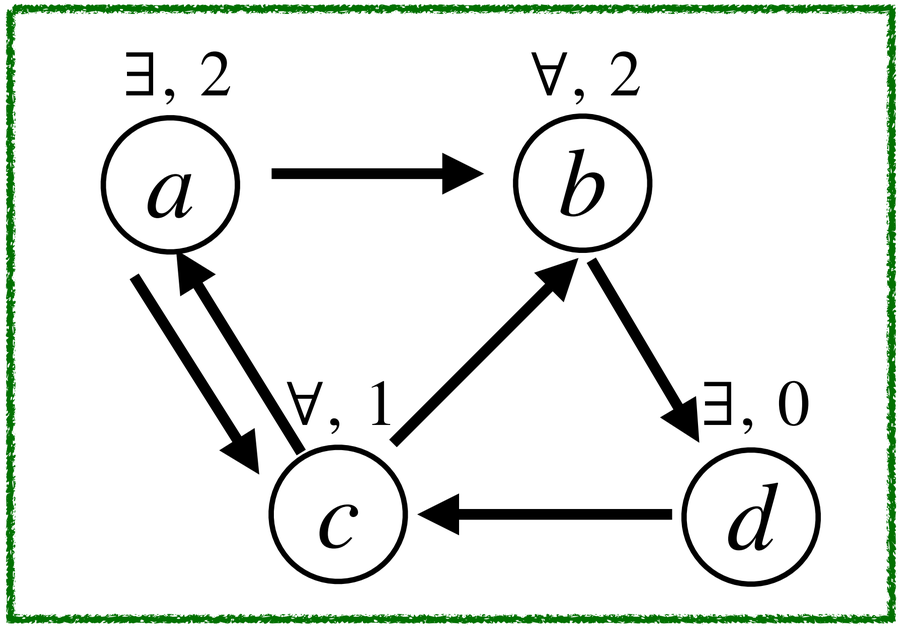
A Compositional Approach to Parity Games

Kazuki Watanabe[a, b] Clovis Eberhart[b, c] Kazuyuki Asada[d] Ichiro Hasuo[a, b, e]

a The Graduate University for Advanced Studies (SOKENDAI)
b National Institute of Informatics, Tokyo, Japan
c Japanese-French Laboratory of Informatics, IRL 3527, CNRS
d Tohoku University
e Ritsumeikan University

MFPS37, September 2021

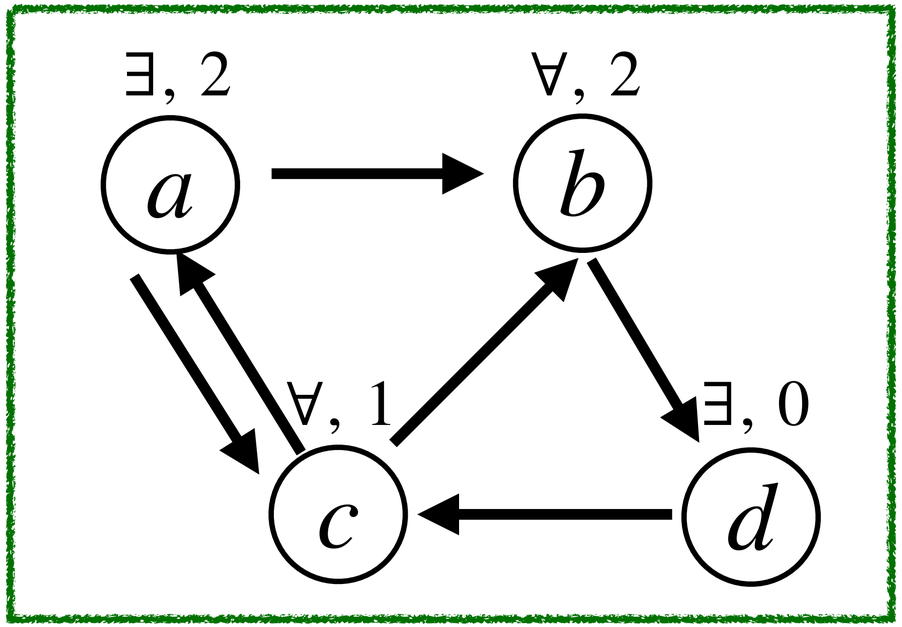
Parity game: verification method for model checking

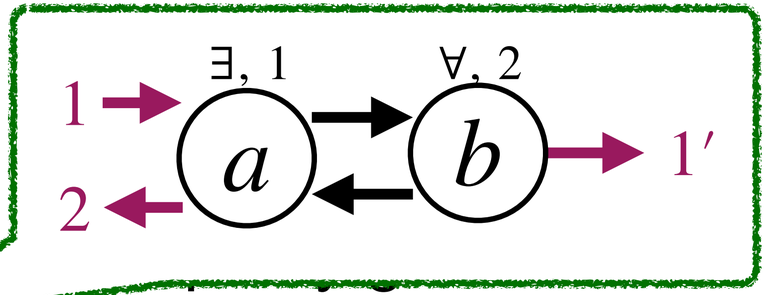


- goal of our paper: building a compositional framework for parity games
  → open parity games: parity games with open ends

- advantage of compositionality: global winning positions can be deduced from local winning positions

Parity game: verification method for model checking





- goal of our paper: building a compositional framew
  → open parity games: parity games with open ends

- advantage of compositionality: global winning positions can be deduced from local winning positions

Background and our idea

Recent work

- prop as a graphical language
  - free symmetric monoidal category suited for 2-dimensional syntax
  - ex) signal flow diagram [Bonchi et al., ’17], networks [Baez et al., ’18]

- **FinScottL$_!$** : model of higher-order model checking [Grellois & Melliès, ’15]
  - higher-order model checking: model checking for functional programs
  - uses parity games implicitly

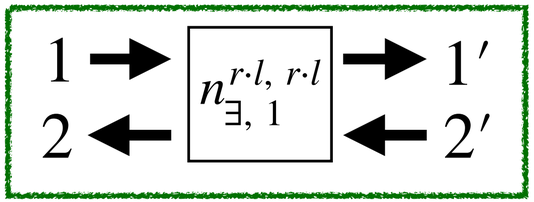
Our idea

- compositionality of **building open parity games**

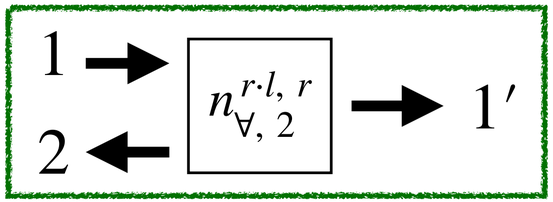
  → syntax category as a prop
- compositionality of **solving winning positions**

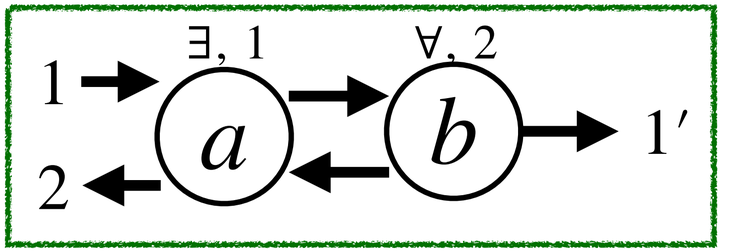
  → semantic category constructed from **FinScottL$_!$**

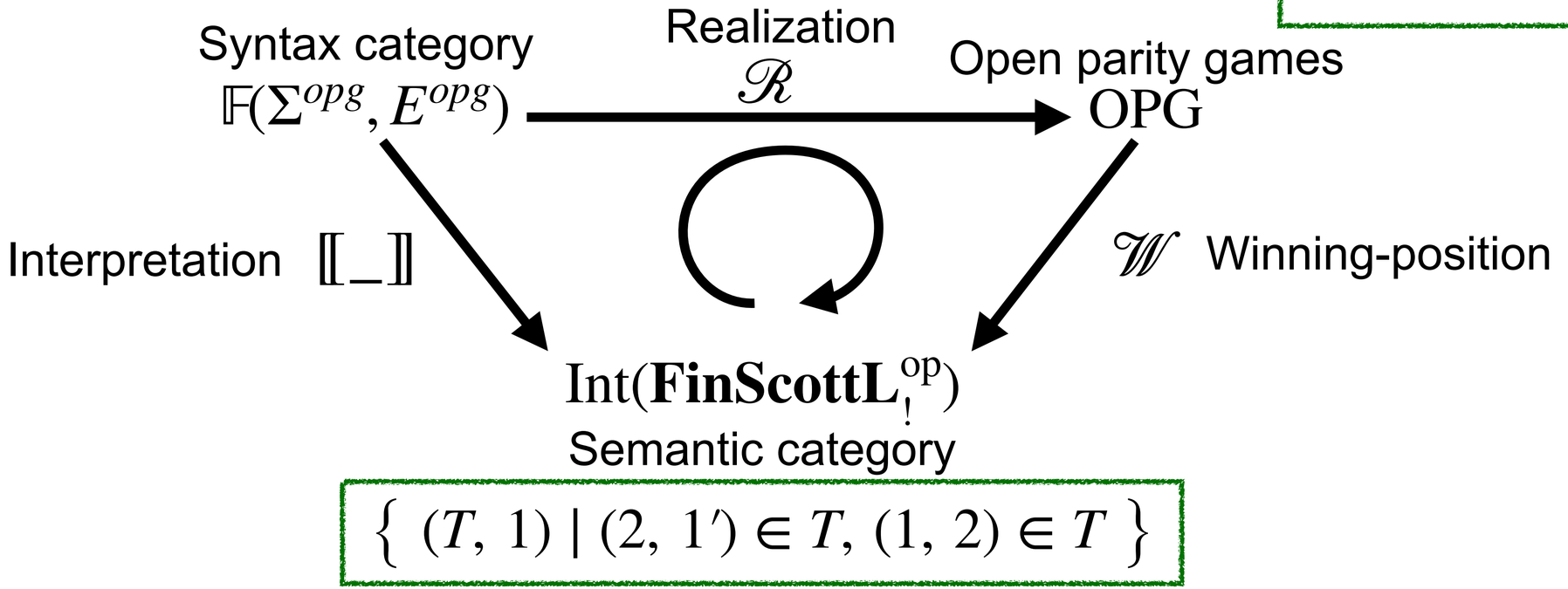
Our contributions



;

