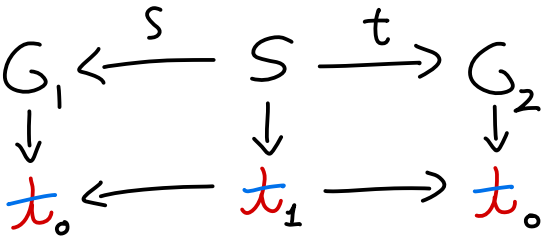
Less "useless" definition

Note that this is very abstract and doesn't demand the usual constraints (winning, deterministic, responsive)

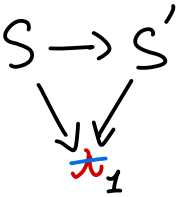
- objects are games $(G \in \mathrm{Cat}, G \xrightarrow{\lambda_G} \mathbb{t}_0)$
- ~~tight maps are reductions~~ $G \xrightarrow{f} G'$ (too tight)



- loose maps are strategies $G_1 \xrightarrow{(S,\lambda,s,t)} G_2$



- 2-cells are



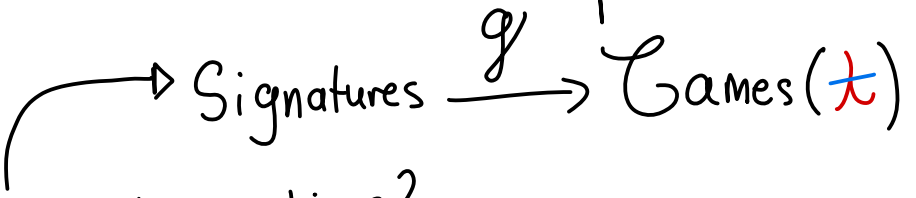
making bla bla bla commute.

$\mathbb{G}\mathrm{ames}(\mathbb{t})$ bicategory of template games.

We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\not{\lambda})$$

We want to define a map



$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

We want to define a map

$$\text{Signatures} \xrightarrow{g} \text{Games}(\not{\lambda})$$

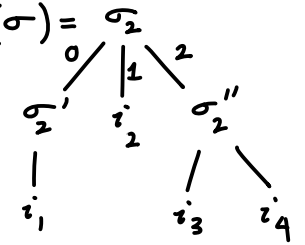
what are the morphisms?

$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

$g\,\varphi$

where (for example)

$\varphi(\sigma) =$



and $i_j \in ar(\sigma)$

We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\not\perp)$$

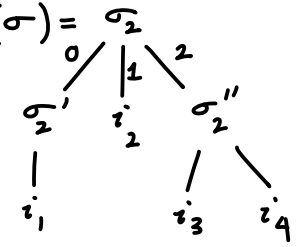
what are the morphisms?

$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

$\mathcal{G}\varphi$

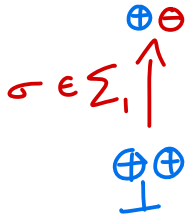
where (for example)

$\varphi(\sigma) =$



and $i_j \in ar(\sigma)$

$\sigma \in \Sigma_1$



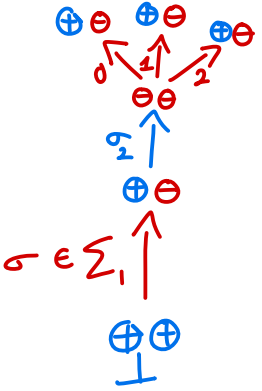
We want to define a map

Signatures $\xrightarrow{g}$ Games($\not\perp$)

what are the morphisms?

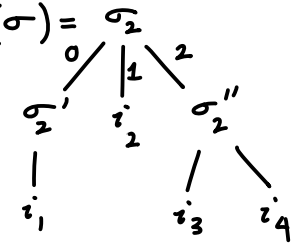
$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

$g\,\varphi$



where (for example)

$\varphi(\sigma) =$



and $i_j \in ar(\sigma)$

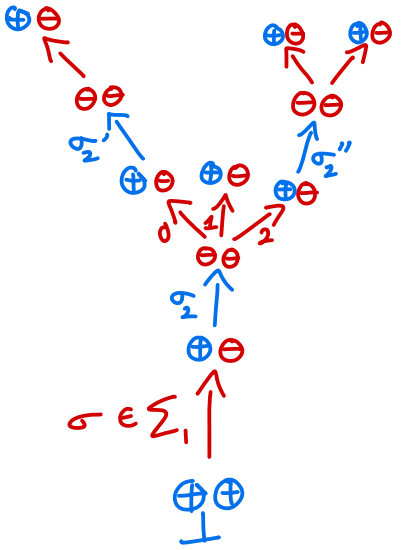
We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\mathcal{A})$$

what are the morphisms?

