Round-bounded control of parameterized systems

Benedikt Bollig[1] Mathieu Lehaut[2] Nathalie Sznajder[2]

[1] CNRS, LSV & ENS Paris-Saclay, Université Paris-Saclay, France

[2] Sorbonne Université, CNRS, LIP6/MoVe, F-75005 Paris, France

Midi de Move

25/10/2018

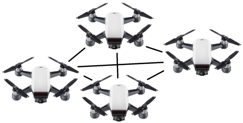
Parameterized system?

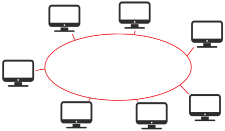
Paremeterized System

Distributed system with no fixed number of processes

Examples

Mobile networks, distributed algorithms, drone swarms, ...





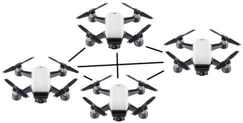
Parameterized system?

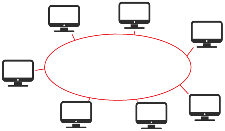
Paremeterized System

Distributed system with no fixed number of processes

Examples

Mobile networks, distributed algorithms, drone swarms, ...





⇒ Should work for any number of processes!

Reachability vs Control

Systems interact with environment: User inputs, Drone sensors, ...

Reachability vs Control

Systems interact with environment: User inputs, Drone sensors, ...

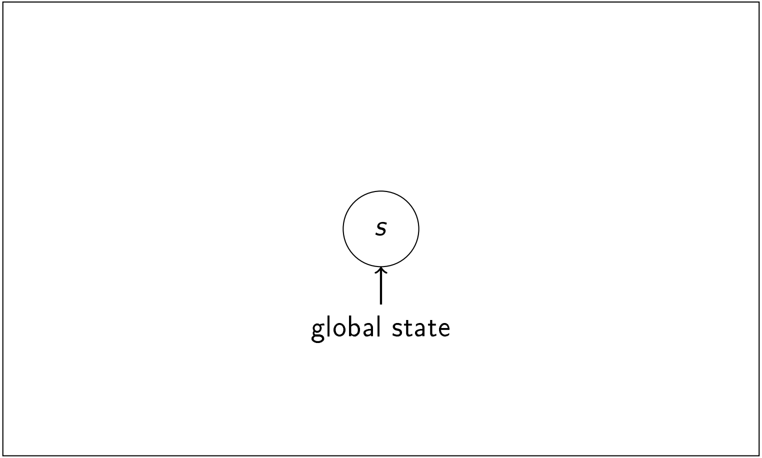
Reachability

Is there a correct behavior?

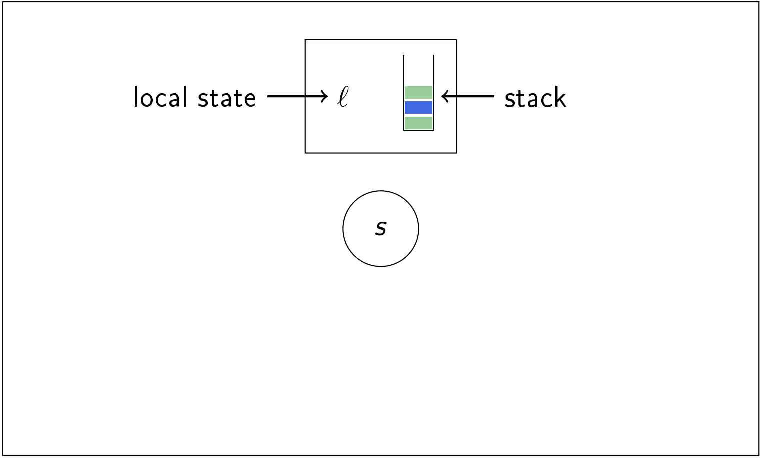
Control

Can we force every behavior to be correct?

