High-Level Message Sequence Charts

Marie Fortin

IRIF, CNRS, Université Paris Cité

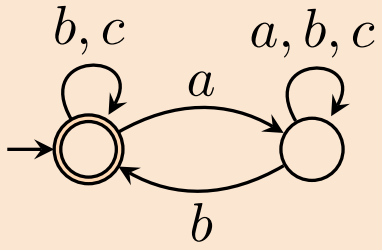
based on joint work with **Benedikt Bollig** and **Paul Gastin**

PaVeDyS meeting, May 13, 2025

Regular Languages of Words

[Kleene 1956]

Finite automata



Regular Expressions

$$((a + b + c)^* b + c)^*$$

[Büchi 1960,
Elgot 1961
Trakhtenbrot 1962]

$$L = \{aab, bcc, acabc, \cdots\}$$

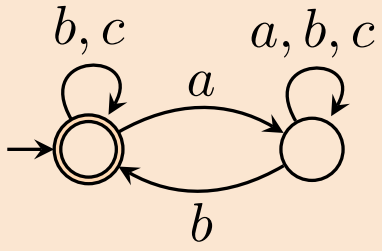
Monadic Second-Order Logic

$$\forall x.\, a(x) \implies \exists y.\, x < y \wedge b(y)$$

Regular Languages of Words

[Kleene 1956]

Finite automata



Regular Expressions

$((a + b + c)^* b + c)^*$

[Büchi 1960,
Elgot 1961
Trakhtenbrot 1962]

$L = \{aab, bcc, acabc, \cdots\}$

Monadic Second-Order Logic

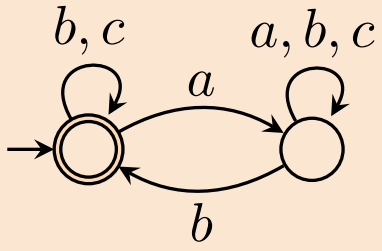
$\forall x.\, a(x) \implies \exists y.\, x < y \wedge b(y)$

What about **concurrent** systems?

Regular Languages of Words

[Kleene 1956]

Finite automata



Regular Expressions

$((a + b + c)^* b + c)^*$

[Büchi 1960,
Elgot 1961
Trakhtenbrot 1962]

