Climbing up a ladder: an *evitcudnioc* approach to contextual refinement

Koko Muroya
(RIMS, Kyoto University)

Shonan meeting #175, 20 May 2024

Climbing up a ladder: an *evitcudnioc* (reverse coinductive) approach to contextual refinement

Koko Muroya
(RIMS, Kyoto University)

Shonan meeting #175, 20 May 2024

Contextual refinement

- "Can the (observable) result of t be reproduced by u?
- variations:
  - $t \preceq_{\downarrow} u \overset{\Delta}{\iff} \forall C \,.\, C[t]\downarrow \implies C[u]\downarrow$
  - $t \preceq_{V} u \overset{\Delta}{\iff} \forall C, v \,.\, C[t] \to^{*} v \implies C[u] \to^{*} v$
  - $t \preceq_{V}^{\geq} u \overset{\Delta}{\iff} \forall C, v \,.\, C[t] \to^{k} v \implies C[u] \to^{m} v \wedge k \geq m$
    - Sands' *improvement*
  - $t \preceq_{V}^{Q} u \overset{\Delta}{\iff} \forall C, v \,.\, C[t] \to^{k} v \implies C[u] \to^{m} v \wedge k \; Q \; m$
    - for a preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$, e.g. $\mathbb{N} \times \mathbb{N}$, $\geq$, …

Abransky's applicative bisimilarity

- *the* coinductive proof methodology for contextual equivalence
  - (1) characterise observational equivalence as "bisimilarity"
  - (2) take a candidate $\bowtie$ of contextual equivalence
  - (3) prove that $\bowtie$ is a "bisimulation"
  - (4) prove that $\bowtie$ is a congruence, typically by Howe's method
- (1) for all $\bowtie$, (2-4) for each $\bowtie$

Abramsky’s applicative bisimilarity

- climbing up a rope of advanced features
  - from applicative to environmental bisimilarity [Koutavas+ ’11]
  - Howe’s method, once for all effects [Dal Lago+ ’17]

A new *evitcudnioc* approach

- yet another coinductive proof methodology for contextual refinement
    - (1) characterise observational refinement as “trace inclusion” of automata
    - (2) design a sound “simulation” notion
    - (3) take a candidate $\triangleleft$ of contextual refinement
    - (4) take the contextual closure $\overline{\triangleleft}$ (i.e. $\forall C\,.\,C[\vec{t}]\ \overline{\triangleleft}\ C[\vec{u}] \iff \forall i\,.\,t_i \triangleleft u_i$)
    - (5) prove that $\overline{\triangleleft}$ is a “simulation”

- (1-2) for all $\triangleleft$, (3-5) for each $\triangleleft$

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence

The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence

The left bar:
(2) "simulation" notions

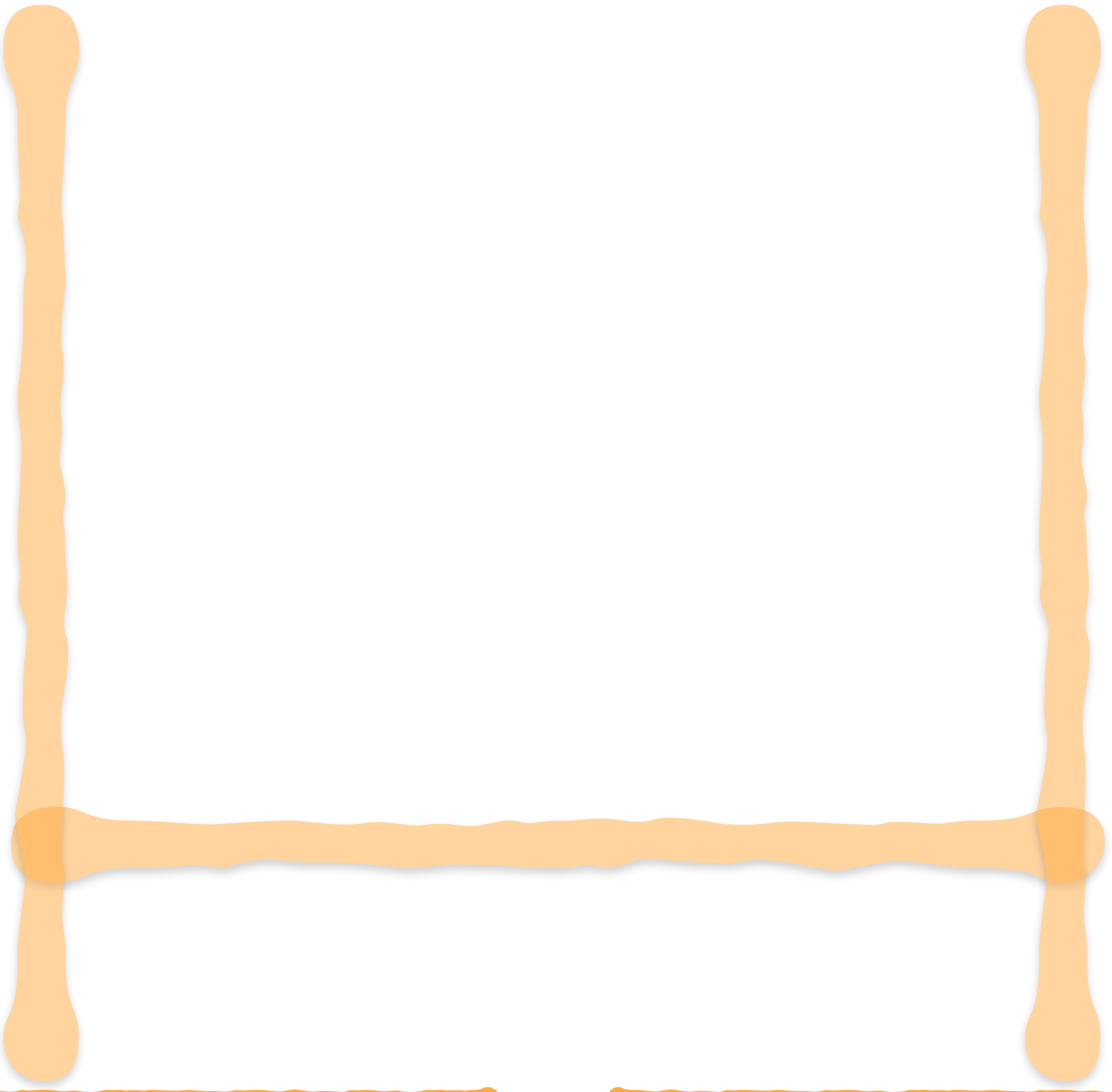
forgetting how each → is defined

The right bar:
(5) "simulation" proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

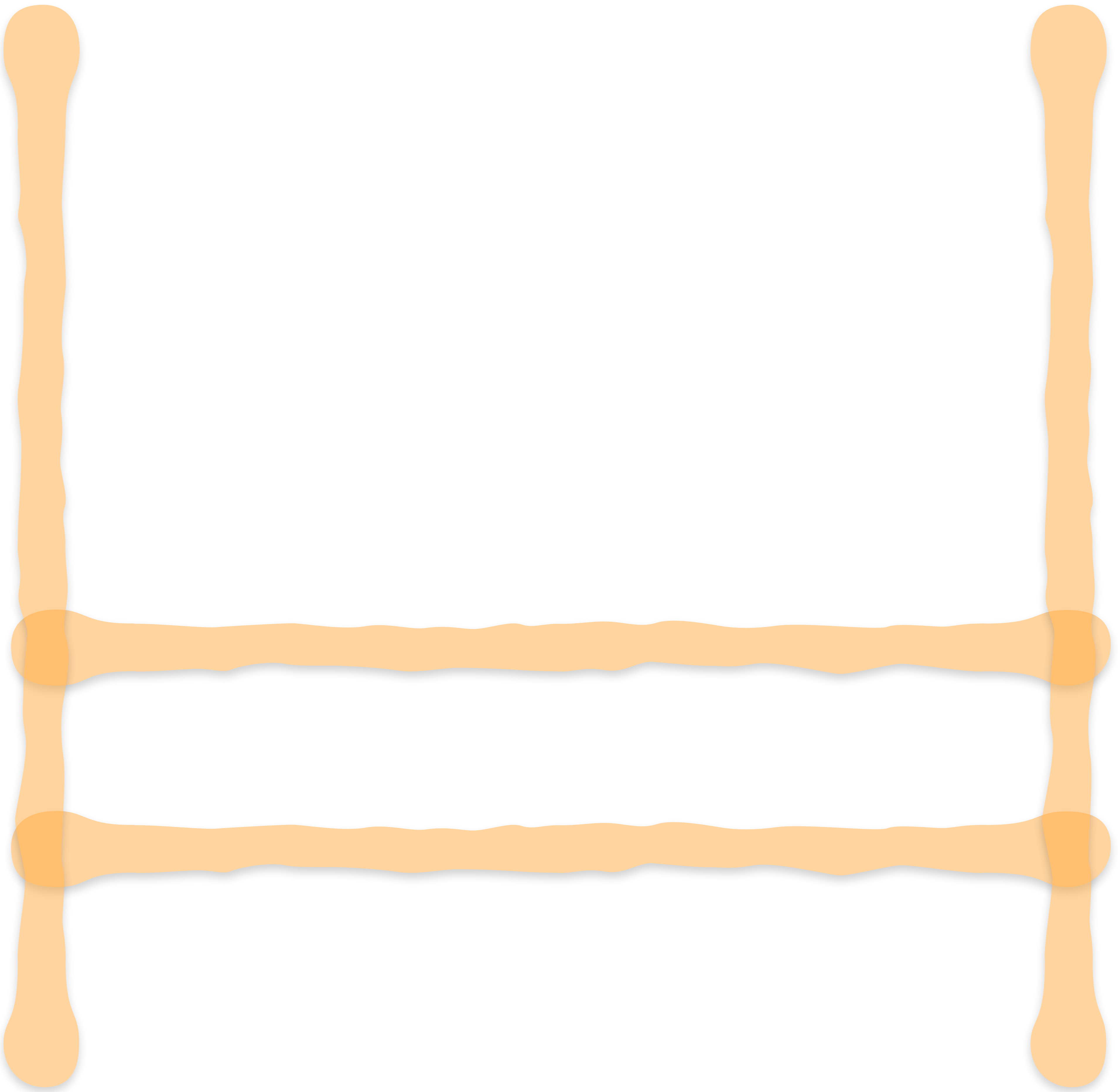
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

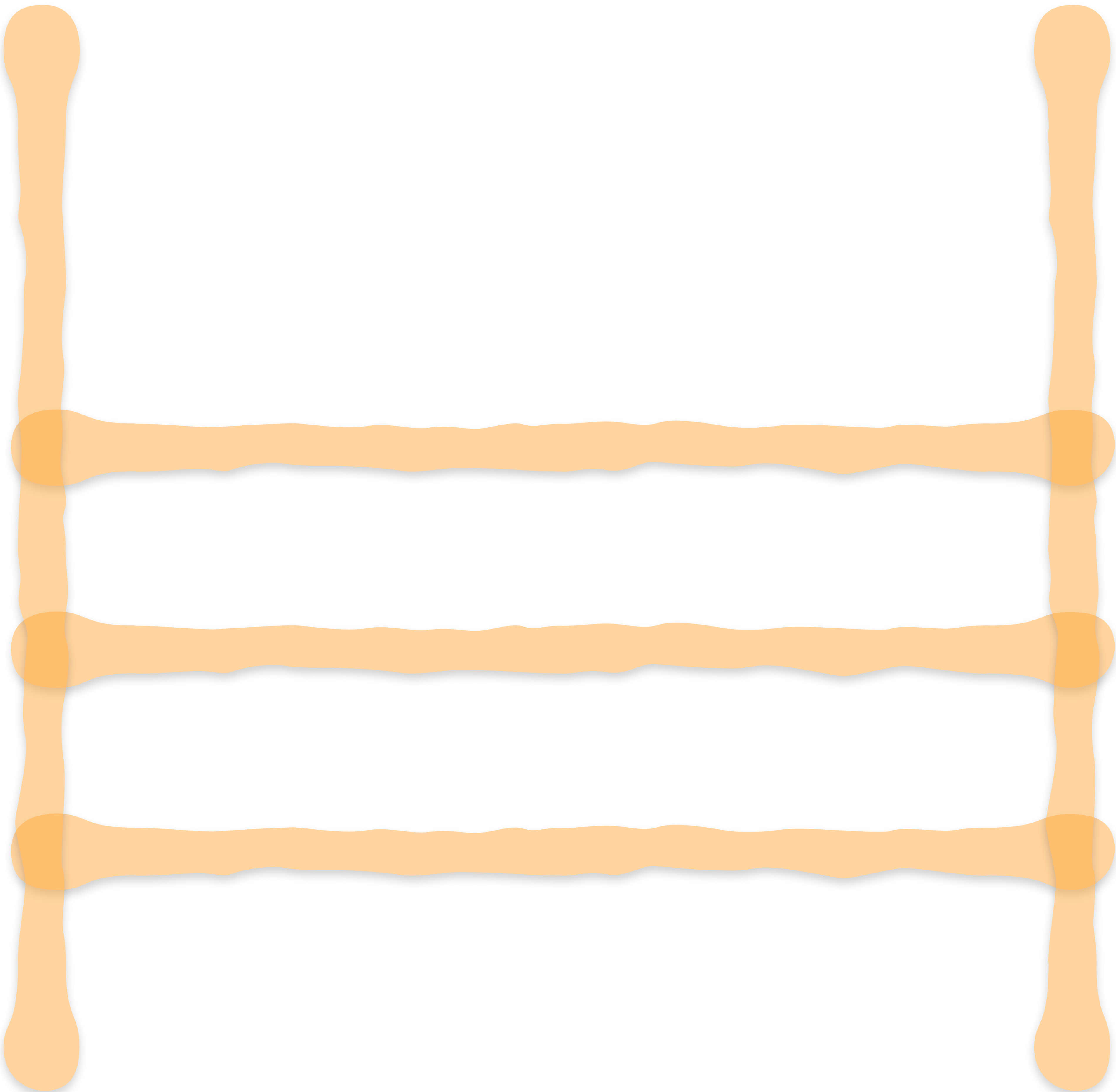
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

