

Realizability of Dynamic MSC Languages

Benedikt Bollig¹ and Loïc Hélouët²

¹LSV, ENS Cachan, CNRS

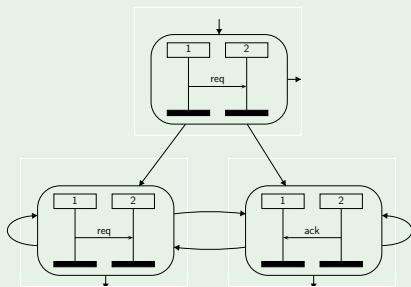
²IRISA, INRIA, Rennes

CSR 2010

Kazan, Russia, June 16–20, 2010

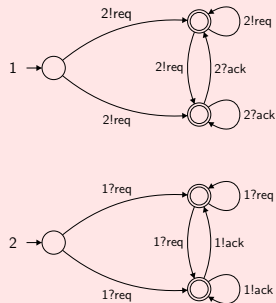
Realizability of Message Sequence Charts

High-Level MSC

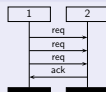


realizable?
⇒

Communicating Automaton



MSC



[AEY'05] Alur & Etassami & Yannakakis. [Realizability and Verification of Message Sequence Graphs](#). 2001/2005.

[HMNST'05] Henriksen & Mukund & Narayan Kumar & Sohoni & Thiagarajan. [A Theory of Regular MSC Languages](#). 2000/2005.

1 Dynamic Communicating Automata

2 Dynamic MSC Grammars

3 Realizability

4 Finite Implementation

Presentation outline

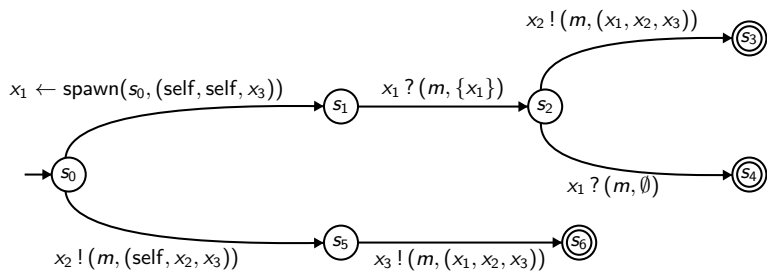
1 Dynamic Communicating Automata

2 Dynamic MSC Grammars

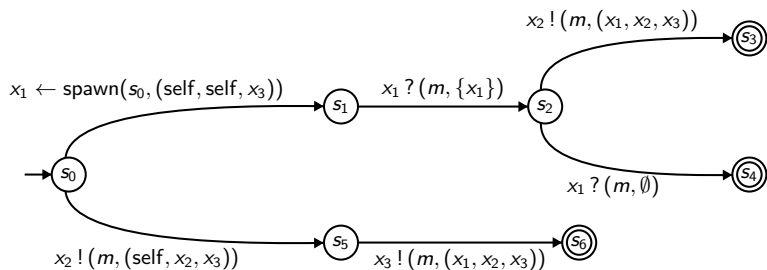
3 Realizability

4 Finite Implementation

Dynamic Communicating Automaton

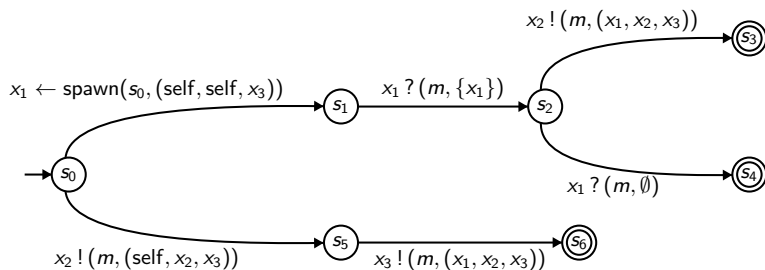


Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3
1	s_0	1	1	1

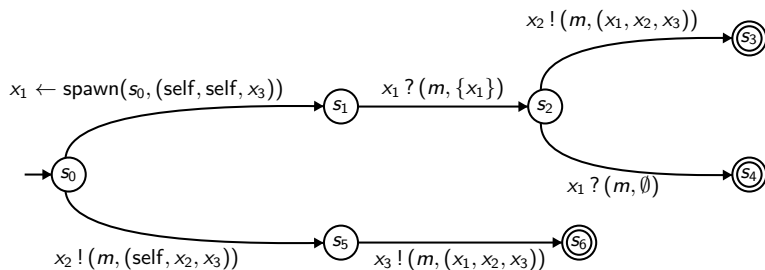
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2
1	s_1	2	1	1	—	—
2	s_0	1	1	1	—	—

spawn(1, 2)