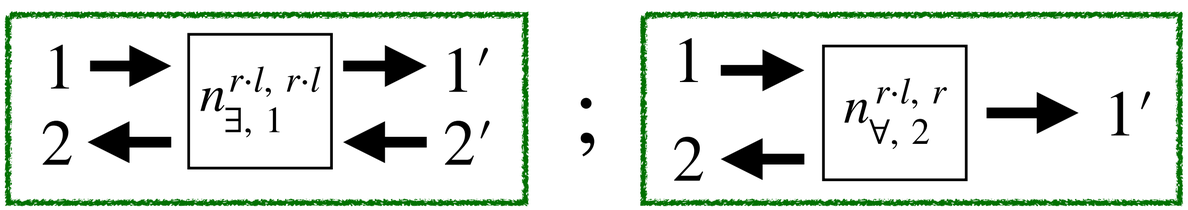




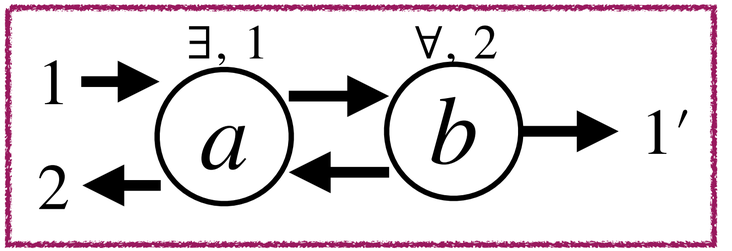


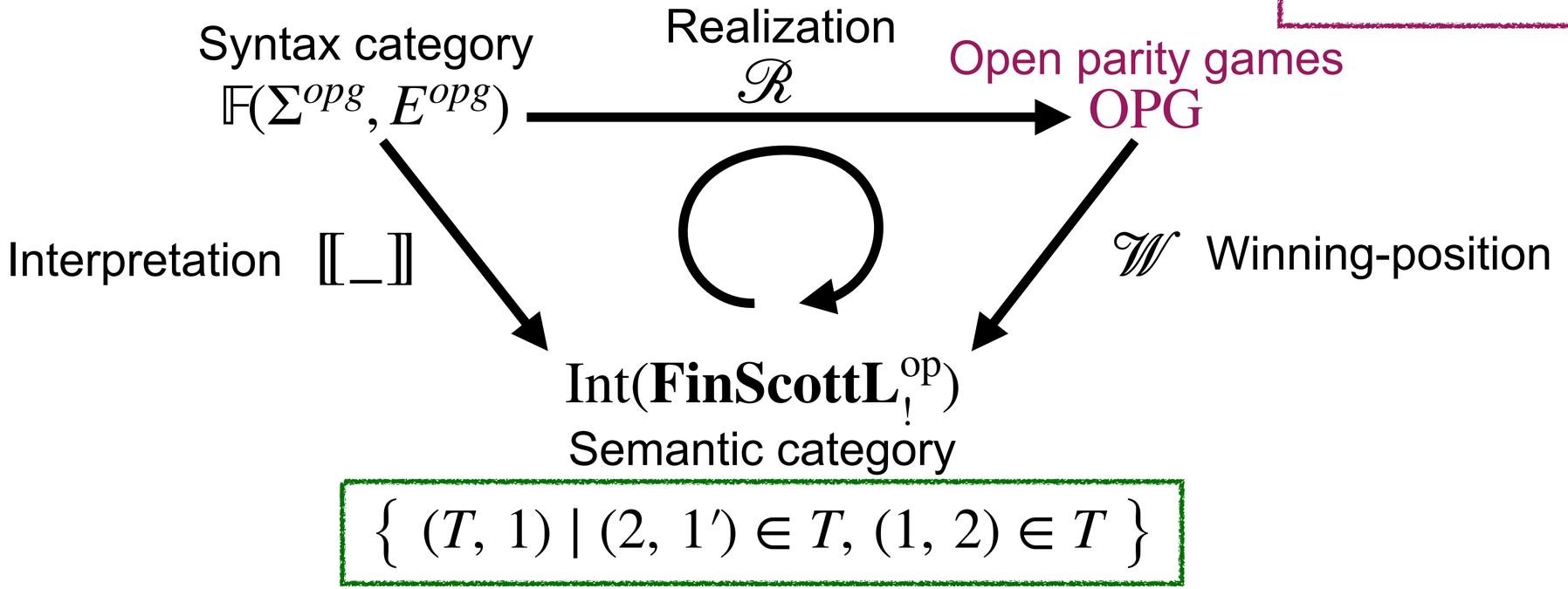
- realization functor $\mathscr{R}$ is full: every open parity game is representable in $\mathbb{F}(\Sigma^{opg}, E^{opg})$
- interpretation functor $[\![_]\!]$ inductively calculates winning positions without explicit construction of strategies
- winning-position functor $\mathscr{W}$ calculates winning positions using strategies

Our contributions



;



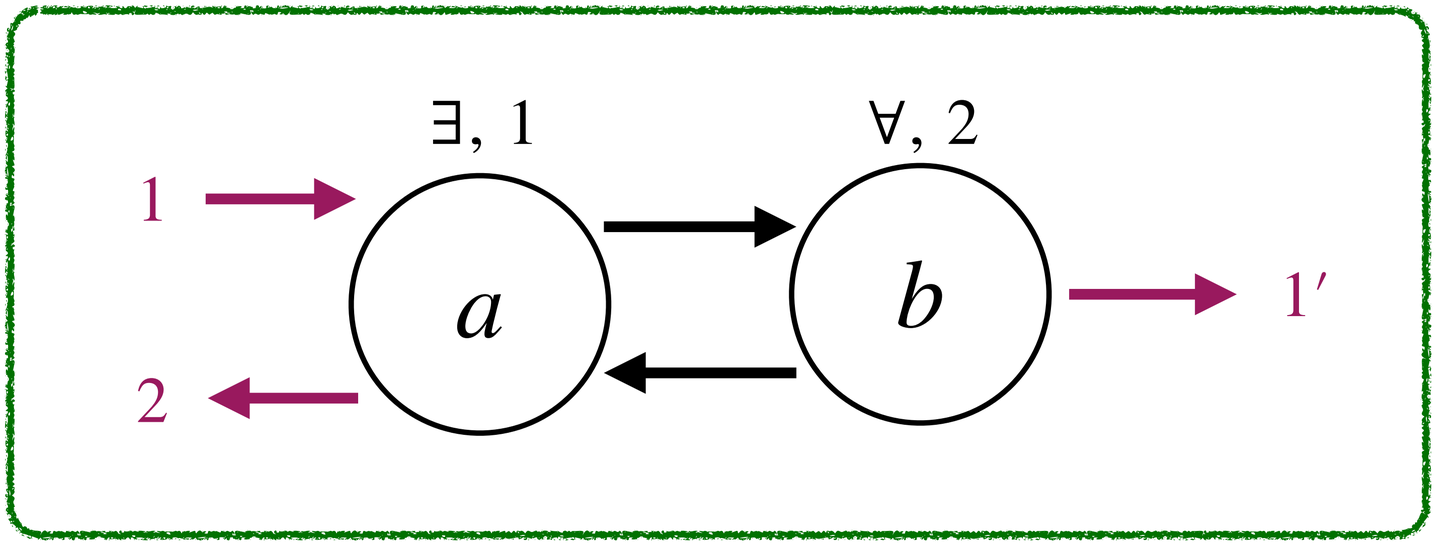


- realization functor $\mathscr{R}$ is full: every open parity game is representable in $\mathbb{F}(\Sigma^{opg}, E^{opg})$
- interpretation functor $[\![_]\!]$ inductively calculates winning positions without explicit construction of strategies
- winning-position functor $\mathscr{W}$ calculates winning positions using strategies