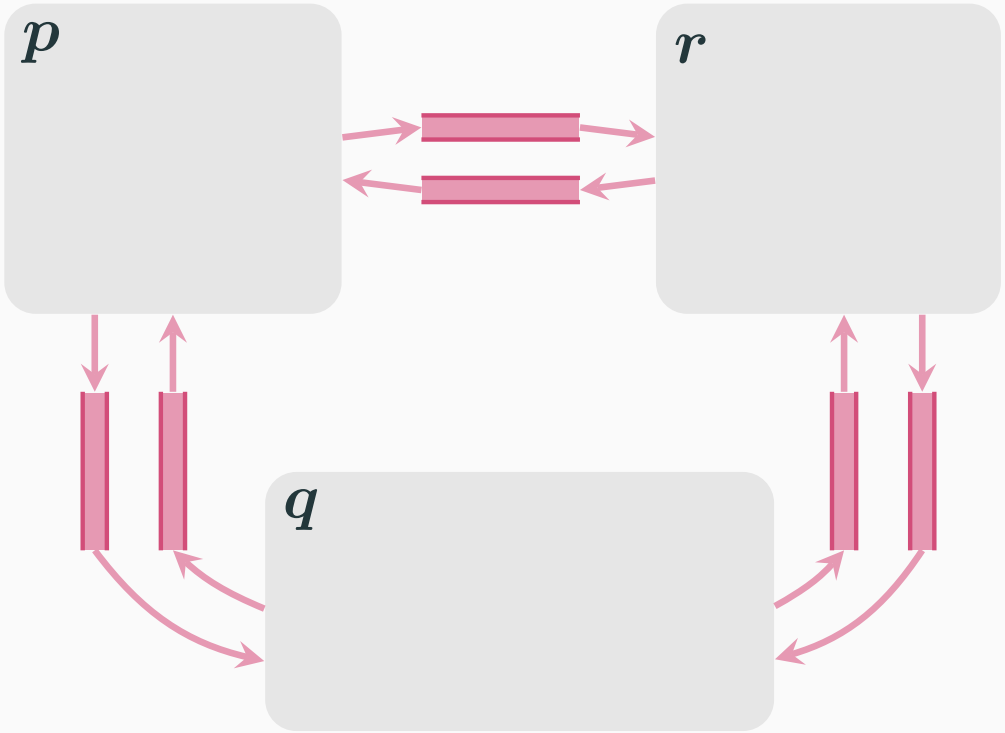
$L = \{aab, bcc, acabc, \cdots\}$

Monadic Second-Order Logic

$\forall x.\, a(x) \implies \exists y.\, x < y \wedge b(y)$

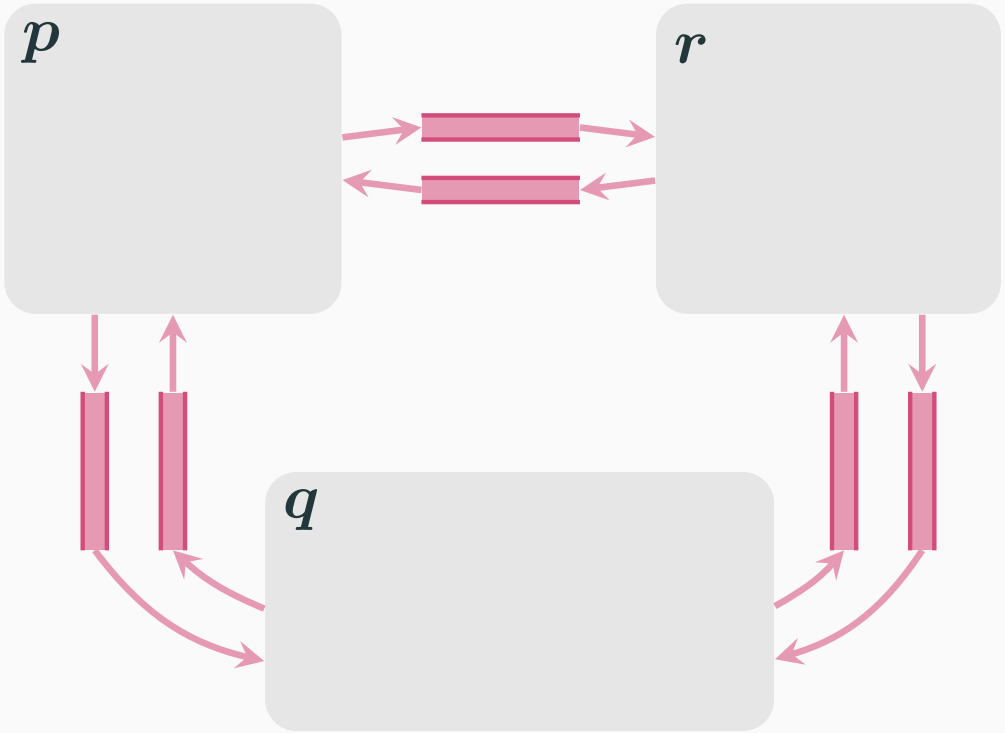
What about **communicating automata**?