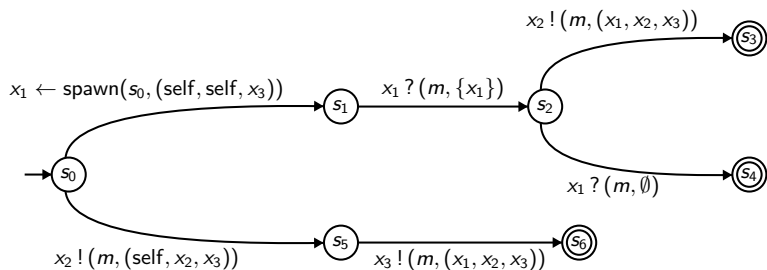
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3
1	s_1	2	1	1	—		
2	s_1	3	1	1		—	
3	s_0	2	2	1			—

`spawn(1, 2) spawn(2, 3)`

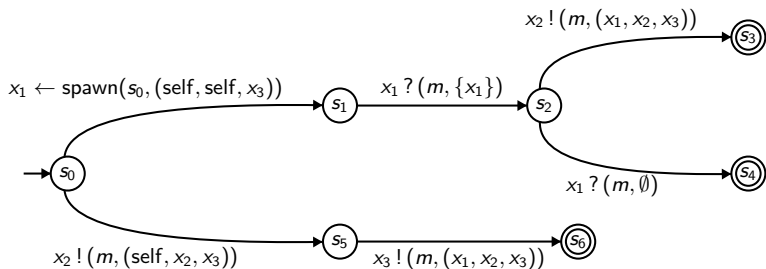
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			
2	s_1	3	1	1		—		
3	s_1	4	2	1			—	
4	s_0	3	3	1				—

spawn(1, 2) spawn(2, 3) spawn(3, 4)

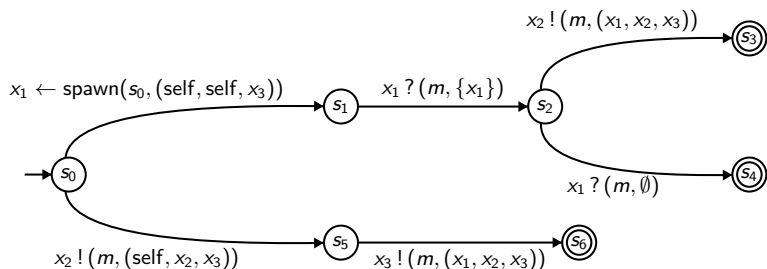
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			
2	s_1	3	1	1		—		
3	s_1	4	2	1			—	$m, (4, 3, 1)$
4	s_5	3	3	1				—

$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) !(4, 3)$

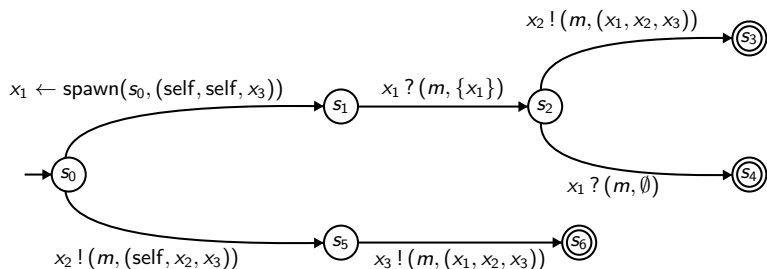
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			$m, (3, 3, 1)$
2	s_1	3	1	1		—		
3	s_1	4	2	1			—	$m, (4, 3, 1)$
4	s_6	3	3	1				—

$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) !(4, 3) !(4, 1)$

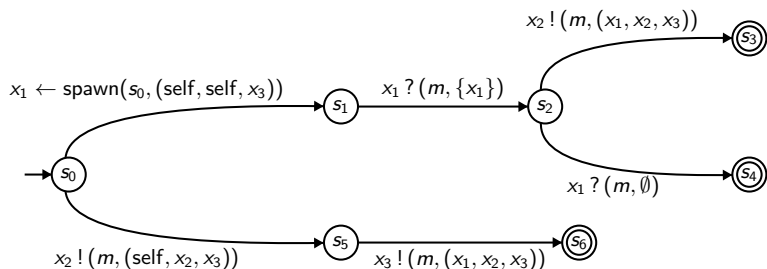
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			$m, (3, 3, 1)$
2	s_1	3	1	1		—		
3	s_2	4	2	1			—	
4	s_6	3	3	1				—