Open parity game

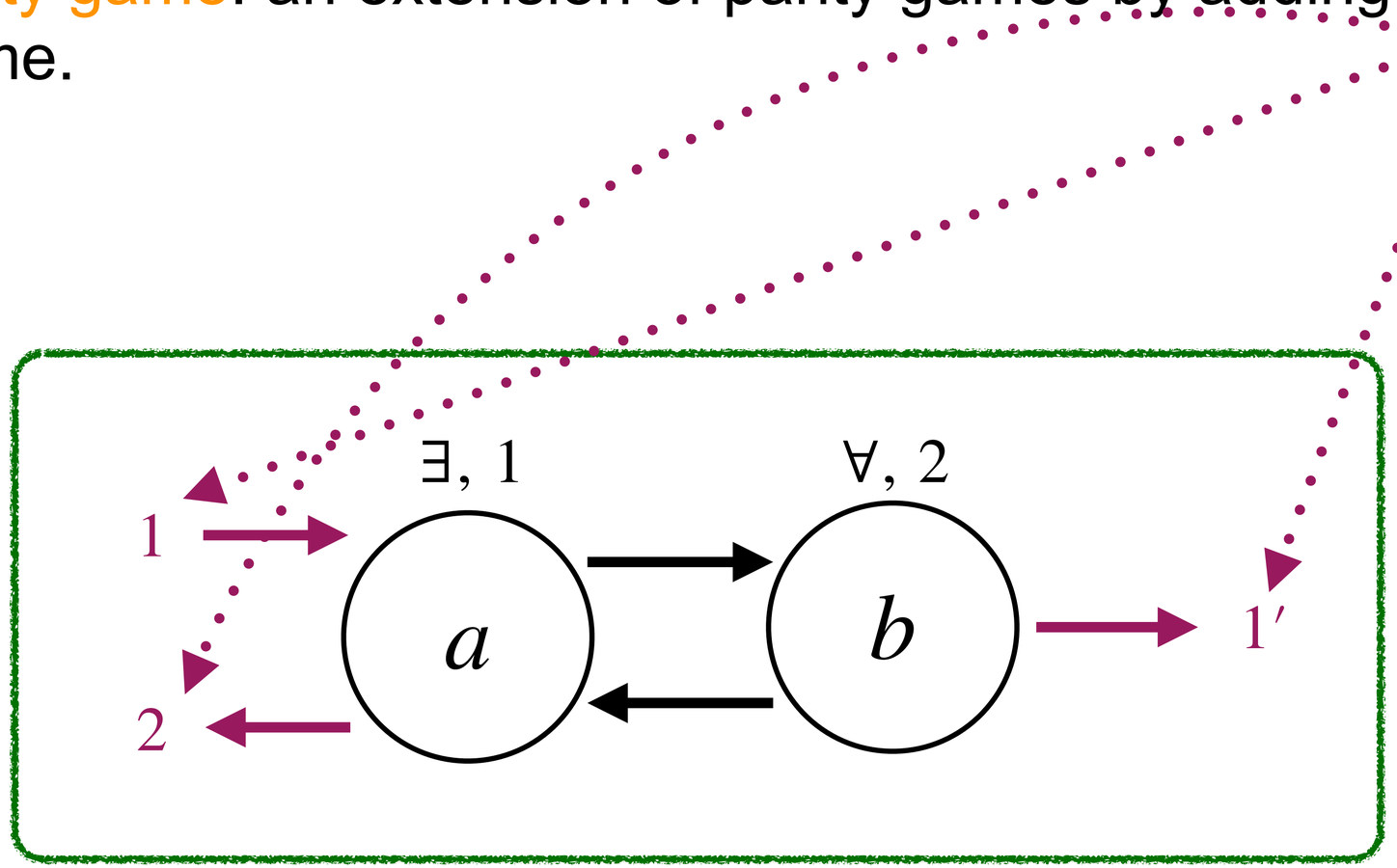
- Open parity game: an extension of parity games by adding open ends to the game.

- Example:



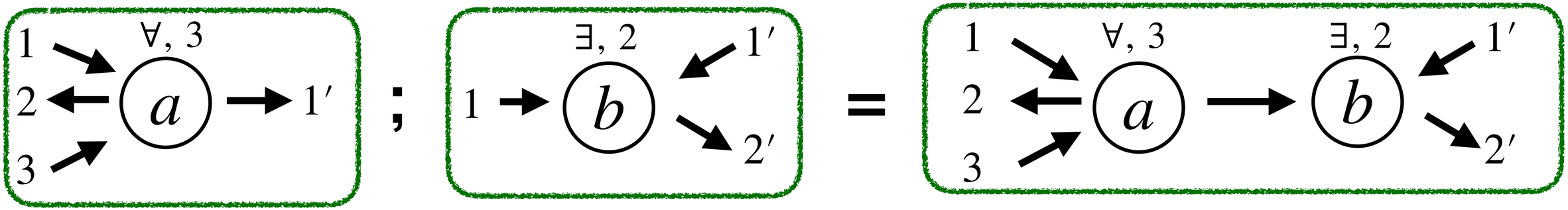
Open parity game

- Open parity game: an extension of parity games by adding open ends to the game.
- Example:

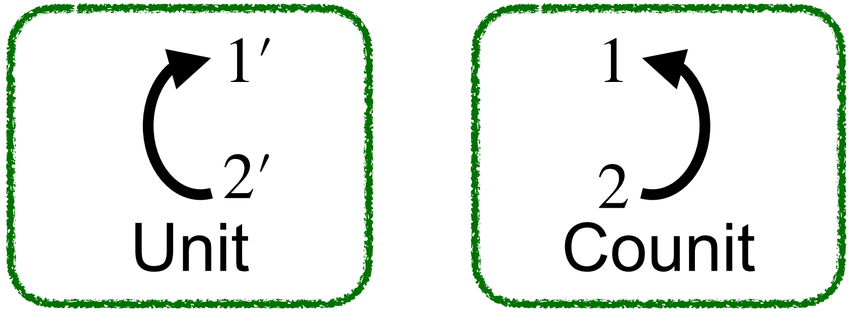


A compact closed category of open parity games OPG

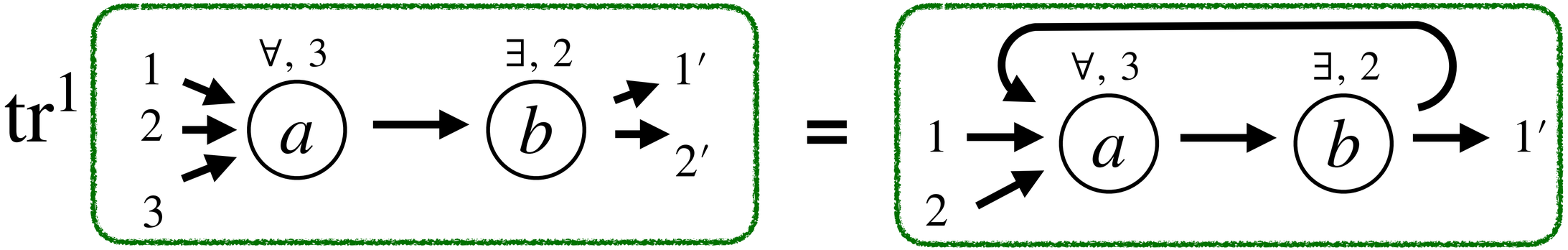
- sequential composition:



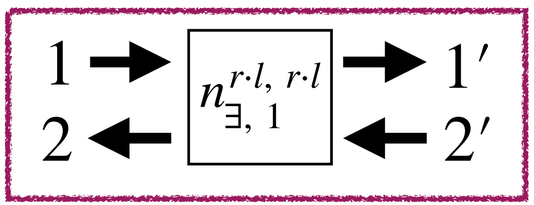
- unit and counit:



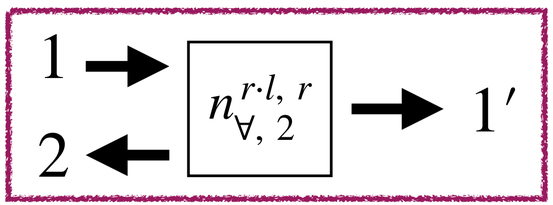
- trace operator:

