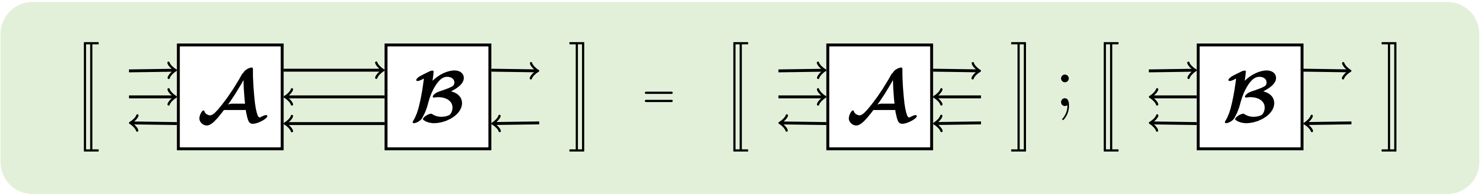
Compositional Probabilistic Model Checking with String Diagrams of MDPs

Kazuki Watanabe[1,2], Clovis Eberhart[1,3], Kazuyuki Asada[4], Ichiro Hasuo[1,2]

1: National Institute of Informatics, Tokyo, Japan
2: SOKENDAI (The Graduate University for Advanced Studies), Japan
3: Japanese-French Laboratory for Informatics (IRL 3527), Tokyo, Japan
4: Research Institute of Electrical Communication, Tohoku University, Sendai, Japan

Supported by ERATO HASUO Metamathematics for Systems Design Project No. JPMJER1603 and the JST Grant No. JPMJFS2136.

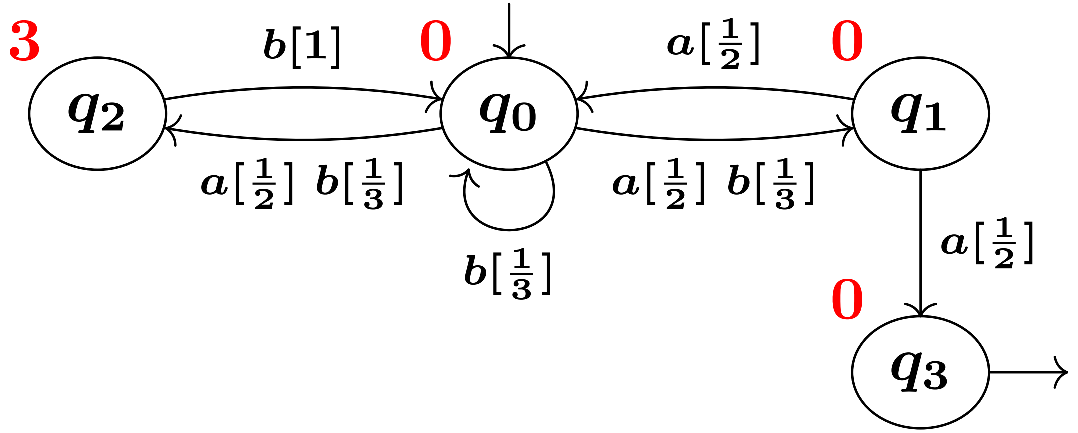
Outline



- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

Optimal Expected Reward of MDPs: Scheduler Synthesis + Its Performance Guarantee

Markov Decision Process (MDP)



- State-based model with **actions** (a, b, …) and **probabilistic uncertainties**
- Basic framework in many research areas (e.g. reinforcement learning)
- General modeling formalism for decision making in an uncertain environment

Goal: Compute Optimal Expected Reward

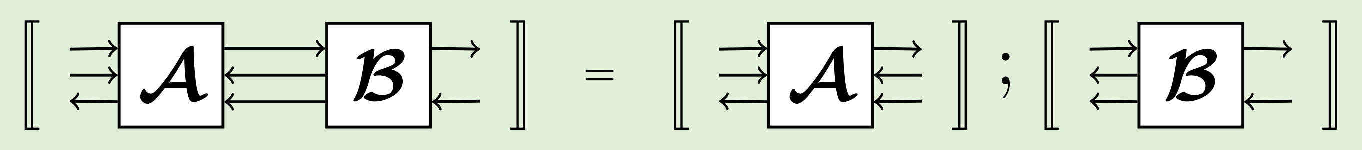
Problem:

- Given an MDP,
- Compute the optimal *scheduler* (~ controller, strategy; it chooses actions) and its expected cumulative reward

Applications:

- **Scheduler synthesis**
  “what is the best strategy?”
- **Formal verificaiton**
  “How much cumulative reward can I expect?”
  “Is the expectation correct?”

Outline



- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

A Paradigm with Conceptual Value, Performance Advantage, and Mathematical Blessing

$$\mathcal{S}(\mathcal{A} \star \mathcal{B}) = \mathcal{S}(\mathcal{A}) \star \mathcal{S}(\mathcal{B})$$

Composition of ***systems*** (seqComp, parComp, sum, ...)

Composition of ***semantics***

Conceptual Value

- "**Divide-and-Conquer**": simplifies a problem into smaller subproblems
- $\mathcal{S}(\mathcal{A})$, $\mathcal{S}(\mathcal{B})$ are **summaries** of components $\mathcal{A}$, $\mathcal{B}$. Unnecessary details get abstracted away

Performance Advantage

- Clear adv. when there are **duplicates** (reuse $\mathcal{S}(\mathcal{A})$ **!**)

$$\begin{aligned}&\mathcal{S}(\mathcal{A} \star \cdots \star \mathcal{A})\\ &\quad = \mathcal{S}(\mathcal{A}) \star \cdots \star \mathcal{S}(\mathcal{A})\end{aligned}$$

- (In some cases you don't need duplicates, e.g. mergesort)

Mathematical Blessing

- Compositionality means that the solution

$$\mathcal{S} : \mathbb{M} \longrightarrow \mathbb{S}$$

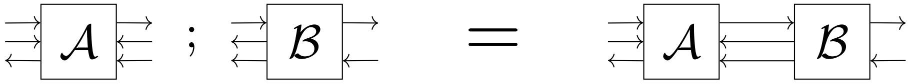
is a **homomorphism**, preserving the operation ★

Watanabe (NII, Tokyo)

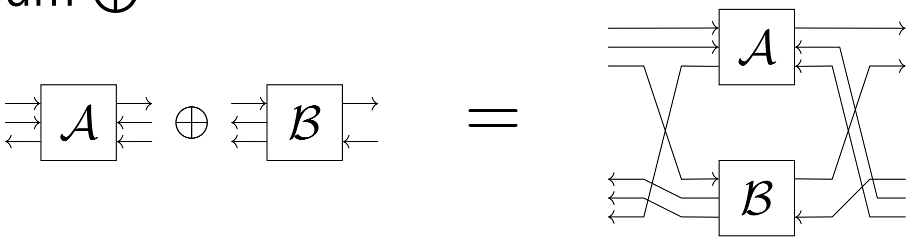
String Diagrams of MDPs: Planar Composition with SeqComp ; and Sum $\oplus$

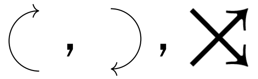
String Diagram of MDPs

- Sequential composition ;



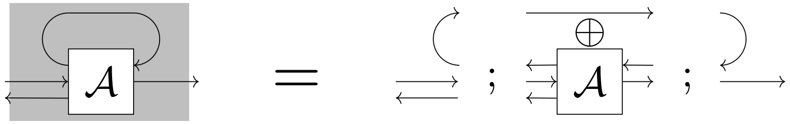
- Sum $\oplus$



- and some "constants"  , ...

→ **planar composition** of MDPs (mostly *sequential* composition; not parallel)

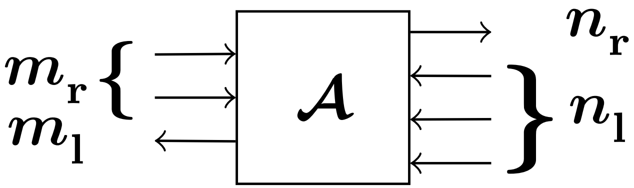
- Loop is a derived operation:



Background: Monoidal Categories

- Well-established topic of category theory (Mac Lane, Kelly, Joyal, Street, ...)
- Used for many applications:
  quantum field theory (Khavanov, ...),
  quantum computation (Abramsky, Coecke, Vicary, Heunen, ...),
  linguistics (Sadrzadeh, Coecke, ...),
  signal flow diagrams (Bonchi, Sobocinski, Zanasi, ...)
- String diagrams as a *graphical syntax* for monoidal categories [Joyal & Street, Adv. Math. 1991]
  - nicely expressive (planar composition, see left)
  - comes with a rich metatheory (see later)

Composition Formalism: String Diagrams of MDPs



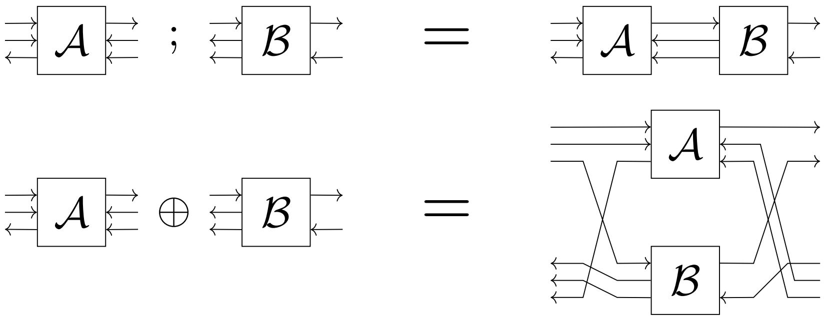
- ***Open MDPs*** extend MDPs with ***open ends***:
  (left, right) × (entrance, exit)
- An open MDP thus comes with an ***arity***.
  E.g. $\mathcal{A}\colon (2,1) \longrightarrow (1,3)$
- Open MDPs are combined with **algebraic operations ; (seqComp) and ⊕ (sum)**

$$\frac{\mathcal{A} : (m_{\mathrm{r}}, m_{\mathrm{l}}) \longrightarrow (n_{\mathrm{r}}, n_{\mathrm{l}}) \qquad \mathcal{B} : (n_{\mathrm{r}}, n_{\mathrm{l}}) \longrightarrow (k_{\mathrm{r}}, k_{\mathrm{l}})}{\mathcal{A} ; \mathcal{B} \quad : (m_{\mathrm{r}}, m_{\mathrm{l}}) \longrightarrow (k_{\mathrm{r}}, k_{\mathrm{l}})}$$

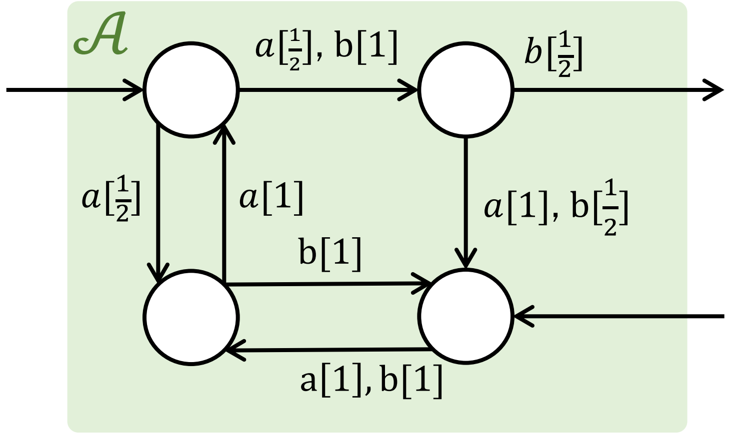
$$\frac{\mathcal{A} : (m_{\mathrm{r}}, m_{\mathrm{l}}) \longrightarrow (n_{\mathrm{r}}, n_{\mathrm{l}}) \qquad \mathcal{B} : (k_{\mathrm{r}}, k_{\mathrm{l}}) \longrightarrow (l_{\mathrm{r}}, l_{\mathrm{l}})}{\mathcal{A} \oplus \mathcal{B} \quad : (m_{\mathrm{r}} + k_{\mathrm{r}}, m_{\mathrm{l}} + k_{\mathrm{l}}) \longrightarrow (n_{\mathrm{r}} + l_{\mathrm{r}}, n_{\mathrm{l}} + l_{\mathrm{l}})}$$

Def. (open MDP, oMDP) Let A be a non-empty finite set, whose elements are called *actions*. An *open MDP* $\mathcal{A}$ (*over* the action set A) is the tuple $(\overline{m}, \overline{n}, Q, A, E, P, R)$ of the following data. We say that it is *from* $\overline{m}$ *to* $\overline{n}$.