Communicating finite-state machines (CFMs)[1]



[1] [Brand, Zafiropulo 1983]

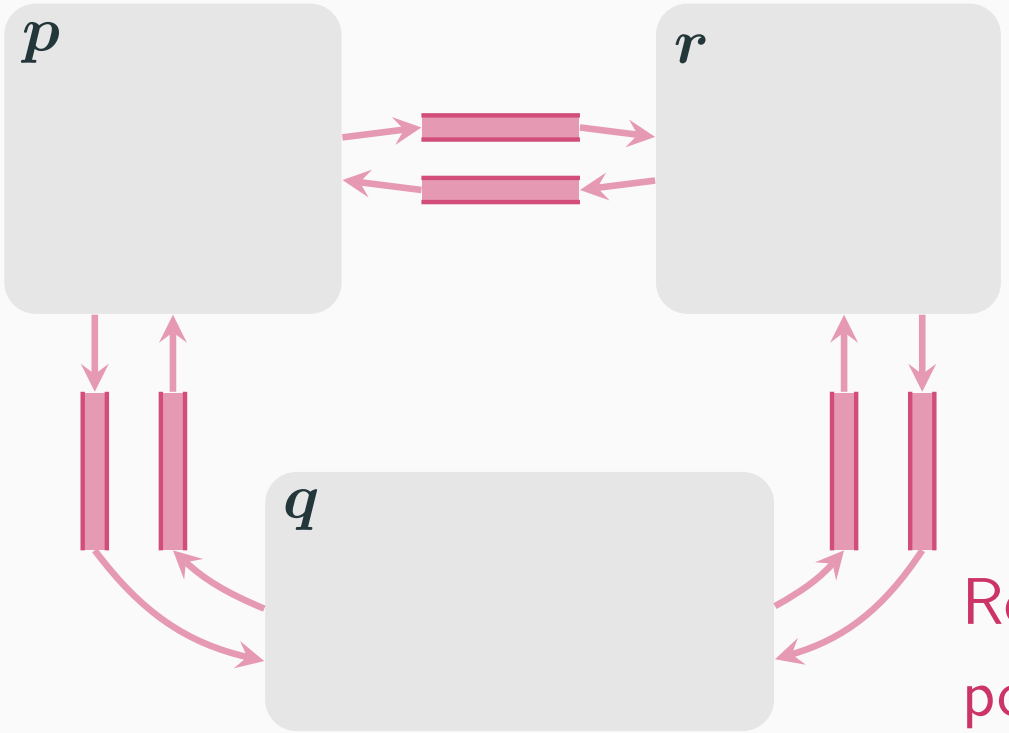
Communicating finite-state machines (CFMs)[1]



Fixed, finite set of processes, e.g. $\{p, q, r\}$

[1] [Brand, Zafiropulo 1983]

Communicating finite-state machines (CFMs)[1]