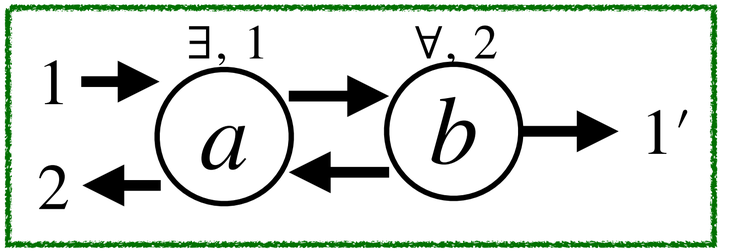


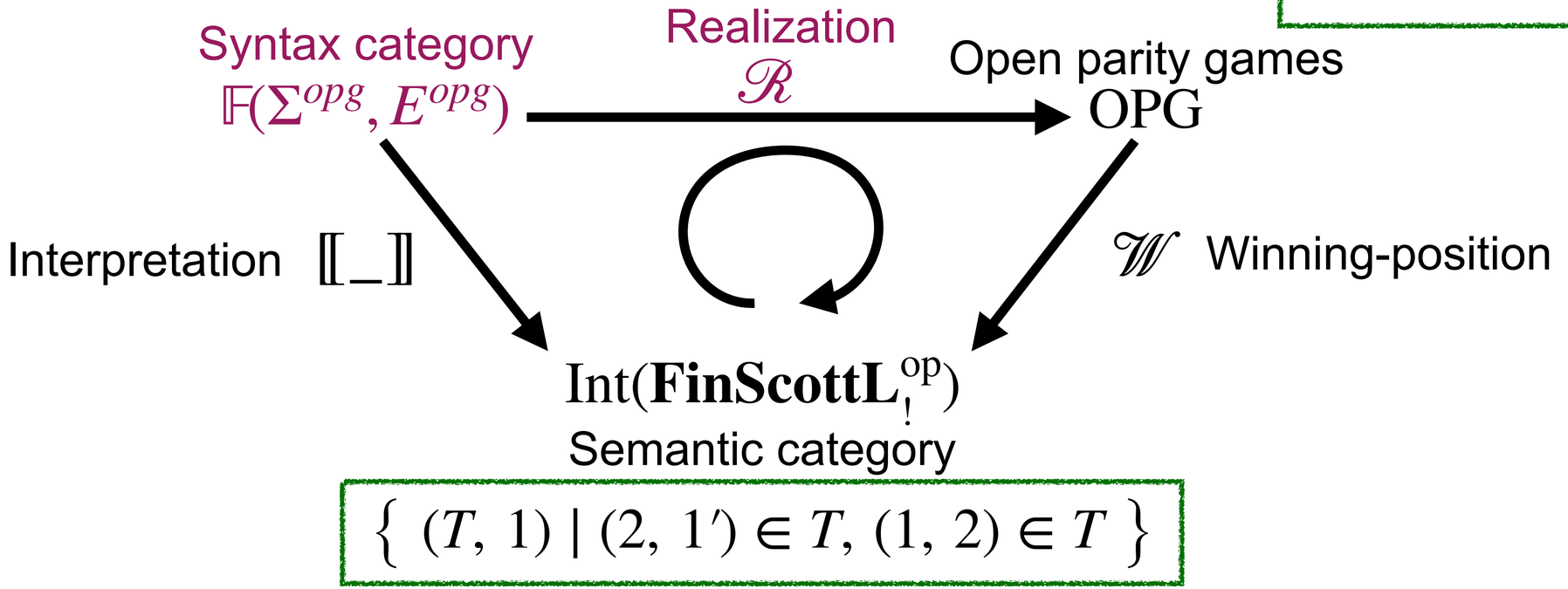
Our contributions



;





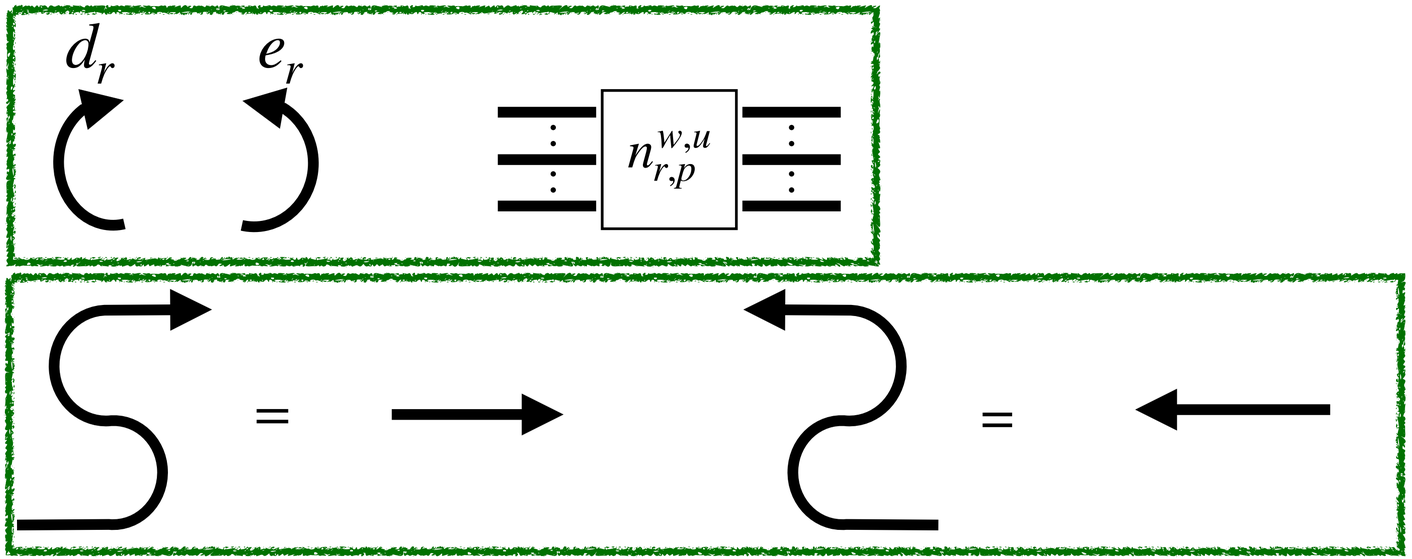


- realization functor $\mathscr{R}$ is full: every open parity game is representable in $\mathbb{F}(\Sigma^{opg}, E^{opg})$
- interpretation functor $[\![_]\!]$ inductively calculates winning positions without explicit construction of strategies
- winning-position functor $\mathscr{W}$ calculates winning positions using strategies

The $\{r, l\}$-prop: $\mathbb{F}(\Sigma^{opg}, E^{opg})$

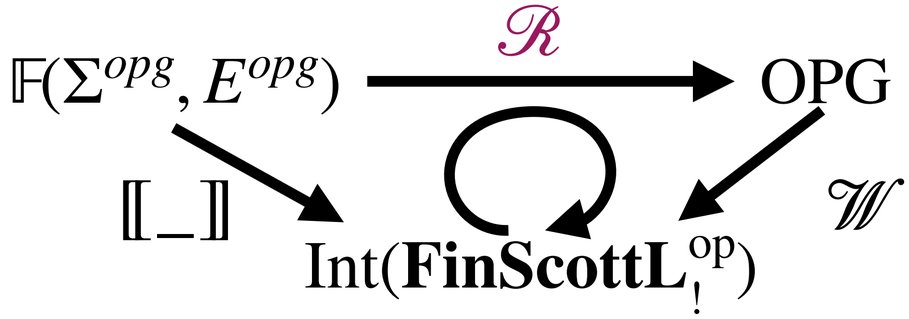
Def (C-prop) : Let C be a set (of *colors*). A C-prop is a strict symmetric monoidal category whose objects form the free monoid C^*.

- $\mathbb{F}(\Sigma^{opg}, E^{opg})$: a free strict symmetric monoidal category generated by a symmetric monoidal theory (Σ^{opg}, E^{opg})
  - signature Σ^{opg}:
  - equations E^{opg}:



Thm: $\mathbb{F}(\Sigma^{opg}, E^{opg})$ is a free compact closed category generated by node generators $n_{r,p}^{w,u}$.

The realization functor $\mathscr{R}$ and fullness

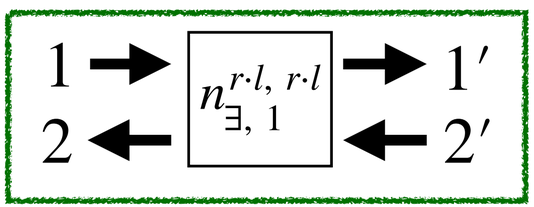


- We define the realization functor $\mathscr{R}$ by the freeness of $\mathbb{F}(\Sigma^{\mathrm{opg}}, E^{\mathrm{opg}})$.
- example

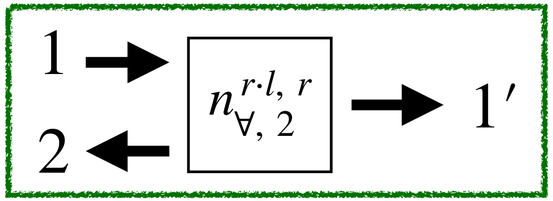
$$\mathscr{R}\left(\rightleftarrows \boxed{n^{r\cdot l, r}_{\forall, 2}} \rightarrow \right) = \begin{array}{c} 1 \rightarrow \\ 2 \leftarrow \end{array} \overset{\forall,\ 2}{\bigcirc\, n} \rightarrow 1'$$

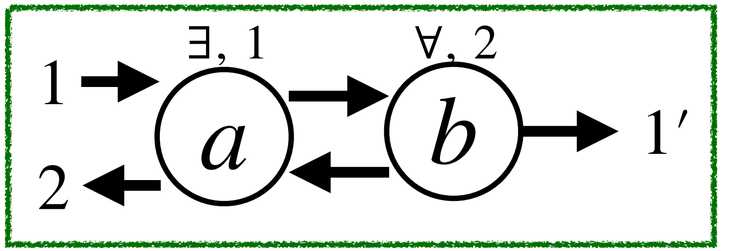
Thm: $\mathscr{R}$ is full.