$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) !(4, 3) !(4, 1) ?(4, 3)$

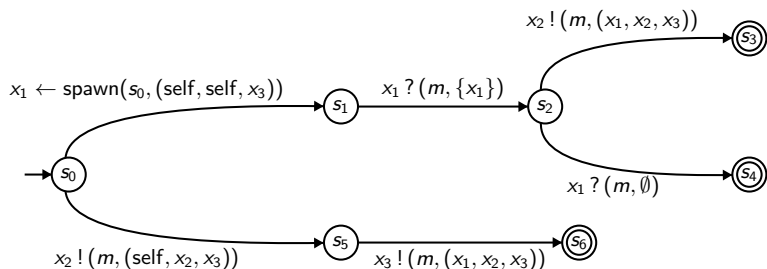
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			$m, (3, 3, 1)$
2	s_1	3	1	1		—	$m, (4, 2, 1)$	
3	s_3	4	2	1			—	
4	s_6	3	3	1				—

spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2)

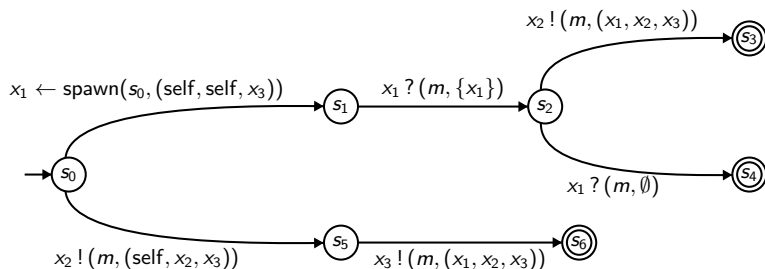
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—			$m, (3, 3, 1)$
2	s_2	4	1	1		—		
3	s_3	4	2	1			—	
4	s_6	3	3	1				—

spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2)

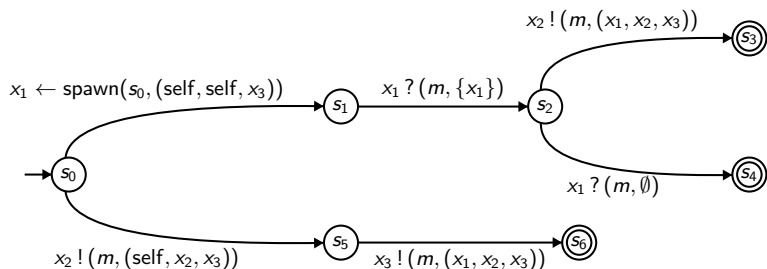
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_1	2	1	1	—	$m, (4, 1, 1)$		$m, (3, 3, 1)$
2	s_3	4	1	1		—		
3	s_3	4	1	1			—	
4	s_6	3	3	1				—

$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1)$

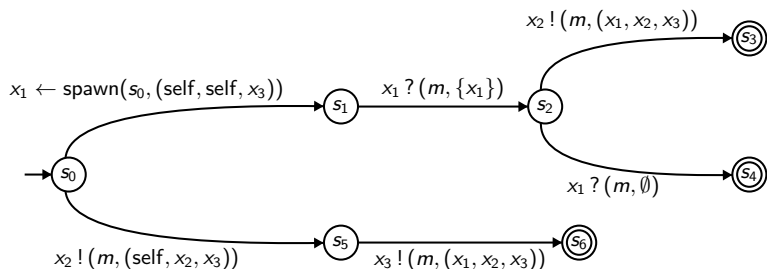
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_2	4	1	1	—			$m, (3, 3, 1)$
2	s_3	4	1	1		—		
3	s_3	4	1	1			—	
4	s_6	3	3	1				—

spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1) ?(2, 1)