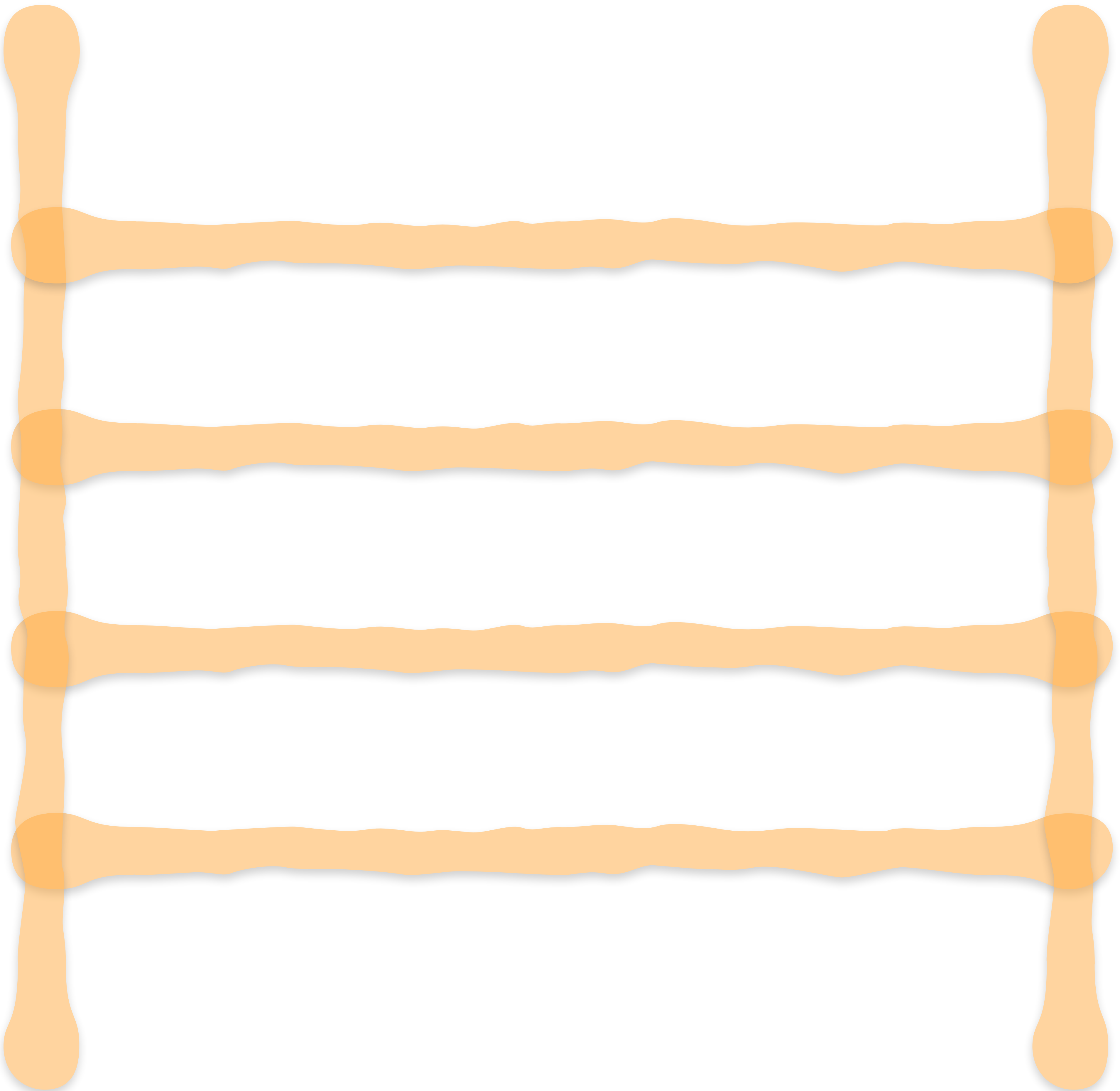
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

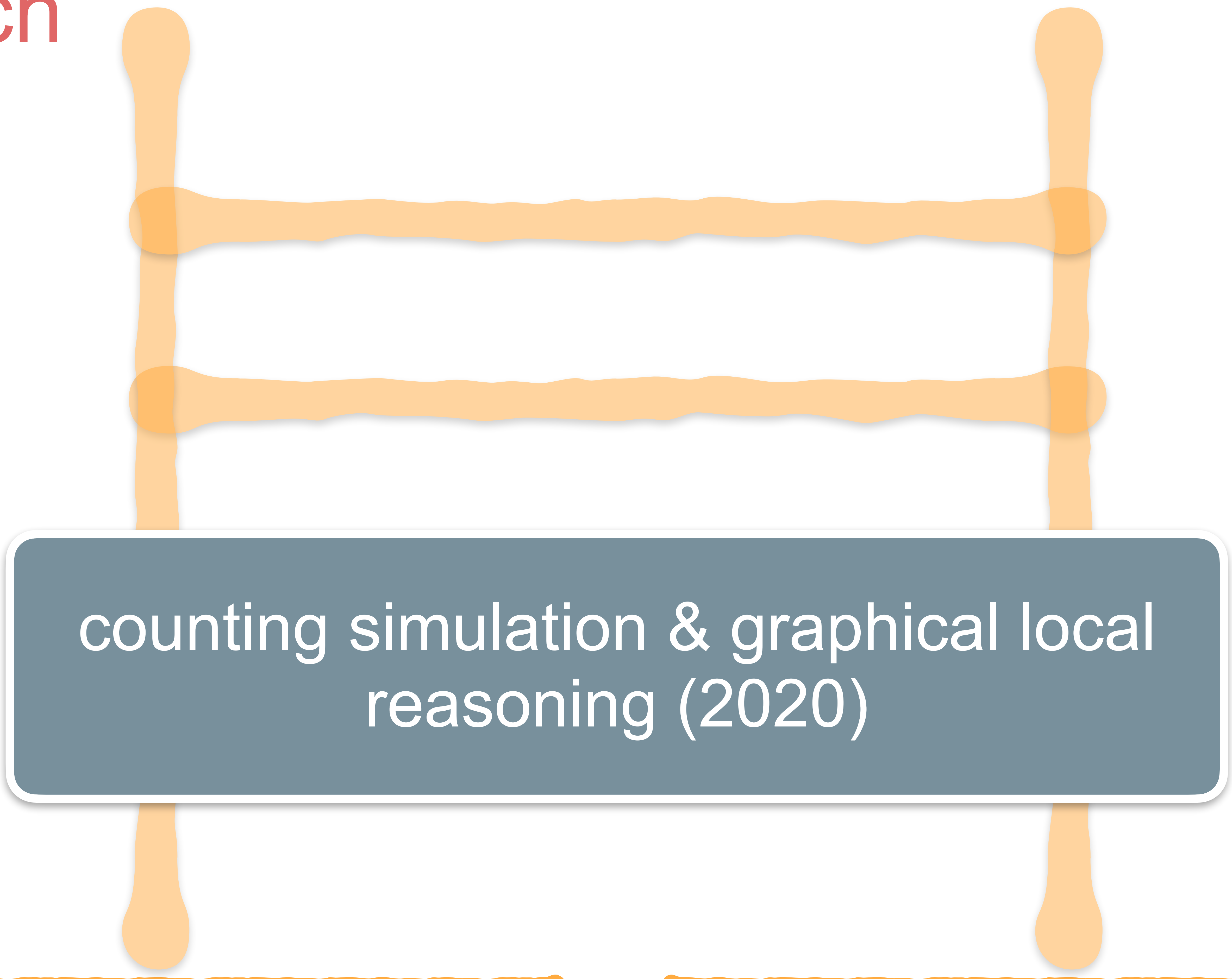
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

Counting simulation

- (2) design a sound "simulation" notion (for observational refinement)
- target: $\preceq^{Q}_{\downarrow}$ for a preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$
    - $t \preceq^{Q}_{\downarrow} u \overset{\Delta}{\iff} \forall C \, . \, C[t] \downarrow^{k} \implies C[u] \downarrow^{m} \wedge k \; Q \; m$
    - Q introduced for a technical reason
        - (will come back to this point)

Counting simulation

- (2) design a sound “simulation” notion (for observational refinement)
- target: $\preceq^{Q}_{\downarrow}$ for a preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$
  - $t \preceq^{Q}_{\downarrow} u \overset{\Delta}{\iff} \forall C\,.\, C[t] \downarrow^{k} \implies C[u] \downarrow^{m} \wedge k \mathrel{Q} m$
- **Def.** (counting simulation)

  R is a Q-counting simulation $\overset{\Delta}{\iff}$
$$
\begin{array}{lcr}
s \to s' \to^{k} s'' & \qquad & s \in F \\
R \,\vdots \qquad\quad \therefore R & & R \,\vdots \\
t \to^{m} t' & & t \in F
\end{array}
$$
- **Prop.** (soundness) If $\overline{\triangleleft}$ is a Q-counting simulation, then $\triangleleft \subseteq \preceq^{Q}_{\downarrow}$.
  - only for deterministic $\to$, to prove by induction

Detour: counting simulation up-to

- namely, for *dependency* of contextual refinements

- Case 1. up to structural congruences
  - e.g. $(\mathtt{let}\ x = t\ \mathtt{in}\ u) \simeq u$ if $x \notin \mathrm{FV}(u)$
  - instead of working with equivalence classes of terms wrt. structural congruences

Detour: counting simulation up-to

- namely, for *dependency* of contextual refinements

- Case 2. up to auxiliary contextual equivalences
  - e.g. $n \simeq m$ for $n, m \in \mathbb{N}$, in the absence of `if`
  - **Q.** Is the call-by-value beta-law $(\lambda x\,.\,t)\ v \triangleleft_\beta t[v/x]$ preserved by `stat`?
    - `stat` inspects memory usage
    - **A1.** No, in the presence of `if`.
      - Try It Online: https://bit.ly/3TqnGOW

Detour: counting simulation up-to

- namely, for *dependency* of contextual refinements

- Case 2. up to auxiliary contextual equivalences
  - e.g. $n \simeq m$ for $n, m \in \mathbb{N}$, in the absence of `if`
  - **Q.** Is the call-by-value beta-law $(\lambda x\,.\,t)\ v \lhd_\beta t[v/x]$ preserved by `stat`?
    - `stat` inspects memory usage
    - **A2.** Yes, in the absence of `if`.
      - The beta-law would depend on the auxiliary law $n \simeq m$.

Detour: counting simulation up-to

- (2) design a sound “simulation” notion (for observational refinement)
- target: $\preceq^{Q}_{\downarrow}$ for a preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$
  - $t \preceq^{Q}_{\downarrow} u \overset{\Delta}{\iff} \forall C\,.\, C[t] \downarrow^{k} \implies C[u] \downarrow^{m} \wedge k \; Q \; m$
- **Def.** (counting simulation up-to)

  R is a Q-counting simulation up to (Q_1, Q_2) $\overset{\Delta}{\iff}$

$$R\,\vdots\;\begin{array}{l} s \to s' \to^{k} s'' \\ \qquad\qquad \therefore\ \dot{\simeq}^{Q_1}_{\downarrow} \circ R \circ \dot{\simeq}^{Q_2}_{\downarrow} \\ t \to^{m} t' \end{array} \qquad R\,\vdots\;\begin{array}{l} s \in F \\ \\ t \in F \end{array}$$

- **Prop.** (soundness) If $\trianglelefteq$ is a Q-counting simulation up to ($\dot{\simeq}^{Q_1}_{\downarrow}$, $\dot{\simeq}^{Q_2}_{\downarrow}$), then $\trianglelefteq \subseteq \preceq^{Q}_{\downarrow}$.
  - only for deterministic $\to$ and *reasonable* (Q, Q_1, Q_2), in particular $Q_1 \subseteq \geq$

Counting simulation

- (2) design a sound "simulation" notion (for observational refinement)
- target: $\preceq^{Q}_{\downarrow}$ for a preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$
  - $t \preceq^{Q}_{\downarrow} u \overset{\Delta}{\iff} \forall C\,.\,C[t]\downarrow^{k} \implies C[u]\downarrow^{m} \wedge k\;Q\;m$
- **Def.** (counting simulation)

  R is a Q-counting simulation $\overset{\Delta}{\iff}$

$$
\begin{array}{ccccccc}
s & \to & s' & \to^{k} & s'' & \qquad & s \in F \\
R \vdots & & & & \therefore R & \qquad & R \vdots \\
t & \to^{m} & t' & & & \qquad & t \in F
\end{array}
$$