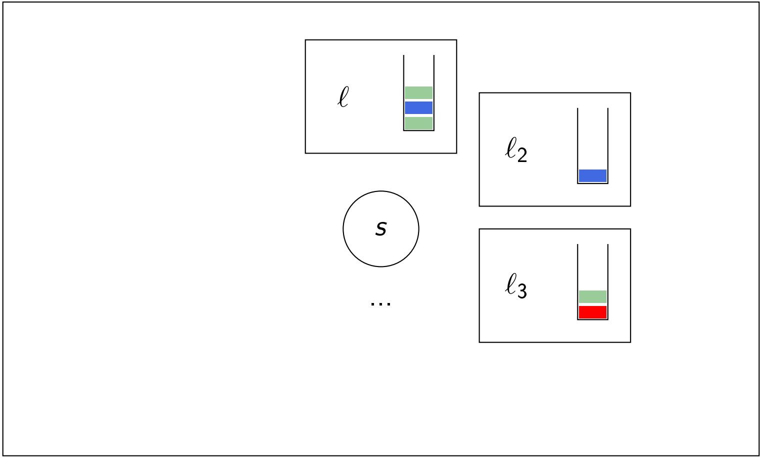
Parameterized Pushdown Systems



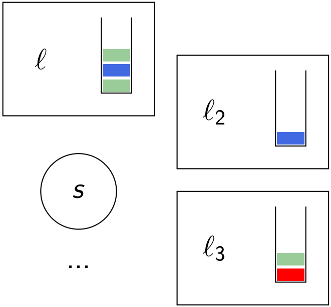
Parameterized Pushdown Systems



Parameterized Pushdown Systems

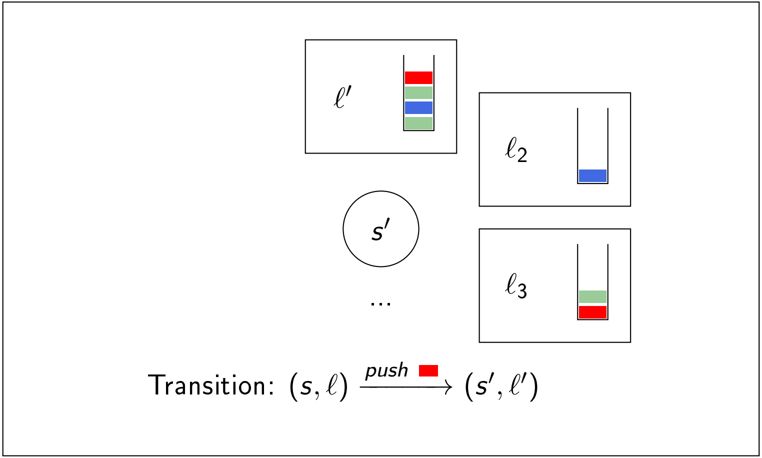


Parameterized Pushdown Systems



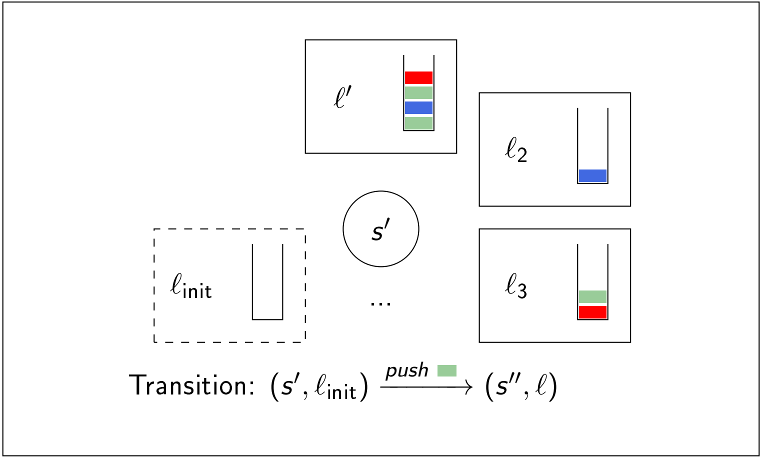
Transition: $(s, \ell) \xrightarrow{push\ 🟥} (s', \ell')$

Parameterized Pushdown Systems



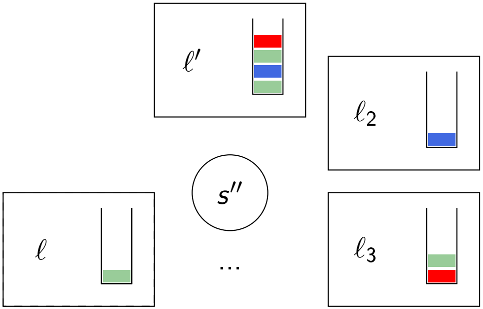
Transition: $(s, \ell) \xrightarrow{\textit{push}\ \text{🟥}} (s', \ell')$

Parameterized Pushdown Systems



$$\text{Transition: } (s', \ell_{\text{init}}) \xrightarrow{\textit{push}\ \text{🟩}} (s'', \ell)$$

Parameterized Pushdown Systems



Transition: $(s', \ell_{\mathrm{init}}) \xrightarrow{\mathit{push}\ 🟩} (s'', \ell)$

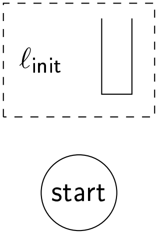
Example: Scheduler model (1/2)

Behavior we want to model:

1. Scheduler sends requests and starts processes,
2. Then each process performs the requested tasks,
3. Then each process stops.

Example: Scheduler model (2/2)

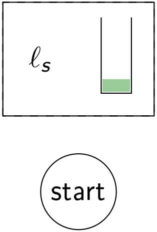
1. Scheduler sends requests and starts processes



$$(\text{start}, \ell_{\text{init}}) \xrightarrow{\mathit{push}\ \text{🟩}} (\text{start}, \ell_s)$$

Example: Scheduler model (2/2)

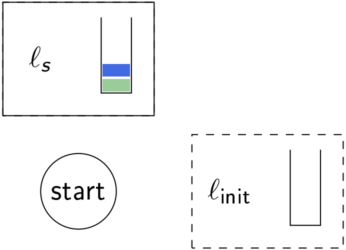
1. Scheduler sends requests and starts processes



$$(\text{start}, \ell_s) \xrightarrow{\textit{push}\ 🟦} (\text{start}, \ell_s)$$

Example: Scheduler model (2/2)

1. Scheduler sends requests and starts processes

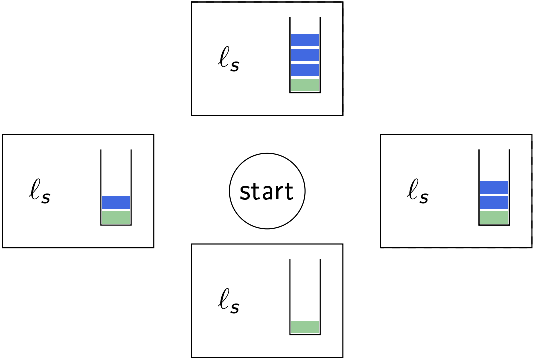


$$(\mathsf{start}, \ell_{\mathsf{init}}) \xrightarrow{\textit{push}\ \text{🟩}} (\mathsf{start}, \ell_s)$$

…

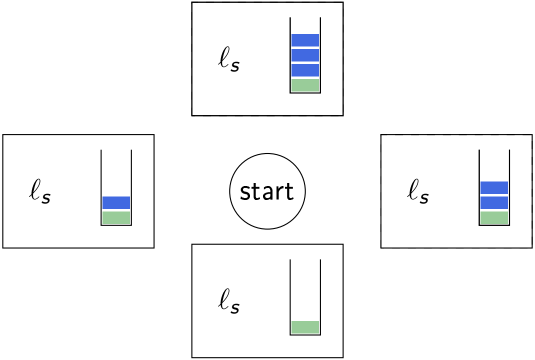
Example: Scheduler model (2/2)

1. Scheduler sends requests and starts processes



Example: Scheduler model (2/2)

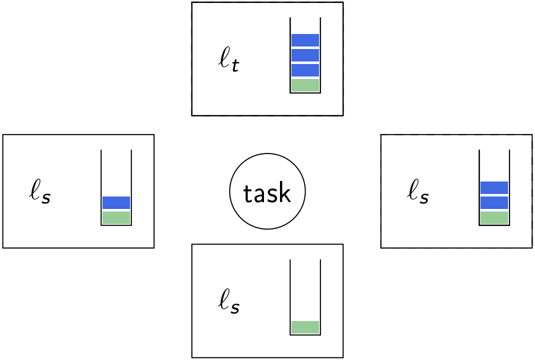
2. Each process performs the requested tasks



$$(\mathsf{start}, \ell_s) \xrightarrow{-} (\mathsf{task}, \ell_t)$$

Example: Scheduler model (2/2)