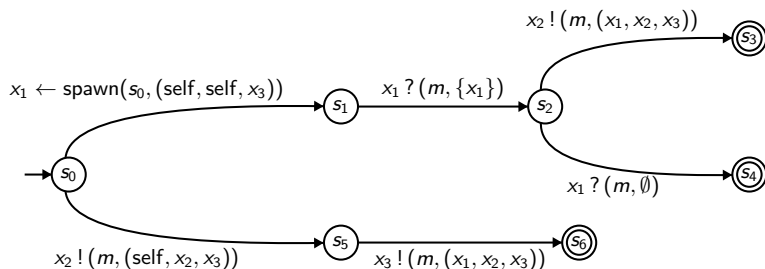
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_4	4	1	1	—			
2	s_3	4	1	1		—		
3	s_3	4	1	1			—	
4	s_6	3	3	1				—

$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1) ?(2, 1) ?(1, 4)$

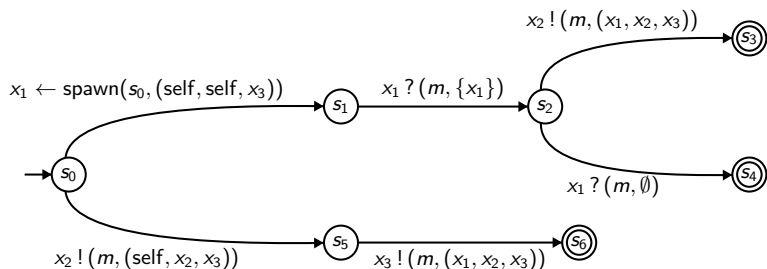
Dynamic Communicating Automaton



Proc	State	x_1	x_2	x_3	1	2	3	4
1	s_4	4	1	1	—			
2	s_3	4	1	1		—		
3	s_3	4	1	1			—	
4	s_6	3	3	1				—

spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1) ?(2, 1) ?(1, 4)

Dynamic Communicating Automaton



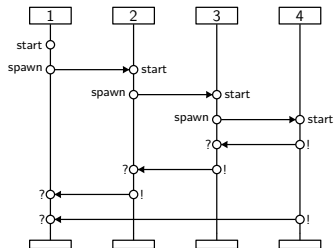
Proc	State	x_1	x_2	x_3	1	2	3	4
9	s_4	4	9	9	—			
7	s_3	4	9	9		—		
2	s_3	4	7	9			—	
4	s_6	2	2	1				—

spawn(9, 7) spawn(7, 2) spawn(2, 4) !(4, 2)!(4, 9)?(4, 2)!(2, 7)?(2, 7)!(7, 9)?(7, 9)?(9, 4)

Message Sequence Charts (MSCs)

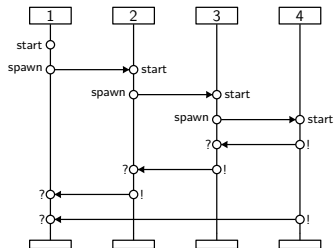
spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1)?(4, 3) !(3, 2)?(3, 2) !(2, 1)?(2, 1)?(1, 4)

Message Sequence Charts (MSCs)



$\text{spawn}(1, 2) \text{ spawn}(2, 3) \text{ spawn}(3, 4) \text{ !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1) ?(2, 1) ?(1, 4)}$

Message Sequence Charts (MSCs)



spawn(1, 2) spawn(2, 3) spawn(3, 4) !(4, 3) !(4, 1) ?(4, 3) !(3, 2) ?(3, 2) !(2, 1) ?(2, 1) ?(1, 4)

$$L_{\text{word}}(\mathcal{A}) \subseteq \Sigma^*$$

$$L(\mathcal{A}) \subseteq \text{MSCs}$$

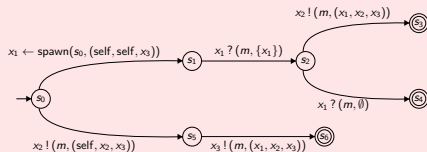
Realizability of Dynamic MSC Languages

Fork-and-Join Grammar [LMM'02]

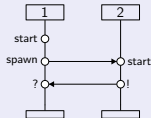
realizable?



Dynamic Communicating Automaton



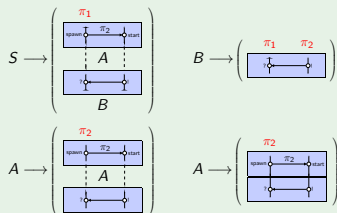
MSC



[LMM'02] Leucker & Madhusudan & Mukhopadhyay. [Dynamic Message Sequence Charts](#). 2002.