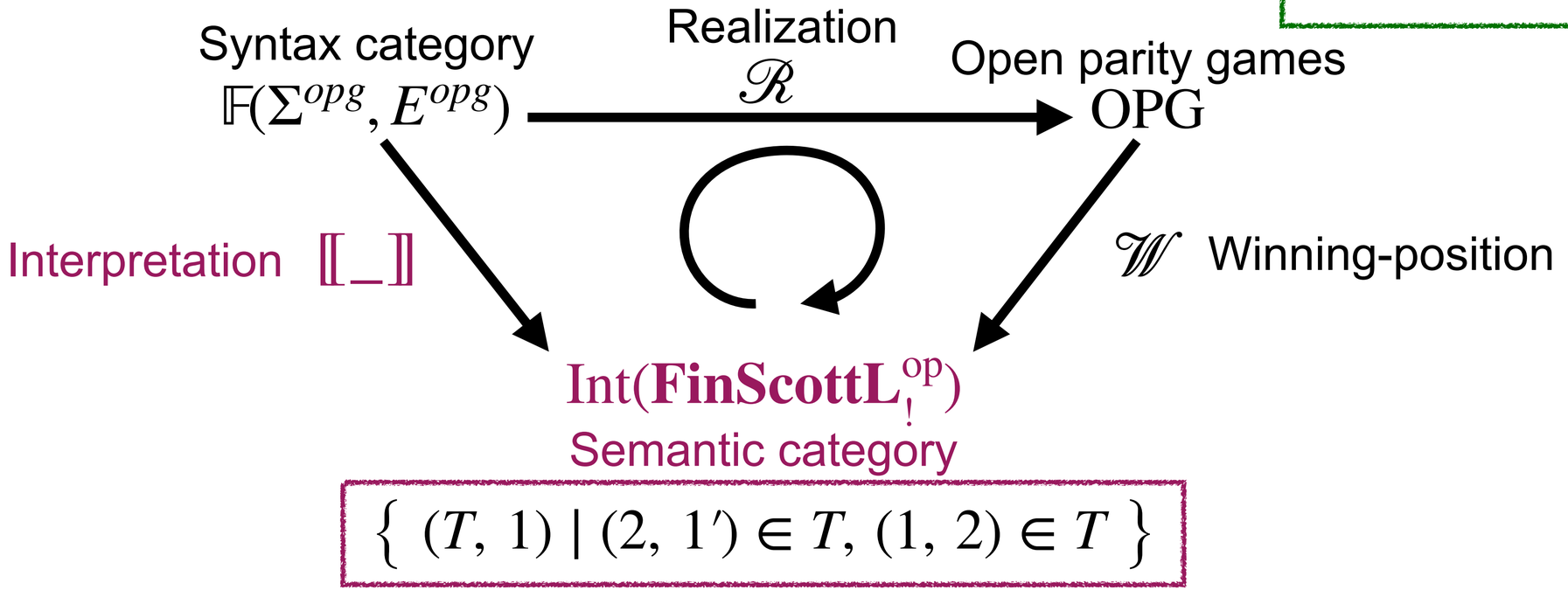
Our contributions



;







- realization functor $\mathscr{R}$ is full: every open parity game is representable in $\mathbb{F}(\Sigma^{opg}, E^{opg})$
- interpretation functor $[\![_]\!]$ inductively calculates winning positions without explicit construction of strategies
- winning-position functor $\mathscr{W}$ calculates winning positions using strategies

$\mathbf{FinScottL}_!$ [Grellois & Melliès, '15]

Def : The category $\mathbf{FinScottL}$ *has as objects finite preordered sets* $A = (|A|, \leq_A)$ *and as morphisms* $R : A \to B$ *(downward closed) binary relations* $R \subseteq A^{op} \times B$.

- finite cartesian products in $\mathbf{FinScottL}$:
  $(|A|, \leq_A) \times (|B|, \leq_B) = (|A| + |B|, \leq_A + \leq_B)$

- comonad $! = \mathscr{P} \circ \square$ on $\mathbf{FinScottL}$
  - induced by the distributive law $\lambda : \mathscr{P} \circ \square \Rightarrow \square \circ \mathscr{P}$
    - the finite powerset comonad $\mathscr{P}$: nondeterminism of a choice by ∀
    - the coloring comonad $\square := \{0, \ldots, M\} \times (-)$: priorities
  - (! is a linear exponential comonad)

- fixed point operator Y in $\mathbf{FinScottL}_!$
  - calculates the parity condition in infinite plays

Semantic category $\mathrm{Int}(\mathbf{FinScottL}_!^{op})$

- Int construction [Joyal et al., '96]: a free construction of a compact closed category $\mathrm{Int}(\mathbf{C})$ from a traced symmetric monoidal category $\mathbf{C}$
  - object of $\mathrm{Int}(\mathbf{C})$ is a pair (X_+, X_-) of objects in $\mathbf{C}$
  - $f : (X_+, X_-) \to (Y_+, Y_-)$ of $\mathrm{Int}(\mathbf{C})$ is $f : X_+ \otimes Y_- \to Y_+ \otimes X_-$ of $\mathbf{C}$
    - This makes morphisms bidirectional
  - Sequential composition in $\mathrm{Int}(\mathbf{C})$ is defined by a trace operator in $\mathbf{C}$.

- trace operator tr in $\mathbf{FinScottL}_!^{op}$

  $\Leftrightarrow$ trace operator tr' in $\mathbf{FinScottL}_!$ (trivially)

  $\Leftrightarrow$ fixed point operator Y in $\mathbf{FinScottL}_!$ (by [Hyland], [Hasegawa, '97])

- semantic category: $\mathrm{Int}(\mathbf{FinScottL}_!^{op})$

Interpretation functor $[\![_]\!]$

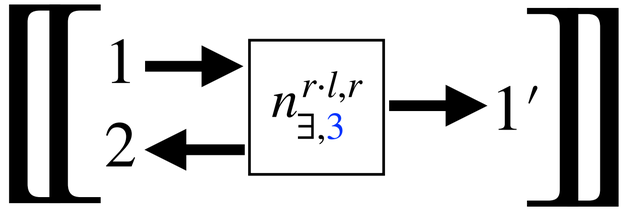
Semantic category $\mathrm{Int}(\mathbf{FinScottL}_!^{op})$

- A morphism is a (downward closed) binary relation
  $R \subseteq \mathscr{P}(\{1,\ldots,M\} \times [m_r + n_l]) \times [n_r + m_l].$

Priority Rightward Leftward

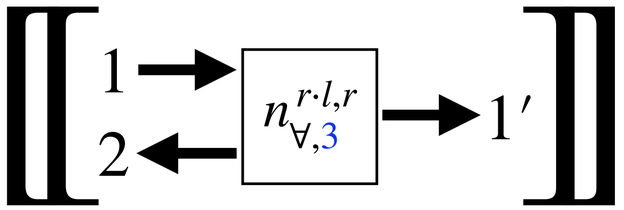
Interpretation functor $[\![_]\!] : \mathbb{F}(\Sigma^{opg}, E^{opg}) \to \mathrm{Int}(\mathbf{FinScottL}_!^{op})$

- defined to be a compact closed functor



Nondeterminism by ∃

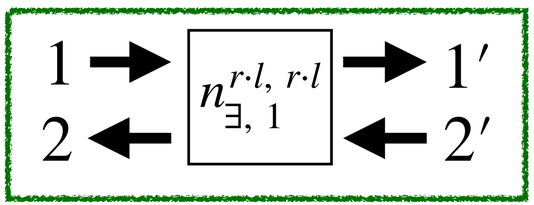
$= \{(T,\ 1) \mid (3,\ 1') \in T \text{ or } (3,\ 2) \in T\}$



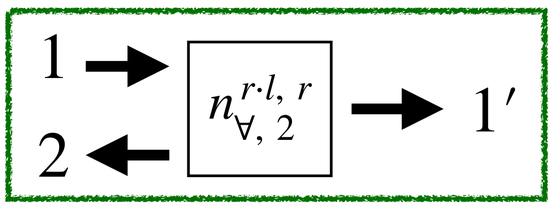
Nondeterminism by ∀

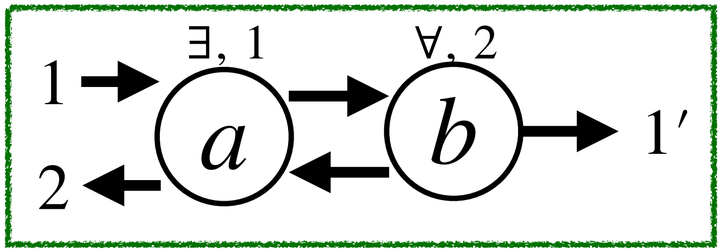
$= \{(T,\ 1) \mid \{(3,\ 1'), (3,\ 2)\} \subseteq T\}$

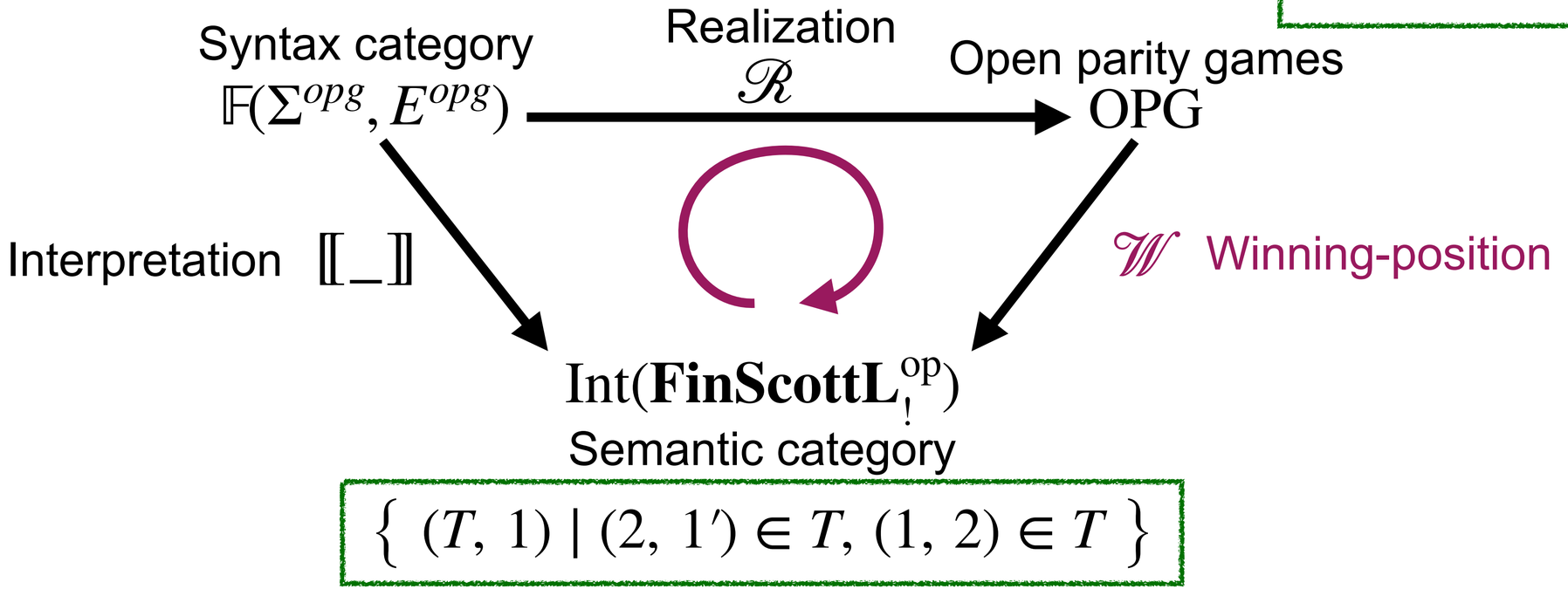
Our contributions



;