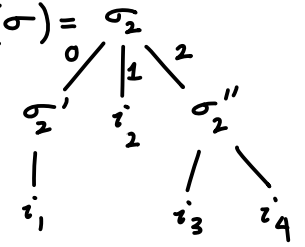
$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

$\mathcal{G}\varphi$



where (for example)

$\varphi(\sigma) =$



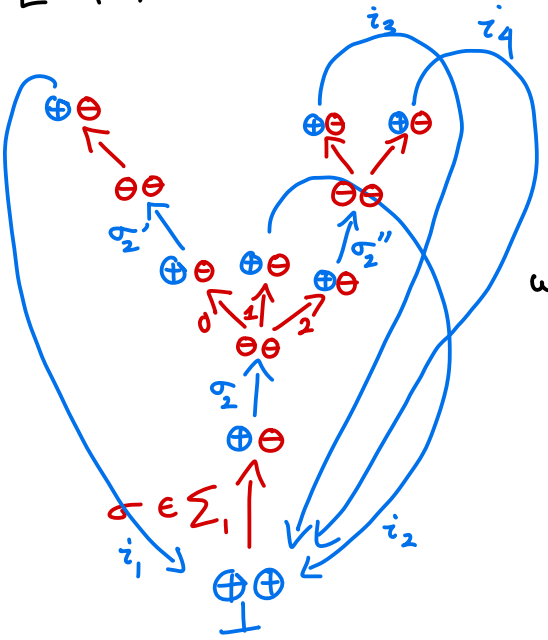
and $i_j \in ar(\sigma)$

We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\mathcal{T})$$

what are the morphisms?

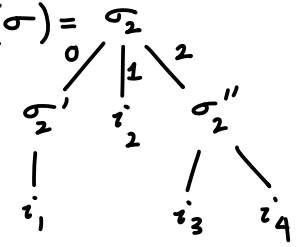
$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map



$\mathcal{G}\varphi$

where (for example)

$\varphi(\sigma) =$



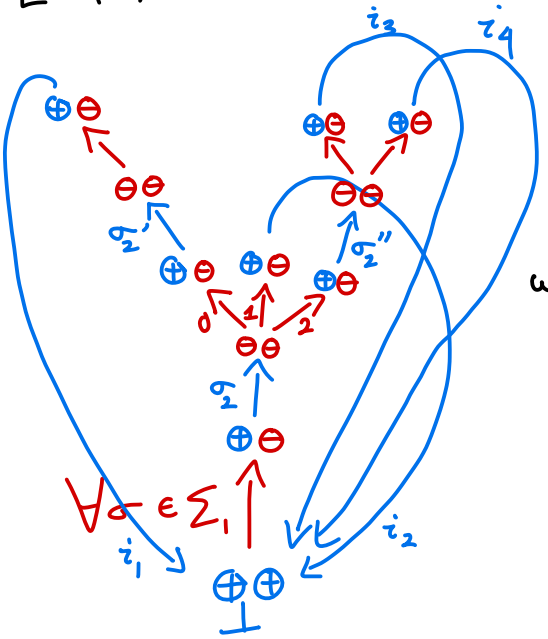
and $i_j \in ar(\sigma)$

We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\mathcal{T})$$

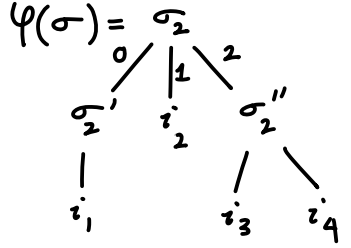
what are the morphisms?

$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map



$\mathcal{G}\varphi$

where (for example)

$\varphi(\sigma) =$ 

and $i_j \in ar(\sigma)$

But also:

$[\Sigma_1 | \emptyset]$-comodel (W, ρ)

where $\rho : TA \longrightarrow (W \times A)^W$

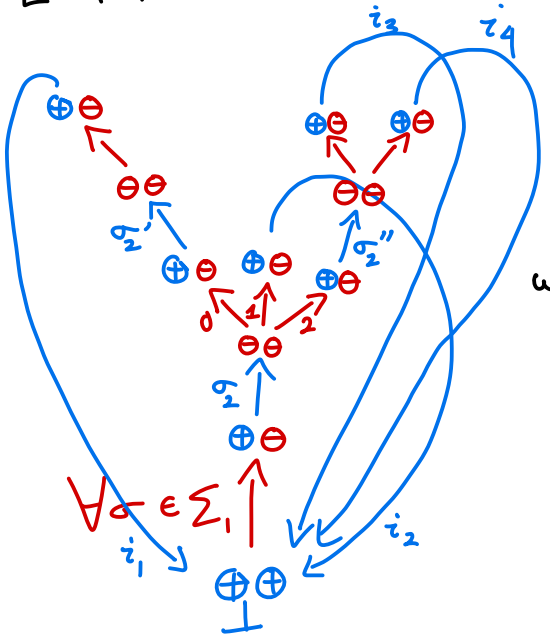
$\rho(\sigma)(w) = (w', i)$

We want to define a map

Signatures $\xrightarrow{\mathcal{G}}$ Games($\mathcal{T}$)

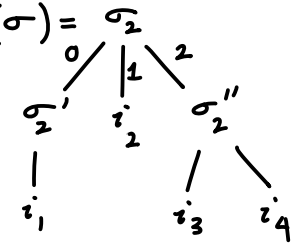
what are the morphisms?