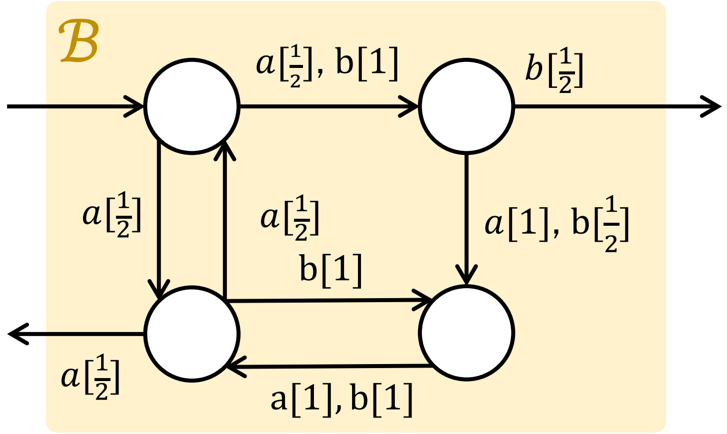
1. $\overline{m} = (m_{\mathrm{r}}, m_{\mathrm{l}})$ and $\overline{n} = (n_{\mathrm{r}}, n_{\mathrm{l}})$ are pairs of natural numbers; they are called the *left-arity* and the *right-arity*, respectively. Moreover, elements of $[m_{\mathrm{r}} + n_{\mathrm{l}}]$ are called *entrances*, and those of $[n_{\mathrm{r}} + m_{\mathrm{l}}]$ are called *exits*.
2. Q is a finite set of *positions*.
3. $E : [m_{\mathrm{r}} + n_{\mathrm{l}}] \to Q + [n_{\mathrm{r}} + m_{\mathrm{l}}]$ is an *entry function*, which maps each entrance to either a position (in Q) or an exit (in $[n_{\mathrm{r}} + m_{\mathrm{l}}]$).
4. $P : Q \times A \times (Q + [n_{\mathrm{r}} + m_{\mathrm{l}}]) \to \mathbb{R}_{\geq 0}$ determines *transition probabilities*, where we require $\sum_{s' \in Q + [n_{\mathrm{r}} + m_{\mathrm{l}}]} P(s, a, s') \in \{0, 1\}$ for each $s \in Q$ and $a \in A$.
5. R is a *reward function* $R : Q \to \mathbb{R}_{\geq 0}$.
6. We impose the following "unique access to each exit" condition. Let $\mathbf{exits} : ([m_{\mathrm{r}} + n_{\mathrm{l}}] + Q) \to \mathcal{P}([n_{\mathrm{r}} + m_{\mathrm{l}}])$ be the *exit function* that collects all immediately reachable exits, that is, 1) for each $s \in Q$, $\mathbf{exits}(s) = \{t \in [n_{\mathrm{r}} + m_{\mathrm{l}}] \mid \exists a \in A. P(s, a, t) > 0\}$, and 2) for each entrance $s \in [m_{\mathrm{r}} + n_{\mathrm{l}}]$, $\mathbf{exits}(s) = \{E(s)\}$ if $E(s)$ is an exit and $\mathbf{exits}(s) = \emptyset$ otherwise.
   - For all $s, s' \in [m_{\mathrm{r}} + n_{\mathrm{l}}] + Q$, if $\mathbf{exits}(s) \cap \mathbf{exits}(s') \neq \emptyset$, then $s = s'$.
   - We further require that each exit is reached from an identical position by at most one action. That is, for each exit $t \in [n_{\mathrm{r}} + m_{\mathrm{l}}]$, $s \in Q$, and $a, b \in A$, if both $P(s, a, t) > 0$ and $P(s, b, t) > 0$, then $a = b$.