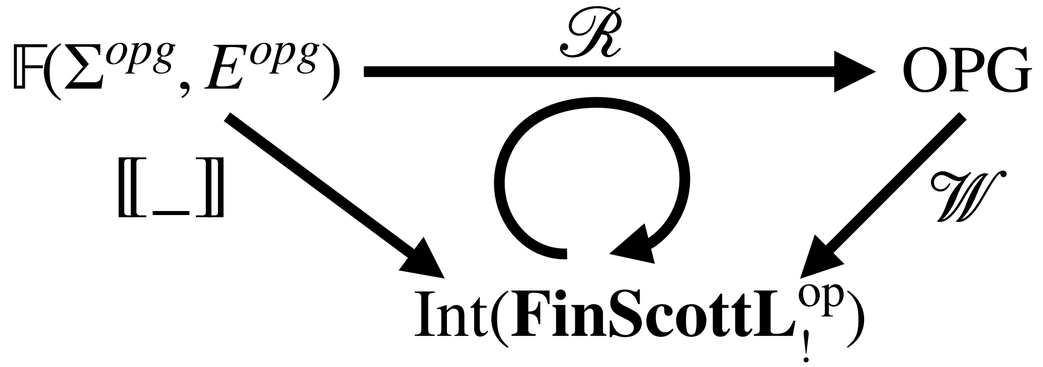
- realization functor $\mathscr{R}$ is full: every open parity game is representable in $\mathbb{F}(\Sigma^{opg}, E^{opg})$
- interpretation functor $[\![_]\!]$ inductively calculates winning positions without explicit construction of strategies
- winning-position functor $\mathscr{W}$ calculates winning positions using strategies

Winning position functor $\mathscr{W}$ and the main theorem

- winning-position functor $\mathscr{W}$ returns whether a start position is winning, losing, or pending.
  - $\mathscr{W}$ is defined in the traditional way by using plays and strategies.
  - pending: intermediate result due to the notion of open ends

Thm: $\mathscr{W}$ is a compact closed functor

Thm (main): The triangle commutes: $[\![_]\!] \simeq \mathscr{W} \circ \mathscr{R}$

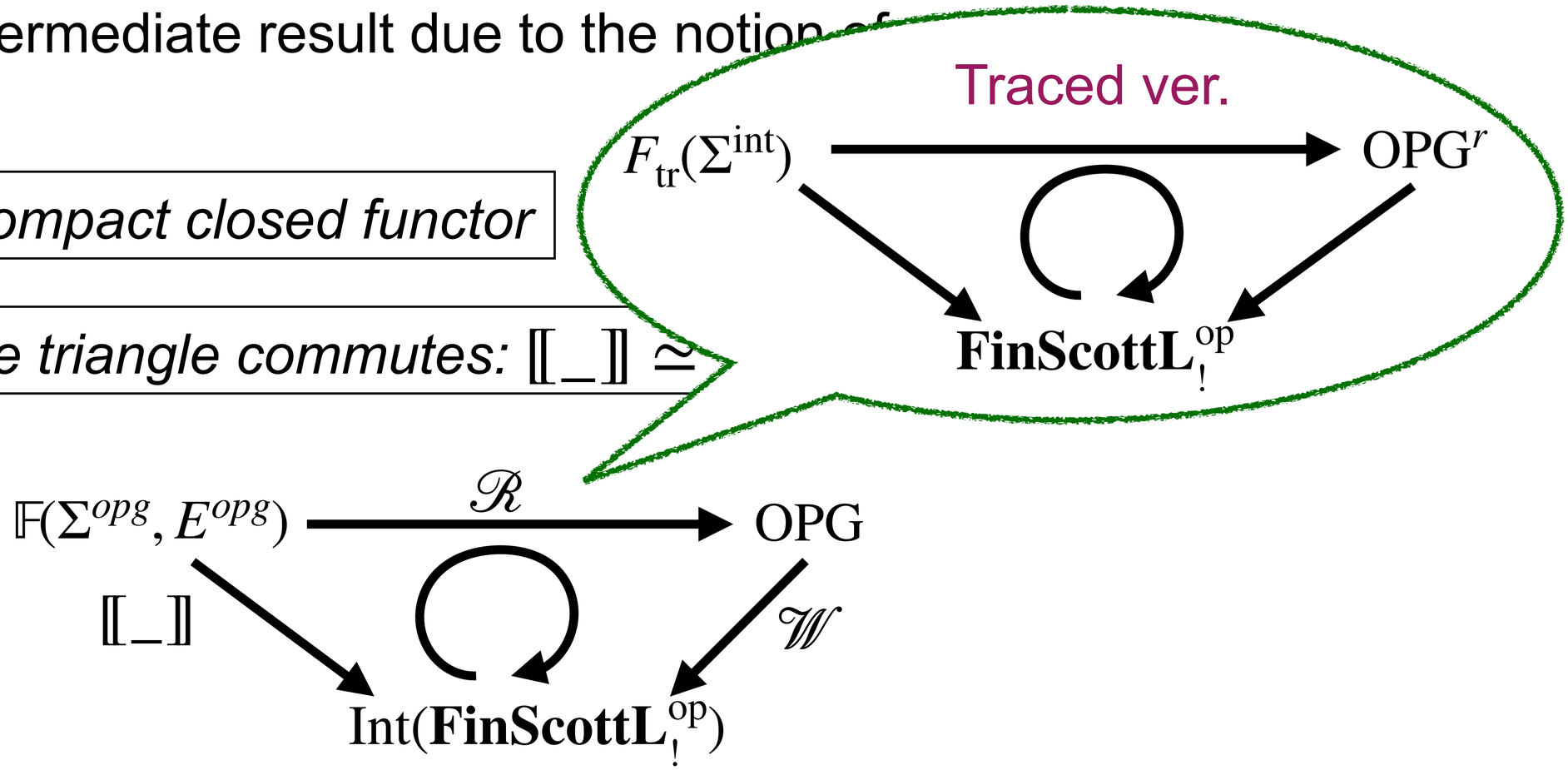


Winning position functor $\mathscr{W}$ and the main theorem

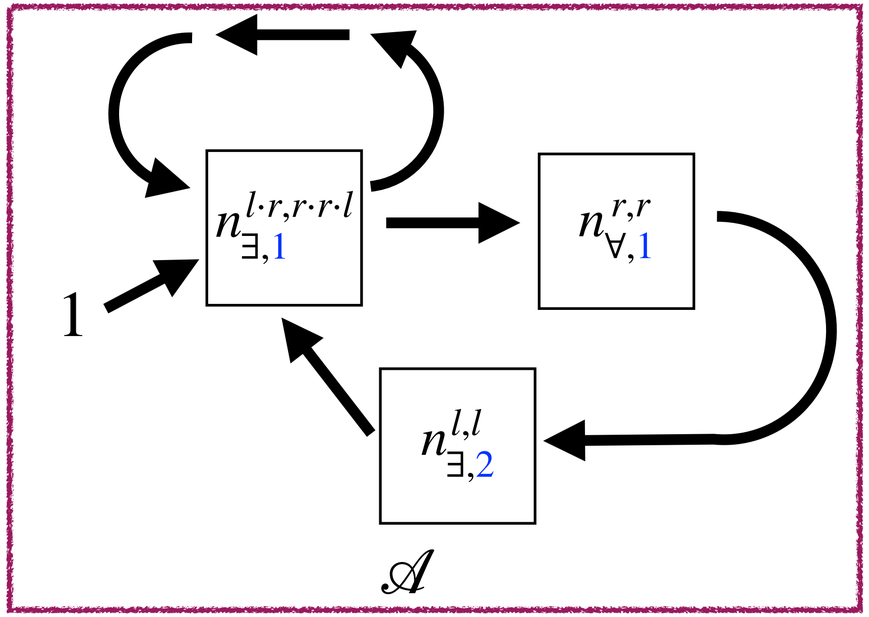
- winning-position functor $\mathscr{W}$ returns whether a start position is winning, losing, or pending.
  - $\mathscr{W}$ is defined in the traditional way by using plays and strategies.
  - pending: intermediate result due to the notion of