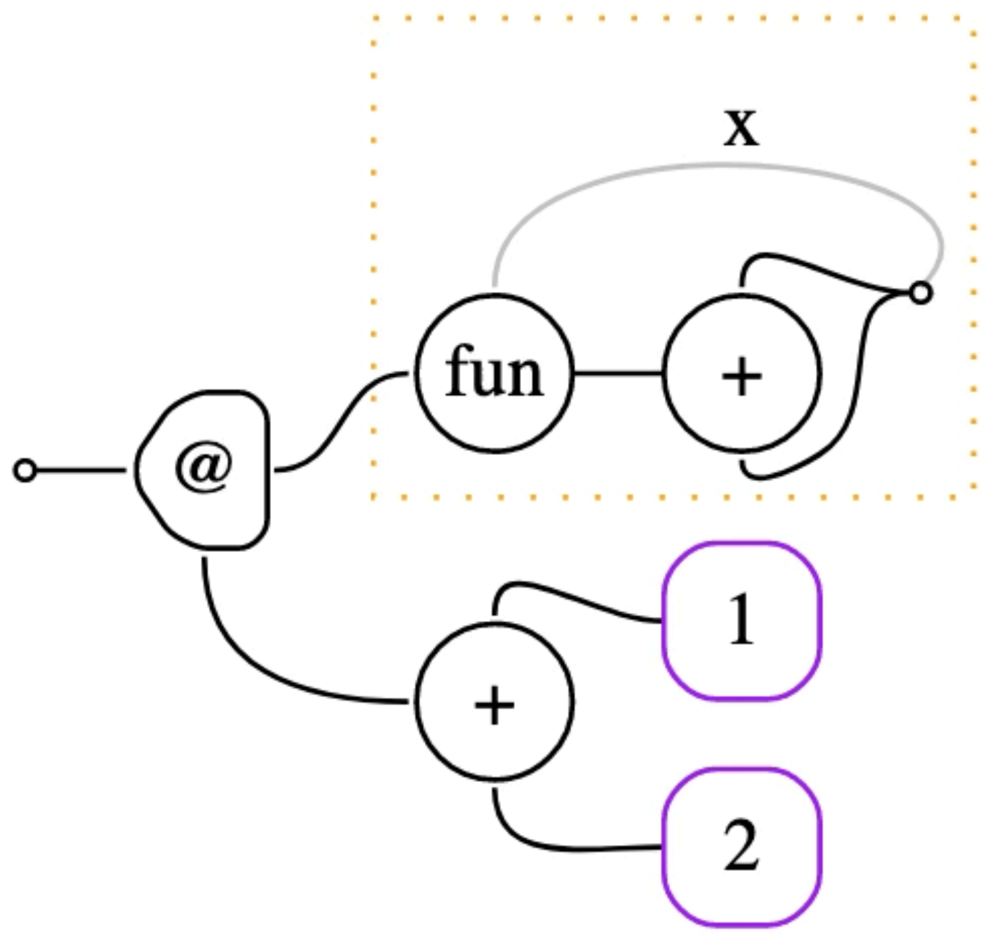
- **Prop.** (soundness) If $\overline{\lhd}$ is a Q-counting simulation, then $\lhd \subseteq \preceq^{Q}_{\downarrow}$.
  - only for deterministic $\to$, to prove by induction
  - **Q.** Can we extend this result to nondeterministic $\to$?

Graphical local reasoning for counting simulation

- (5) prove that $\overline{\triangleleft}$ is a Q-simulation

$$
\begin{array}{ccc}
 & C[\vec{t}] \to s \to^{k} C'[\vec{t'}] \\
\overline{\triangleleft}\vdots & \quad\quad \therefore\,\overline{\triangleleft} \\
 & C[\vec{u}] \to^{m} C'[\overrightarrow{u'}]
\end{array}
$$

- now examining how each $\to$ is defined
  - namely: *token-guided graph rewriting*
    - A token, moving around a graph, substitutes evaluation contexts.

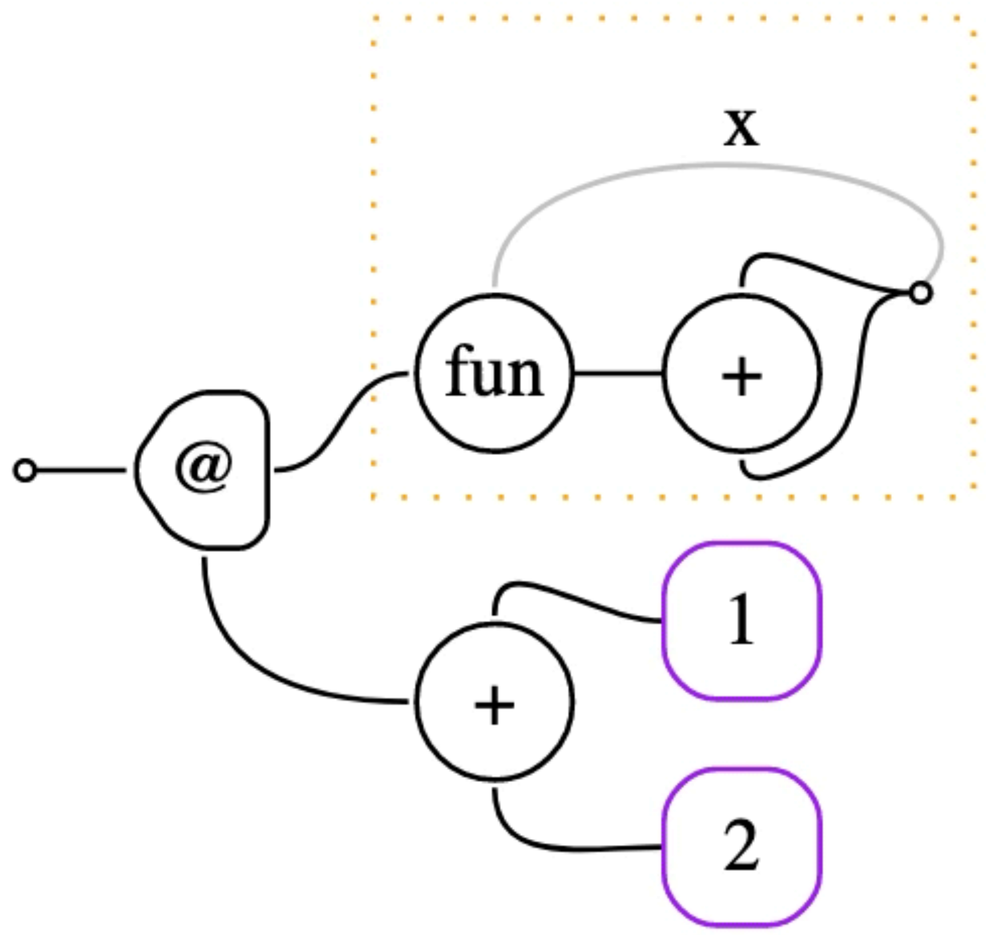


Graphical local reasoning for counting simulation

- (5) prove that $\overline{\triangleleft}$ is a Q-simulation

$$\begin{array}{lcl} & C[\vec{t}] \to s \to^{k} C'[\vec{t'}] \\ \overline{\triangleleft}\,\vdots & & \quad\quad\quad\quad \therefore\, \overline{\triangleleft} \\ & C[\vec{u}] \to^{m} C'[\overrightarrow{u'}] \end{array}$$

- now examining how each $\to$ is defined
  - namely: *token-guided graph rewriting*
    - A token, moving around a graph, substitutes evaluation contexts.

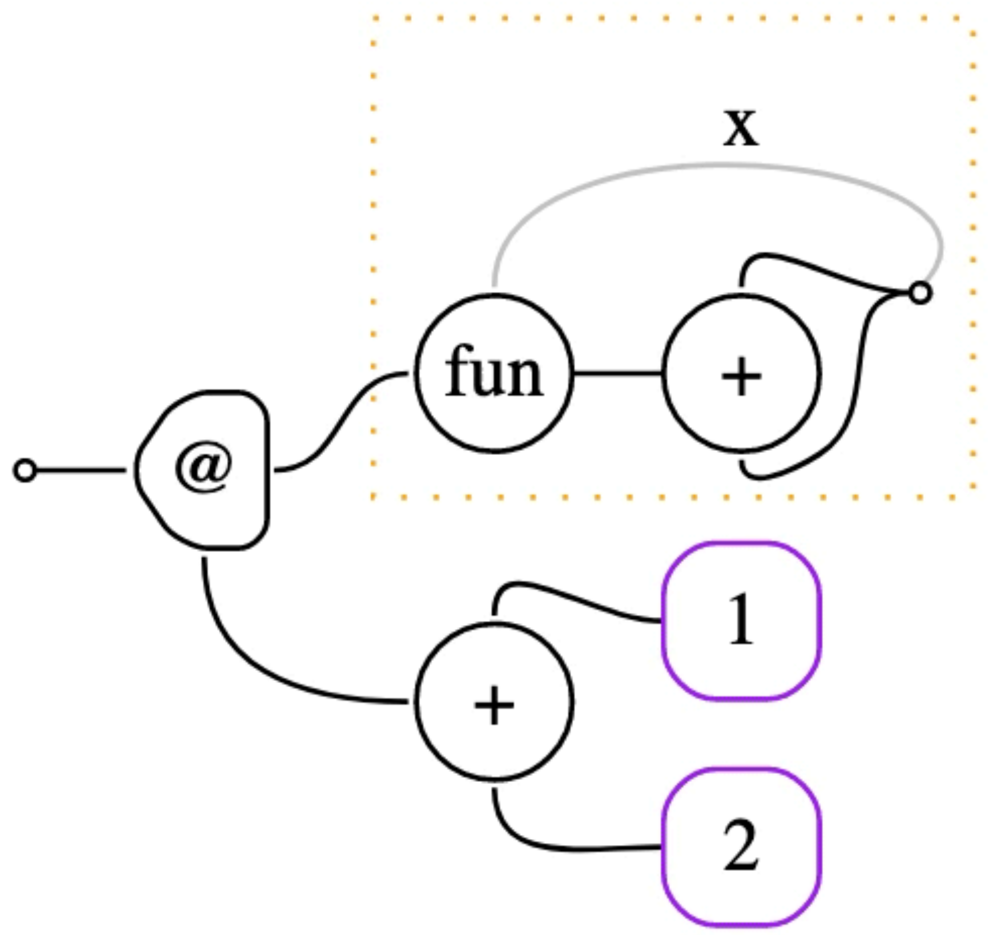


Token-guided graph rewriting

1. A token does depth-first traversal, searching for a redex.

2. The token triggers rewrite of the found redex.

3. Go back to 1.

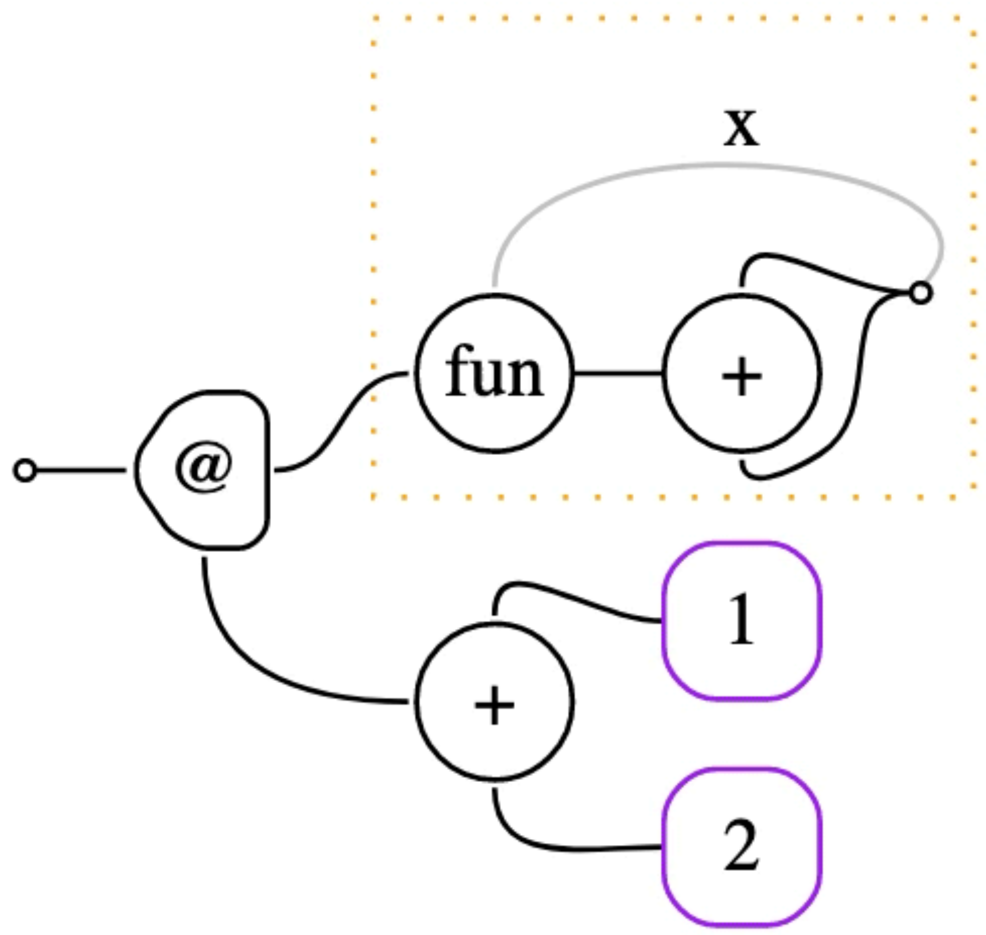


Token-guided graph rewriting

1. A token does depth-first traversal, searching for a redex.

2. The token triggers rewrite of the found redex.

3. Go back to 1.



Graphical local reasoning for counting simulation

- (5) prove that $\overline{\triangleleft}$ is a Q-simulation

$$\begin{array}{lll} & \dot{\mathscr{C}}[\overrightarrow{N}] \to \dot{P} \to^{k} & \dot{\mathscr{C}}'[\overrightarrow{N'}] \\ \overline{\triangleleft}\,\vdots & & \iddots\ \overline{\triangleleft} \\ & \dot{\mathscr{C}}[\overrightarrow{H}] \to^{m} \dot{\mathscr{C}}'[\overrightarrow{H'}] & \end{array}$$

- case analysis on $\dot{\mathscr{C}}[\overrightarrow{N}] \to \dot{P}$ in terms of the token behaviour
  - The token moves inside the context $\dot{\mathscr{C}}$. ==> Always OK.
  - The token visits N_i. ==> OK if $\triangleleft$ is Q-*safe*.
  - The token triggers rewrite. ==> OK if $\triangleleft$ is Q-*robust*.
- **Prop.** If $\triangleleft$ is Q-safe and Q-robust, then $\overline{\triangleleft}$ is a Q-simulation.