; (seqComp) of String Diagrams of MDPs

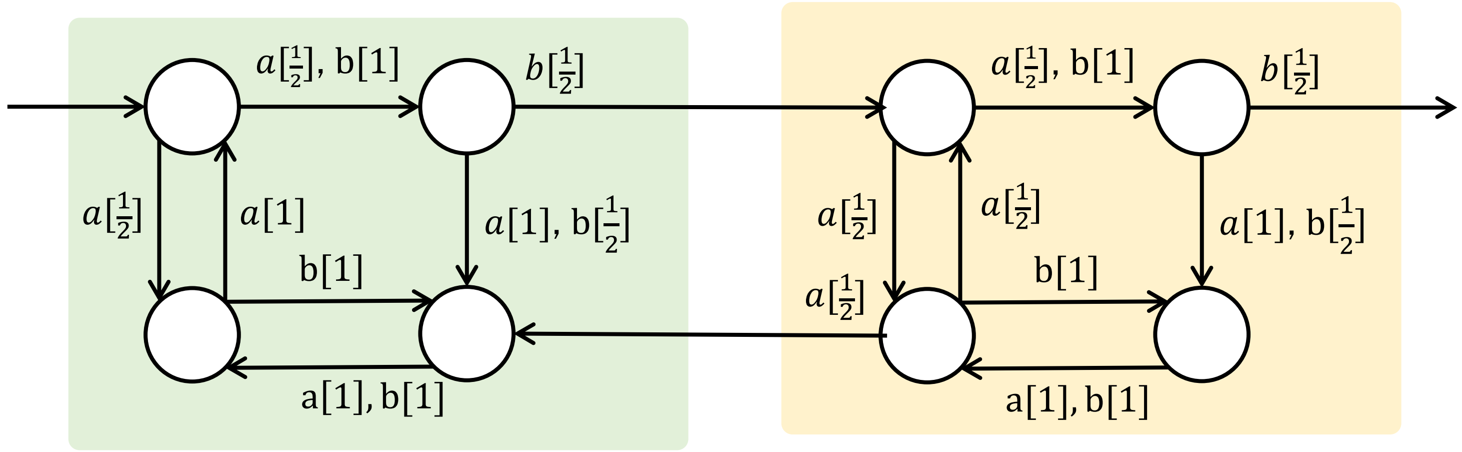


$: (1,0) \to (1,1)$



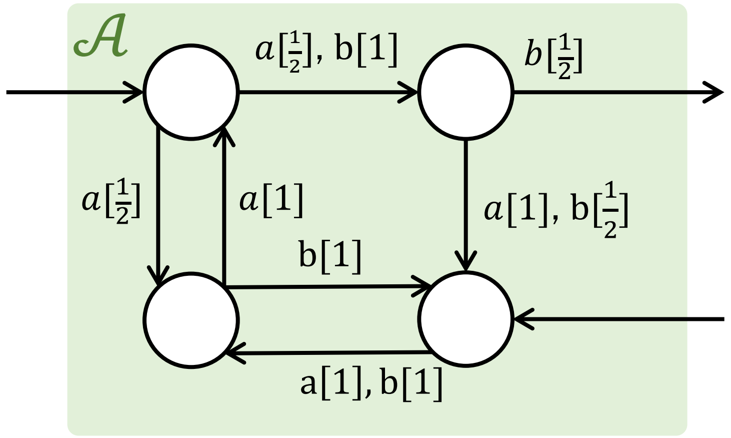
$: (1,1) \to (1,0)$

$\mathcal{A}\,;\,\mathcal{B} =$

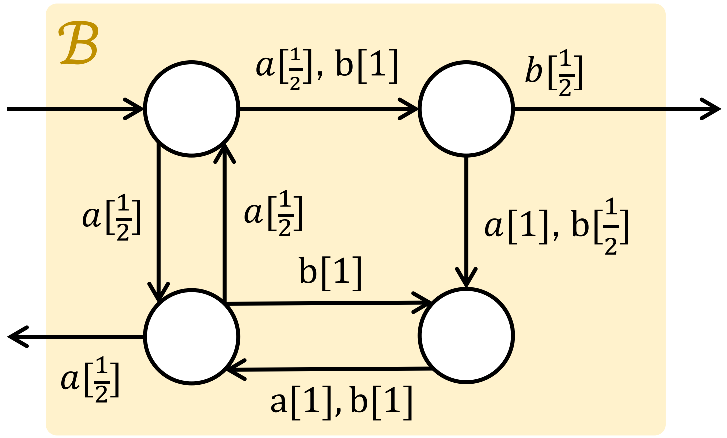


$: (1,0) \to (1,0)$

$\oplus$ (sum) of String Diagrams of MDPs

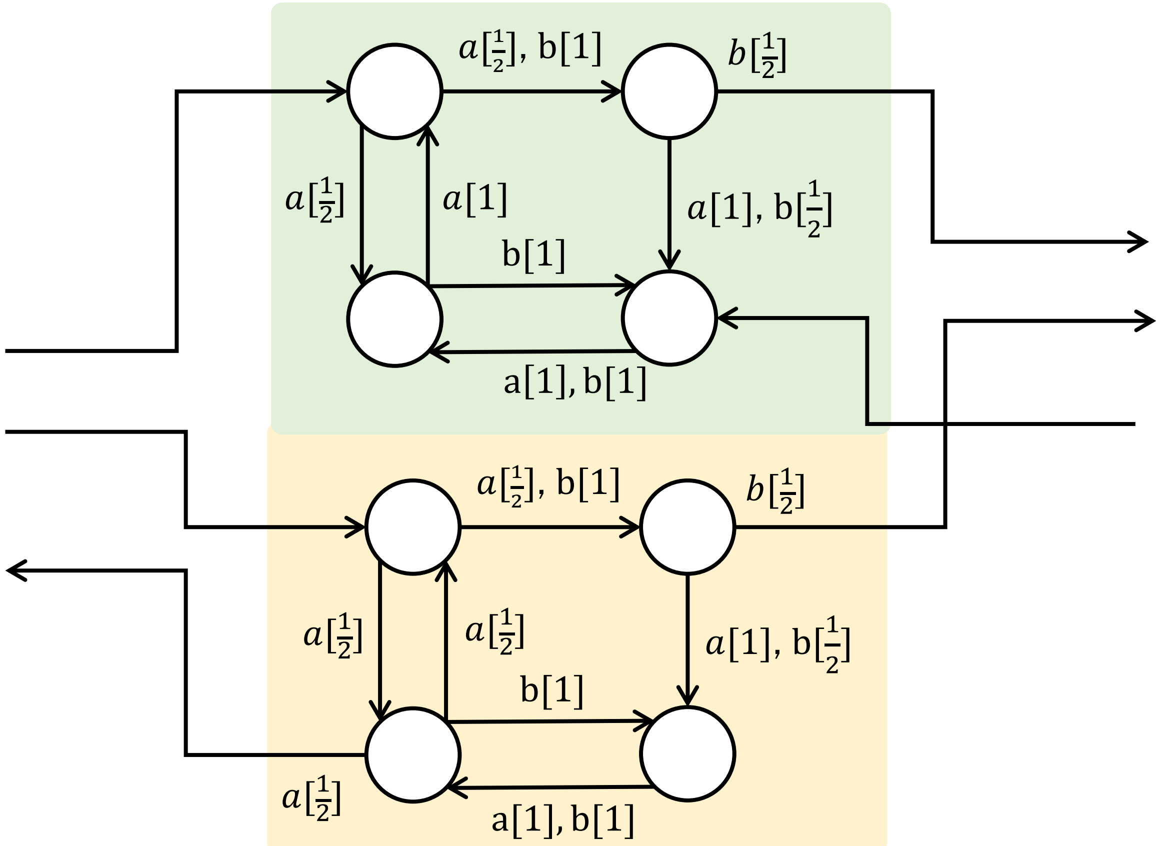


$: (1,0) \to (1,1)$



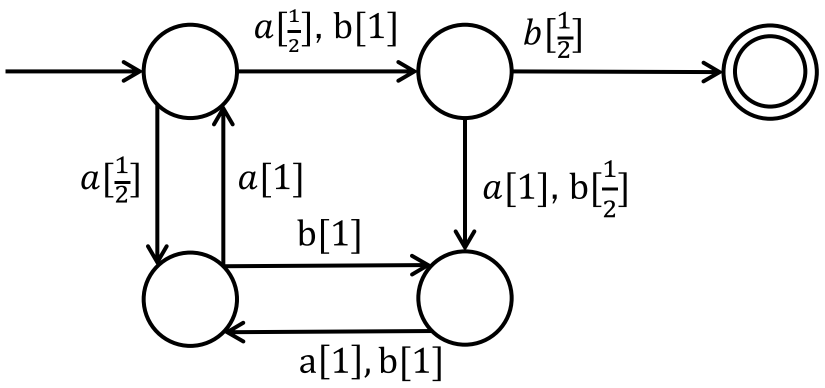
$: (1,1) \to (1,0)$

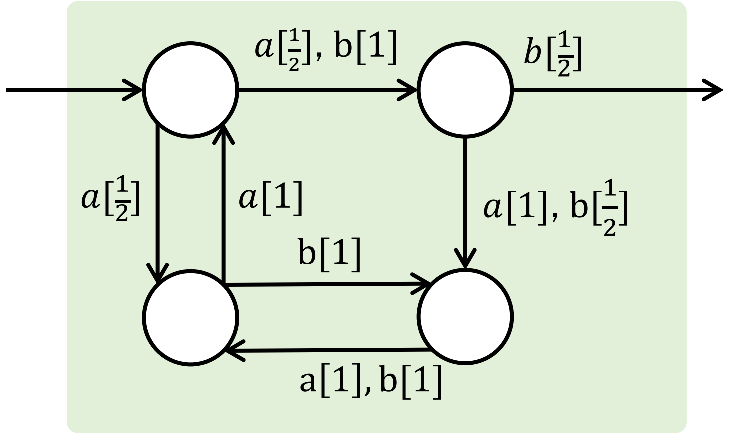
$\mathcal{A} \oplus \mathcal{B} =$



$: (2,1) \to (2,1)$

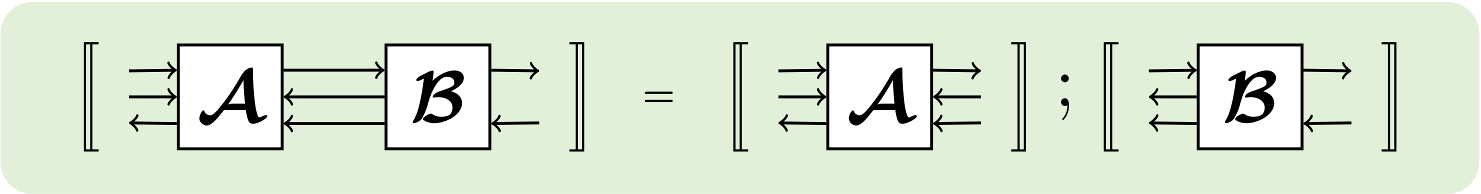
String Diagrams of MDPs: (Usual) MDPs as Open MDPs





$: (1,0) \to (1,0)$

Outline



- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

CompMDP: a Compositional MDP Model Checking Algorithm

function CompMDP($\mathcal{A}$)

Input:

a "string diagram" $\mathcal{A}$: $(m_{\mathrm{r}}, m_{\mathrm{l}}) \to (n_{\mathrm{r}}, n_{\mathrm{l}})$ of open MDPs, composed with ; (seqComp) and $\oplus$ (sum)

Output:

a set $\left\{ \left(p^{\tau}_{i,j}, r^{\tau}_{i,j}\right)_{i\in[m_{\mathrm{r}}+n_{\mathrm{l}}],\ j\in[n_{\mathrm{r}}+m_{\mathrm{l}}]} \right\}_{\tau}$

if $\mathcal{A}$ is atomic **then**

return $\left\{ \Big(\mathbf{RPr}(\mathcal{A}^{\tau})(i,j),\ \mathbf{ERw}(\mathcal{A}^{\tau})(i,j)\Big)_{i\in[m_{\mathrm{r}}+n_{\mathrm{l}}],\, j\in[n_{\mathrm{r}}+m_{\mathrm{l}}]} \;\middle|\; \begin{array}{l} \tau \text{ is a memoryless} \\ \text{scheduler of } \mathcal{A} \end{array} \right\}_{\tau}$

elsif $\mathcal{A} = \mathcal{B} \,;\, \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) ; CompMDP($\mathcal{C}$)

elsif $\mathcal{A} = \mathcal{B} \oplus \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) $\oplus$ CompMDP($\mathcal{C}$)

CompMDP: a Compositional MDP Model Checking Algorithm

function CompMDP($\mathcal{A}$)

Input:

a “string diagram” $\mathcal{A}$: $(m_{\mathrm{r}}, m_{\mathrm{l}}) \to$??? MDPs,

composed with ; (seqComp) and $\oplus$ ( )

Output:

a set $\boxed{\left\{ \left(p^{\tau}_{i,j}, r^{\tau}_{i,j}\right)_{i\in[m_{\mathrm{r}}+n_{\mathrm{l}}],\ j\in[n_{\mathrm{r}}+m_{\mathrm{l}}]} \right\}_{\tau}}$

if $\mathcal{A}$ is atomic **then**

return $\left\{ \left(\mathbf{RPr}(\mathcal{A}^{\tau})(i,j),\ \mathbf{ERw}(\mathcal{A}^{\tau})(i,j)\right)_{i\in[m_{\mathrm{r}}+n_{\mathrm{l}}],\, j\in[n_{\mathrm{r}}+m_{\mathrm{l}}]} \;\middle|\; \begin{array}{l}\tau \text{ is a memoryless} \\ \text{scheduler of } \mathcal{A}\end{array} \right\}_{\tau}$

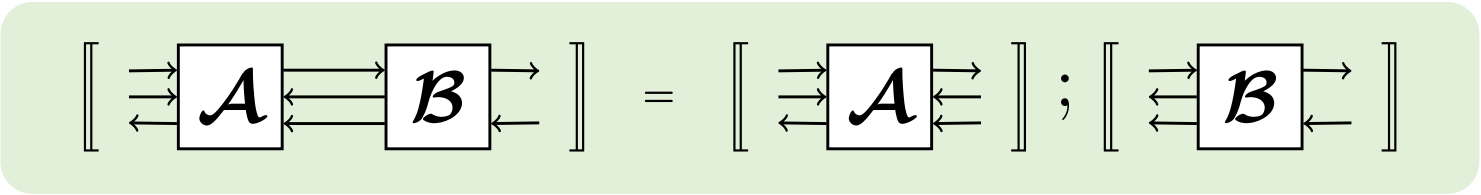
elsif $\mathcal{A} = \mathcal{B}\, ;\, \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) ; CompMDP($\mathcal{C}$)

elsif $\mathcal{A} = \mathcal{B} \oplus \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) $\oplus$ CompMDP($\mathcal{C}$)

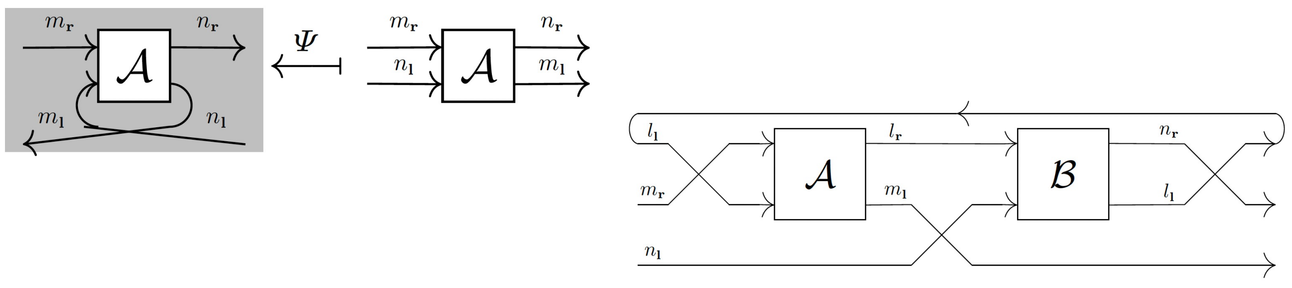
Outline



- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

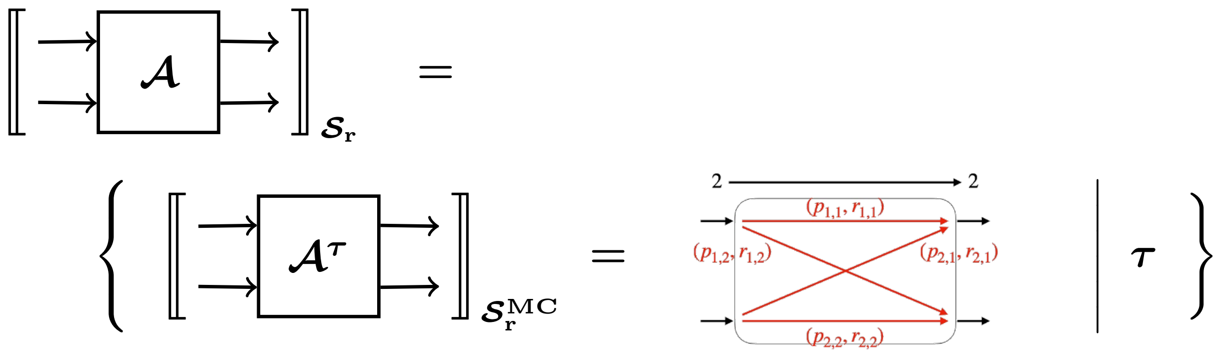
Upgrading Frameworks for Free: Compositional Solutions for MC, MDP, and bi-dir. MDP

$$\mathbf{oMDP} \xrightarrow{\mathcal{S}} \mathbb{S}$$



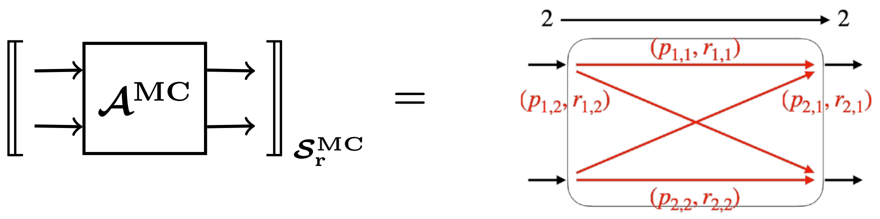
the Int construction ▲ bidirectional

$$\mathbf{roMDP} \xrightarrow[\mathcal{S}_{\mathbf{r}}]{} \mathbb{S}_{\mathbf{r}}$$



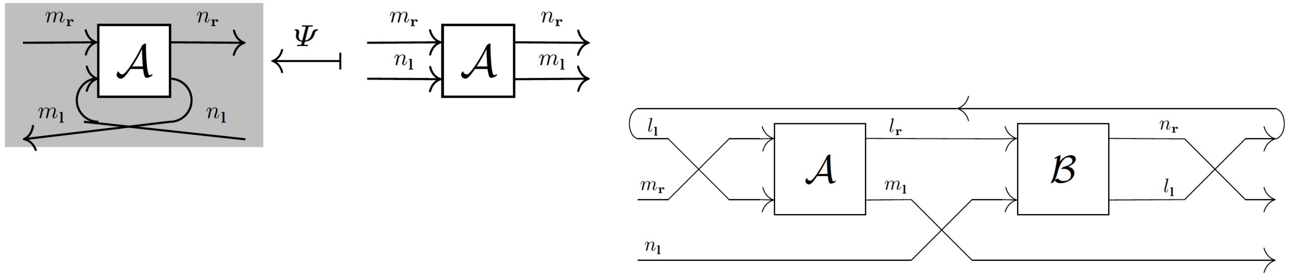
change of base w/ $\mathcal{P}$ ▲ non-determinism

$$\mathbf{roMC} \xrightarrow[\mathcal{S}^{\mathrm{MC}}_{\mathbf{r}}]{} \mathbb{S}^{\mathrm{MC}}_{\mathbf{r}}$$