$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map



$\mathcal{G}\varphi$

where (for example)

$\varphi(\sigma) =$ 

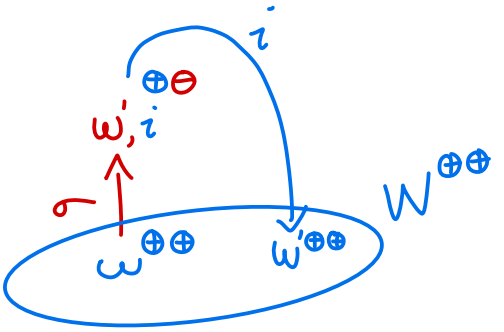
and $i_j \in ar(\sigma)$

But also:

$[\Sigma_1 | \emptyset]$-comodel (W, ρ)

where $\rho : TA \longrightarrow (W \times A)^W$

$\rho(\sigma)(w) = (w', i)$



We want to define a map

$$\text{Signatures} \xrightarrow{\mathcal{G}} \text{Games}(\mathcal{T})$$

what are the morphisms?

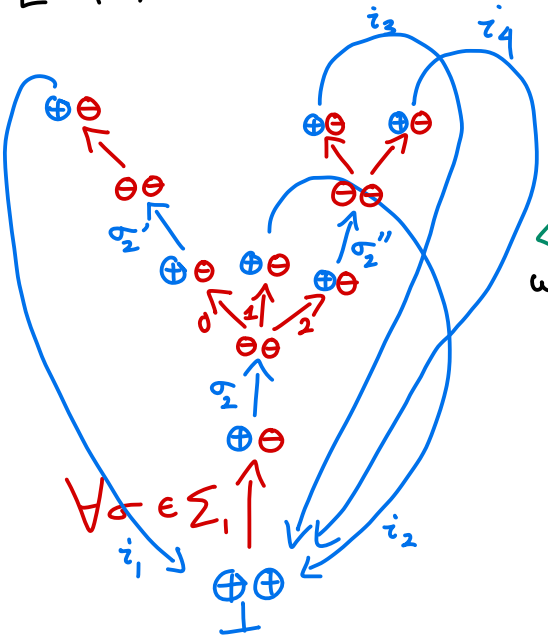
$[\Sigma_1 | \emptyset] \xrightarrow{\varphi} [\Sigma_2 | \emptyset]$ monad map

But also:

$[\Sigma_1 | \emptyset]$-comodel (W, ρ)

where $\rho : TA \longrightarrow (W \times A)^W$

There is $\rho(\sigma)(w) = (w', i)$

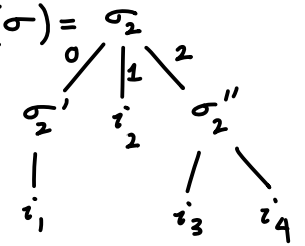


$\mathcal{G}\varphi$

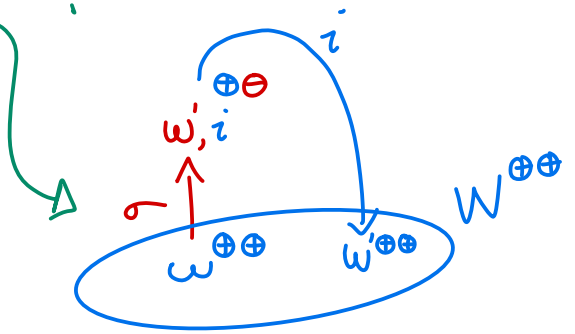
a common picture!

where (for example)

$\varphi(\sigma) =$



and $i_j \in ar(\sigma)$



The bicategory $\mathcal{G}ig$

- Objects are signatures Σ

The bicategory $\mathcal{Sig}$

- Objects are signatures Σ
- maps $\Sigma_2 \xrightarrow{(W,\rho)} \Sigma_1$ are Σ_2-residual Σ_1-comodels [Ahman-Bauer 2020] Garner 2023

$$\rho(\sigma \in \Sigma_1) : W \longrightarrow T_{\Sigma_2}(W \times ar(\sigma))$$