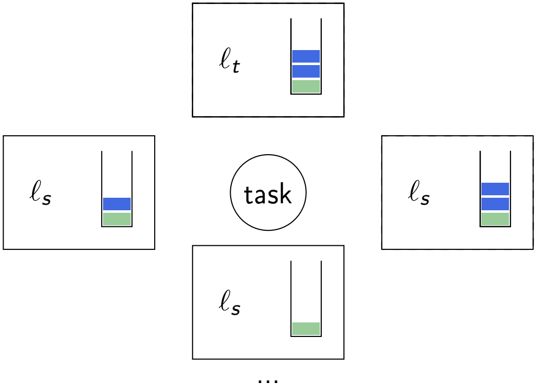
2. Each process performs the requested tasks



$$(\text{task}, \ell_t) \xrightarrow{pop\ \blacksquare} (\text{task}, \ell_t)$$

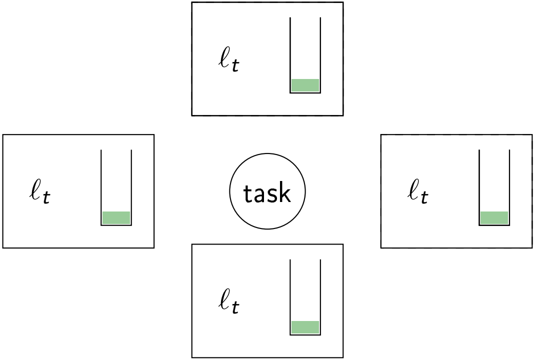
Example: Scheduler model (2/2)

2. Each process performs the requested tasks



Example: Scheduler model (2/2)

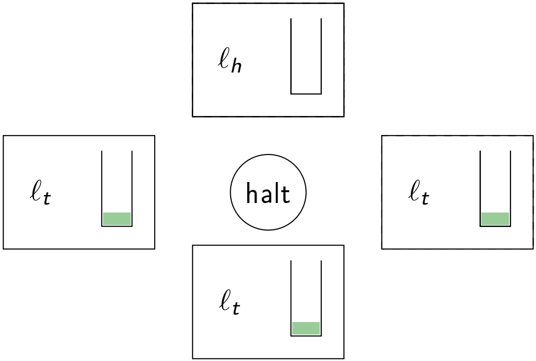
3. Each process stops



$$(\text{task}, \ell_t) \xrightarrow{pop\ \text{🟩}} (\text{halt}, \ell_h)$$

Example: Scheduler model (2/2)

3. Each process stops



$$(\text{halt}, \ell_t) \xrightarrow{pop\ 🟩} (\text{halt}, \ell_h)$$

Round-bounded behaviors

Even Reachability is undecidable!

Round-bounded behaviors

Even Reachability is undecidable!

Restriction: rounds [La Torre et al., 2010]

One round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

Round-bounded behaviors

Even Reachability is undecidable!

Restriction: rounds [La Torre et al., 2010]

One round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...

Round-bounded behaviors

Even Reachability is undecidable!

Restriction: rounds [La Torre et al., 2010]

One round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...

✗ 1 1 1 2 2 3 3 1 4 4 6 ...

Round-bounded behaviors

Even Reachability is undecidable!

Restriction: rounds [La Torre et al., 2010]

One round:

- ✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...
- ✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...
- X 1 1 1 2 2 3 3 1 4 4 6 ...

N rounds:

1 1 1 2 3 4 4 | 1 3 3 4 5 | 3 3 5 6 6 6 7 | ...

Round-bounded behaviors

Even Reachability is undecidable!

Restriction: rounds [La Torre et al., 2010]

One round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...

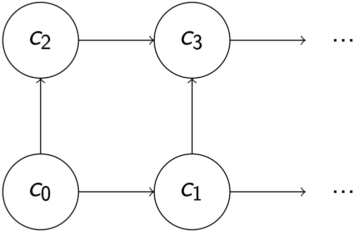
X 1 1 1 2 2 3 3 1 4 4 6 ...

N rounds:

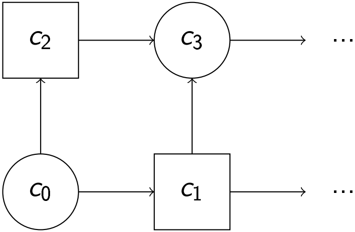
1 1 1 2 3 4 4 | 1 3 3 4 5 | 3 3 5 6 6 6 7 | ...

Round-bounded Reachability is PSPACE-complete.

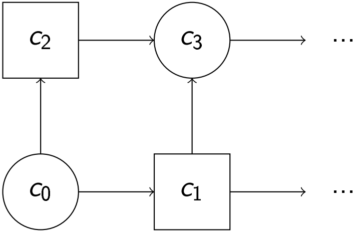
Control: Parameterized Pushdown Games



Control: Parameterized Pushdown Games



Control: Parameterized Pushdown Games



Control problem

Is there a winning strategy for the Controller?

Main result

Decidable, but inherently non-elementary.

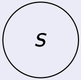
Decidability

Reduction to *phase-bounded multi-pushdown games*
[Atig et al., 2017]

Decidability

Reduction to *phase-bounded multi-pushdown games*
[Atig et al., 2017]

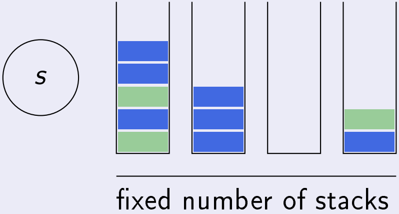
Multi-pushdown systems



Decidability

Reduction to *phase-bounded multi-pushdown games* [Atig et al., 2017]

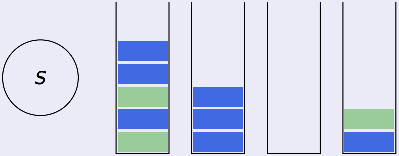
Multi-pushdown systems



Decidability

Reduction to *phase-bounded multi-pushdown games*
[Atig et al., 2017]

Multi-pushdown systems

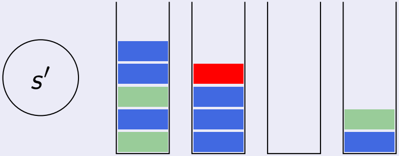


$$s \xrightarrow{push\ 2\ 🟥} s'$$

Decidability

Reduction to *phase-bounded multi-pushdown games*
[Atig et al., 2017]