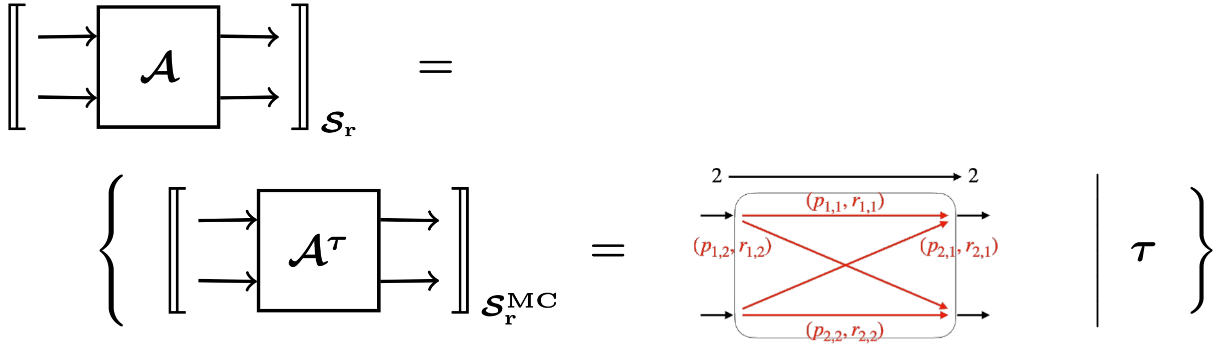
Upgrading Frameworks for Free: Compositional Solutions for MC, MDP, and bi-dir. MDP

$$\mathbf{oMDP} \xrightarrow{\mathcal{S}} \mathbb{S}$$



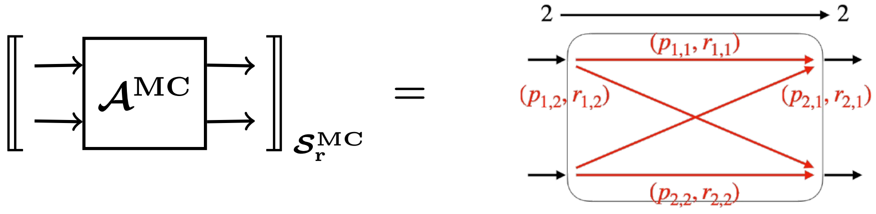
the Int construction — bidirectional

$$\mathbf{roMDP} \xrightarrow[\mathcal{S}_{\mathbf{r}}]{} \mathbb{S}_{\mathbf{r}}$$

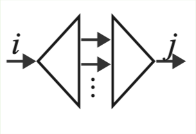
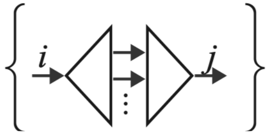
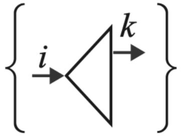
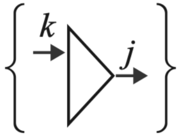
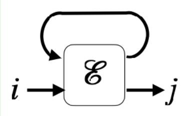


change of base w/ $\mathcal{P}$ — non-determinism

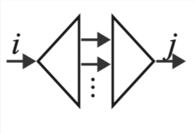
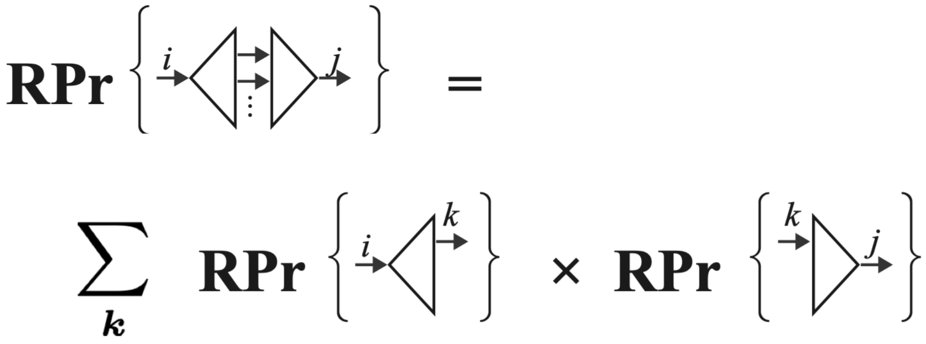
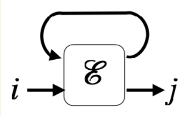
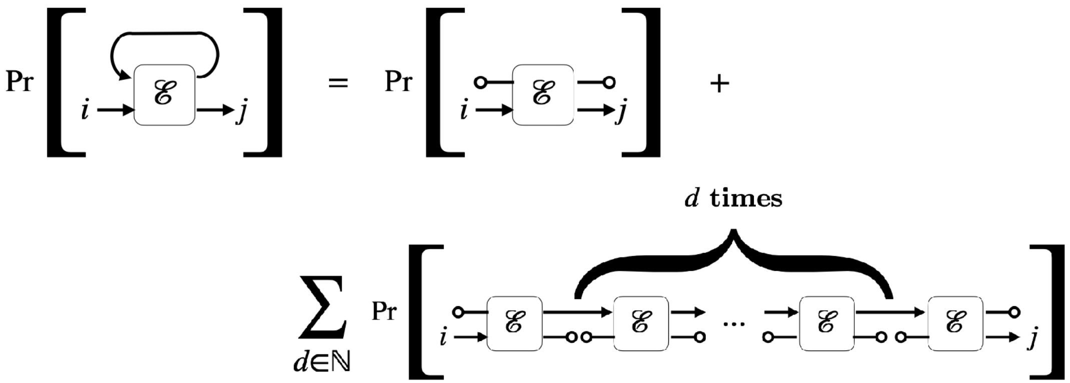
$$\mathbf{roMC} \xrightarrow[\mathcal{S}_{\mathbf{r}}^{\mathrm{MC}}]{} \mathbb{S}_{\mathbf{r}}^{\mathrm{MC}}$$



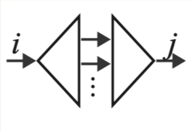
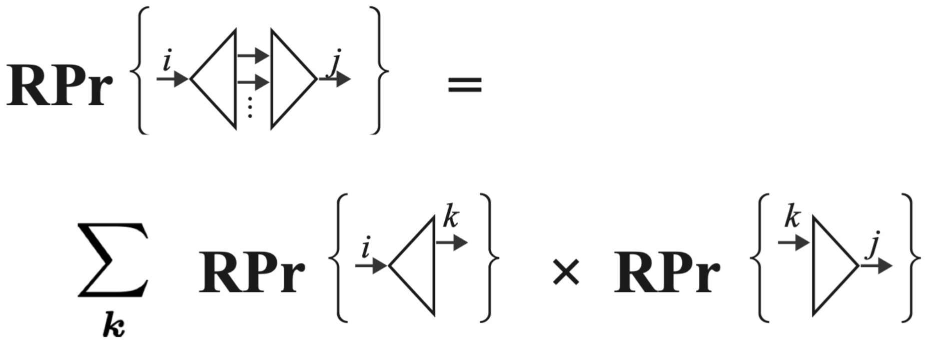
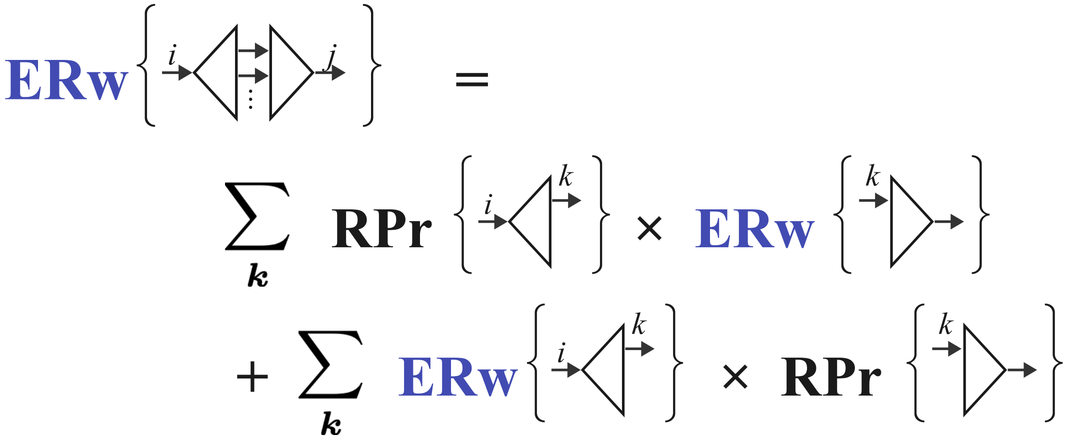
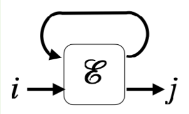
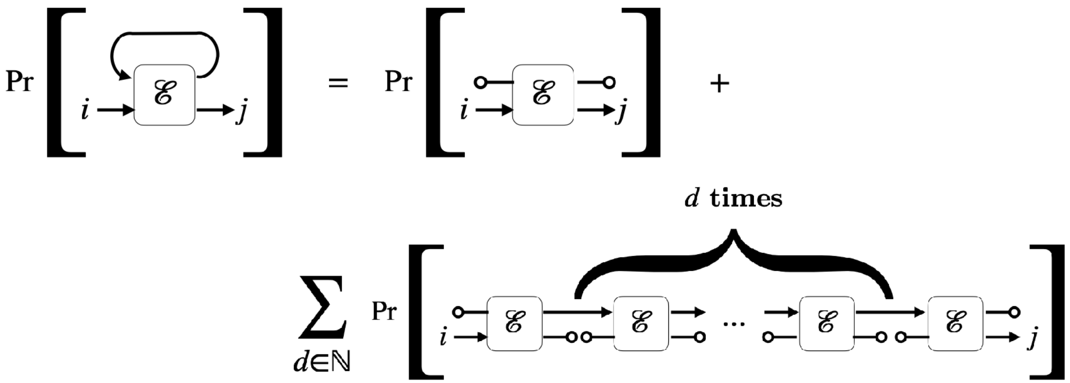
Decomposition Equalities for (rightward open) Markov Chains

	reachability probability	expected reward
seq. comp. 	$\mathbf{RPr}$  $=$ $\sum_{\boldsymbol{k}}$ $\mathbf{RPr}$  $\times$ $\mathbf{RPr}$  (Folklore)	
trace  (trace is primitive in a uni-dir. setting)		