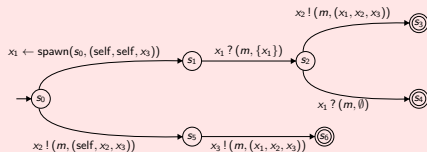
Realizability of Dynamic MSC Languages

Dynamic MSC Grammar

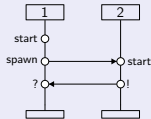


realizable?
 \Rightarrow

Dynamic Communicating Automaton



MSC



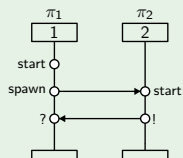
[LMM'02] Leucker & Madhusudan & Mukhopadhyay. [Dynamic Message Sequence Charts](#). 2002.

Presentation outline

- 1 Dynamic Communicating Automata
- 2 Dynamic MSC Grammars**
- 3 Realizability
- 4 Finite Implementation

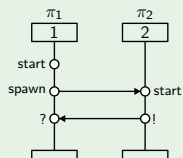
Building Blocks of Dynamic MSC Grammars

Named MSC $\{\pi_1, \pi_2\} \rightarrow \{1, 2\}$



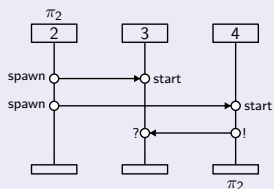
Building Blocks of Dynamic MSC Grammars

Named MSC $\{\pi_1, \pi_2\} \rightarrow \{1, 2\}$



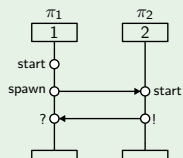
○