The bicategory $\mathcal{S}ig$

- Objects are signatures Σ
- maps $\Sigma_2 \xrightarrow{(W,\rho)} \Sigma_1$ are Σ_2-residual Σ_1-comodels [Ahman-Bauer 2020] Garner 2023

$$\rho(\sigma \in \Sigma_1) : W \longrightarrow T_{\Sigma_2}(W \times ar(\sigma))$$

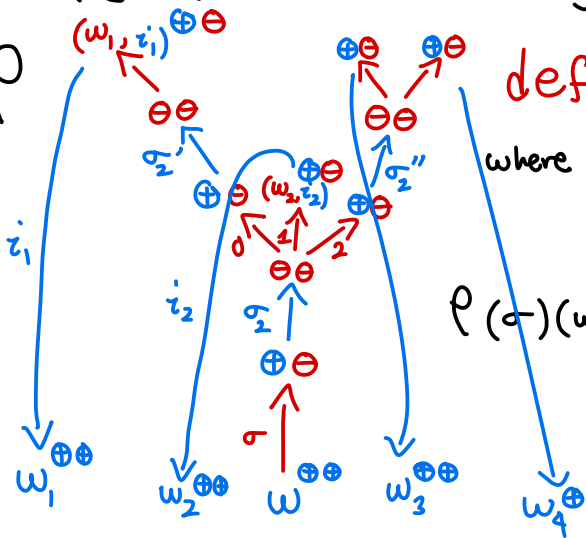
- 2-cells are $W \to W'$ making bla bla bla commute.

The bicategory $\mathcal{S}ig$

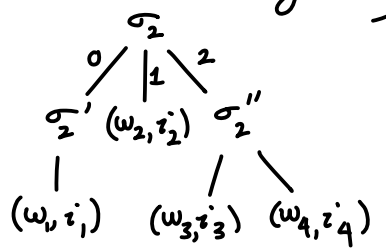
- Objects are signatures Σ
- maps $\Sigma_2 \xrightarrow{(W,\rho)} \Sigma_1$ are Σ_2-residual Σ_1-comodels [Ahman-Bauer 2020] Garner 2023

$$\rho(\sigma \in \Sigma_1) : W \longrightarrow T_{\Sigma_2}(W \times ar(\sigma))$$

- 2-cells are $W \to W'$ making bla bla bla commute.

prop $\mathcal{G}\rho$  defines a pseudofunctor

where (for example) $\mathcal{G} : \mathcal{S}ig \to \mathcal{G}ames(\mathcal{T})$

$\rho(\sigma)(\omega) =$ 

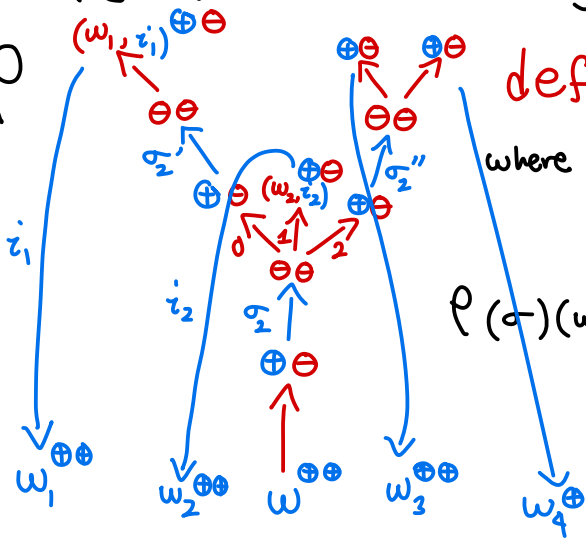
and $i_j \in ar(\sigma)$

The bicategory $\mathcal{S}ig$

- Objects are signatures Σ
- maps $\Sigma_2 \xrightarrow{(W,\rho)} \Sigma_1$ are Σ_2-residual Σ_1-comodels [Ahman-Bauer 2020] Garner 2023

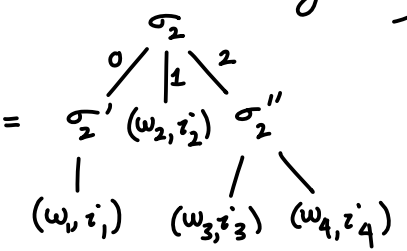
$$\rho(\sigma \in \Sigma_1) : W \longrightarrow T_{\Sigma_2}(W \times ar(\sigma))$$

- 2-cells are $W \to W'$ making bla bla bla commute.

prop $\mathcal{G}\rho$ 

monoidal defines a pseudofunctor

where (for example) $\mathcal{G} : \mathcal{S}ig \to \mathcal{G}ames(\mathcal{T})$

$\rho(\sigma)(\omega) =$ 

and $i_j \in ar(\sigma)$

The bicategory $\mathcal{S}ig$

- Objects are signatures Σ
- maps $\Sigma_2 \xrightarrow{(W,\rho)} \Sigma_1$ are Σ_2-residual Σ_1-comodels [Ahman-Bauer 2020] Garner 2023