Graphical local reasoning for counting simulation

- (5) prove that $\overline{\triangleleft}$ is a Q-simulation

$$
\begin{tikzcd}
\dot{\mathscr{C}}[\overrightarrow{N}] \arrow[r] \arrow[d, dotted, "\overline{\triangleleft}"'] & \dot{P} \arrow[r, "k"] & \dot{\mathscr{C}}'[\overrightarrow{N'}] \\
\dot{\mathscr{C}}[\overrightarrow{H}] \arrow[r, "m"] & \dot{\mathscr{C}}'[\overrightarrow{H'}] \arrow[ur, dotted, "\overline{\triangleleft}"'] &
\end{tikzcd}
$$

- **<u>Prop.</u>** If $\triangleleft$ is Q-safe and Q-robust, then $\overline{\triangleleft}$ is a Q-simulation.
- **Q.** How to prove safety and robustness?
- **A.** By hand.
  - Safety: by feasible pen-and-paper proof.
  - Robustness: by tedious, involved, error-prone, case analysis.
  - **Q'.** Can somebody help the case analysis?

Graphical local reasoning for counting simulation

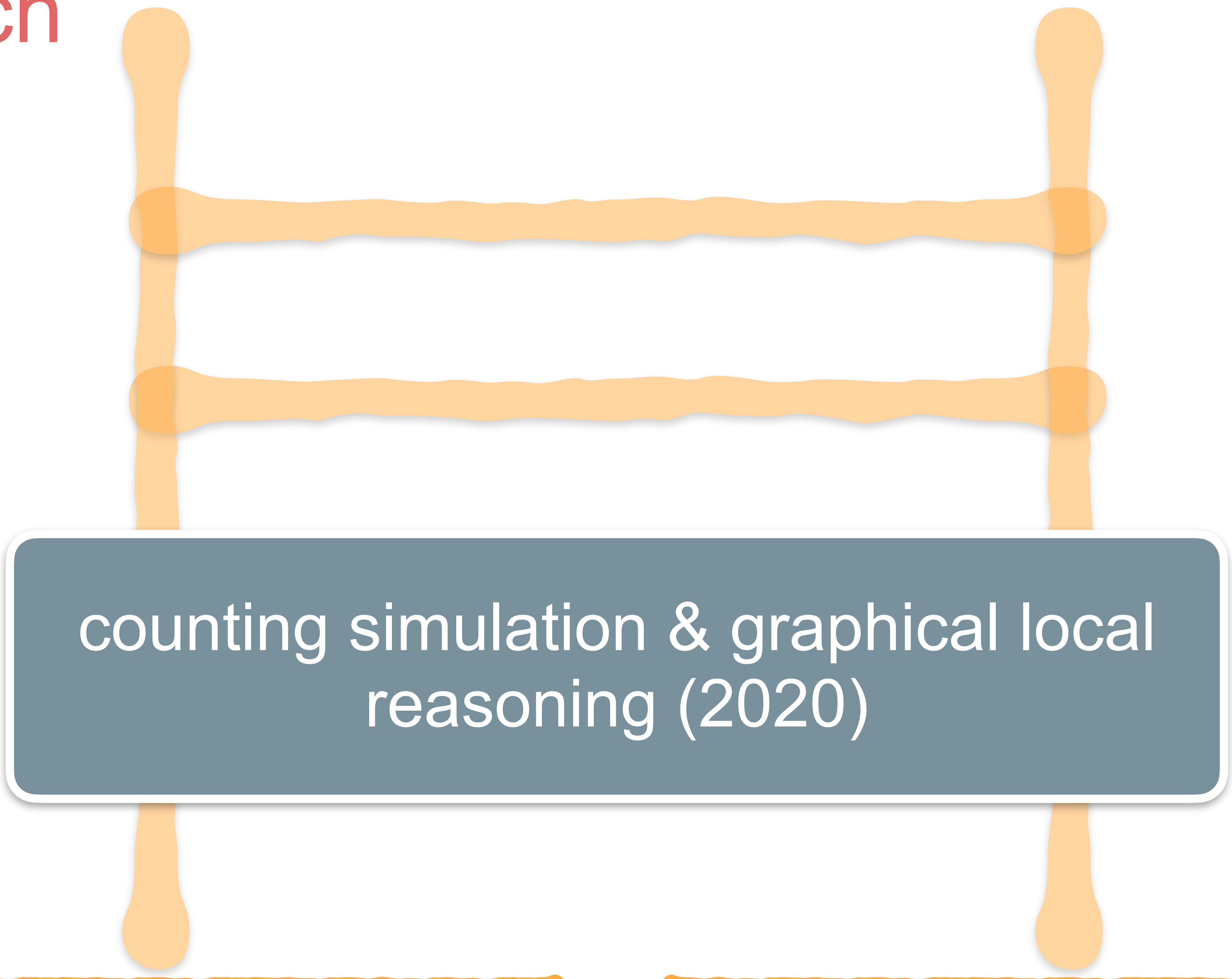
- (5) prove that $\overline{\lhd}$ is a Q-simulation

$$
\begin{array}{llll}
 & \dot{\mathscr{C}}[\overrightarrow{N}] \to \dot{P} & \to^{k} & \dot{\mathscr{C}}'[\overrightarrow{N'}] \\
\overline{\lhd} \vdots & & & \therefore\ \overline{\lhd} \\
 & \dot{\mathscr{C}}[\overrightarrow{H}] & \to^{m} & \dot{\mathscr{C}}'[\overrightarrow{H'}]
\end{array}
$$

- **Prop.** If $\lhd$ is Q-safe and Q-robust, then $\overline{\lhd}$ is a Q-simulation.
- **Q.** How to prove safety and robustness?
- **Q".** Can we do everything with terms and conventional reduction semantics?

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

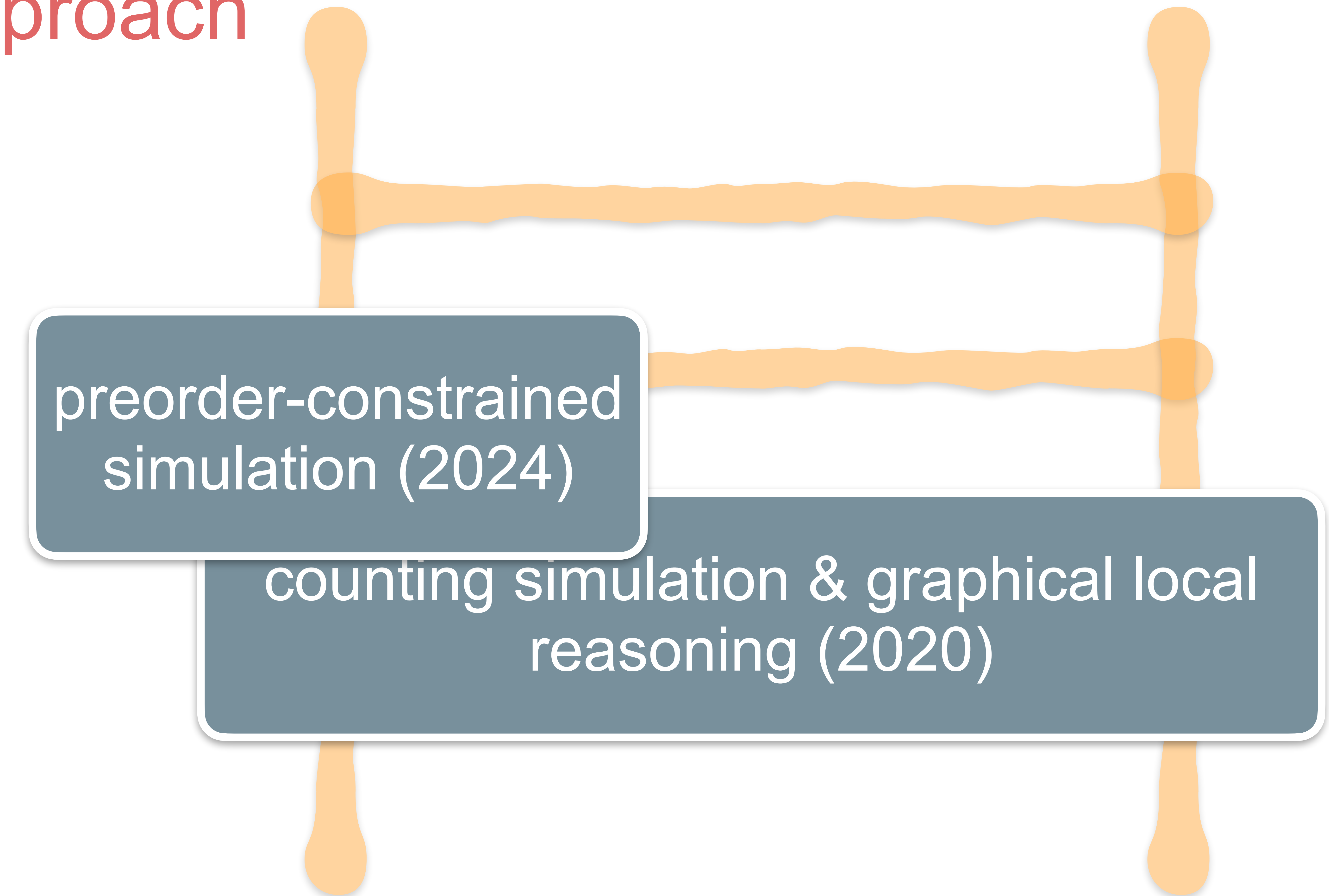
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

Preorder-constrained simulation

- (1) characterise observational refinement as "trace inclusion" of automata
- **Def.** (reduction semantics as NA)

$$\frac{}{E[(\lambda x\,.\,t)\ v] \xrightarrow{\tau} E[t\{v/x\}]} \qquad \frac{}{\mathtt{or}(t_1, t_2) \xrightarrow{\mathtt{or}_i} t_i} \qquad \frac{}{n \xrightarrow{n} \checkmark}$$

$$\frac{}{\mathtt{in}(t_1, t_2) \xrightarrow{\mathtt{in}_i} t_i}$$

- **Def.** ($\mathcal{Q}$-trace inclusion) $x \sqsubseteq^{\mathcal{Q}} y \overset{\Delta}{\iff} \forall w \in L_{\mathcal{A}_1}(x)\,.\,\exists w' \in L_{\mathcal{A}_2}(y)\,.\,w\ \mathcal{Q}\ w'$.
  - for a preorder $\mathcal{Q} \subseteq \Sigma^* \times \Sigma^*$ on words
    - *lifted* preorder $w\,|\,Q\,|\,w' \overset{\Delta}{\iff} |\,w\,|\,Q\,|\,w'\,|$ for $Q \subseteq \mathbb{N} \times \mathbb{N}$
    - *filtered equality* $a\tau\tau b\tau c\tau =_{\mathrm{rem}(\tau)} abc$

Preorder-constrained simulation

- (1) characterise observational refinement as "trace inclusion" of automata
- **Def.** (reduction semantics as NA)

$$\frac{}{E[(\lambda x\,.\,t)\ v] \xrightarrow{\tau} E[t\{v/x\}]} \qquad \frac{}{\mathtt{or}(t_1, t_2) \xrightarrow{\mathtt{or}_i} t_i} \quad \frac{}{\mathtt{in}(t_1, t_2) \xrightarrow{\mathtt{in}_i} t_i} \qquad \frac{}{n \xrightarrow{n} \checkmark}$$

- **Def.** ($\mathcal{Q}$-trace inclusion) $x \sqsubseteq^{\mathcal{Q}} y \overset{\Delta}{\iff} \forall w \in L_{\mathcal{A}_1}(x)\,.\,\exists w' \in L_{\mathcal{A}_2}(y)\,.\,w\ \mathcal{Q}\ w'$.
- **Lem.** $\sqsubseteq^{|Q|\cup =_{\mathrm{rem}(\tau,\mathtt{or})}} = \dot{\preceq}^{Q}_{V}$ for nondeterminism
- **Lem.** $\sqsubseteq^{|Q|\cup =_{\mathrm{rem}(\tau)}} = \dot{\preceq}^{Q}_{V}$ for I/O

$\mathtt{or}(1,2) \dot{\preceq}^{=}_{V} \mathtt{or}(2,1)$ … YES.

$\mathtt{in}(1,2) \dot{\preceq}^{=}_{V} \mathtt{in}(2,1)$ … NO.

Preorder-constrained simulation

- (2) design a sound “simulation” notion (for observational refinement

- **<u>Def.</u>** (counting simulation)

$$R \text{ is a } Q\text{-counting simulation} \overset{\Delta}{\iff} \begin{tikzcd} s \arrow[r] \arrow[d, dotted, no head, "R"'] & s' \arrow[r, "k"] & s'' \\ t \arrow[r, "m"] & t' \arrow[ur, dotted, no head, "R"'] & \end{tikzcd} \qquad \begin{tikzcd} s \in F \arrow[d, dotted, no head, "R"'] \\ t \in F \end{tikzcd}$$

- **<u>Def.</u>** (preorder-constrained simulation)

$$R \text{ is an } M\text{-lookahead } \mathcal{Q}\text{-constrained simulation} \overset{\Delta}{\iff} \begin{tikzcd} s \arrow[r, two heads, "w_1"] \arrow[d, dotted, no head, "R"'] & s_k \arrow[r, two heads, "w_2"] & s_M \\ t \arrow[r, two heads, "w'"] & t' \arrow[u, dotted, no head, "R"'] & \end{tikzcd} \qquad \begin{tikzcd} s \arrow[r, two heads, "w"] \arrow[d, dotted, no head, "R"'] & s_k \in F \\ t \arrow[r, two heads, "w'"] & t' \in F \end{tikzcd}$$

  - idea: swap ∀ and ∃ to fully inspect branches

Preorder-constrained simulation

- (2) design a sound “simulation” notion (for observational refinement

- **<u>Def.</u>** (counting simulation)

  R is a Q-counting simulation $\overset{\Delta}{\iff}$
  $$\begin{array}{ccccc} & s & \to & s' & \to^{k} & s'' \\ R\,\vdots & & & & \therefore\, R & \\ & t & \to^{m} & t' & & \end{array} \qquad \begin{array}{cc} & s \in F \\ R\,\vdots & \\ & t \in F \end{array}$$

- **<u>Def.</u>** (preorder-constrained simulation)

  R is an M-lookahead $\mathcal{Q}$-constrained simulation $\overset{\Delta}{\iff}$
  $$\begin{array}{cccccc} & s & \overset{w_1}{\twoheadrightarrow} & s_k & \overset{w_2}{\twoheadrightarrow} & s_M \\ R\,\vdots & & & \vdots\, R & & \\ & t & \overset{w'}{\twoheadrightarrow} & t' & & \end{array} \qquad \begin{array}{cccc} & s & \overset{w}{\twoheadrightarrow} & s_k \in F \\ R\,\vdots & & & \\ & t & \overset{w'}{\twoheadrightarrow} & t' \in F \end{array}$$

- **<u>Prop.</u>** (soundness) If $\overline{\lhd}$ is an M-lookahead $|Q| \cup =_{\mathrm{rem}(\tau,\mathtt{or})}$ -constrained simulation, then $\lhd \subseteq \preceq^{Q}_{V}$.