Partial MSC $\pi_2 \mapsto (2, 4)$



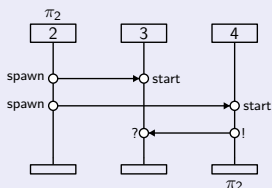
Building Blocks of Dynamic MSC Grammars

Named MSC $\{\pi_1, \pi_2\} \rightarrow \{1, 2\}$



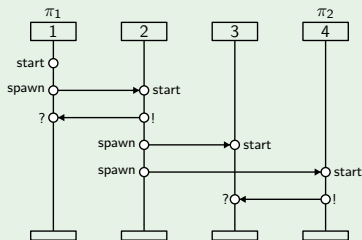
\circ

Partial MSC $\pi_2 \mapsto (2, 4)$

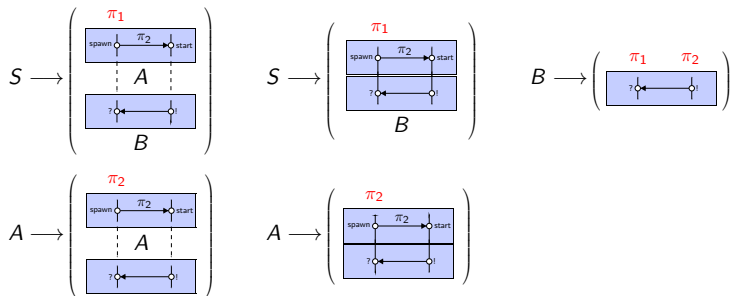


$=$

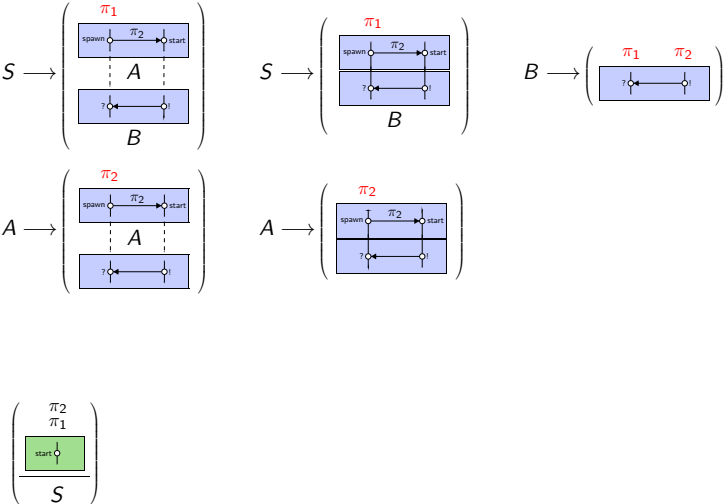
Product: Named MSC



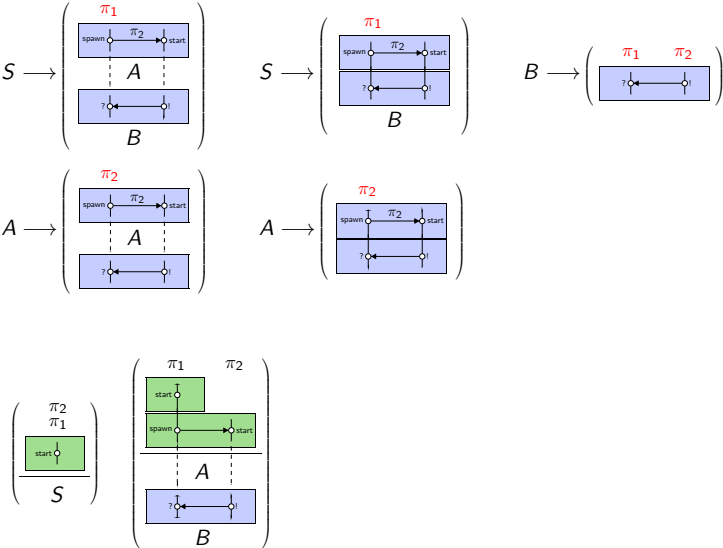
Dynamic MSC Grammar



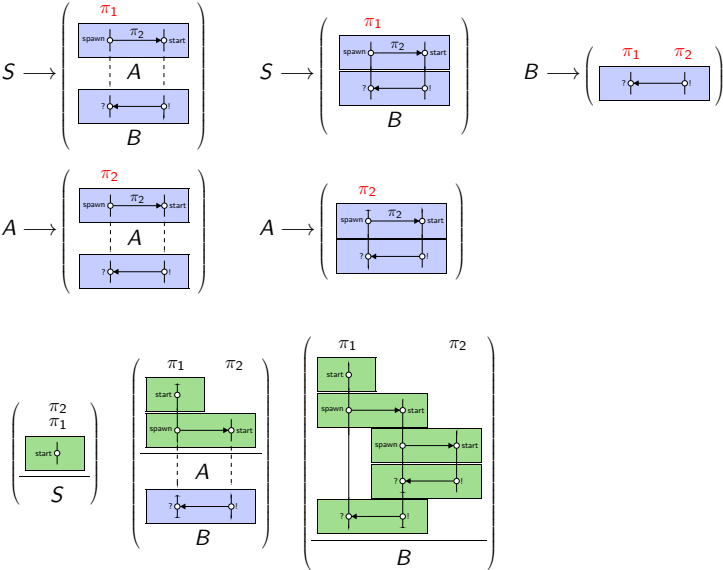
Dynamic MSC Grammar



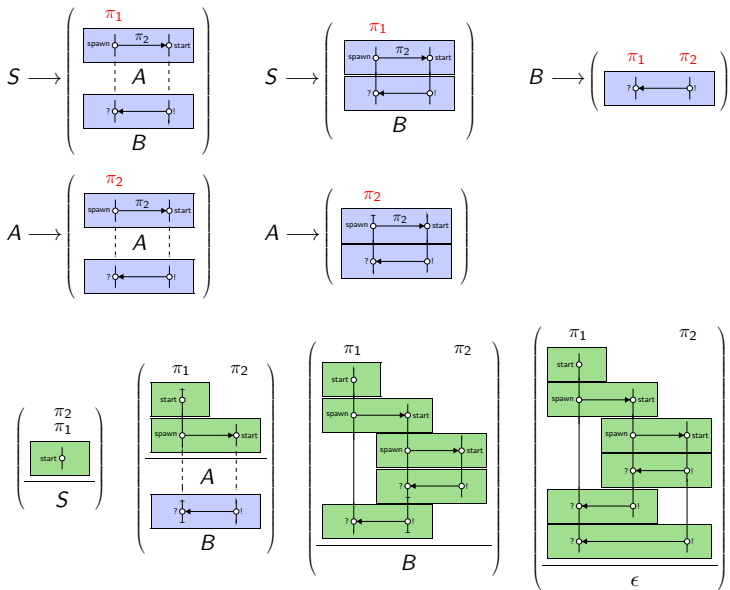
Dynamic MSC Grammar



Dynamic MSC Grammar



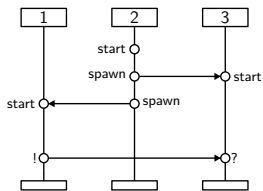
Dynamic MSC Grammar



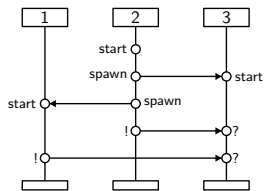
Presentation outline

- 1 Dynamic Communicating Automata
- 2 Dynamic MSC Grammars
- 3 Realizability**
- 4 Finite Implementation