Multi-pushdown systems

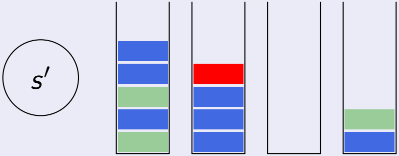


$$s \xrightarrow{push\ 2\ 🟥} s'$$

Decidability

Reduction to *phase-bounded multi-pushdown games*
[Atig et al., 2017]

Multi-pushdown systems

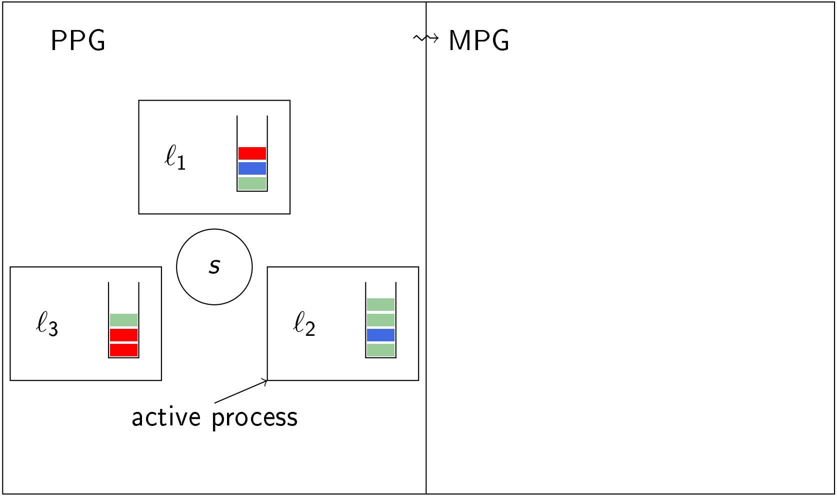


$$s \xrightarrow{\textit{push}\ 2\ 🟥} s'$$

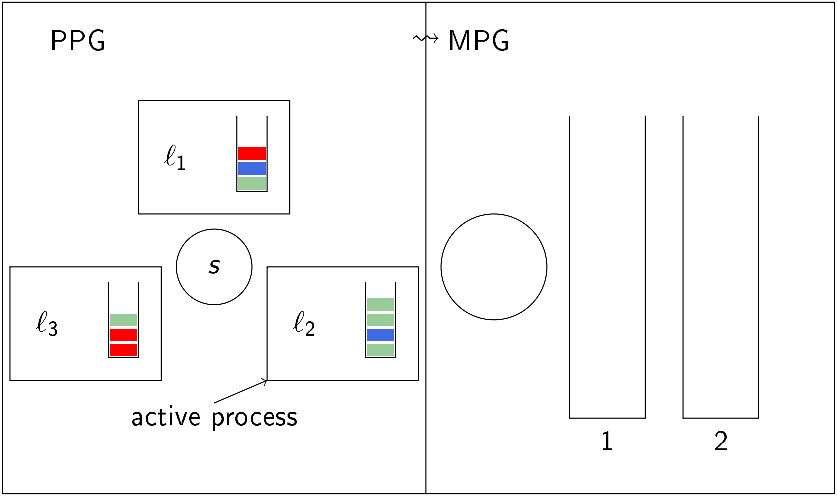
Phase-bounded

Phase: pop only from one stack, push unrestricted.

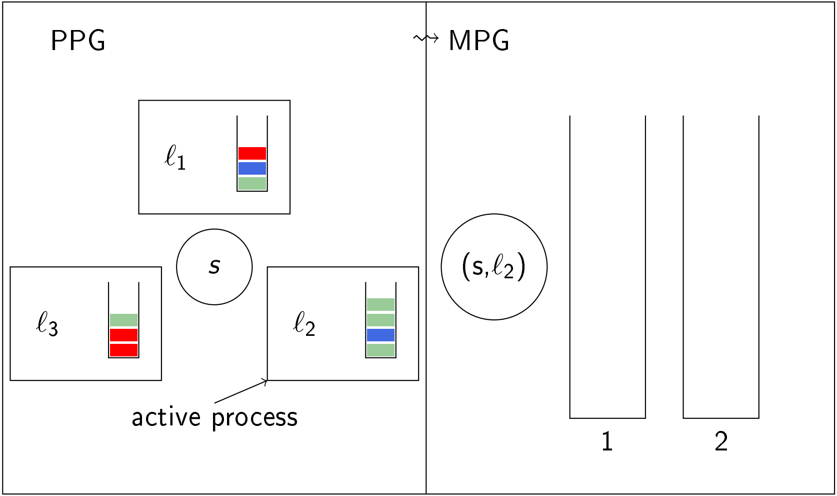
Sketch of the reduction - encoding configurations



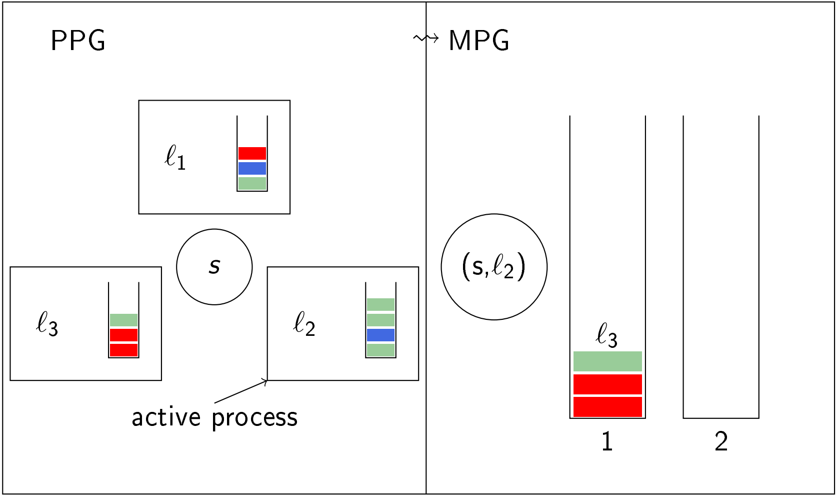
Sketch of the reduction - encoding configurations