Decomposition Equalities for (rightward open) Markov Chains

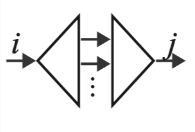
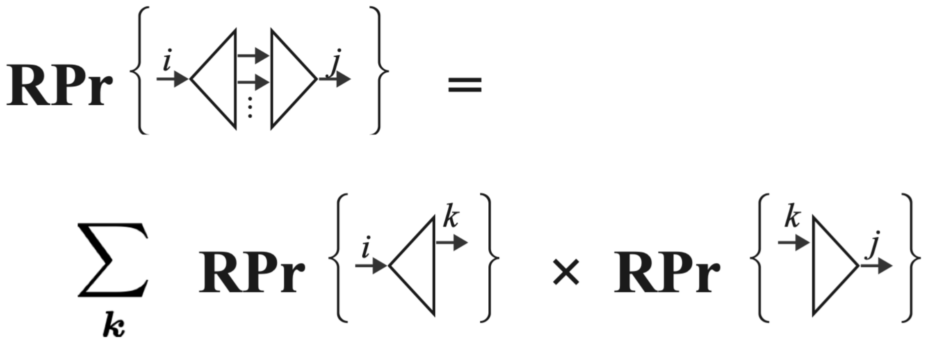
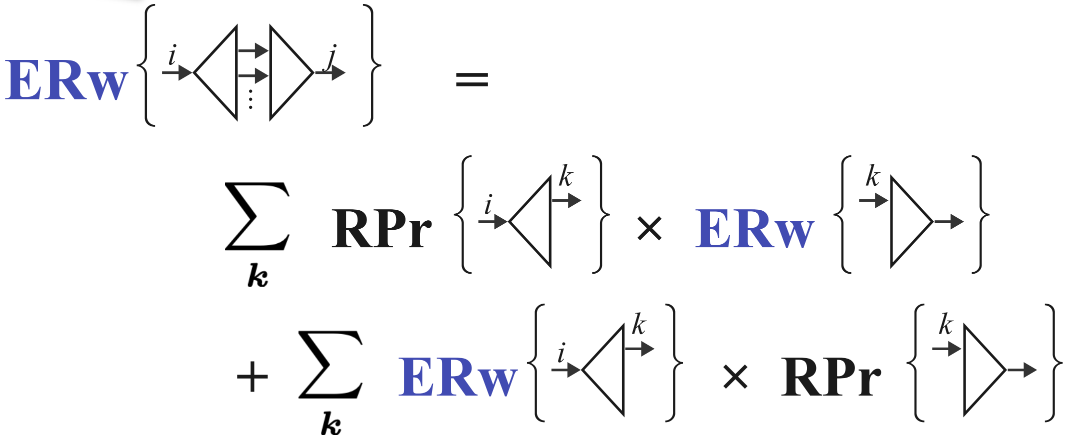
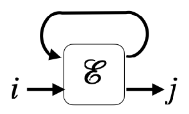
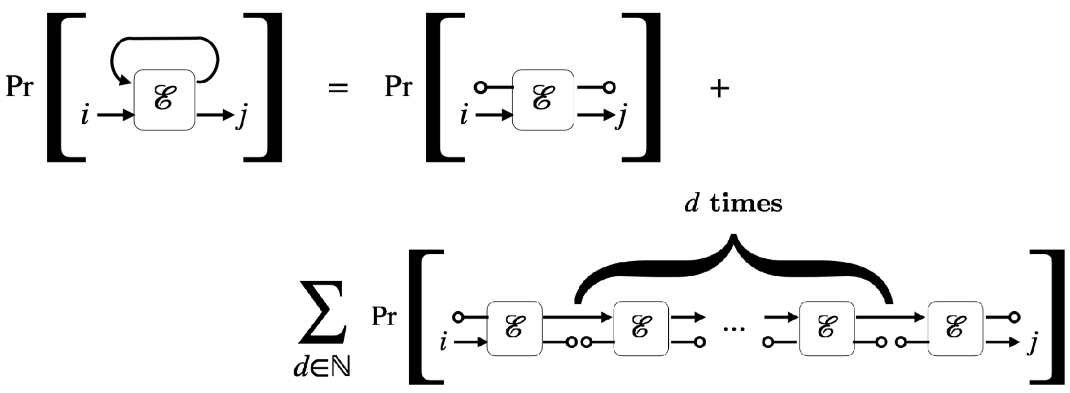
	reachability probability	expected reward
seq. comp. 	 (Folklore)	
trace  (trace is primitive in a uni-dir. setting)	 (Girard's *execution formula*)	

Decomposition Equalities for (rightward open) Markov Chains

	reachability probability	expected reward
seq. comp. 	 (Folklore)	 (Prop. 3.2)
trace  (trace is primitive in a uni-dir. setting)	 (Girard's *execution formula*)	$\left[\mathbf{ERw}^{\mathrm{tr}_{l;m,n}(\mathcal{E})}(i,j)\right]_{i,j} = \left[\mathbf{ERw}^{\mathcal{E}}(l+i,l+j)\right]_{i,j} + \sum_{d\in\mathbb{N}} \left[\begin{pmatrix} \left[\mathbf{RPr}^{\mathcal{E}}(l+i,k)\right]_{i,k} & \left[\mathbf{ERw}^{\mathcal{E}}(l+i,k)\right]_{i,k} \end{pmatrix} \cdot \begin{pmatrix} \left[\mathbf{RPr}^{\mathcal{E}}(k,k')\right]_{k,k'} & \left[\mathbf{ERw}^{\mathcal{E}}(k,k')\right]_{k,k'} \\ \left[0\right]_{k,k'} & \left[\mathbf{RPr}^{\mathcal{E}}(k,k')\right]_{k,k'} \end{pmatrix}^{d} \cdot \begin{pmatrix} \left[\mathbf{ERw}^{\mathcal{E}}(k',l+j)\right]_{k',j} \\ \left[\mathbf{RPr}^{\mathcal{E}}(k',l+j)\right]_{k',j} \end{pmatrix} \right]$ (Prop. 3.2)

Decomposition Equalitie en) Markov Chains

For ERw,
record RPr as well!

	reachability pro	expected reward
seq. comp. 	 (Folklore)	 (Prop. 3.2)
trace  (trace is primitive in a uni-dir. setting)	 (Girard's *execution formula*)	$$\left[\mathrm{ERw}^{\mathrm{tr}_{l;m,n}(\mathcal{E})}(i,j)\right]_{i,j} = \left[\mathrm{ERw}^{\mathcal{E}}(l+i,l+j)\right]_{i,j} + \sum_{d\in\mathbb{N}} \left[\left(\left[\mathrm{RPr}^{\mathcal{E}}(l+i,k)\right]_{i,k} \;\; \left[\mathrm{ERw}^{\mathcal{E}}(l+i,k)\right]_{i,k}\right) \cdot \begin{pmatrix} \left[\mathrm{RPr}^{\mathcal{E}}(k,k')\right]_{k,k'} & \left[\mathrm{ERw}^{\mathcal{E}}(k,k')\right]_{k,k'} \\ \left[0\right]_{k,k'} & \left[\mathrm{RPr}^{\mathcal{E}}(k,k')\right]_{k,k'} \end{pmatrix}^{d} \cdot \begin{pmatrix} \left[\mathrm{ERw}^{\mathcal{E}}(k',l+j)\right]_{k',j} \\ \left[\mathrm{RPr}^{\mathcal{E}}(k',l+j)\right]_{k',j} \end{pmatrix} \right]$$ (Prop. 3.2)

CompMDP: a Compositional MDP Model Checking Algorithm

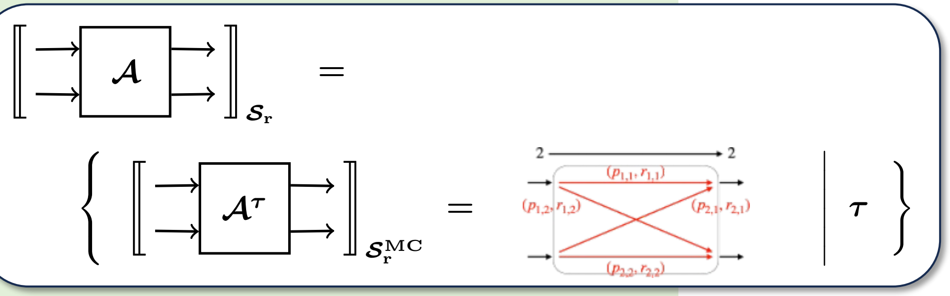
function CompMDP($\mathcal{A}$)

Input:

a "string diagram" $\mathcal{A}$: $(m_{\mathrm{r}}, m_{\mathrm{l}}) \to (n_{\mathrm{r}}, n_{\mathrm{l}})$ of open MDPs, composed with ; (seqComp) and $\oplus$ (sum)

Output:

a set $\boxed{\left\{ \left(p^{\tau}_{i,j}, r^{\tau}_{i,j}\right)_{i \in [m_{\mathrm{r}}+n_{\mathrm{l}}],\ j \in [n_{\mathrm{r}}+m_{\mathrm{l}}]} \right\}_{\tau}}$



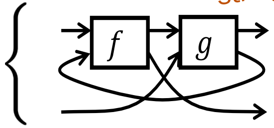
if $\mathcal{A}$ is atomic **then**

return $\left\{ \left(\mathbf{RPr}(\mathcal{A}^{\tau})(i,j),\ \mathbf{ERw}(\mathcal{A}^{\tau})(i,j) \right)_{i \in [m_{\mathrm{r}}+n_{\mathrm{l}}],\, j \in [n_{\mathrm{r}}+m_{\mathrm{l}}]} \;\middle|\; \begin{array}{l} \tau \text{ is a memoryless} \\ \text{scheduler of } \mathcal{A} \end{array} \right\}_{\tau}$

elsif $\mathcal{A} = \mathcal{B} \,;\, \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) ; CompMDP($\mathcal{C}$)

concretely, string diagram in $\mathbb{S}^{\mathrm{MC}}_{\mathbf{r}}$

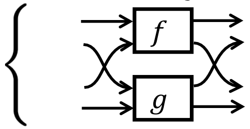


$\left| \begin{array}{l} f \in \mathrm{CompMDP}(\mathcal{B}) \\ g \in \mathrm{CompMDP}(\mathcal{C}) \end{array} \right\}$

elsif $\mathcal{A} = \mathcal{B} \oplus \mathcal{C}$ **then**

return CompMDP($\mathcal{B}$) $\oplus$ CompMDP($\mathcal{C}$)

concretely,

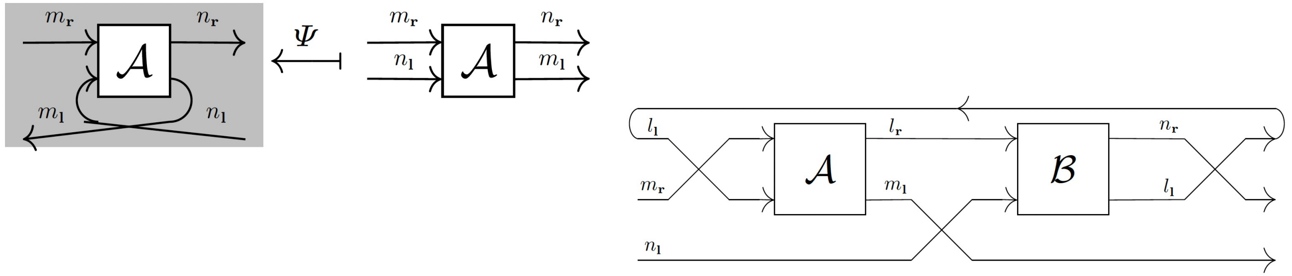


$\left| \begin{array}{l} f \in \mathrm{CompMDP}(\mathcal{B}) \\ g \in \mathrm{CompMDP}(\mathcal{C}) \end{array} \right\}$

Watanabe (NII, Tokyo)

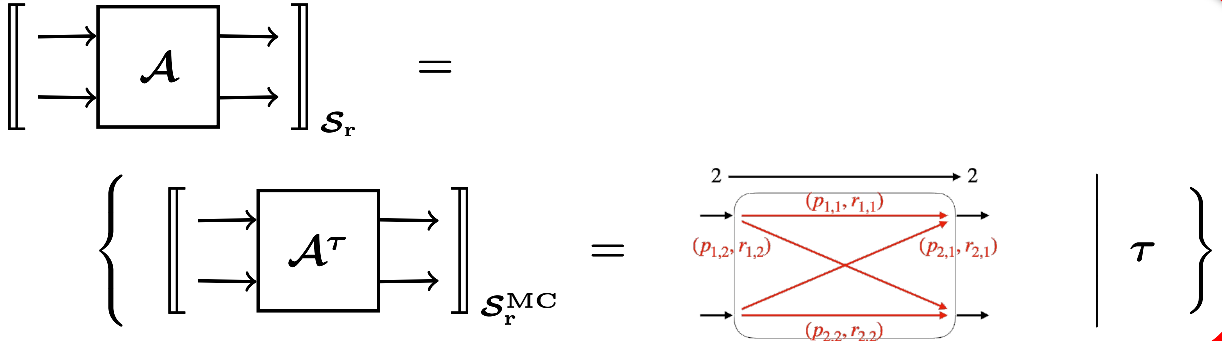
Upgrading Frameworks for Free: Compositional Solutions for MC, MDP, and bi-dir. MDP

$$\mathbf{oMDP} \xrightarrow{\ \mathcal{S}\ } \mathbb{S}$$



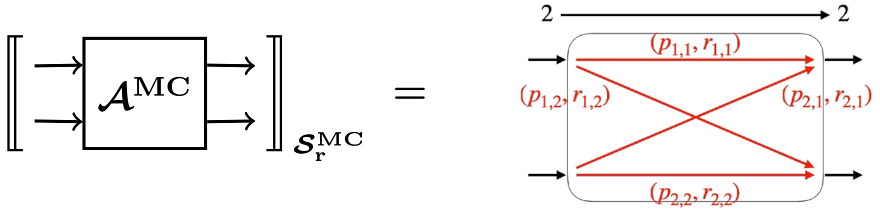
the Int construction — bidirectional

$$\mathbf{roMDP} \xrightarrow[\ \mathcal{S}_{\mathbf{r}}\ ]{} \mathbb{S}_{\mathbf{r}}$$