Realizability

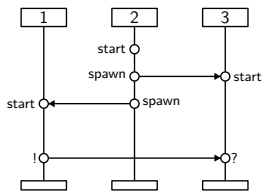


non-realizable

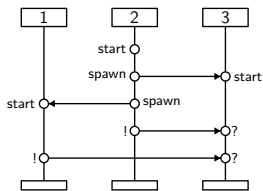


2-realizable

Realizability



non-realizable



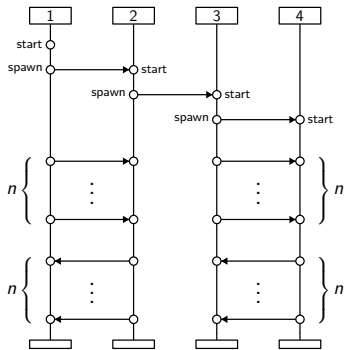
2-realizable

Definition

Let L be a set of MSCs and $b \in \mathbb{N} \cup \{\infty\}$.

- L is **realizable** if there is a DCA \mathcal{A} such that $L = L(\mathcal{A})$.
- L is **b -realizable** if \mathcal{A} uses no more than b process variables.

A realizable MSC language



Realizability

Theorem

The following problem is EXPTIME-complete:

INPUT: Dynamic MSC grammar G .

QUESTION: Is $L(G)$ realizable?

Proof

- Build tree automaton \mathcal{A}_G accepting the valid parse trees of G that are not realizable.
- $L(G)$ realizable $\iff L(\mathcal{A}_G) = \emptyset$
- $L(G)$ realizable $\implies L(G)$ is $(|Proc(G)| + \ell \cdot |\Pi|)$ -realizable
- Hardness: reduction from intersection problem for tree automata

Realizability

Theorem

The following problem is EXPTIME-complete:

INPUT: Dynamic MSC grammar G .

QUESTION: Is $L(G)$ realizable?

Proof

- Build tree automaton \mathcal{A}_G accepting the valid parse trees of G that are not realizable.
- $L(G)$ realizable $\iff L(\mathcal{A}_G) = \emptyset$
- $L(G)$ realizable $\implies L(G)$ is $(|Proc(G)| + \ell \cdot |\Pi|)$ -realizable
- Hardness: reduction from intersection problem for tree automata

Realizability

Theorem

The following problem is EXPTIME-complete:

INPUT: Dynamic MSC grammar G .

QUESTION: Is $L(G)$ realizable?

Proof

- Build tree automaton \mathcal{A}_G accepting the valid parse trees of G that are not realizable.
- $L(G)$ realizable $\iff L(\mathcal{A}_G) = \emptyset$
- $L(G)$ realizable $\implies L(G)$ is $(|Proc(G)| + \ell \cdot |\Pi|)$ -realizable
- Hardness: reduction from intersection problem for tree automata

Realizability

Theorem

The following problem is EXPTIME-complete:

INPUT: Dynamic MSC grammar G .

QUESTION: Is $L(G)$ realizable?

Proof

- Build tree automaton \mathcal{A}_G accepting the valid parse trees of G that are not realizable.
- $L(G)$ realizable $\iff L(\mathcal{A}_G) = \emptyset$
- $L(G)$ realizable $\implies L(G)$ is $(|Proc(G)| + \ell \cdot |\Pi|)$ -realizable
- Hardness: reduction from intersection problem for tree automata

Realizability

Theorem

The following problem is EXPTIME-complete:

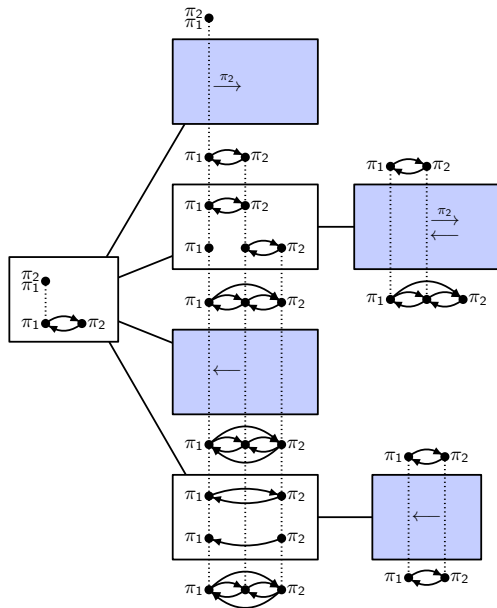
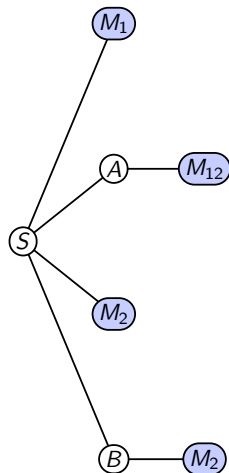
INPUT: Dynamic MSC grammar G .