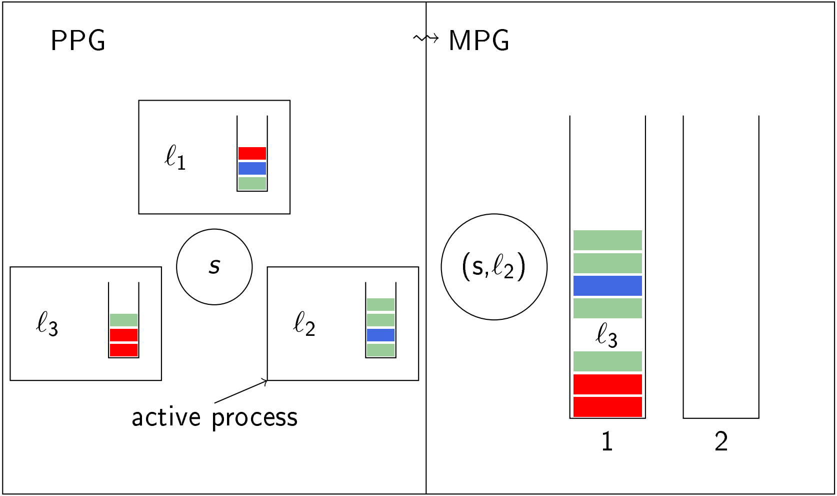
Sketch of the reduction - encoding configurations



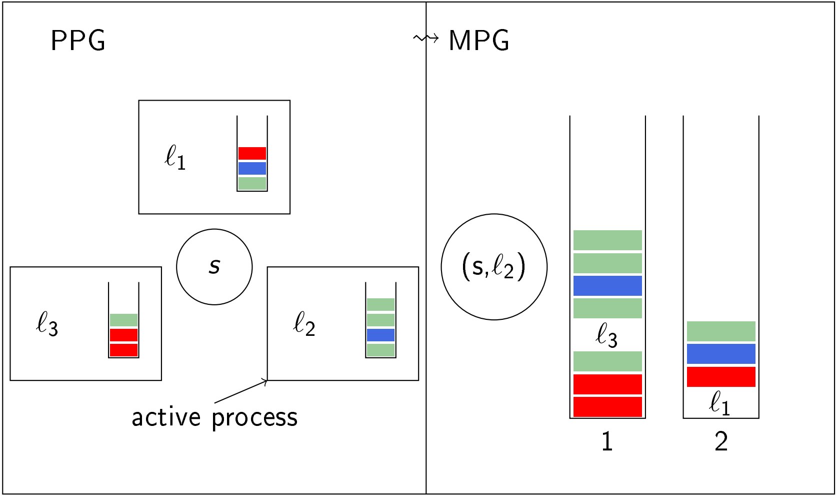
Sketch of the reduction - encoding configurations



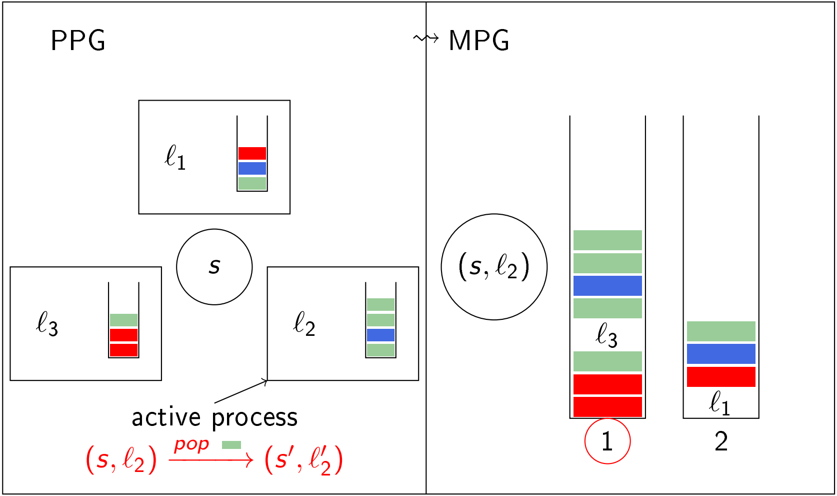
Sketch of the reduction - encoding configurations