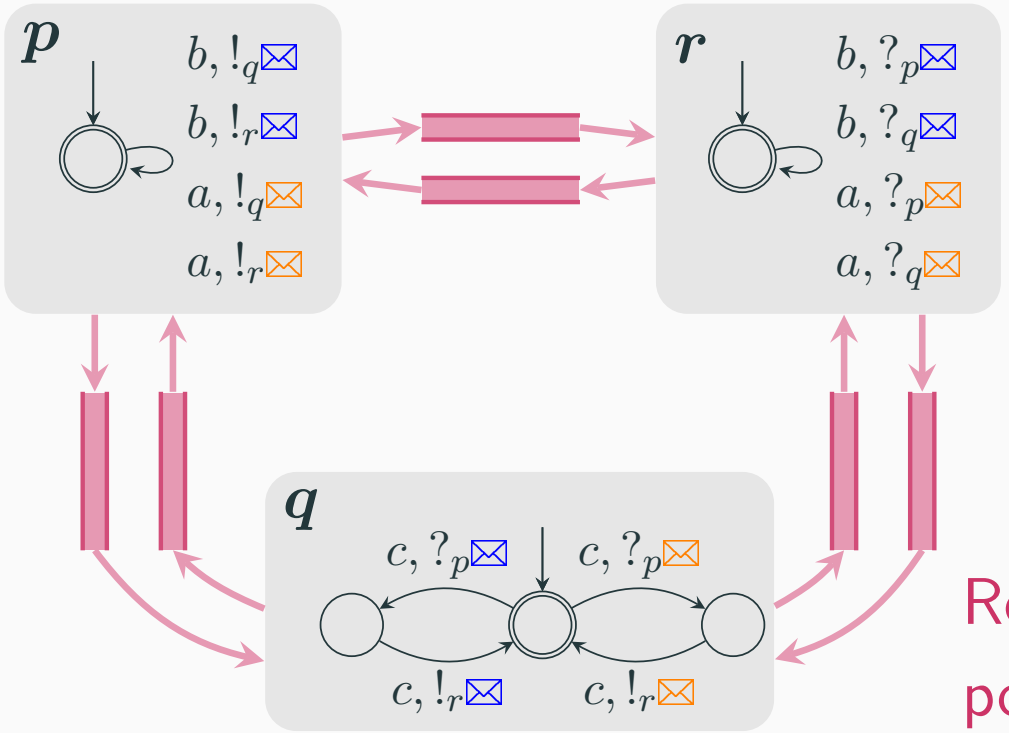


Reliable unbounded
point-to-point FIFO channels

Fixed, finite set of processes, e.g. $\{p, q, r\}$

[1] [Brand, Zafiropulo 1983]

Communicating finite-state machines (CFMs)[1]



One finite automaton for each process

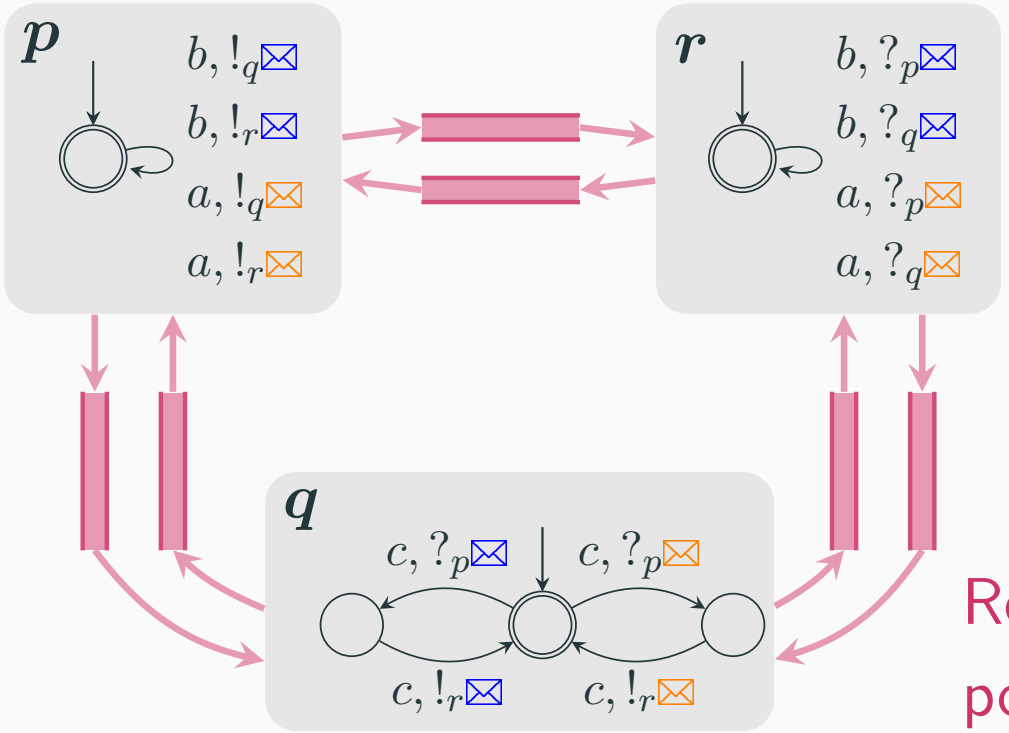
- Finite input alphabet, e.g. $\Sigma = \{a, b, c\}$
- Sends/receives from a finite message alphabet, e.g. $\{$✉, ✉$\}$

Reliable unbounded point-to-point FIFO channels

Fixed, finite set of processes, e.g. $\{p, q, r\}$

[1] [Brand, Zafiropulo 1983]