  - now for nondeterministic $\to$, but not for probabilistic $\to$ yet (e.g. $\mathtt{or}_{0.5}(1,1) \lhd_{\leq_+} \mathtt{or}_{0.5}(0,1)$)

Preorder-constrained simulation, as a reachability game

- two-player reachability game between Challenger & Simulator

Position	Player	Move	Guard	
$(w, x, y) \in \Sigma^* \times X_1 \times X_2$	Challenger	(wa, x', y)	$x \overset{a}{\rightsquigarrow}_1 x'$	①
		$(\checkmark, w, x, y)$	$x \in F_1$	②
$(w, x', y) \in \Sigma^* \times X_1 \times X_2$	Simulator	(w, x', y)	$\lvert w \rvert < M$	③
		(ε, x', y')	$\exists w' \in \Sigma^*.\ \lvert w' \rvert < N \wedge y \overset{w'}{\twoheadrightarrow}_2 y' \wedge w\mathbf{Q}w'$	④
$(\checkmark, w, x, y) \in \{\checkmark\} \times \Sigma^* \times X_1 \times X_2$	Simulator	sim-win	$\exists w' \in \Sigma^*. \exists y' \in F_2.\ \lvert w' \rvert < N \wedge y \overset{w'}{\twoheadrightarrow}_2 y' \wedge w\mathbf{Q}w'$	⑤

① Challenger chooses $x \overset{a}{\rightsquigarrow}_1 x'$ from the current state x and enqueues the label a.

② Challenger is at an accepting state $x \in F_1$. Challenger forces Simulator to check whether an accepting state is reachable from $y \in X_2$.

③ Simulator skips the turn. This move is always possible when $M = \infty$.

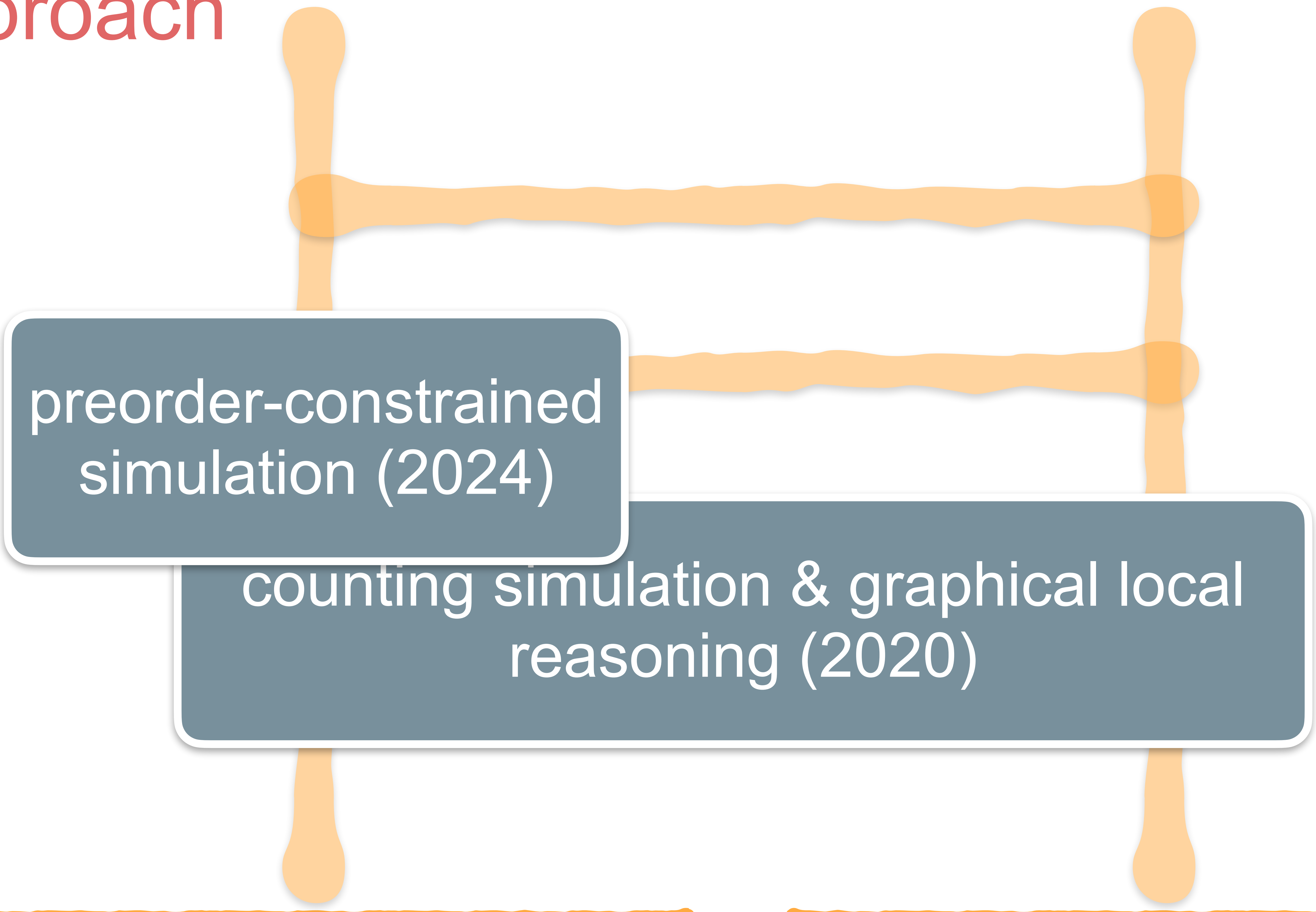
④ Simulator simulates Challenger's moves in the queue w, in less than N transitions.

⑤ Simulator simulates Challenger's moves in the queue w and reaches an accepting state, in less than N transitions.

Figure 9 Two-player game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ characterising (M-bounded) $\mathbf{Q}$-constrained simulation.

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) "simulation" notions

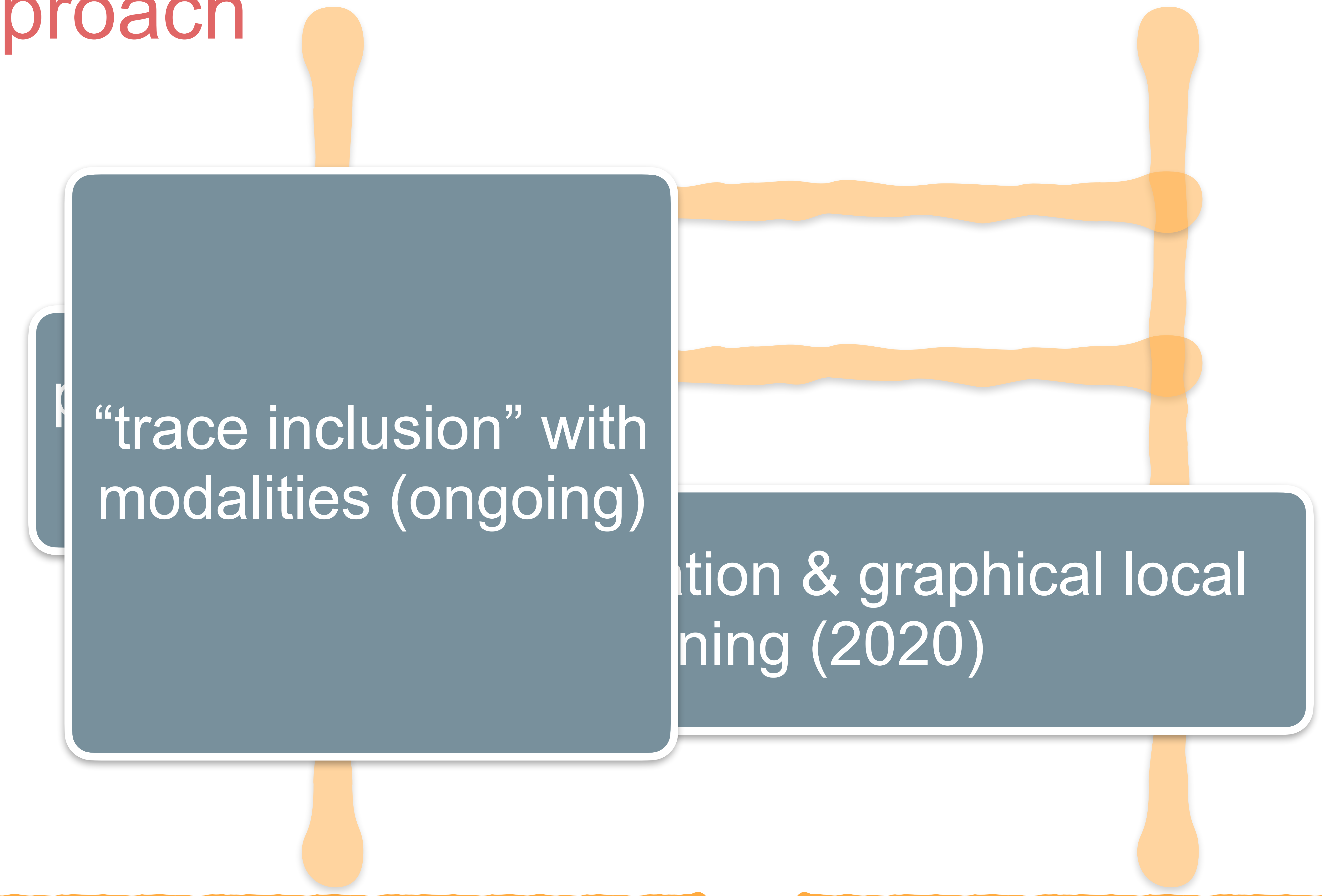
forgetting how each → is defined

The right bar:
(5) "simulation" proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

"Trace inclusion" with modalities

- (1) characterise observational refinement as "trace inclusion" of automata
- **Def.** (reduction semantics as NA)

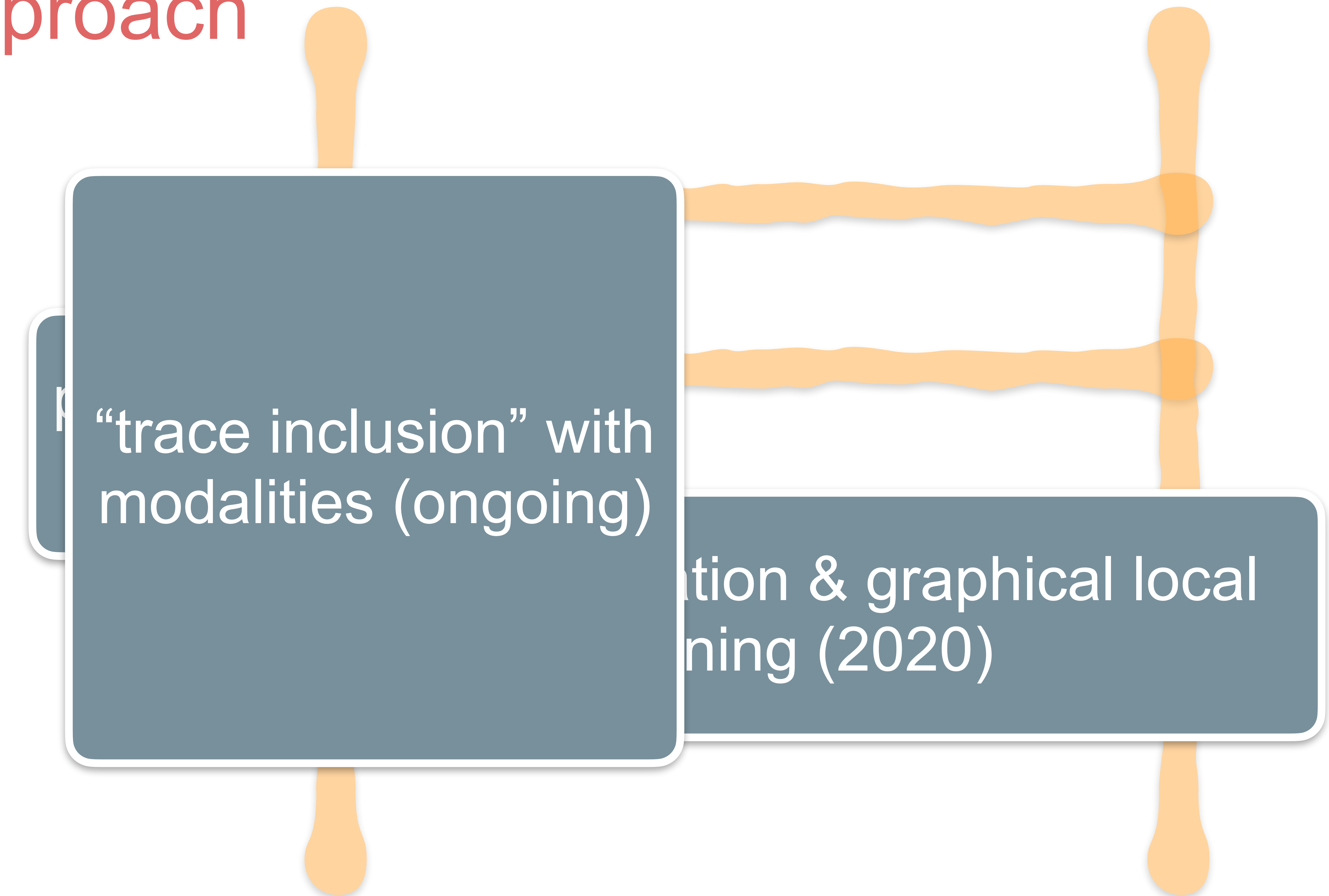
$$\frac{}{E[(\lambda x\,.\,t)\ v] \xrightarrow{\tau} E[t\{v/x\}]} \qquad \frac{}{\mathtt{or}(t_1, t_2) \xrightarrow{\mathtt{or}_i} t_i} \qquad \frac{}{n \xrightarrow{n} \checkmark}$$

$$\frac{}{\mathtt{in}(t_1, t_2) \xrightarrow{\mathtt{in}_i} t_i}$$

- **Def.** ($\mathcal{O}$-trace inclusion) $x \sqsubseteq^{\mathcal{O}} y \overset{\Delta}{\iff} \cdots$
- **Goal** $\sqsubseteq^{\mathcal{O}} = \dot{\preceq}^{\mathcal{O}}_{V}$ for various algebraic effects
    - $\dot{\preceq}^{\mathcal{O}}_{V}$: observational refinement with *modalities* [Simpson+ '18]
    - altering modalities —> adjusting observation to various effects