Sketch of the reduction - encoding configurations

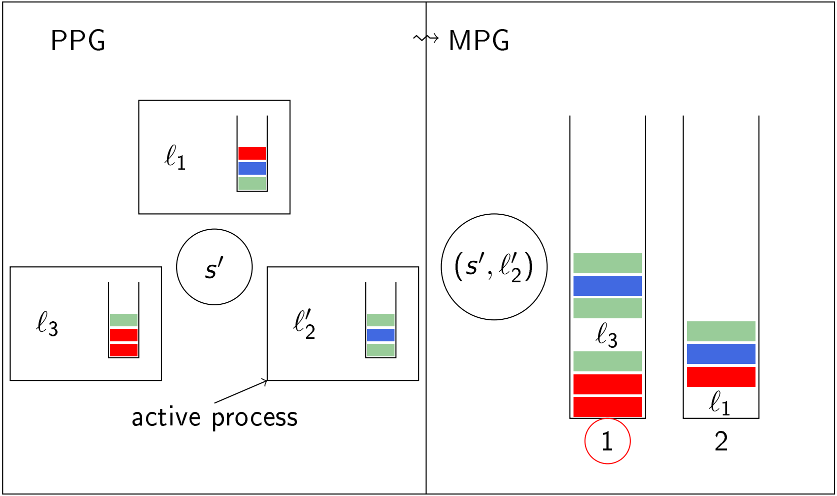


Sketch of the reduction - simulating transitions

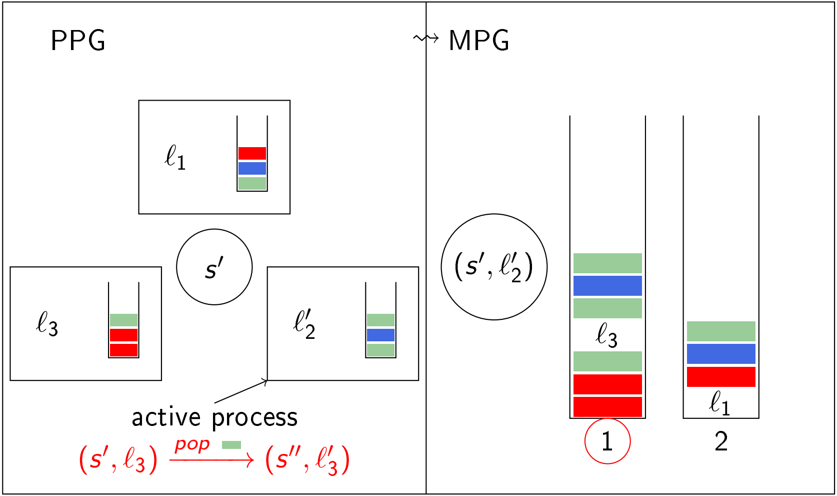


$(s, \ell_2) \xrightarrow{pop\ 🟩} (s', \ell'_2)$

Sketch of the reduction - simulating transitions



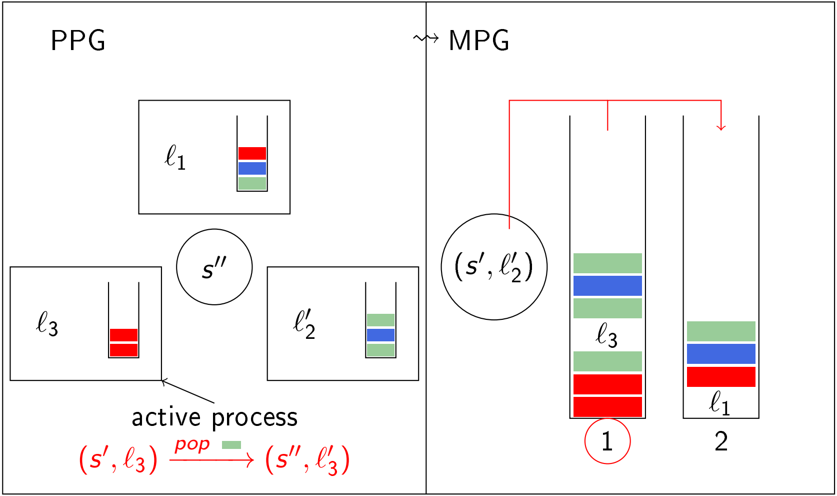
Sketch of the reduction - simulating transitions



active process

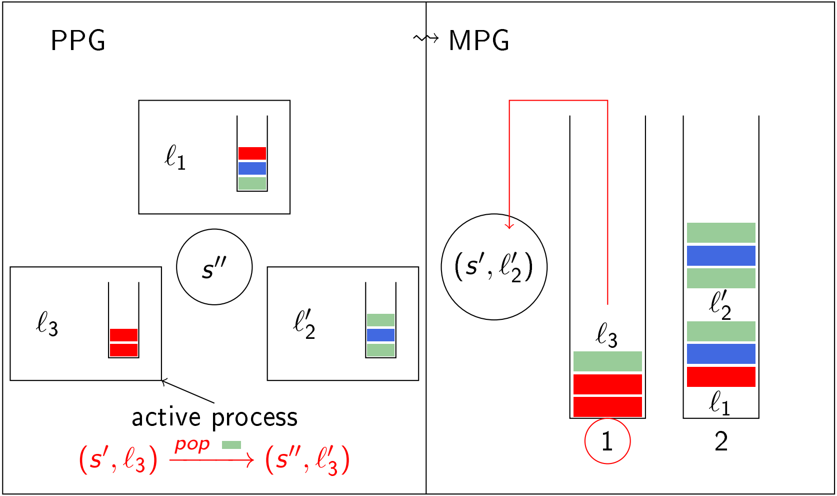
$(s', \ell_3) \xrightarrow{\text{pop 🟩}} (s'', \ell'_3)$