Thm: $\mathscr{W}$ is a compact closed functor

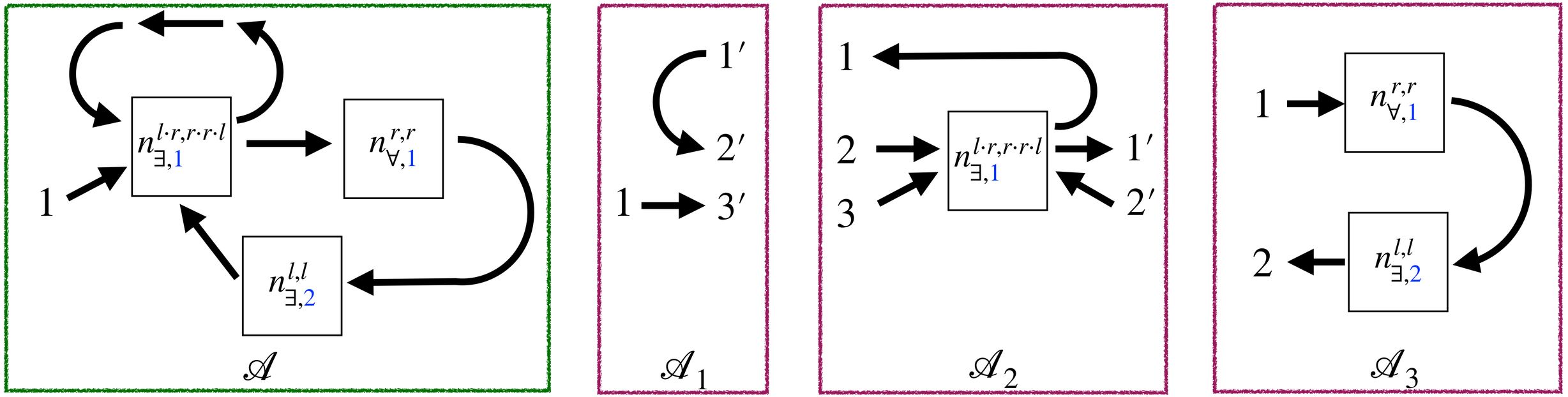
Thm (main): The triangle commutes: $[\![_]\!] \simeq$



Example


$n^{l\cdot r, r\cdot r\cdot l}_{\exists,1}$
$n^{r,r}_{\forall,1}$
$n^{l,l}_{\exists,2}$
1
$\mathscr{A}$

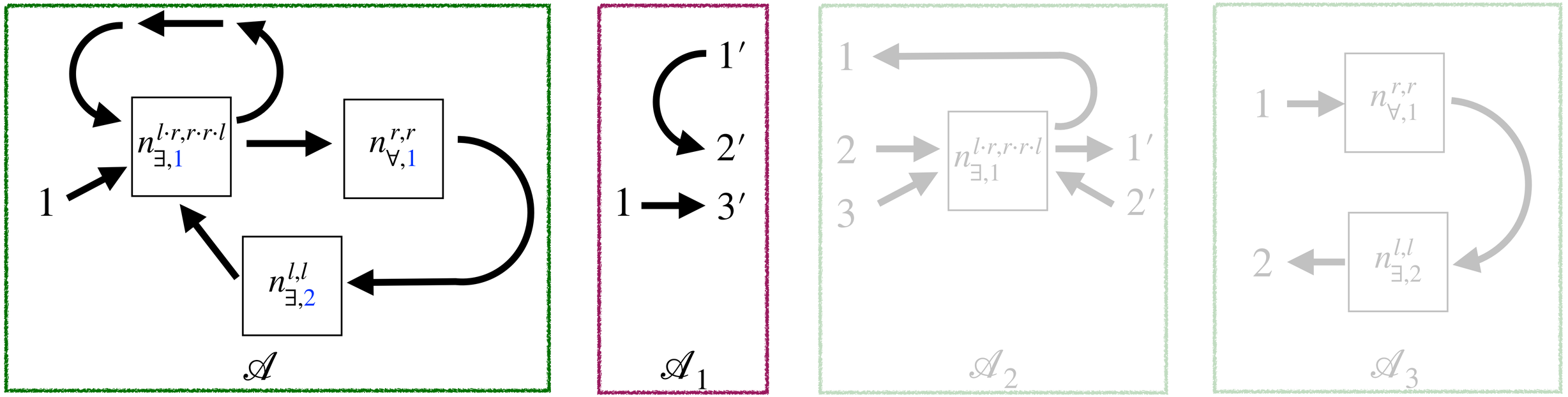
Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T, 1) \mid (0, 3') \in T\} \cup \{(T, 1') \mid (0, 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \{(T, i) \mid i \in \{2, 3, 2'\}, \exists j \in \{1, 1'\}, (1, j) \in T\}$,
- $[\![\mathscr{A}_3]\!] = \{(T, 1) \mid (2, 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T, i) \mid i \in \{1, 2'\}, (1, 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing, 1)\}$.

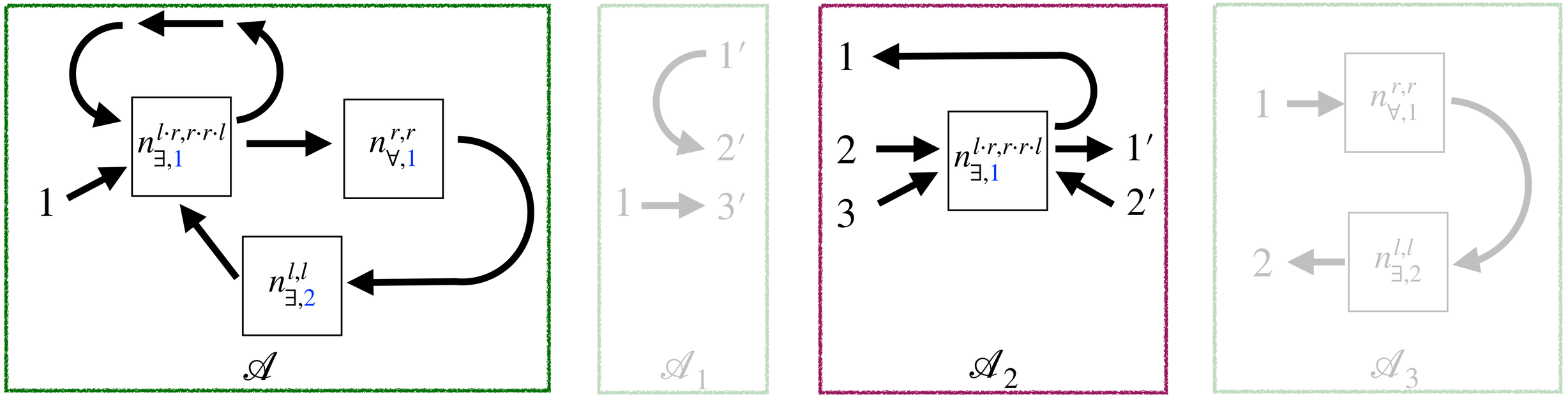
Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T, 1) \mid (0, 3') \in T\} \cup \{(T, 1') \mid (0, 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \{(T, i) \mid i \in \{2, 3, 2'\}, \ ((1, 1) \in T \text{ or } (1, 1') \in T)\}$,
- $[\![\mathscr{A}_3]\!] = \{(T, 1) \mid (2, 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T, i) \mid i \in \{1, 2'\}, (1, 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing, 1)\}$.

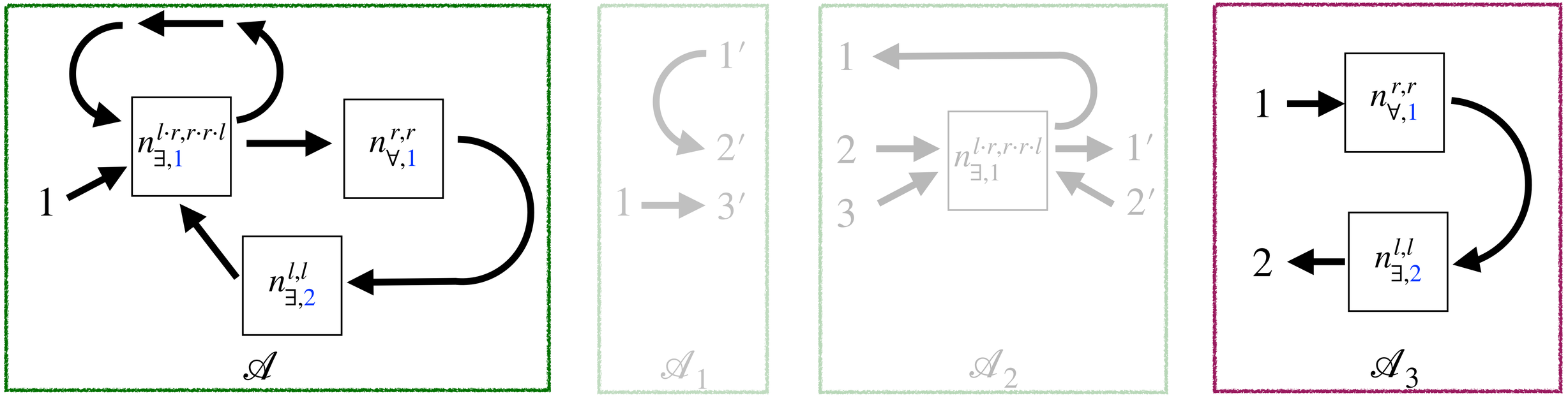
Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T, 1) \mid (0, 3') \in T\} \cup \{(T, 1') \mid (0, 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \{(T, i) \mid i \in \{2, 3, 2'\},\ ((1, 1) \in T \text{ or } (1, 1') \in T)\}$,
- $[\![\mathscr{A}_3]\!] = \{(T, 1) \mid (2, 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T, i) \mid i \in \{1, 2'\}, (1, 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing, 1)\}$.