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

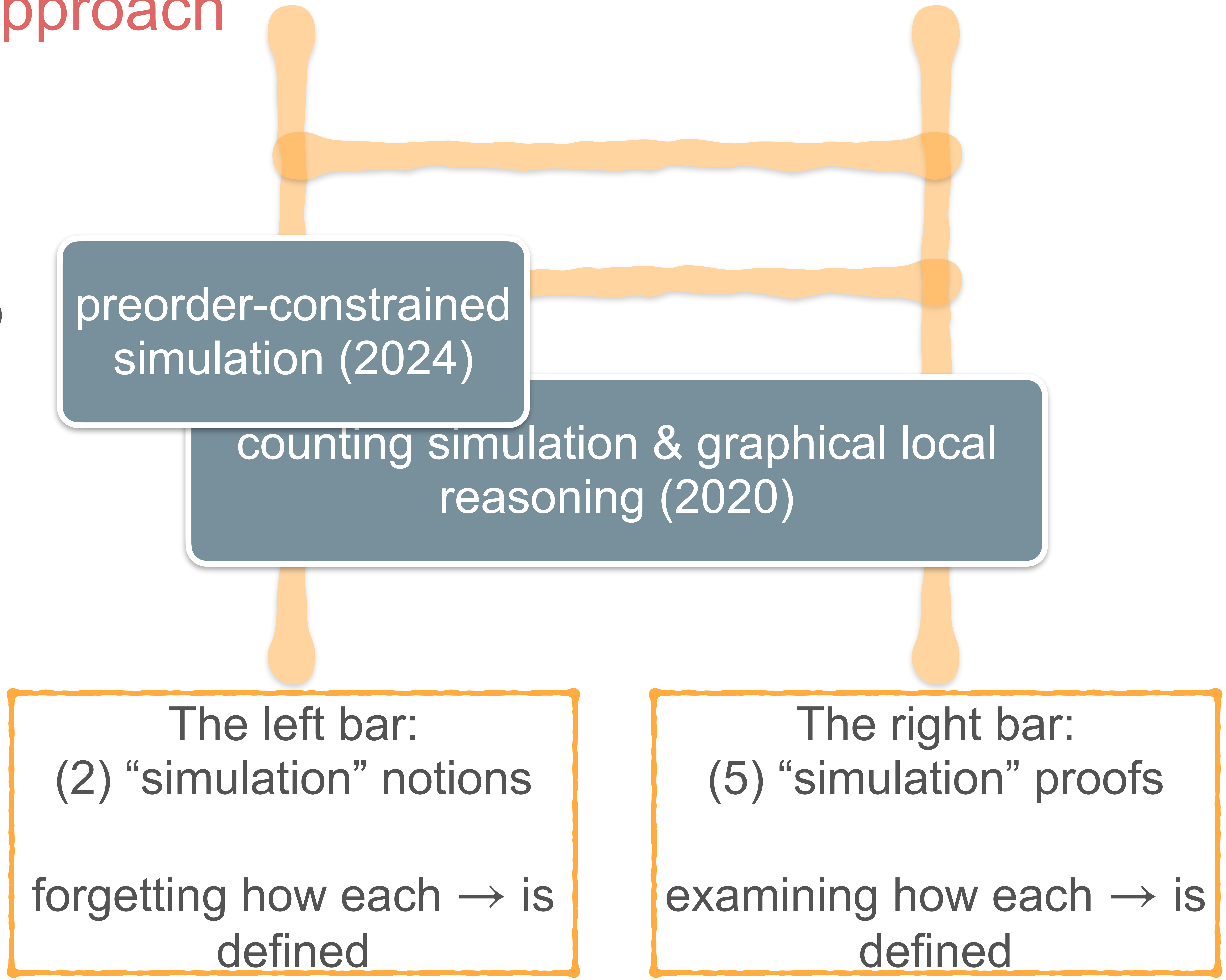
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

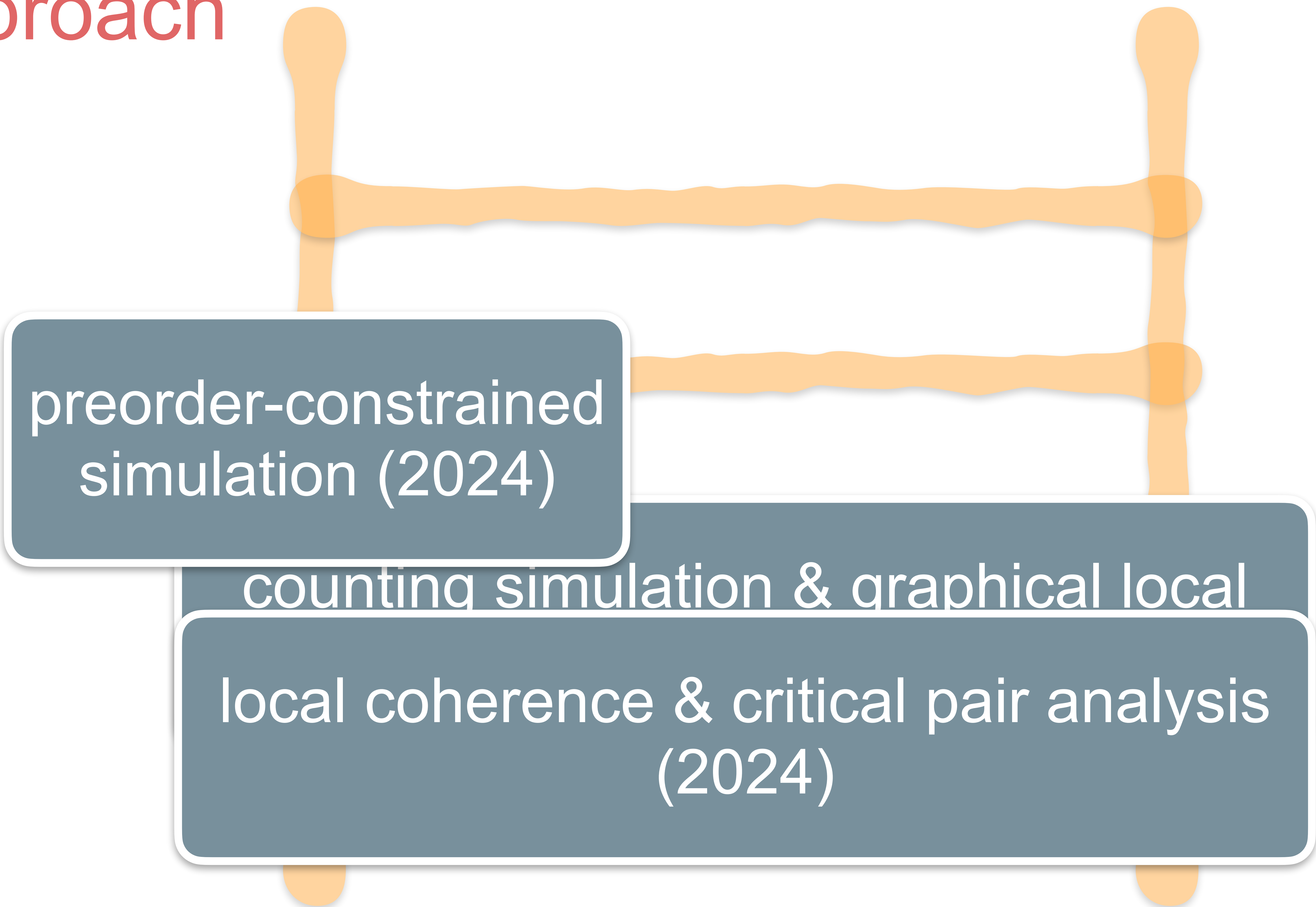
A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion (for observational refinement)
  - (3) take a candidate $\lhd$ of contextual refinement
  - (4) take the contextual closure $\overline{\lhd}$ (i.e. $\forall C\,.\,C[\vec{t}]\ \overline{\lhd}\ C[\vec{u}] \iff \forall i\,.\,t_i \lhd u_i$)
  - (5) prove that $\overline{\lhd}$ is a “simulation”
- (1-2) for all $\lhd$, (3-5) for each $\lhd$

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion (for observational refinement)
  - (3) take a candidate $\triangleleft$ of contextual refinement
  - (4) take the contextual closure $\overline{\triangleleft}$ (i.e. $\forall C\,.\,C[\vec{t}]\ \overline{\triangleleft}\ C[\vec{u}] \iff \forall i\,.\,t_i \triangleleft u_i$)
  - (5) prove that $\overline{\triangleleft}$ is a counting simulation
- only for deterministic $\rightarrow$

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as "trace inclusion" of automata
  - (2) design a sound "simulation" notion (for observational refinement)
  - (3) take a *refinement rule* $(l \Rightarrow r) \in \mathscr{R}$, a candidate of ctx. refinement
  - (4) take the contextual closure $\overline{\lhd}$ (i.e. $\forall C \,.\, C[\vec{t}] \mathrel{\overline{\lhd}} C[\vec{u}] \iff \forall i \,.\, t_i \lhd u_i$)
  - (5) prove that $\overline{\lhd}$ is a counting simulation
- only for deterministic $\rightarrow$

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion (for observational refinement)
  - (3) take a *refinement rule* $(l \Rightarrow r) \in \mathscr{R}$, a candidate of ctx. refinement
  - (4) take the *refinement* relation $\Rightarrow_{\mathscr{R}}$ (i.e. $\dfrac{(l \Rightarrow r) \in \mathscr{R} \quad C \in Ctx}{C[l\theta] \Rightarrow_{\mathscr{R}} C[r\theta]}$)
  - (5) prove that $\overline{\lhd}$ is a counting simulation
- only for deterministic $\to$

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion (for observational refinement)
  - (3) take a *refinement rule* $(l \Rightarrow r) \in \mathscr{R}$, a candidate of ctx. refinement
  - (4) take the *refinement* relation $\Rightarrow_{\mathscr{R}}$ (i.e. $\dfrac{(l \Rightarrow r) \in \mathscr{R} \quad C \in Ctx}{C[l\theta] \Rightarrow_{\mathscr{R}} C[r\theta]}$)
  - (5) prove that $\Rightarrow_{\mathscr{R}}$ is *locally coherent*
- only for deterministic $\rightarrow$

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion (for observational refinement)
  - (3) take a *refinement rule* $(l \Rightarrow r) \in \mathscr{R}$, a candidate of ctx. refinement
  - (4) take the *refinement* relation $\Rightarrow_{\mathscr{R}}$ (i.e. $\dfrac{(l \Rightarrow r) \in \mathscr{R} \quad C \in Ctx}{C[l\theta] \Rightarrow_{\mathscr{R}} C[r\theta]}$)
  - (5) prove that $\Rightarrow_{\mathscr{R}}$ is *locally coherent*
- only for the deterministic *evaluation relation* $\rightarrow_{\mathscr{E}}$ (i.e. $\dfrac{(l \rightarrow r) \in \mathscr{E} \quad E \in Ectx}{E[l\theta] \rightarrow_{\mathscr{E}} E[r\theta]}$)

A new *evitcudnioc* approach from rewriting perspective

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as "trace inclusion" of automata
  - (2) design a sound "simulation" notion (for observational refinement)
  - (3) take a *refinement rule* $(l \Rightarrow r) \in \mathscr{R}$, a candidate of ctx. refinement
  - (4) take the *refinement* relation $\Rightarrow_{\mathscr{R}}$ (i.e. $\dfrac{(l \Rightarrow r) \in \mathscr{R} \quad C \in Ctx}{C[l\theta] \Rightarrow_{\mathscr{R}} C[r\theta]}$)
  - (5) prove that $\Rightarrow_{\mathscr{R}}$ is *locally coherent*
- only for the deterministic *evaluation relation* $\rightarrow_{\mathscr{E}}$ (i.e. $\dfrac{(l \rightarrow r) \in \mathscr{E} \quad E \in Ectx}{E[l\theta] \rightarrow_{\mathscr{E}} E[r\theta]}$)

standard term rewriting

new kind of term rewriting

Local coherence

- (2) design a sound "simulation" notion (for observational refinement)
- target: $\preceq^{\geq}_{V}$ (Sands' *improvement*)
  - $t \preceq^{\geq}_{V} u \overset{\Delta}{\iff} \forall C \,.\, C[t] \to^{k} v \implies C[u] \to^{m} v \wedge k \geq m$
  - $\geq$ chosen for a technical reason
- **Def.** (local coherence)

  $\Rightarrow_{\mathscr{R}}$ is locally coherent $\overset{\Delta}{\iff}$

$$
\begin{array}{ccccc}
s & \to_{\mathscr{E}} & s' & \to^{k}_{\mathscr{E}} & s'' \\
\Downarrow_{\mathscr{R}} & & & \swarrow\!\!\swarrow^{*}_{\mathscr{R}} & \\
t & \to^{m}_{\mathscr{E}} & t' & &
\end{array}
$$

a notion taken from equational rewriting

- **Prop.** (soundness) If $\Rightarrow_{\mathscr{R}}$ is locally coherent, then $\Rightarrow_{\mathscr{R}} \subseteq \preceq^{\geq}_{V}$.
  - only for deterministic $\to_{\mathscr{E}}$ and *value-invariant* $\Rightarrow_{\mathscr{R}}$

Critical pair analysis for local coherence

- (5) prove that $\Rightarrow_{\mathscr{R}}$ is locally coherent

- <u>**Thm.**</u> (critical pair theorem) $\Rightarrow_{\mathscr{R}}$ is locally coherent iff every critical pair is joinable.