Communicating finite-state machines (CFMs)[1]



One finite automaton for each process

- Finite input alphabet, e.g. $\Sigma = \{a, b, c\}$
- Sends/receives from a finite message alphabet, e.g. {✉, ✉}

Reliable unbounded point-to-point FIFO channels

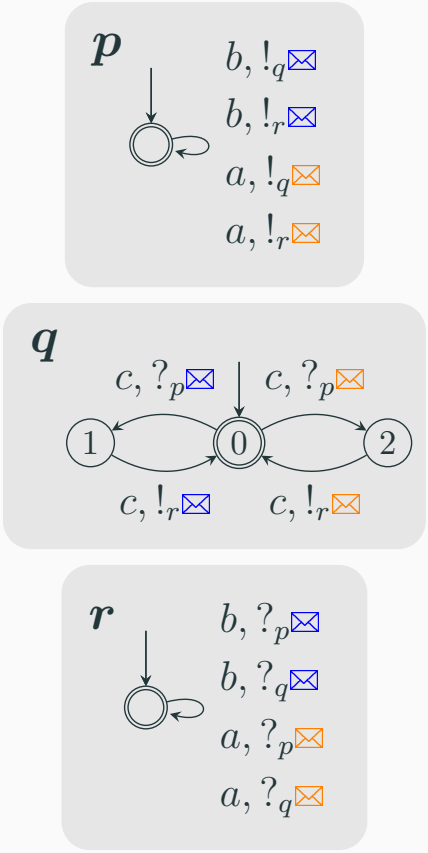
Fixed, finite set of processes, e.g. $\{p, q, r\}$

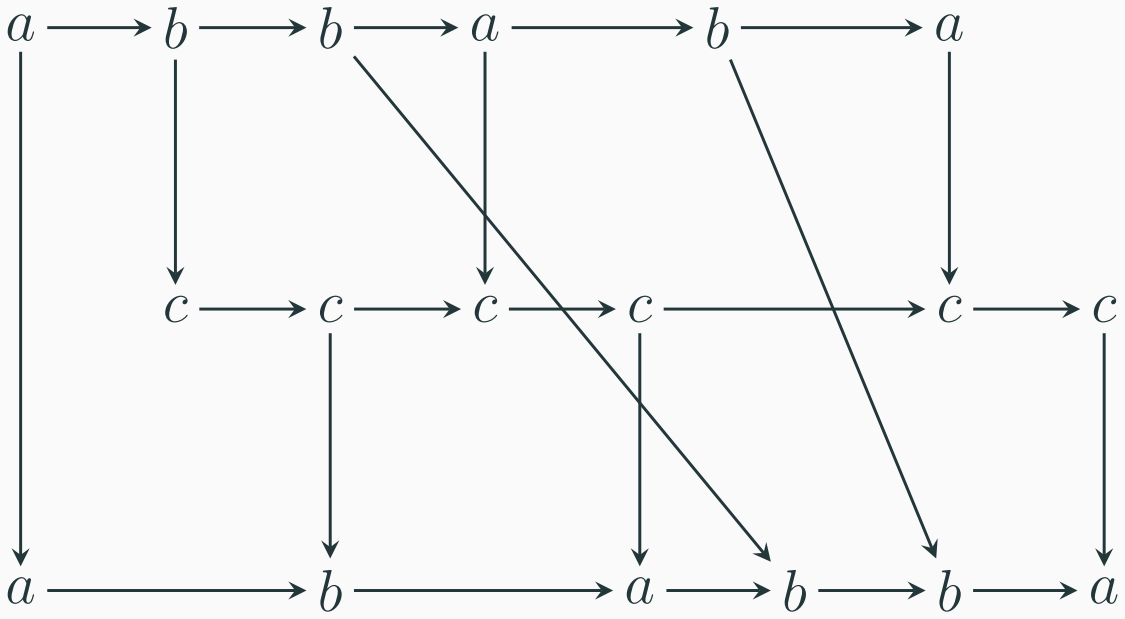
Global acceptance condition

[1] [Brand, Zafiropulo 1983]