Sketch of the reduction - simulating transitions

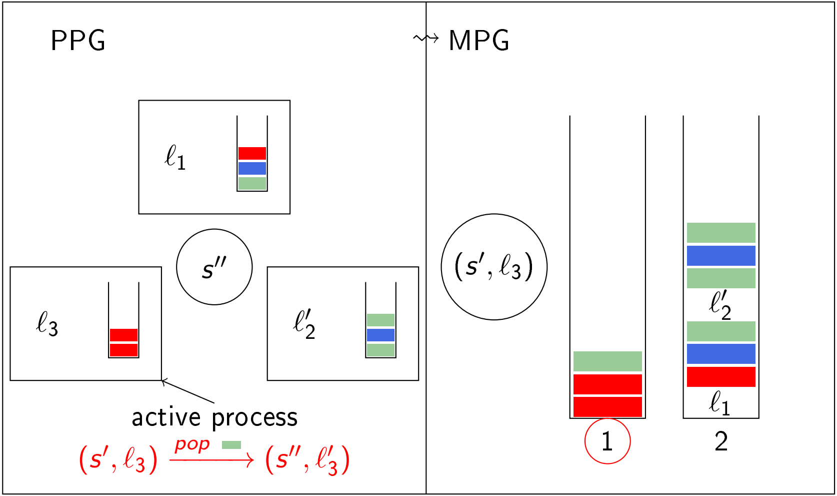


$$(s', \ell_3) \xrightarrow{pop\ 🟩} (s'', \ell_3')$$

Sketch of the reduction - simulating transitions

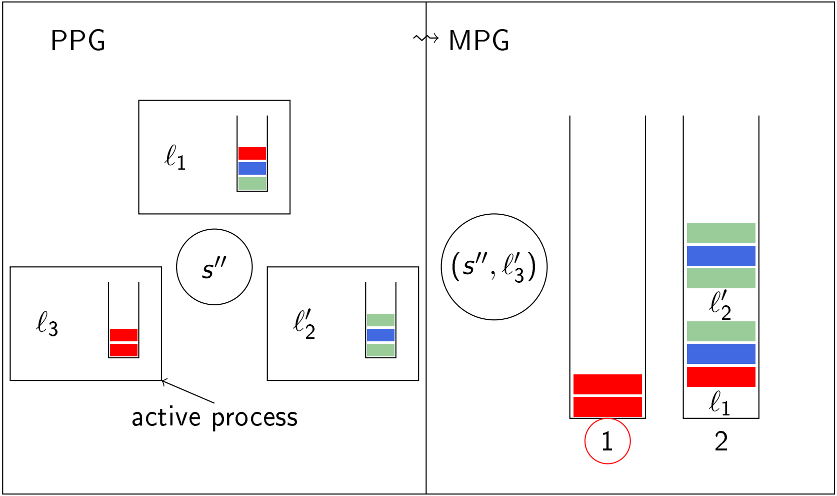


Sketch of the reduction - simulating transitions

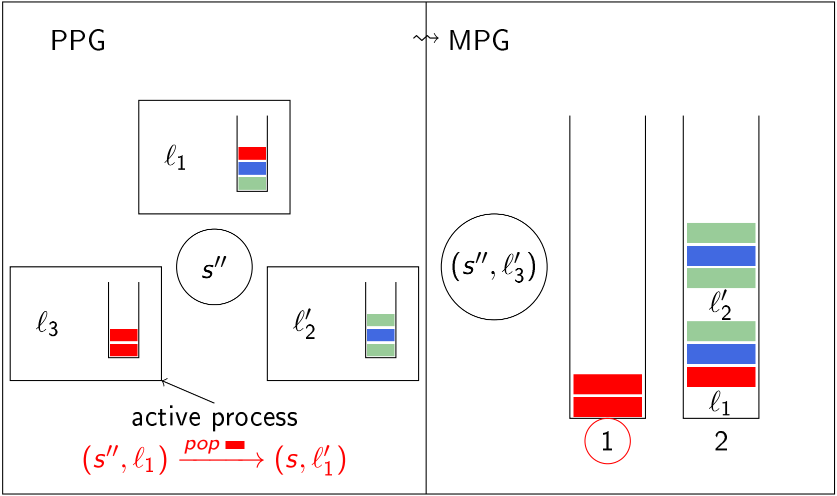


$(s', \ell_3) \xrightarrow{pop\ 🟩} (s'', \ell'_3)$

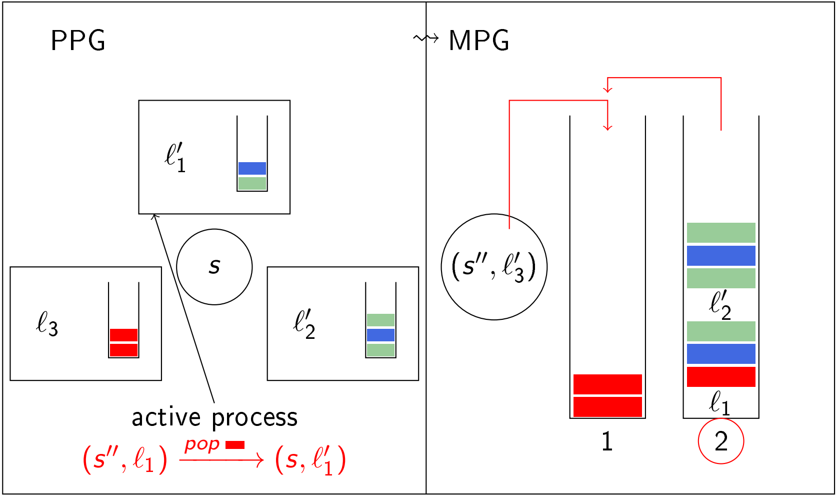
Sketch of the reduction - simulating transitions



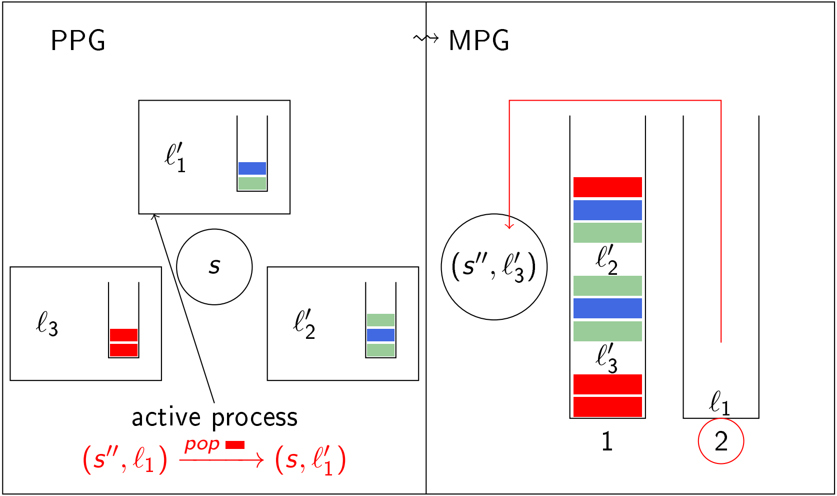
Sketch of the reduction - simulating transitions



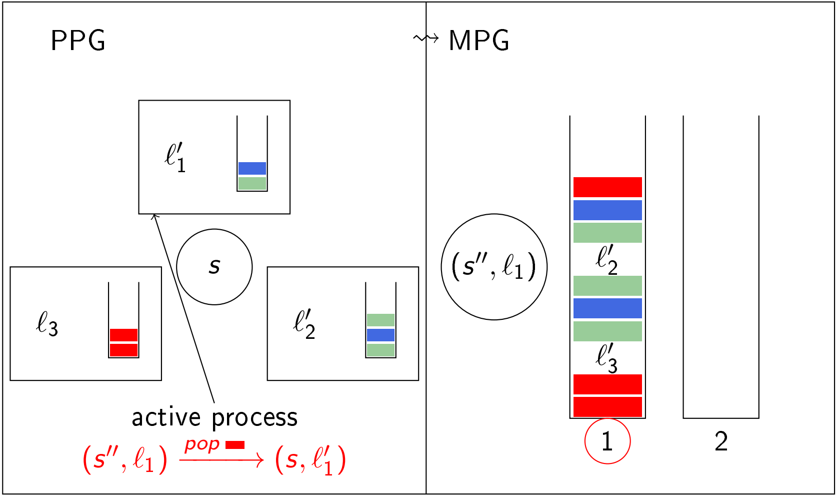
Sketch of the reduction - simulating transitions