  - joinability:

$$\begin{tikzcd} s \arrow[r, "\mathscr{E}"'] \arrow[d, Rightarrow, "\mathscr{R}"] & s' \arrow[r, "k"] \arrow[r, "\mathscr{E}"'] & s'' \arrow[dl, Rightarrow, "*"', "\mathscr{R}"] \\ t \arrow[r, "m"] \arrow[r, "\mathscr{E}"'] & t' & \end{tikzcd}$$

  - only for evaluation-context-preserving $\Rightarrow_{\mathscr{R}}$, linear refinement rules, left-linear evaluation rules, …

Critical pairs can be **automatically** enumerated & checked for joinability!

- <u>**Ex.**</u> the call-by-value λ-calculus,

  the computational λ-calculus with (shallow) effect handlers

2 critical pairs

10 critical pairs

Critical pair analysis for local coherence

- **Ex.** the call-by-value λ-calculus

Signature Σ	$\lambda : \langle 1 \rangle,\ \ @ : \langle 0, 0 \rangle$
Syntax class *Sclass*	**values** $\quad V ::= \lambda x.t$
Evaluation contexts *Ectx*	$E ::= \square \mid E\ t \mid v\ E$
Evaluation rules $\mathcal{E}$	**Refinement rules** $\mathcal{R}$
$(\lambda x.M[x])\ V \to M[V]$	$(\lambda x.M[x])\ V \Rightarrow M[V]$
	$\lambda x.V\ x \Rightarrow V$

$$\begin{tikzcd}
 & (\lambda x.V\ x)\ V' \arrow[dl, Rightarrow] \arrow[dr] & \\
V\ V' \arrow[rr, equal] & & V\ V'
\end{tikzcd}$$

$$\begin{tikzcd}
 & (\lambda x'.M'[x'])\ (\lambda x.V\ x) \arrow[dl, Rightarrow] \arrow[dr] & \\
(\lambda x'.M'[x'])\ V \arrow[dr] & & M'[\lambda x.V\ x] \arrow[dl, Rightarrow] \\
 & M'[V] &
\end{tikzcd}$$

Critical pair analysis for local coherence

- **Ex.** the computational λ-calculus with (shallow) effect handlers

Signature Σ

$\mathtt{true}\colon\langle 0\rangle$, $\mathtt{false}\colon\langle 0\rangle$, $\mathtt{fun}\colon\langle 1\rangle$, $@\colon\langle 0,0\rangle$, $\mathtt{return}\colon\langle 0\rangle$, $\mathtt{op}_1\colon\langle 0,1\rangle$, $\mathtt{op}_2\colon\langle 0,1\rangle$,
$\mathtt{handler}_1\colon\langle 1,2\rangle$, $\mathtt{handler}_0\colon\langle 1\rangle$, $\mathtt{do}\colon\langle 0,1\rangle$, $\mathtt{if}\colon\langle 0,0,0\rangle$, $\mathtt{with\_handle}\colon\langle 0,0\rangle$

Syntax class *Sclass*

functions $F ::= x \mid \mathtt{fun}(x.P)$

values $V ::= \mathtt{true} \mid \mathtt{false} \mid F \mid H$

handlers $H ::= \mathtt{handler}_1(x.P, x.k.P_1) \mid \mathtt{handler}_0(x.P)$

computations $P, P_1, P_2 ::= \mathtt{return}(V) \mid \mathtt{op}(V, y.P) \mid \mathtt{do}(P_1, x.P_2)$
$\mid \mathtt{if}(V, P_1, P_2) \mid F\ V \mid \mathtt{with\_handle}(H, P)$

Evaluation contexts *Ectx* $E ::= \square \mid \mathtt{do}(E, x.P) \mid \mathtt{with\_handle}(H, E)$

Evaluation rules $\mathcal{E}$ where $i \in [2]$

$$\mathtt{do}(\mathtt{return}(V), x.P[x]) \to P[V] \tag{1}$$
$$\mathtt{do}(\mathtt{op}_i(V, y.P_1[y]), x.P_2[x]) \to \mathtt{op}_i(V, y.\mathtt{do}(P_1[y], x.P_2[x])) \tag{2}$$
$$\mathtt{if}(\mathtt{true}, P_1, P_2) \to P_1 \tag{3}$$
$$\mathtt{if}(\mathtt{false}, P_1, P_2) \to P_2 \tag{4}$$
$$\mathtt{fun}(x.P[x])\ V \to P[V] \tag{5}$$

In the following three rules, $h_1 \equiv \mathtt{handler}_1(x.P[x], x.k.P_1[x,k])$.

$$\mathtt{with\_handle}(h_1, \mathtt{return}(V)) \to P[V] \tag{6}$$
$$\mathtt{with\_handle}(h_1, \mathtt{op}_1(V, y.P'[y])) \to P_1[V, \mathtt{fun}(y.P'[y])] \tag{7}$$
$$\mathtt{with\_handle}(h_1, \mathtt{op}_2(V, y.P'[y])) \to \mathtt{op}_2(V, y.\mathtt{with\_handle}(h_1, P'[y])) \tag{8}$$

In the following two rules, $h_0 \equiv \mathtt{handler}_0(x.P[x])$.

$$\mathtt{with\_handle}(h_0, \mathtt{return}(V)) \to P[V] \tag{9}$$
$$\mathtt{with\_handle}(h_0, \mathtt{op}_i(V, y.P'[y])) \to \mathtt{op}_i(V, y.\mathtt{with\_handle}(h_0, P'[y])) \tag{10}$$

Refinement rules $\mathcal{R}$

$$\mathtt{do}(P, x.\mathtt{return}(x)) \Rightarrow P \tag{r3}$$
$$\mathtt{do}(\mathtt{do}(P_1, x_1.P_2[x_1]), x_2.P_3[x_2]) \Rightarrow \mathtt{do}(P_1, x_1.\mathtt{do}(P_2[x_1], x_2.P_3[x_2])) \tag{r4}$$
$$\mathtt{fun}(x.F\ x) \Rightarrow F \tag{r9}$$
$$\mathtt{with\_handle}(\mathtt{handler}_0(x.P[x]), P') \Rightarrow \mathtt{do}(P', x.P[x]) \tag{r13}$$

Critical pair analysis for local coherence

- **<u>Ex.</u>** the computational λ-calculus with (shallow) effect handlers

$$
\begin{tikzcd}[column sep=small]
& \mathsf{do}(\mathtt{return}(V), x.\mathtt{return}(x)) \arrow[dl, Rightarrow, "r3"'] \arrow[dr, "1"] & \\
\mathtt{return}(V) \arrow[rr, equal] & & \mathtt{return}(V)
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& \mathsf{do}(\mathsf{op}_i(V, y.P[y]), x.\mathtt{return}(x)) \arrow[dl, Rightarrow, "r3"'] \arrow[dr, "2"] & \\
\mathsf{op}_i(V, y.P[y]) & & \mathsf{op}_i(V, y.\mathsf{do}(P[y], x.\mathtt{return}(x))) \arrow[ll, Rightarrow, "r3"']
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& & \mathsf{do}(\mathsf{do}(\mathtt{return}(V), x.P[x]), x'.P'[x']) \arrow[dll, Rightarrow, "r4"'] \arrow[dr, "1"] & \\
\mathsf{do}(\mathtt{return}(V), x.\mathsf{do}(P[x], x'.P'[x'])) \arrow[dr, "1"'] & & & \mathsf{do}(P[V], x'.P'[x']) \arrow[dll, equal] \\
& \mathsf{do}(P[V], x'.P'[x']) & &
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& \mathsf{do}(\mathsf{do}(\mathsf{op}_i(V, x.P[x]), y.P_2[y]), z.P_3[z]) \arrow[dl, Rightarrow, "r4"'] \arrow[dr, "2"] & \\
\mathsf{do}(\mathsf{op}_i(V, x.P[x]), y.\mathsf{do}(P_2[y], z.P_3[z])) \arrow[d, "2"'] & & \mathsf{do}(\mathsf{op}_i(V, x.\mathsf{do}(P[x], y.P_2[y])), z.P_3[z]) \arrow[d, "2"] \\
\mathsf{op}_i(V, x.\mathsf{do}(P[x], y.\mathsf{do}(P_2[y], z.P_3[z]))) & & \mathsf{op}_i(V, x.\mathsf{do}(\mathsf{do}(P[x], y.P_2[y]), z.P_3[z])) \arrow[ll, Rightarrow, "r4"']
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& \mathsf{fun}(x.V\ x)\ V' \arrow[dl, Rightarrow, "r9"'] \arrow[dr, "5"] & \\
V\ V' \arrow[rr, equal] & & V\ V'
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& & \mathtt{with\_handle}(h_0, \mathtt{return}(V)) \arrow[dll, Rightarrow, "r13"'] \arrow[dr, "9"] & \\
\mathsf{do}(\mathtt{return}(V), x.P[x]) \arrow[dr, "1"'] & & & P[V] \arrow[dll, equal] \\
& P[V] & &
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
& \mathtt{with\_handle}(h_0, \mathsf{op}_i(V, y.P'[y])) \arrow[dl, Rightarrow, "r13"'] \arrow[dr, "10"] & \\
\mathsf{do}(\mathsf{op}_i(V, y.P'[y]), x.P[x]) \arrow[dr, "2"'] & & \mathsf{op}_i(V, y.\mathtt{with\_handle}(h_0, P'[y])) \arrow[dl, Rightarrow, "r13"] \\
& \mathsf{op}_i(V, y.\mathsf{do}(P'[y], x.P[x])) &
\end{tikzcd}
$$

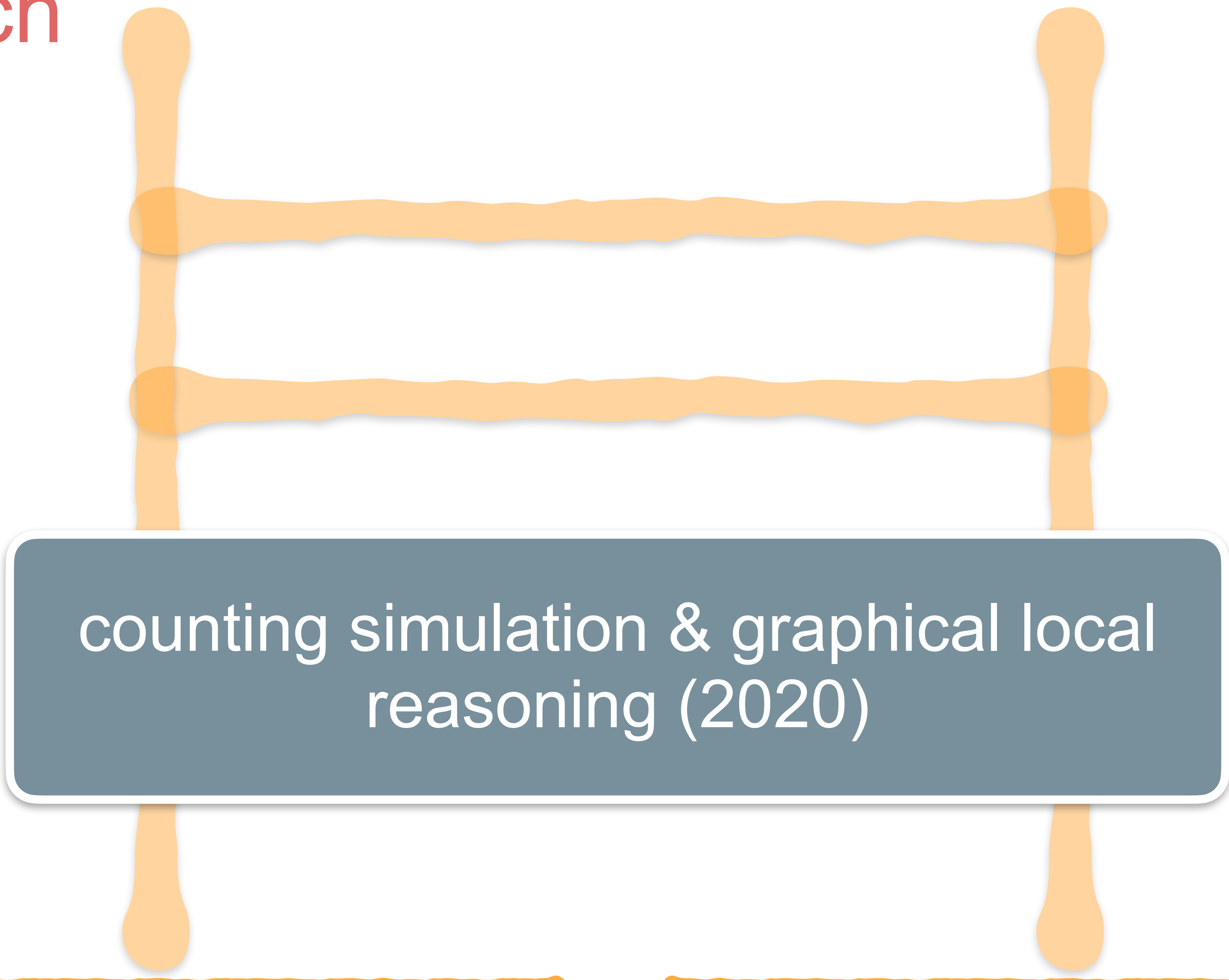
A new *evitcudnioc* approach

- yet another coinductive proof methodology for contextual refinement
  - (1) characterise observational refinement as “trace inclusion” of automata
  - (2) design a sound “simulation” notion
  - (3) take a candidate $\triangleleft$ of contextual refinement
  - (4) take the contextual closure $\overline{\triangleleft}$ (i.e. $\forall C \,.\, C[\vec{t}] \mathrel{\overline{\triangleleft}} C[\vec{u}] \iff \forall i \,.\, t_i \triangleleft u_i$)
  - (5) prove that $\overline{\triangleleft}$ is a “simulation”