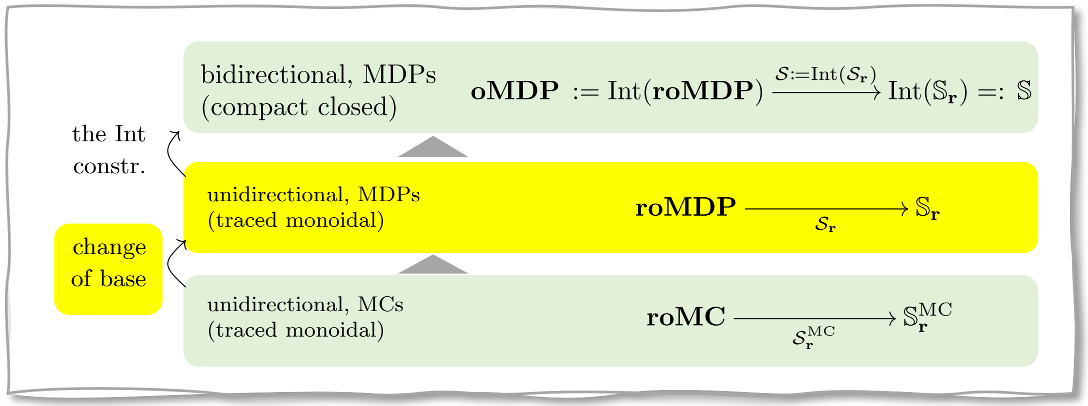
change of base w/ $\mathcal{P}$ — non-determinism

$$\mathbf{roMC} \xrightarrow[\ \mathcal{S}^{\mathrm{MC}}_{\mathbf{r}}\ ]{} \mathbb{S}^{\mathrm{MC}}_{\mathbf{r}}$$



Change of Base for Accommodating Actions, Schedulers, Optimality

[Eilenberg & Kelly '66] [Cruttwell, PhD thesis, '08] ...



We apply change of base,
wrt. the powerset functor $\mathcal{P} : \mathbf{Set} \longrightarrow \mathbf{Set}$,
to *upgrade* $\mathbb{S}^{\mathrm{MC}}_\mathbf{r}$ (for MCs) to $\mathbb{S}_\mathbf{r}$ (for MDPs).

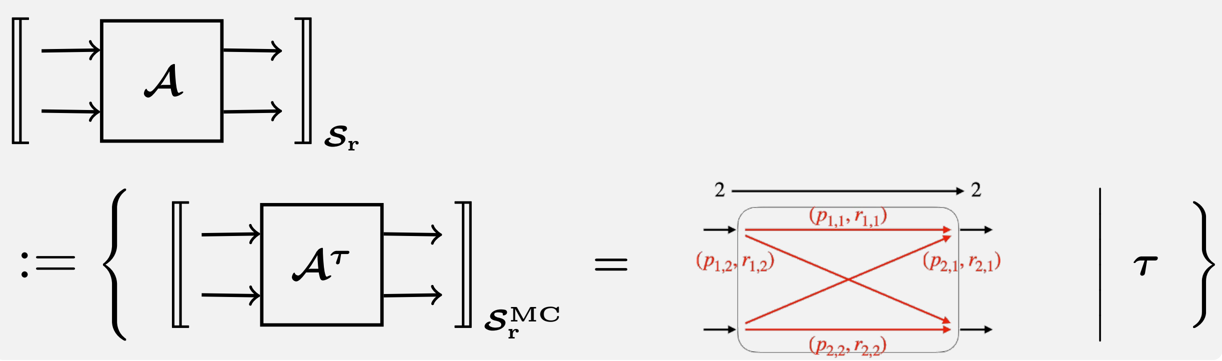
Concretely,

Def. ($\mathbb{S}_\mathbf{r}$)
Object: natural number m
Arrow:

$$\frac{F : m \longrightarrow n \quad \text{in } \mathbb{S}_\mathbf{r}}{F \text{ is a set } \{f_i \mid f_i : m \to n \text{ in } \mathbb{S}^{\mathrm{MC}}_\mathbf{r}\}}$$

$\mathbb{S}_\mathbf{r}$ is a TSMC with pointwise extension of opr. of $\mathbb{S}^{\mathrm{MC}}_\mathbf{r}$.
E.g. $F \circ G = \{f_i \circ g_j\}_{i,j}$
(Being a category: by general theory of change of base.
Being traced monoidal: not covered, but easy.)

The solution functor $\mathcal{S}_\mathbf{r} : \mathbf{roMDP} \longrightarrow \mathbb{S}_\mathbf{r}$
is defined by bunding up different schedulers' behaviors



RPr and ERw,
under a scheduler τ

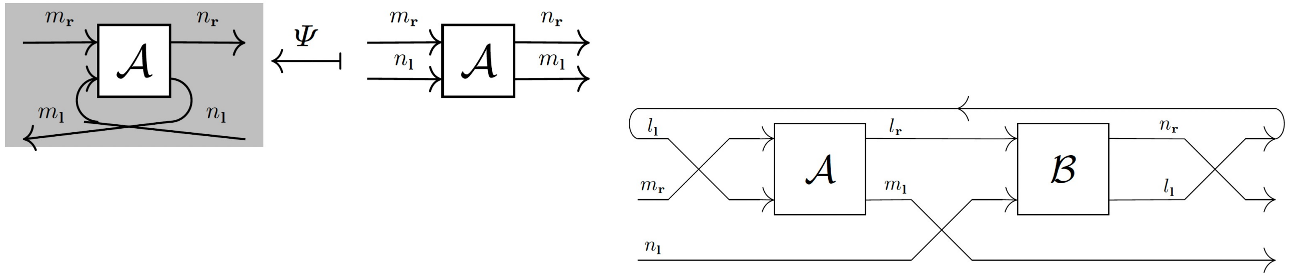
Thm.
$\mathcal{S}_\mathbf{r} : \mathbf{roMDP} \longrightarrow \mathbb{S}_\mathbf{r}$ is a traced symmetric monoidal functor, i.e. a homomorphism of TSMCs. □

➔ **compositional model checking** of MDPs!

Watanabe (NII, Tokyo)

Upgrading Frameworks for Free: Compositional Solutions for MC, MDP, and bi-dir. MDP

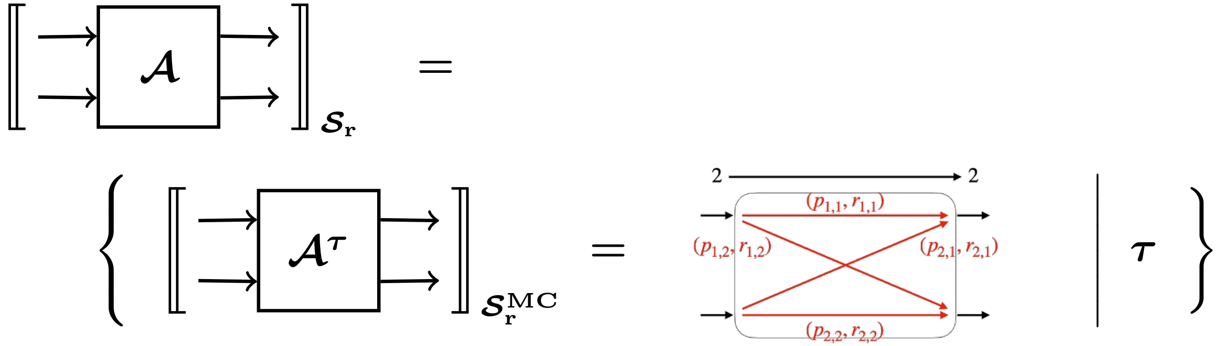
$$\mathbf{oMDP} \xrightarrow{\mathcal{S}} \mathbb{S}$$



the Int construction

bidirectional

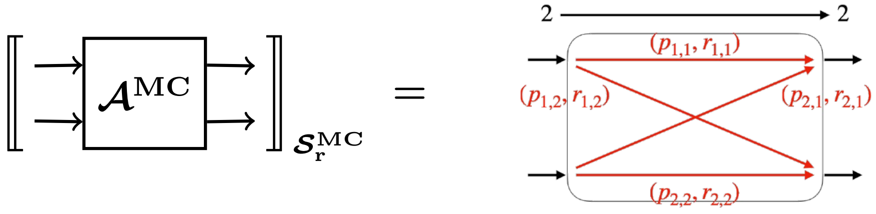
$$\mathbf{roMDP} \xrightarrow[\mathcal{S}_{\mathbf{r}}]{} \mathbb{S}_{\mathbf{r}}$$



change of base w/ $\mathcal{P}$

non-determinism

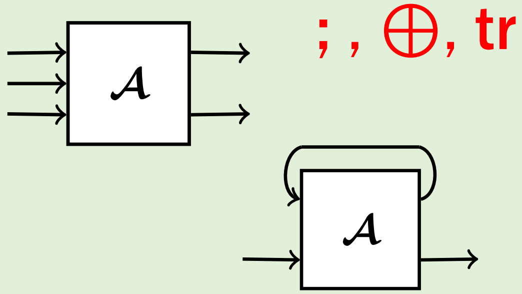
$$\mathbf{roMC} \xrightarrow[\mathcal{S}^{\mathrm{MC}}_{\mathbf{r}}]{} \mathbb{S}^{\mathrm{MC}}_{\mathbf{r}}$$



The Int Construction from Unidirectional to Bidirectional

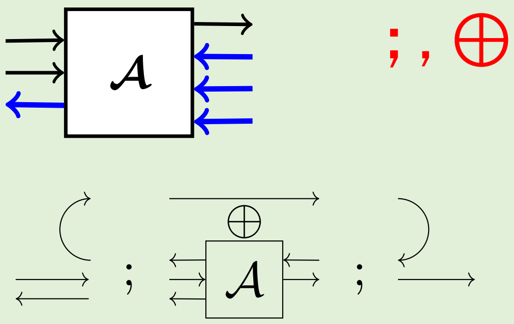
The *Int construction*: [Joyal, Street & Verity '96]
a general construction that turns

unidirectional string diagrams with loops



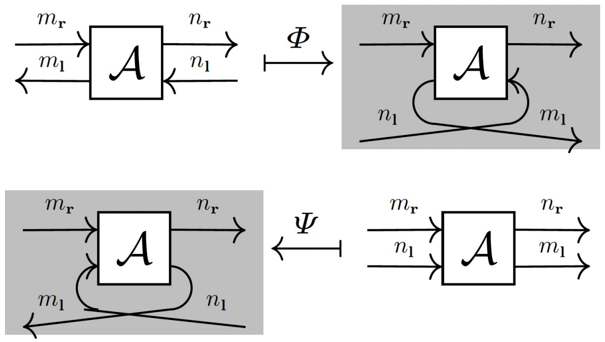
(traced sym. monoidal categories (TSMC))