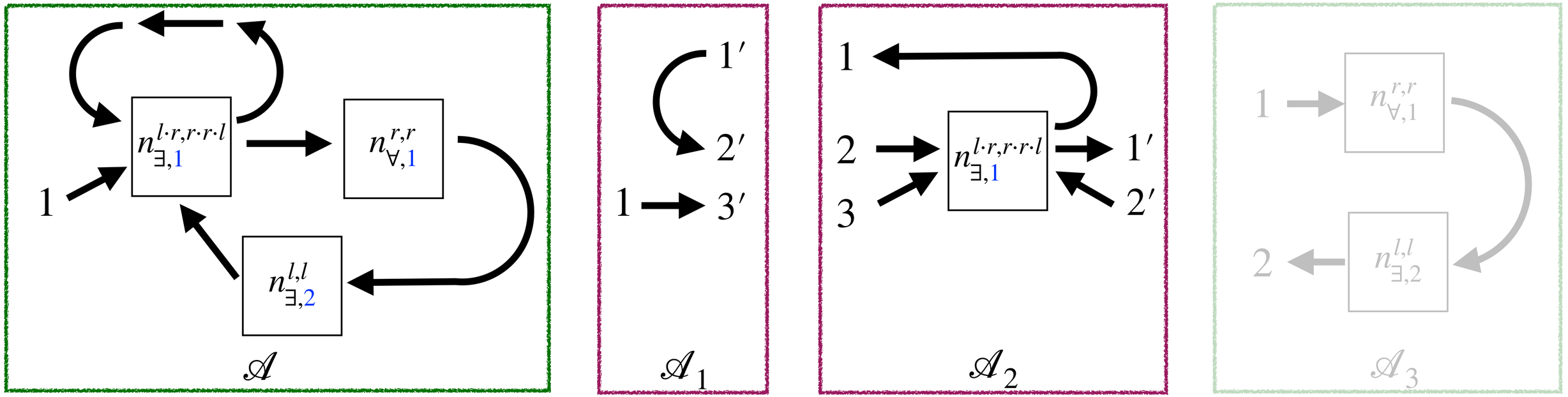
Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T, 1) \mid (0, 3') \in T\} \cup \{(T, 1') \mid (0, 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \{(T, i) \mid i \in \{2, 3, 2'\}, \ ((1, 1) \in T \text{ or } (1, 1') \in T)\}$,
- $[\![\mathscr{A}_3]\!] = \{(T, 1) \mid (2, 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T, i) \mid i \in \{1, 2'\}, (1, 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing, 1)\}$.

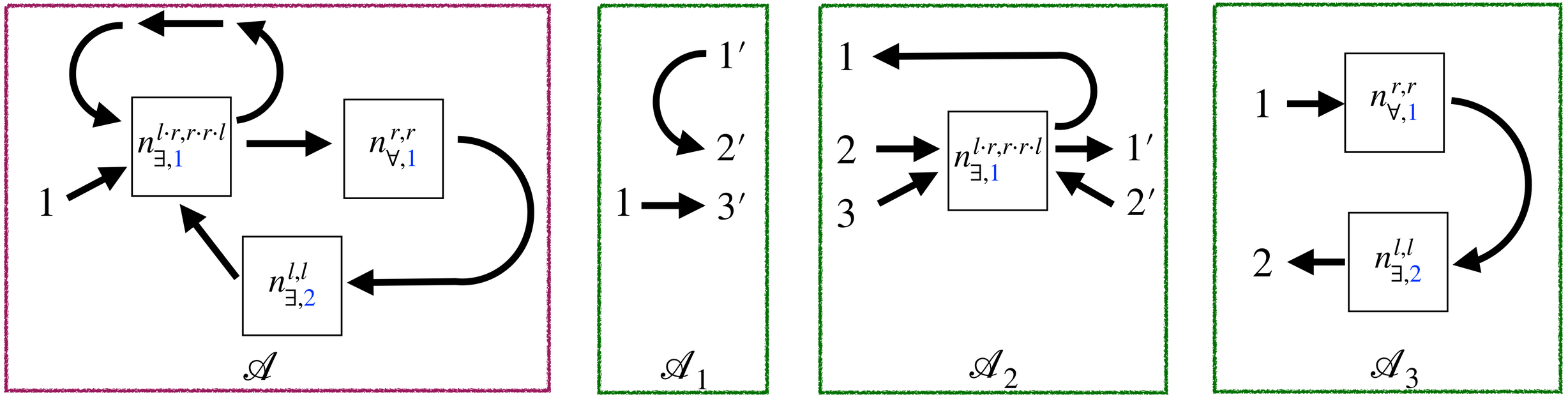
Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T, 1) \mid (0, 3') \in T\} \cup \{(T, 1') \mid (0, 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \left\{(T, i) \mid i \in \{2, 3, 2'\},\ \left((1, 1) \in T \text{ or } (1, 1') \in T\right)\right\}$,
- $[\![\mathscr{A}_3]\!] = \{(T, 1) \mid (2, 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T, i) \mid i \in \{1, 2'\}, (1, 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing, 1)\}$.

Example



$\mathscr{A}$ can be divided as $\mathscr{A}_3 \circ \mathscr{A}_2 \circ \mathscr{A}_1$.

- $[\![\mathscr{A}_1]\!] = \{(T,\ 1) \mid (0,\ 3') \in T\} \cup \{(T,\ 1') \mid (0,\ 2') \in T\}$,
- $[\![\mathscr{A}_2]\!] = \big\{(T,\ i) \mid i \in \{2,\ 3,\ 2'\},\ \ \big((1,\ 1) \in T \text{ or } (1,\ 1') \in T\big)\big\}$,
- $[\![\mathscr{A}_3]\!] = \{(T,\ 1) \mid (2,\ 2) \in T\}$,
- $[\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = [\![\mathscr{A}_2]\!] \circ [\![\mathscr{A}_1]\!] = \{(T,\ i) \mid i \in \{1,\ 2'\},\ (1,\ 1') \in T\}$,
- $[\![\mathscr{A}]\!] = [\![\mathscr{A}_3]\!] \circ [\![\mathscr{A}_2 \circ \mathscr{A}_1]\!] = \{(\varnothing,\ 1)\}$.

Related work

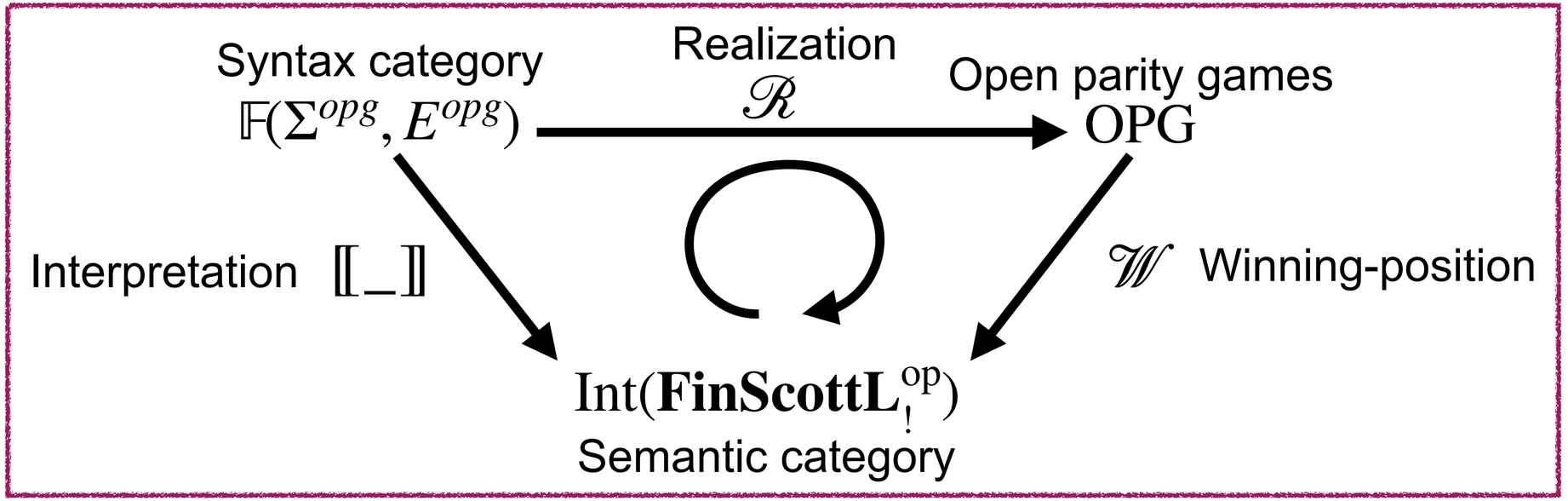
- Prop as a graphical language:
  - signal flow diagram [Bonchi et al., ’17]
  - networks [Baez et al., ’18]
  - petri nets [Bonchi et al., ’19]
  - directed acyclic graph [Fiore & Campos, ’13]
  - finite-state automata [Piedeleu & Zanasi, ’21]
  - ZX-calculus [Carette et al., ’19]
  - network games [Lavore et al., ’21]

- Compositional higher-order model checking
  - relational model [Grellois & Melliès, ’15]
  - game semantics [Tsukada & Ong, ’14]

- Open games [Ghani et al., ’18]
  - different kind of games (for economics)

Conclusion and future work

- Conclusion: we established a compositional approach to parity games



- Future work:
  - compositional model checking
  - history dependent strategy sensitive model
  - equivalence between a (different) graphical language and the category of open parity games
  - coalgebraic treatment of open parity games