- (1-2) for all $\triangleleft$, (3-5) for each $\triangleleft$

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



M. PhD thesis 2020

The left bar:
(2) “simulation” notions

forgetting how each → is defined

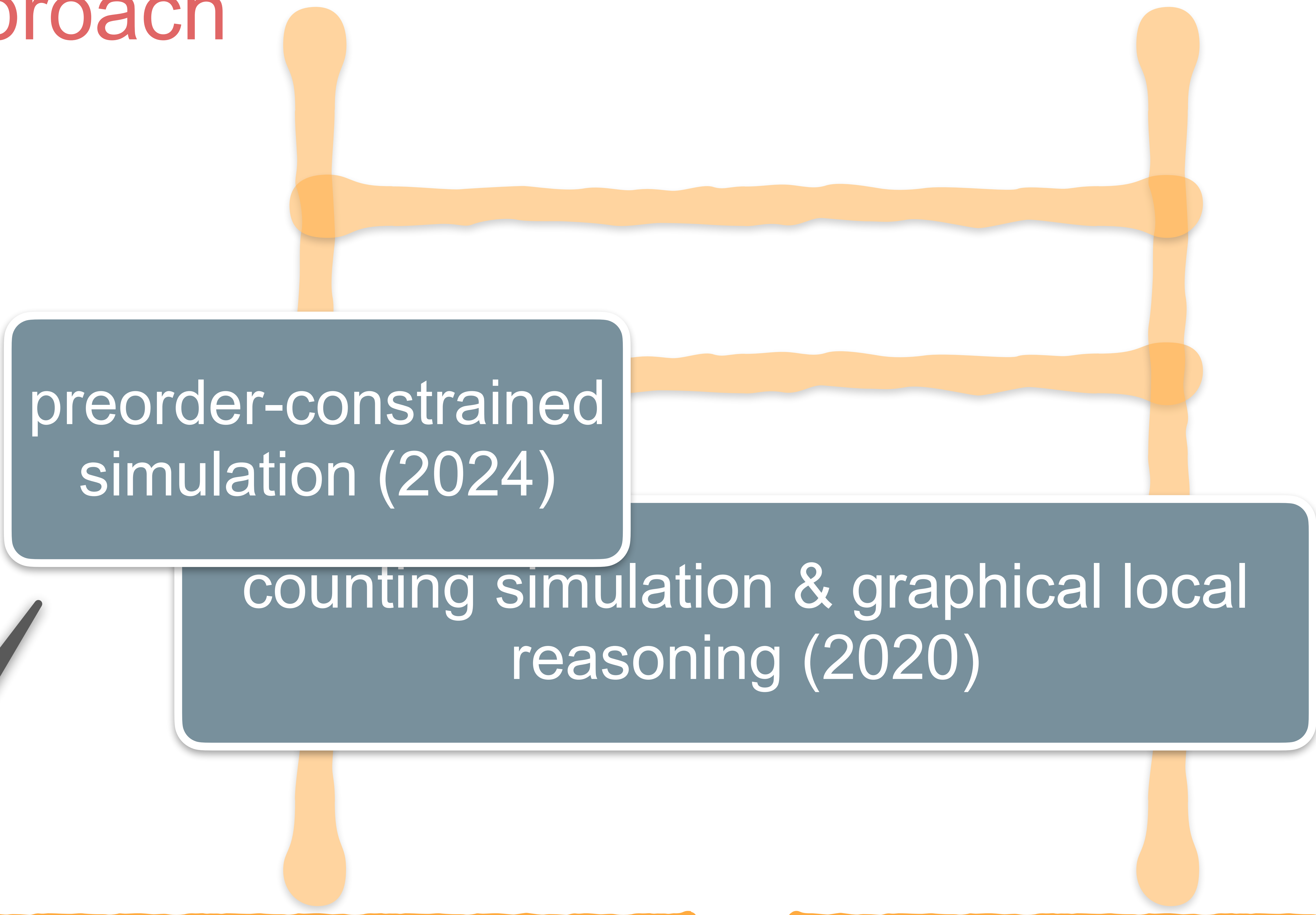
The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence

M., Sanada & Urabe
CMCS 2024



The left bar:
(2) “simulation” notions

forgetting how each → is defined

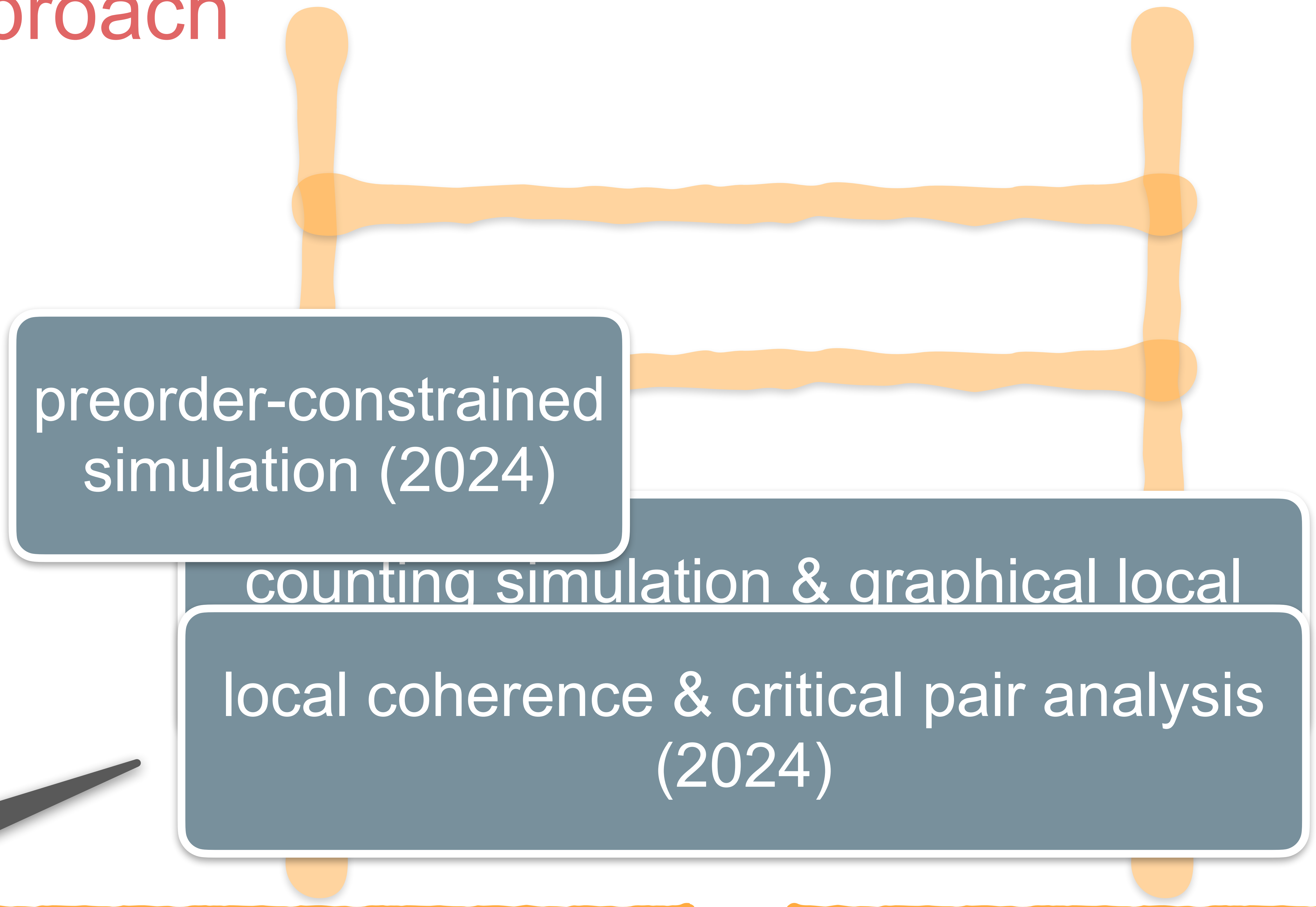
The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence

M. & Hamana
FLOPS 2024



The left bar:
(2) “simulation” notions

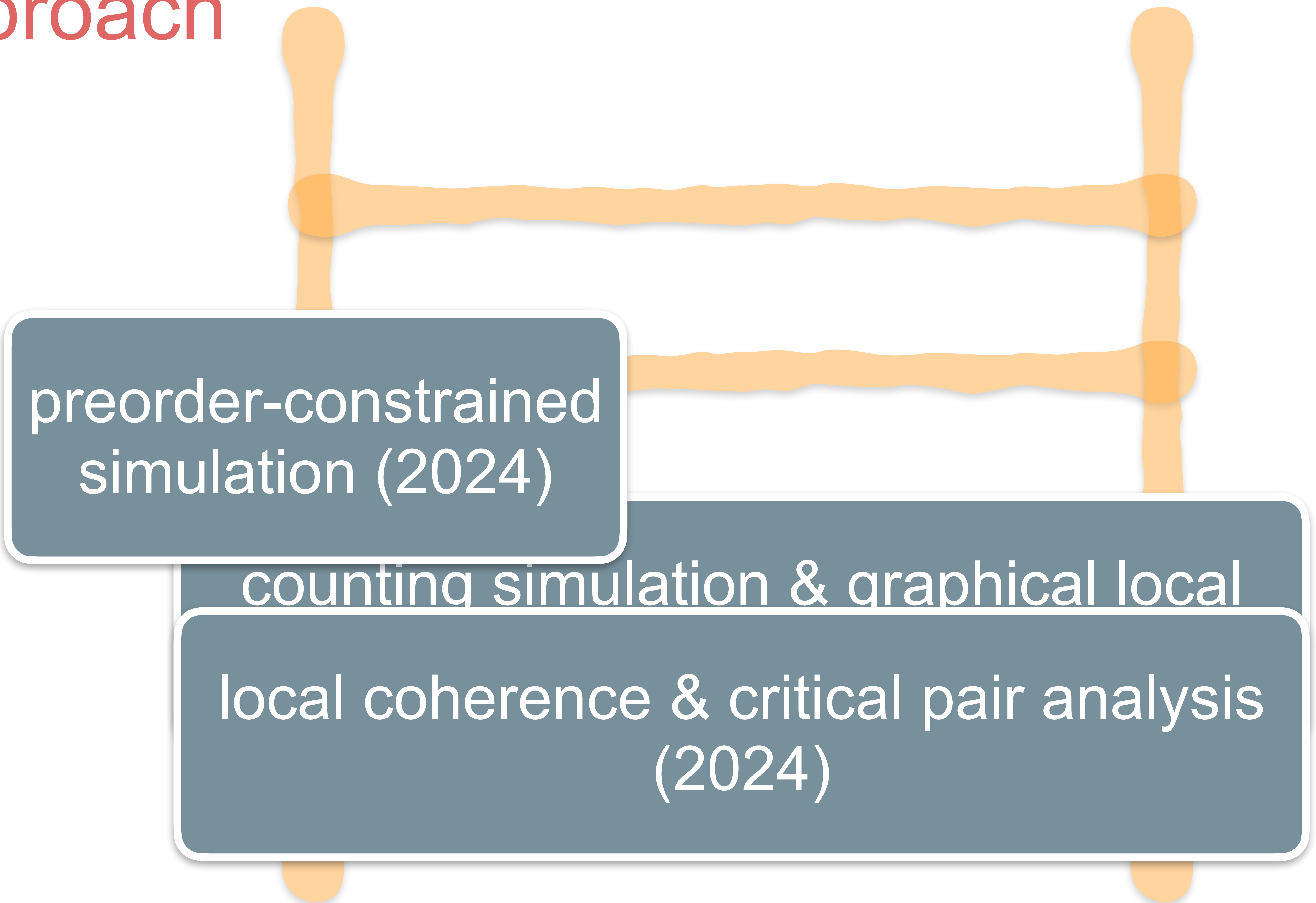
forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined

A new *evitcudnioc* approach

- climbing up a ladder
  - probability
  - nondeterminism, I/O
  - state
  - divergence



- potential side-steps
  - call-by-need
  - continuation

The left bar:
(2) “simulation” notions

forgetting how each → is defined

The right bar:
(5) “simulation” proofs

examining how each → is defined