Message Sequence Charts (MSC)

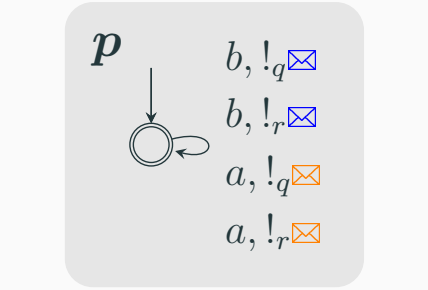
The language of a CFM is a set of Message Sequence Charts.

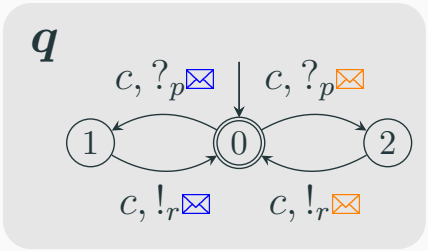


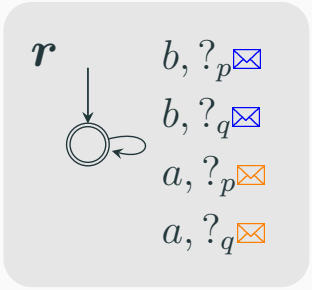


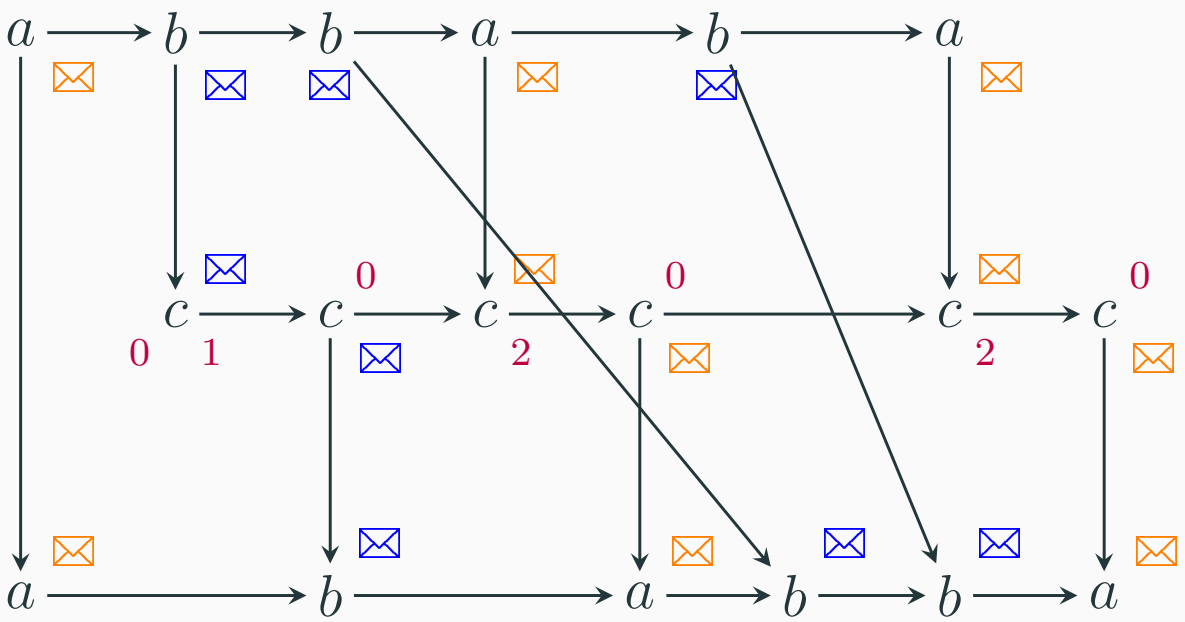
Message Sequence Charts (MSC)

The language of a CFM is a set of Message Sequence Charts.