QUESTION: Is $L(G)$ realizable?

Proof

- Build tree automaton \mathcal{A}_G accepting the valid parse trees of G that are not realizable.
- $L(G)$ realizable $\iff L(\mathcal{A}_G) = \emptyset$
- $L(G)$ realizable $\implies L(G)$ is $(|Proc(G)| + \ell \cdot |\Pi|)$ -realizable
- Hardness: reduction from intersection problem for tree automata

The tree automaton

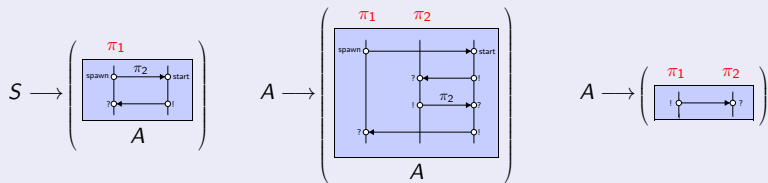


Presentation outline

- 1 Dynamic Communicating Automata
- 2 Dynamic MSC Grammars
- 3 Realizability
- 4 Finite Implementation

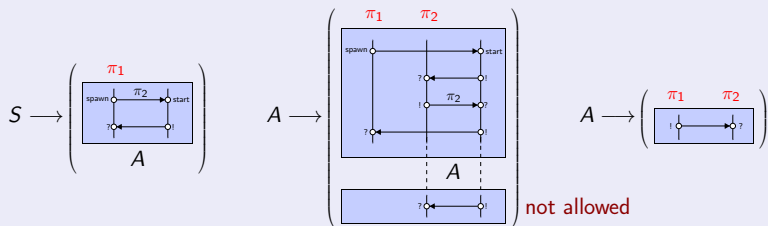
Local Dynamic MSC Grammars

Definition (Local Grammar [GMSZ'06])



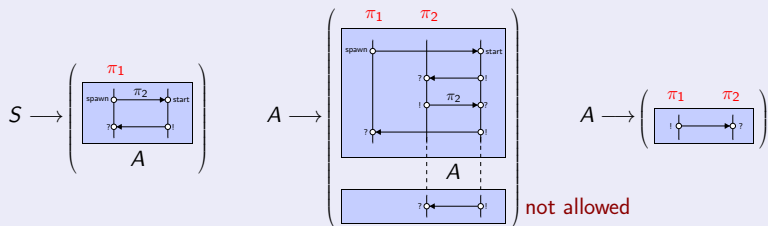
Local Dynamic MSC Grammars

Definition (Local Grammar [GMSZ'06])



Local Dynamic MSC Grammars

Definition (Local Grammar [GMSZ'06])



Theorem

- *Emptiness and realizability for local grammars are PSPACE-complete.*
- *For every **realizable** local grammar G , there is a finite DCA \mathcal{A} such that $L(\mathcal{A}) = L(G)$.*

Future work

- More classes of implementable dynamic MSC grammars
- Extended grammars (regular expressions on right-hand sides)
- Dynamic regular MSC languages [HMNST'05]
- DCA and logic [BL'05, GKM'06, HMNST'05]
- Connection with π -calculus, series-parallel languages [LW'00], ...

[BL'06] B. & Leucker. [Message-Passing Automata are expressively equivalent to EMSO Logic](#). 2006.

[GKM'06] Genest & Kuske & Muscholl. [A Kleene theorem and model checking algorithms for existentially bounded communicating automata](#). 2006.

[HMNST'05] Henriksen & Mukund & Narayan Kumar & Sohoni & Thiagarajan. [A Theory of Regular MSC Languages](#). 2005.

[LW'00] Lodaya & Weil. [Series-parallel languages and the bounded-width property](#). 2000.