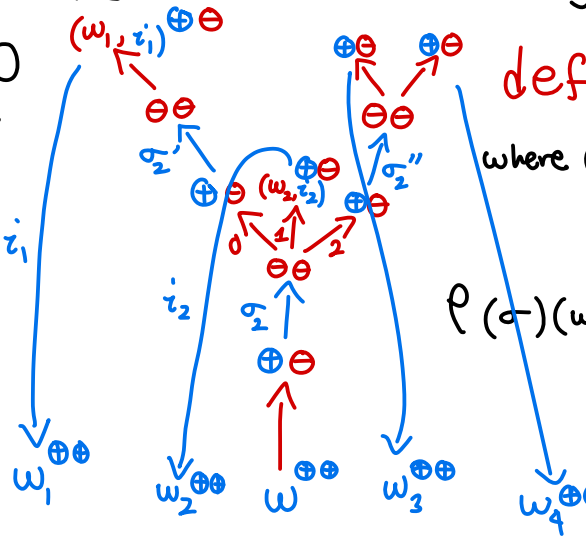
$$\rho(\sigma \in \Sigma_1) : W \longrightarrow T_{\Sigma_2}(W \times ar(\sigma))$$

- 2-cells are $W \to W'$ making bla bla bla commute.

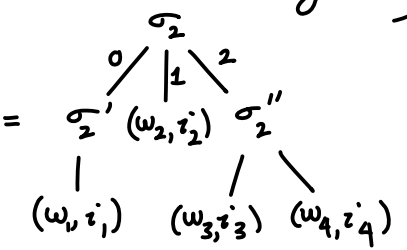
remark these are particular kinds of monad bimodules

prop $\mathcal{G}\rho$ defines a monoidal pseudofunctor where (for example)

$$\mathcal{G} : \mathcal{S}ig \to \mathcal{G}ames(\not{1})$$



$$\rho(\sigma)(\omega) =$$



and $i_j \in ar(\sigma)$

We want to characterize the <u>essential image</u> of

$$\mathcal{G} : \mathcal{G}ig \longrightarrow \mathcal{G}ames(\mathcal{X})$$

We want to characterize the essential image of

$$\mathcal{G} : \mathcal{S}ig \longrightarrow \mathcal{G}ames(\pm)$$

① Σ induces always a game with one $\oplus$ state.
Other than this, is fully general (up to "bisimulation").

We want to characterize the essential image of

$$\mathcal{G} : \mathcal{S}ig \longrightarrow \mathcal{G}ames(\mathcal{X})$$

① Σ induces always a game with one $\oplus$ state.
Other than this, is fully general (up to "bisimulation").

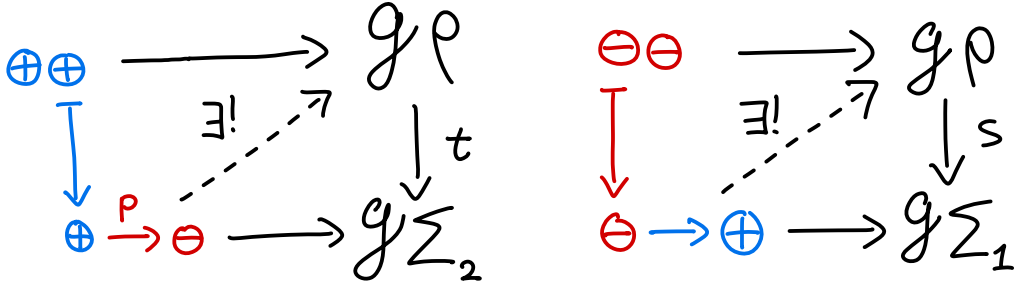
② $\Sigma_1 \overset{(W,P)}{\nrightarrow} \Sigma_2$ induces strategies which are deterministic and "winning"

We want to characterize the <u>essential image</u> of

$$\mathcal{G} : \mathcal{G}ig \longrightarrow \mathcal{G}ames(\dagger)$$

① Σ induces always a game with one $\oplus$ state.
Other than this, is fully general (up to "bisimulation").