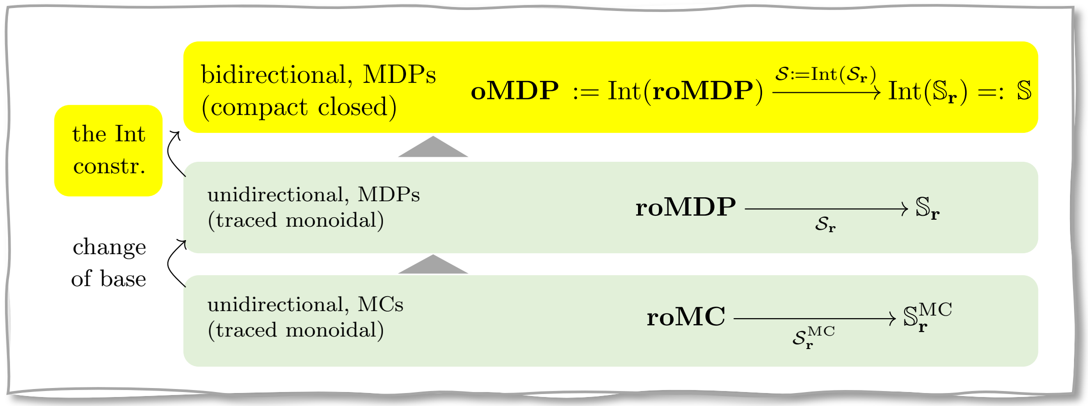
bidirectional string diagrams



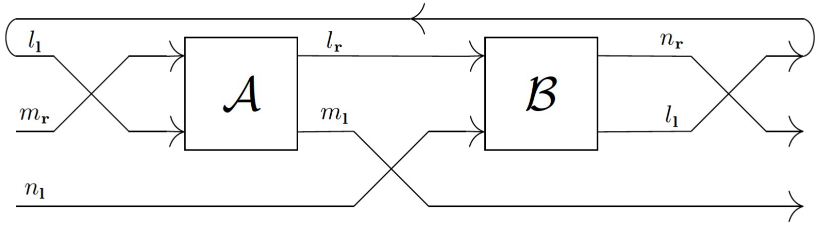
(compact closed categories (compCC))

by *twisting*:





In particular, the bidirectional seqComp $\mathcal{A}\,;\,\mathcal{B}$ is



$\mathbf{Int}$ extends to functors (and 2-cells):

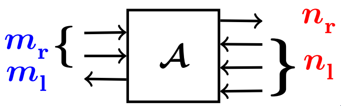
$$\mathbf{Int} \colon \mathbf{TSMC} \longrightarrow \mathbf{CompCC}$$

We thus apply $\mathbf{Int}$ to $\mathcal{S}_r \colon \mathbf{roMDP} \longrightarrow \mathbb{S}_r$ and get

$$\mathcal{S} \colon \mathbf{oMDP} \longrightarrow \mathbb{S}$$

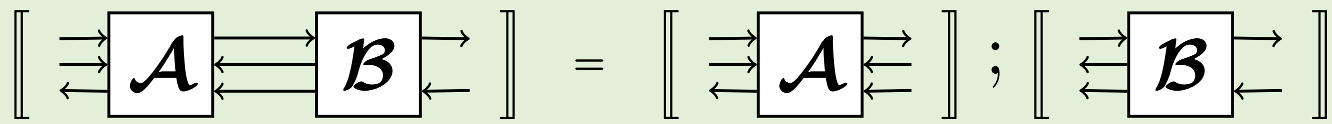
__Object__: (m_r, m_l)
__Arrow__: $\mathcal{A} : (m_r, m_l) \longrightarrow (n_r, n_l)$ in oMDP

an open MDP



This is a compact closed functor.

Outline



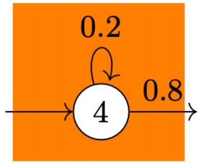
- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

Experiment Results

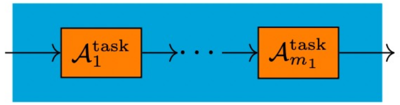
- Compositional algorithm can be **arbitrary faster** (reuse $\mathcal{S}(\mathcal{A})$!)

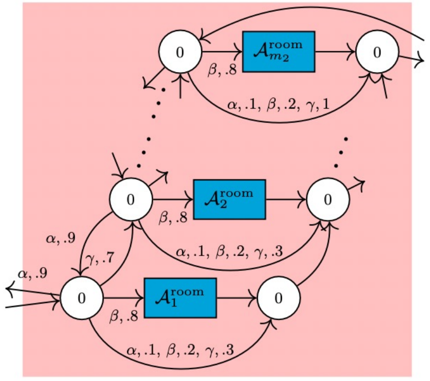
$$\mathcal{S}(\mathcal{A} \star \cdots \star \mathcal{A}) = \mathcal{S}(\mathcal{A}) \star \cdots \star \mathcal{S}(\mathcal{A})$$

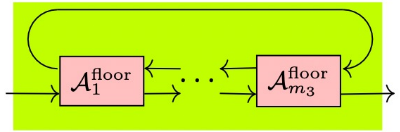
- Overall, we do indeed witness the performance advantage of compositionality
- We need MDPs given in a compositional formalism. This is realistic. Our *Patrol* benchmark:

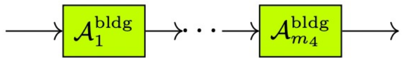


(a) A *task* $\mathcal{A}_i^{\text{task}}$.


(b) A *room* $\mathcal{A}_i^{\text{room}}$ combines tasks.


(c) A *floor* $\mathcal{A}_i^{\text{floor}}$ combines rooms.


(d) A *building* $\mathcal{A}_i^{\text{bldg}}$ combines floors.


(e) A *neighborhood* $\mathcal{A}^{\text{nbd}}$ combines buildings.

benchmark	$\lvert Q\rvert$	$\lvert E\rvert$	exec. time [s] DI-high	DI-mid	DI-low
Patrol1	$\mathbf{10^8}$	$\mathbf{10^8}$	21	42	83
Patrol2	$\mathbf{10^8}$	$\mathbf{10^8}$	23	48	90
Patrol3	$\mathbf{10^9}$	$\mathbf{10^9}$	22	43	89
Patrol4	$\mathbf{10^9}$	$\mathbf{10^9}$	30	60	121
Wholesale1	$\mathbf{10^8}$	$\mathbf{2 \cdot 10^8}$	130	260	394
Wholesale2	$\mathbf{10^8}$	$\mathbf{2 \cdot 10^8}$	92	179	274
Wholesale3	$\mathbf{2 \cdot 10^8}$	$\mathbf{4 \cdot 10^8}$	6	12	23
Wholesale4	$\mathbf{2 \cdot 10^8}$	$\mathbf{4 \cdot 10^8}$	129	260	393

benchmark	$\lvert Q\rvert$	$\lvert E\rvert$	exec. time [s] FZ-none	FZ-int.	FZ-all (PRISM)
Packets1	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	65
Packets2	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	64
Packets3	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	56
Packets4	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	56
Patrol5	$\mathbf{10^8}$	$\mathbf{10^8}$	**22**	**22**	TO
Wholesale5	$\mathbf{5 \cdot 10^7}$	$\mathbf{10^8}$	TO	14	TO

$\lvert Q\rvert$ is the number of positions; $\lvert E\rvert$ is the number of transitions (only counting action branching, not probabilistic branching); execution time is the average of five runs, in sec.; timeout (TO) is 1200 sec.

Apple MacBook Pro 2.3 GHz Dual-Core Intel Core i5 with 16GB of RAM

Experiment Results

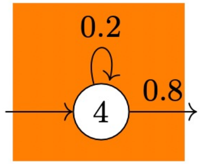
Watanabe (NII, Tokyo)

DI (degree of identification):
how much the same components are
indeed recognized to be identical

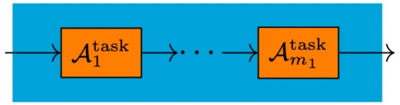
- Compositional algorithm can be **arbitrary faster** (reuse $\mathcal{S}(\mathcal{A})$!)

$$\mathcal{S}(\mathcal{A} \star \cdots \star \mathcal{A}) = \mathcal{S}(\mathcal{A}) \star \cdots \star \mathcal{S}(\mathcal{A})$$

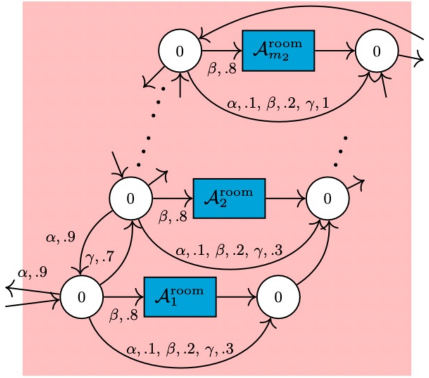
- Overall, we do indeed witness the performance advantage of compositionality
- We need MDPs given in a compositional formalism. This is realistic. Our *Patrol* benchmark:



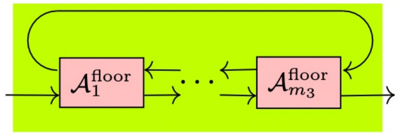
(a) A *task* $\mathcal{A}_i^{\text{task}}$.



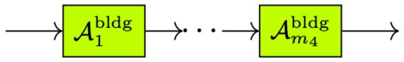
(b) A *room* $\mathcal{A}_i^{\text{room}}$ combines tasks.



(c) A *floor* $\mathcal{A}_i^{\text{floor}}$ combines rooms.



(d) A *building* $\mathcal{A}_i^{\text{bldg}}$ combines floors.



(e) A *neighborhood* $\mathcal{A}^{\text{nbd}}$ combines buildings.

benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	exec. time [s] DI-high	DI-mid	DI-low
Patrol1	10^8	10^8	21	42	83
Patrol2	10^8	10^8	23	48	90
Patrol3	10^9	10^9	22	43	89
Patrol4	10^9	10^9	30	60	121
Wholesale1	10^8	$2 \cdot 10^8$	130	260	394
Wholesale2	10^8	$2 \cdot 10^8$	92	179	274
Wholesale3	$2 \cdot 10^8$	$4 \cdot 10^8$	6	12	23
Wholesale4	$2 \cdot 10^8$	$4 \cdot 10^8$	129	260	393

performance improves

benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	FZ-none	FZ-int.	FZ-all (PRISM)
Packets1	$2.5 \cdot 10^5$	$5 \cdot 10^5$	TO	1	65
Packets2	$2.5 \cdot 10^5$	$5 \cdot 10^5$	TO	3	64
Packets3	$2.5 \cdot 10^5$	$5 \cdot 10^5$	TO	1	56
Packets4	$2.5 \cdot 10^5$	$5 \cdot 10^5$	TO	3	56
Patrol5	10^8	10^8	**22**	**22**	TO
Wholesale5	$5 \cdot 10^7$	10^8	TO	14	TO

$\lvert Q \rvert$ is the number of positions; $\lvert E \rvert$ is the number of transitions (only counting action branching, not probabilistic branching); execution time is the average of five runs, in sec.; timeout (TO) is 1200 sec.