Sketch of the reduction - simulating transitions



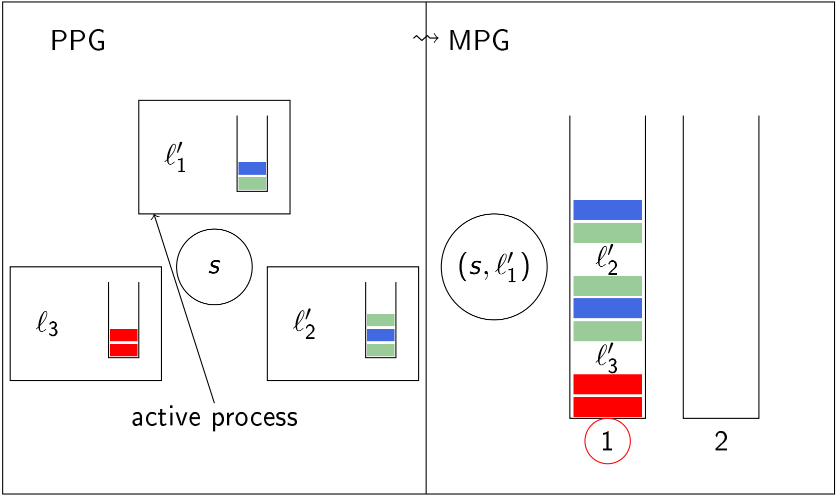
Sketch of the reduction - simulating transitions



active process

$(s'', \ell_1) \xrightarrow{pop\ 🟥} (s, \ell_1')$

Sketch of the reduction - simulating transitions



Hardness (1/2)

Reduction of satisfiability of FO($<$) on finite words. [Stockmeyer, 1970]

Example: $\exists x.\forall y.\mathrm{a}(x) \wedge (\mathrm{b}(y) \Rightarrow y < x)$

✓ a b c b a c c a

✗ a a b a b c

Hardness (1/2)

Reduction of satisfiability of FO(<) on finite words. [Stockmeyer, 1970]

Example: $\exists x.\forall y.\mathrm{a}(x) \wedge (\mathrm{b}(y) \Rightarrow y < x)$

✓ a b c b a c c a

X a a b a b c

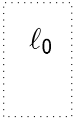
Syntax

$$\begin{aligned} t &::= \mathrm{a}(x) \mid x < y \mid x = y \\ \varphi &::= t \mid \neg t \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x.\varphi \mid \forall x.\varphi \end{aligned}$$

with $x, y \in \mathrm{Var}$ and $a \in \Sigma$.

Hardness (2/2)

Step 1: Controller creates the witness



$$\exists x. \forall y. \mathrm{a}(x) \land (\neg \mathrm{b}(y) \lor y < x)$$

Hardness (2/2)

Step 1: Controller creates the witness

a	b	c	b	a	c	c	a

$$\exists x . \forall y . \mathrm{a}(x) \wedge (\neg \mathrm{b}(y) \vee y < x)$$

Hardness (2/2)

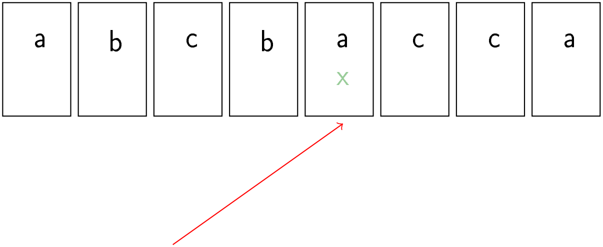
Step 2: Controller and Environment assign values to variables

a	b	c	b	a	c	c	a

$$\underline{\exists x}.\forall y.\mathrm{a}(x) \wedge (\neg \mathrm{b}(y) \vee y < x)$$

Hardness (2/2)

Step 2: Controller and Environment assign values to variables



$$\exists x.\forall y.\mathrm{a}(x) \wedge (\neg \mathrm{b}(y) \vee y < x)$$

Hardness (2/2)

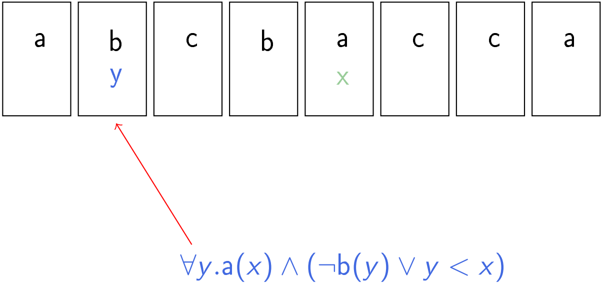
Step 2: Controller and Environment assign values to variables

a	b	c	b	a	c	c	a
				x			

$$\underline{\forall y}.\mathrm{a}(x) \wedge (\neg \mathrm{b}(y) \vee y < x)$$

Hardness (2/2)

Step 2: Controller and Environment assign values to variables



$$\forall y.\mathrm{a}(x) \wedge (\neg \mathrm{b}(y) \vee y < x)$$

Hardness (2/2)

Step 3: Disjunctions and Conjunctions

a	b y	c	b	a x	c	c	a

$$\mathsf{a}(x) \underline{\wedge} (\neg \mathsf{b}(y) \vee y < x)$$

Hardness (2/2)

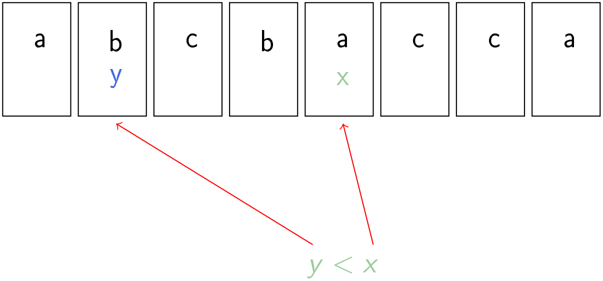
Step 3: Disjunctions and Conjunctions

a	b	c	b	a	c	c	a
	y			x			

$$\neg \mathrm{b}(y) \underline{\lor} y < x$$

Hardness (2/2)

Step 4: Checking the remaining term



Hardness (2/2)

Step 4: Checking the remaining term

a	b	c	b	a	c	c	a
	y			x			

Win!

Conclusion

Round-bounded is an interesting restriction for parameterized pushdown systems, and leads to decidability for both Reachability and Control.

Conclusion

Round-bounded is an interesting restriction for parameterized pushdown systems, and leads to decidability for both Reachability and Control.

Future works:

- Relax round-bounded restriction (not fixed order)
- Use game frameworks for model-checking properties for data logic
- ...

Conclusion

Round-bounded is an interesting restriction for parameterized pushdown systems, and leads to decidability for both Reachability and Control.

Future works:

- Relax round-bounded restriction (not fixed order)
- Use game frameworks for model-checking properties for data logic
- ...

Thanks for listening, questions?