② $\Sigma_1 \overset{(W,P)}{\nrightarrow} \Sigma_2$ induces strategies which are <u>deterministic</u> and <u>"winning"</u>

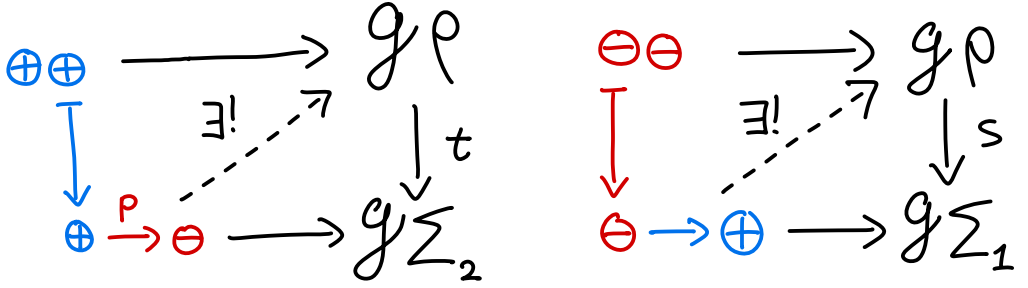


We want to characterize the <u>essential image</u> of

$$g : \mathcal{G}ig \longrightarrow \mathcal{G}ames(\pm)$$

① Σ induces always a game with one $\oplus$ state.
Other than this, is fully general (up to "bisimulation").

② $\Sigma_1 \overset{(W,P)}{\nrightarrow} \Sigma_2$ induces strategies which are <u>deterministic</u> and <u>"winning"</u>



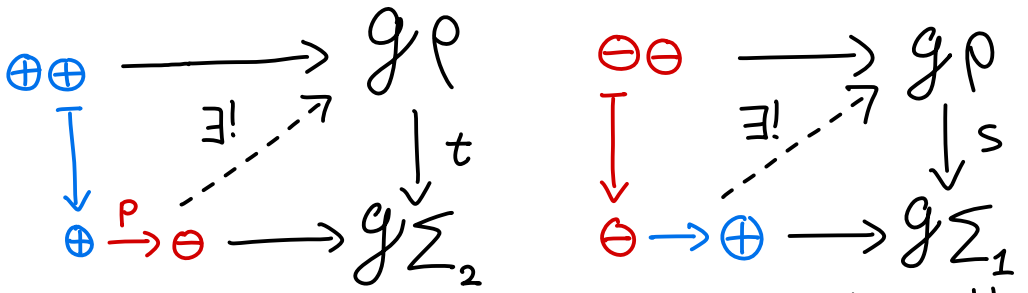
For this, need to attach data of winning infinite plays
i.e.
$\mathcal{G}ames(\pm)(0^{\oplus} \to 1^{\ominus} \to 2^{\oplus} \to 3^{\ominus} \ldots, G) \to \{win, lose\}$

We want to characterize the essential image of

$$\mathcal{G} : \mathcal{G}ig \longrightarrow \mathcal{G}ames(\not{\tau})$$

① Σ induces always a game with one $\oplus$ state.
Other than this, is fully general (up to "bisimulation").

② $\Sigma_1 \overset{(W,P)}{+\!\!\rightarrow} \Sigma_2$ induces strategies which are deterministic and "winning"

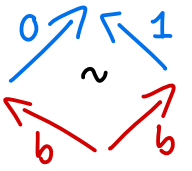


For this, need to attach data of winning infinite plays
i.e.
$\mathcal{G}ames(\not{\tau})(0^{\oplus} \to 1^{\ominus} \to 2^{\oplus} \to 3^{\ominus} \ldots, G) \to \{win, lose\}$

Question What structure do we need on the LHS to get adjoint construction to $\mathcal{G}$? Statefully indexed family of signatures? 13/15

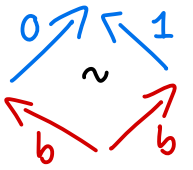
And what about the equations?

to model $b(x,y)=b(y,x)$, we intuitively wanted



And what about the equations?

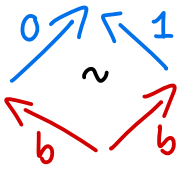
to model $b(x,y)=b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

And what about the equations?

to model $b(x,y)=b(y,x)$, we intuitively wanted



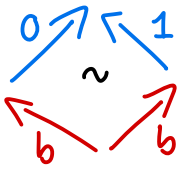
Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

But what about $b(x,y) = b(x,x)$?

This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

And what about the equations?

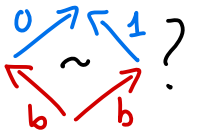
to model $b(x,y)=b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

But what about $b(x,y) = b(x,x)$?

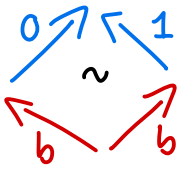
also



This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

And what about the equations?

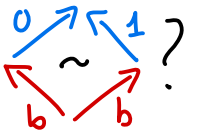
to model $b(x,y) = b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

But what about $b(x,y) = b(x,x)$?

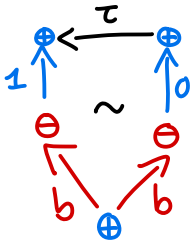
also



?

This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

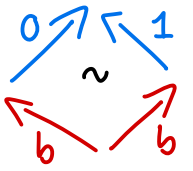
Idea 1 (Silent Transitions)



mark applications of ~ with a silent transition τ.