Apple MacBook Pro 2.3 GHz Dual-Core Intel Core i5 with 16GB of RAM

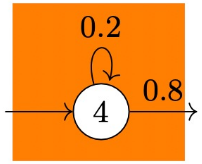
Experiment Results

Scalability:
big MDPs are model checked in realistic time

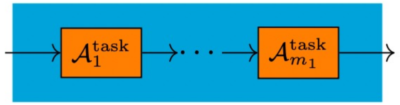
- Compositional algorithm can be **arbitrary faster** (reuse $\mathcal{S}(\mathcal{A})$!)

$$\mathcal{S}(\mathcal{A} \star \cdots \star \mathcal{A})$$
$$= \mathcal{S}(\mathcal{A}) \star \cdots \star \mathcal{S}(\mathcal{A})$$

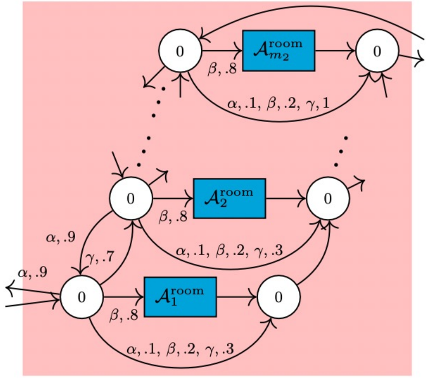
- Overall, we do indeed witness the performance advantage of compositionality
- We need MDPs given in a compositional formalism. This is realistic. Our *Patrol* benchmark:



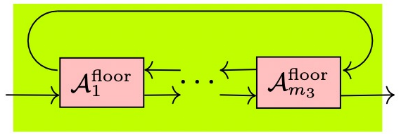
(a) A *task* $\mathcal{A}_i^{\mathrm{task}}$.



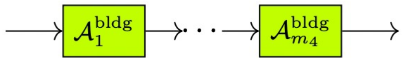
(b) A *room* $\mathcal{A}_i^{\mathrm{room}}$ combines tasks.



(c) A *floor* $\mathcal{A}_i^{\mathrm{floor}}$ combines rooms.



(d) A *building* $\mathcal{A}_i^{\mathrm{bldg}}$ combines floors.



(e) A *neighborhood* $\mathcal{A}^{\mathrm{nbd}}$ combines buildings.

benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	exec. time [s] DI-high	DI-mid	DI-low
Patrol1	$\mathbf{10^8}$	$\mathbf{10^8}$	21	42	83
Patrol2	$\mathbf{10^8}$	$\mathbf{10^8}$	23	48	90
Patrol3	$\mathbf{10^9}$	$\mathbf{10^9}$	22	43	89
Patrol4	$\mathbf{10^9}$	$\mathbf{10^9}$	30	60	121
Wholesale1	$\mathbf{10^8}$	$\mathbf{2 \cdot 10^8}$	130	260	394
Wholesale2	$\mathbf{10^8}$	$\mathbf{2 \cdot 10^8}$	92	179	274
Wholesale3	$\mathbf{2 \cdot 10^8}$	$\mathbf{4 \cdot 10^8}$	6	12	23
Wholesale4	$\mathbf{2 \cdot 10^8}$	$\mathbf{4 \cdot 10^8}$	129	260	393

benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	exec. time [s] FZ-none	FZ-int.	FZ-all (PRISM)
Packets1	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	65
Packets2	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	64
Packets3	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	56
Packets4	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	56
Patrol5	$\mathbf{10^8}$	$\mathbf{10^8}$	**22**	**22**	TO
Wholesale5	$\mathbf{5 \cdot 10^7}$	$\mathbf{10^8}$	TO	14	TO

$\lvert Q \rvert$ is the number of positions; $\lvert E \rvert$ is the number of transitions (only counting action branching, not probabilistic branching); execution time is the average of five runs, in sec.; timeout (TO) is 1200 sec.

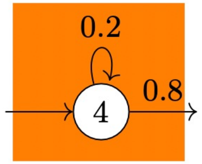
Apple MacBook Pro 2.3 GHz Dual-Core Intel Core i5 with 16GB of RAM

Experiment Results

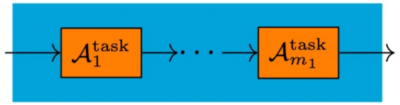
- Compositional algorithm can be **arbitrary faster** (reuse $\mathcal{S}(\mathcal{A})$!)

$$\mathcal{S}(\mathcal{A} \star \cdots \star \mathcal{A}) = \mathcal{S}(\mathcal{A}) \star \cdots \star \mathcal{S}(\mathcal{A})$$

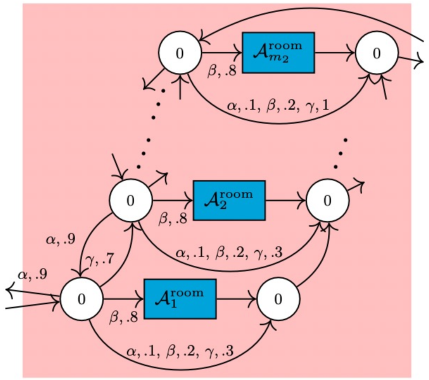
- Overall, we do indeed witness the performance advantage of compositionality
- We need MDPs given in a compositional format. This is realistic. Our *Patrol* benchmark:



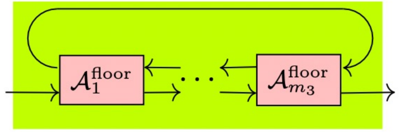
(a) A *task* $\mathcal{A}_i^{\text{task}}$.



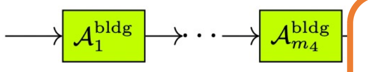
(b) A *room* $\mathcal{A}_i^{\text{room}}$ combines tasks.



(c) A *floor* $\mathcal{A}_i^{\text{floor}}$ combines rooms.



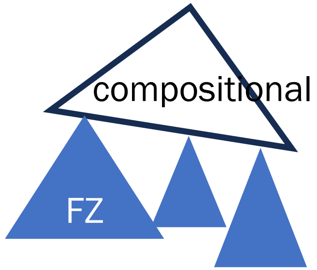
(d) A *building* $\mathcal{A}_i^{\text{bldg}}$ combines floors.



(e) A *neighborhood* $\mathcal{A}^{\text{nbd}}$ co... buildings.

benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	exec. time [s] DI-high	DI-mid	DI-low
Patrol1	$\mathbf{10^8}$	$\mathbf{10^8}$	21	42	83
Patrol2	$\mathbf{10^8}$	$\mathbf{10^8}$	23	48	90
Patrol3	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

FZ (freezing):
We can stop doing compositionally at a certain depth
(FZ-all = no compositionality; we used PRISM)

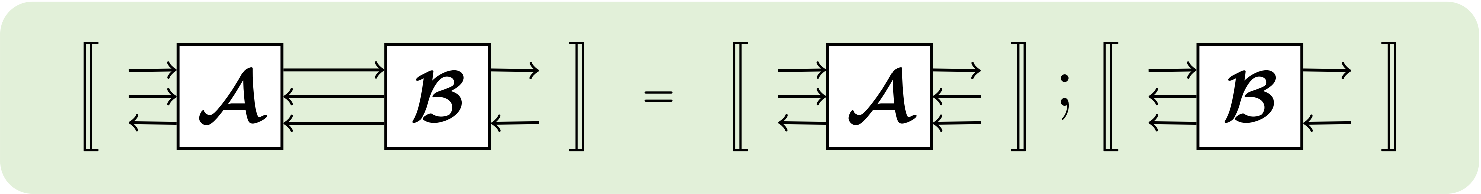


benchmark	$\lvert Q \rvert$	$\lvert E \rvert$	exec. time [s] FZ-none	FZ-int.	FZ-all (PRISM)
Packets1	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	65
Packets2	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	64
Packets3	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	1	56
Packets4	$\mathbf{2.5 \cdot 10^5}$	$\mathbf{5 \cdot 10^5}$	TO	3	56
Patrol5	$\mathbf{10^8}$	$\mathbf{10^8}$	**22**	**22**	TO
Wholesale5	$\mathbf{5 \cdot 10^7}$	$\mathbf{10^8}$	TO	14	TO

$\lvert Q \rvert$ is the number of positions; $\lvert E \rvert$ is the number of tr... tions (only counting ... average of

- Compositionality helps
- But going all the way down may not be a good idea

Outline



- Target problem: optimal expected reward of MDPs
- Composition formalism: string diagrams of MDPs
- Compositional solution of MDPs
- Upgrading compositional solution for free
- Experimental evaluation
- Conclusions

Related Work (Compositional Probabilistic MC)

- **Probabilistic** model checking is an active field (Baier, Larsen, Katoen, Kwiatkowska, Parker, Raskin, …)
- **Compositionality** in model checking is an old problem [Clarke, Long & McMillan, LICS'89] [Tsukada & Ong, LICS'14] …
- Two closely related works on **compositional probabilistic** model checking:

Probabilistic Model Checking wrt. Parallel Composition ∥

[Kwiatkowska, Norman, Parker & Qu, Inf. Comp. '13]

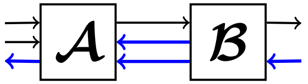
- Compositional model checking of **parallel composition** $\mathcal{A} \parallel \mathcal{B}$
- … but **assume-guarantee "contracts"** betw. $\mathcal{A}$ and $\mathcal{B}$ must be devised
- Harder problem in general

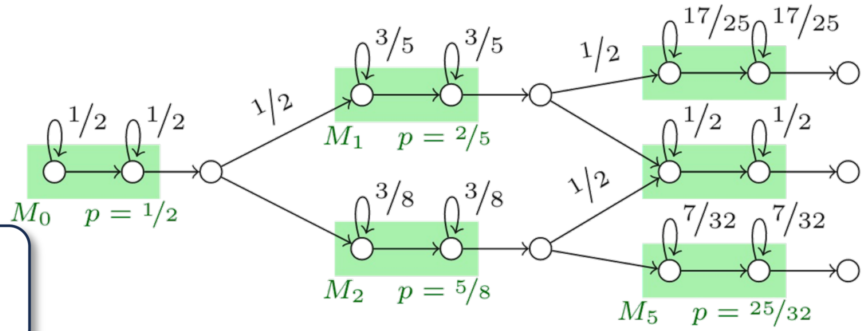
Parametric MDP Model Checking for Sequential Composition

[Junges & Spaan, CAV'22]

- Sequential composition of **parametric MDPs**
- **Unidirectional** composition

Ours is **bidirectional**





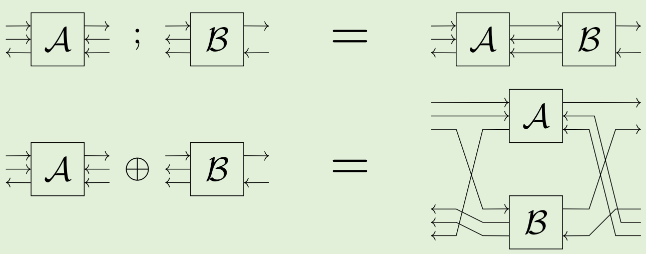
- Assumption: locally optimal schedulers are globally optimal, too
  (It holds if component exits are unique. We don't need this assumption)
- Compositional solution of **parametric** components $\mathcal{A}(p)$
  (We don't do this)

Watanabe (NII, Tokyo)

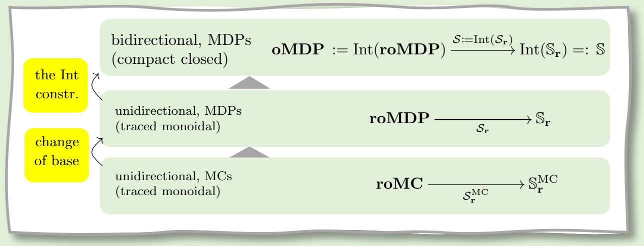
Monoidal Categories Guiding Planer-Compositional Model Checking

<u>“Our general methodology”</u>:

- Composition by **string diagrams**

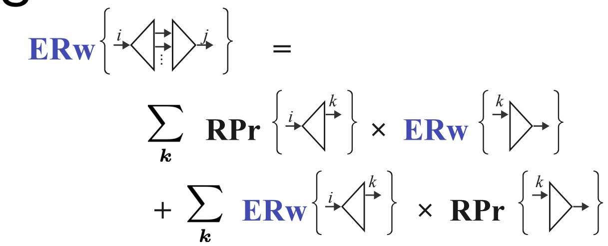


- Semantic domains from **category theory**
- **Upgrading frameworks for free**



We applied it to **MDP model checking**

- semantic categories by decomposition equalities



- a **compositional algorithm** with clear performance advantage

Future work

- parallel composition
- other problems
  - mean payoff games [Watanabe+, arXiv’23]