And what about the equations?

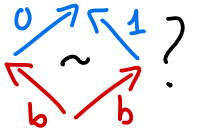
to model $b(x,y)=b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\text{Cat}, \boxtimes)$, replacing $(\text{Cat}, \times)$.

But what about $b(x,y) = b(x,x)$?

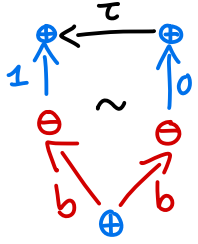
also



?

This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

Idea 1 (Silent Transitions)

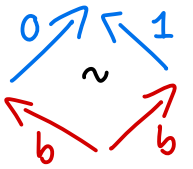


mark applications of ~ with a silent transition τ.

- How to generalize to other equations?
- Have to restrict to "noisy" β only.

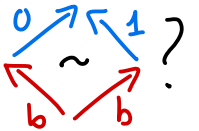
And what about the equations?

to model $b(x,y)=b(y,x)$, we intuitively wanted



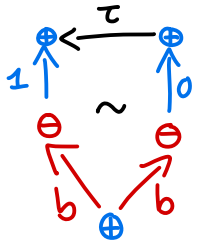
Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

But what about $b(x,y)=b(x,x)$? also ?



This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

Idea 1 (Silent Transitions)



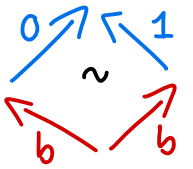
mark applications of ~ with a silent transition τ.

- How to generalize to other equations?
- Have to restrict to "noisy" β only.

Idea 2 (Replace 2Cat by Mon2Cat)

And what about the equations?

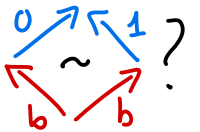
to model $b(x,y)=b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

But what about $b(x,y)=b(x,x)$?

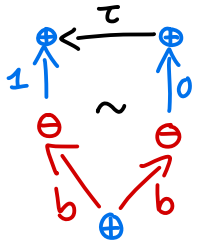
also



?

This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

Idea 1 (Silent Transitions)



mark applications of ~ with a silent transition τ.

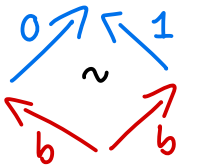
- How to generalize to other equations?
- Have to restrict to "noisy" β only.

Idea 2 (Replace 2Cat by Mon2Cat)

View $[\Sigma \mid \mathcal{E}]$ as a 3-computad generating a monoidal 2-category.

And what about the equations?

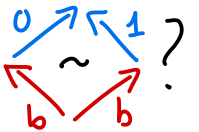
to model $b(x,y)=b(y,x)$, we intuitively wanted



Formally, [Mellies2021] tells us how to define games internal to $(2\mathrm{Cat}, \boxtimes)$, replacing $(\mathrm{Cat}, \times)$.

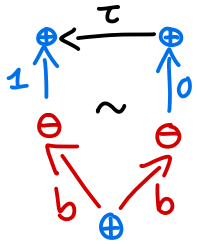
But what about $b(x,y)=b(x,x)$?

also



This selects for $\{0000\ldots\} \subseteq 2^{\omega}$.

Idea 1 (Silent Transitions)

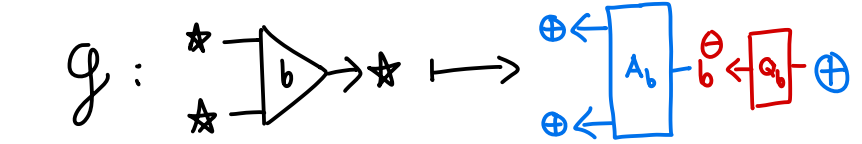


mark applications of ~ with a silent transition τ.

- How to generalize to other equations?
- Have to restrict to "noisy" β only.

Idea 2 (Replace 2Cat by Mon2Cat)

View $[\Sigma \mid \mathcal{E}]$ as a 3-computad generating a monoidal 2-category. Then



polarizes the 2-category.

And even further into the future

- Generalization beyond Monad on Set?

And even further into the future

- Generalization beyond Monad on Set?
- Establish functorial relationship with $\mathbb{B}\mathrm{T}$

And even further into the future

- Generalization beyond Monad on Set?
- Establish bifunctorial relationship with $\mathbb{BT}$

this can also be made bicategorical! (WIP)

retronatural transformations are 2-cells in $\mathbb{K}\ell_{\text{retro}}(\$\text{Pan}(\mathbb{T}))$

And even further into the future

- Generalization beyond Monad on Set?
- Establish bifunctorial relationship with $\mathbb{BT}$

this can also be made bicategorical! (WIP)

retronatural transformations are 2-cells in $\mathbb{K}\ell_{\text{retro}}(\$\text{Pan}(\mathbb{T}))$

- (semi-)relatedly, we have an adjunction

$$\text{Mnd}_{\mathbb{Z}}(\text{Set}) \underset{\perp}{\rightleftarrows} \text{Quantale}$$

$T0 \neq \emptyset$

Seems related to Cameron Calk's work on ω-Quantales...

What is the relationship?

And even further into the future

- Generalization beyond Monad on Set?
- Establish bifunctorial relationship with $\mathbb{BT}$

this can also be made bicategorical! (WIP)

retronatural transformations are 2-cells in $\mathbb{K}\ell_{\text{retro}}(\$\text{Pan}(\mathbb{T}))$

- (semi-)relatedly, we have an adjunction

$$\text{Mnd}_z(\text{Set}) \rightleftarrows \text{Quantale}$$

Seems related to Cameron Calk's work on ω-Quantales...

$T0 \neq \emptyset$

What is the relationship?

Functor of Points, sure, but Computational Meaning?

- What does a theory of game schemes look like?

And even further into the future

- Commutative AG is ultimately about solving polynomial zeros <=> polynomial eqns.

Polynomial ~ program syntax

where are $p = q$? ~ where are $P \simeq Q$? (Full Abstraction)

- Can we get presentation-invariant structures out of $\mathcal{G}[\Sigma|E]$?

BT is one example.

What about (co)homology?

- C^*-algebras are also often talked about in terms of localic groupoids. Can we get game spectra of presentations for C^*-algebras, and build a theory of non-commutative algebraic geometry? What is the relationship with Connes-style NCAG?

References

[Garner, R, Wu 2026] "Stone Duality for Monads"

[Koenig 2021] "Grounding Game Semantics in Categorical Algebra"

[Mellies 2019] "Categorical Combinatorics of Scheduling and Synchronization in Game Semantics"

[—11— 2019] "Template Games and differential linear logic"

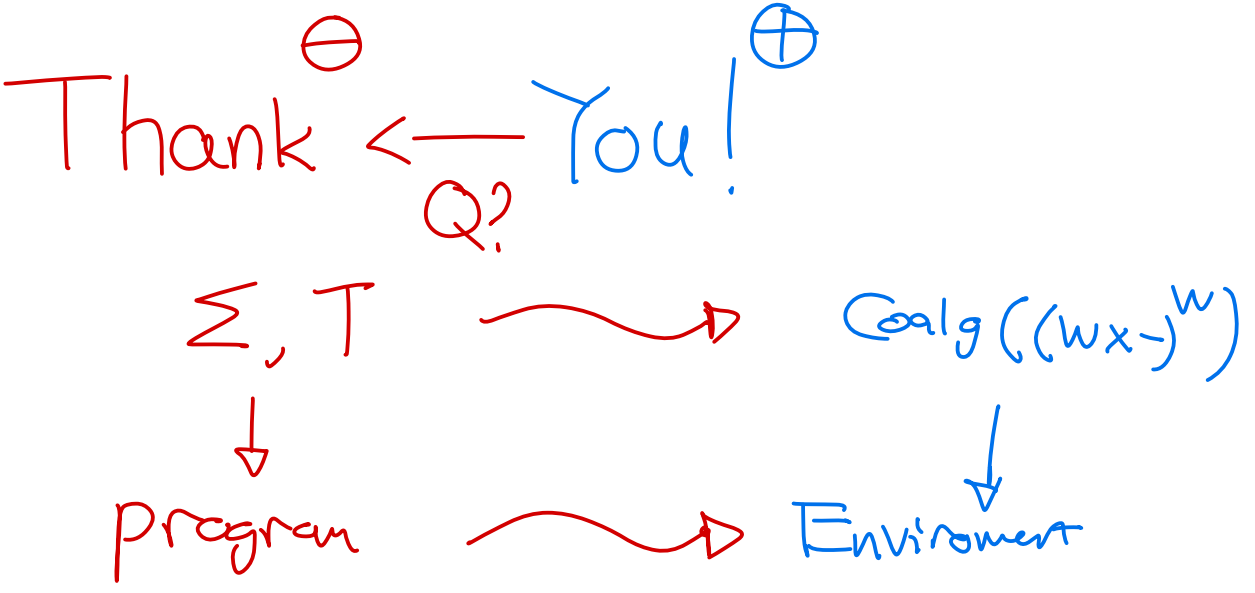
[—11— 2021] "Asynchronous Template Games and the Gray Tensor Product of 2-categories"

[Ahman, Bauer 2020] "Runners in Action"

[Garner 2023] "Stream Processors and comodels"

[Calk 2025] "Higher Catoids, Higher Quantales and their correspondences"

Thank ⊖ You! ⊕


Thank ⊖
You! ⊕
Q?
Σ, T
Coalg((Wx-)^W)
program
Environment