Undecidability

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting?

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

- (un)decidability vs. expressiveness

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

- (un)decidability vs. expressiveness
- positive results for implementability

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable: emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

- (un)decidability vs. expressiveness
- positive results for implementability
- decidable restrictions/sub-approximations: e.g., bounded channels

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

$\rightarrow$ Most decision problems for CFMs are undecidable: emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

- (un)decidability vs. expressiveness
- positive results for implementability
- decidable restrictions/sub-approximations: e.g., bounded channels

Undecidability

CFM with ≥ 2 processes can easily simulate Turing machines

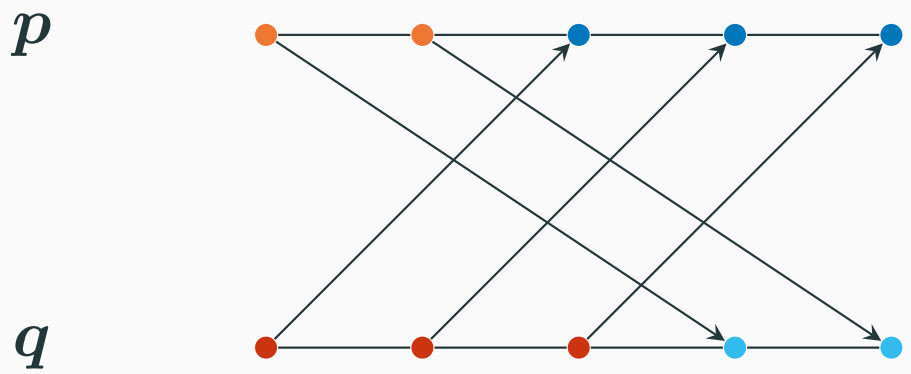
$\rightarrow$ Most decision problems for CFMs are undecidable:
emptiness, reachability, model-checking...

Does it mean CFMs are uninteresting? **No!**

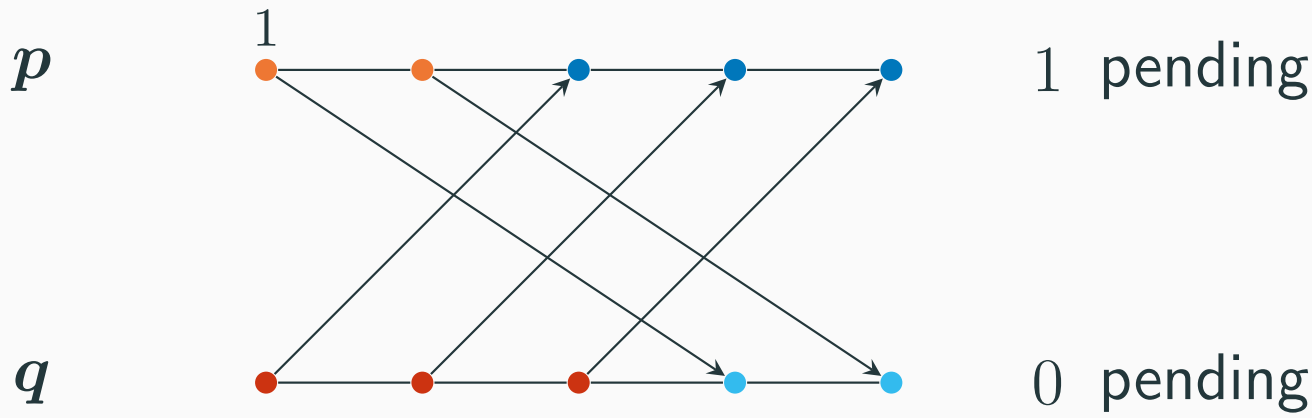
- (un)decidability vs. expressiveness
- positive results for implementability
- decidable restrictions/sub-approximations: e.g., bounded channels

In this talk: mainly unbounded, sometimes bounded

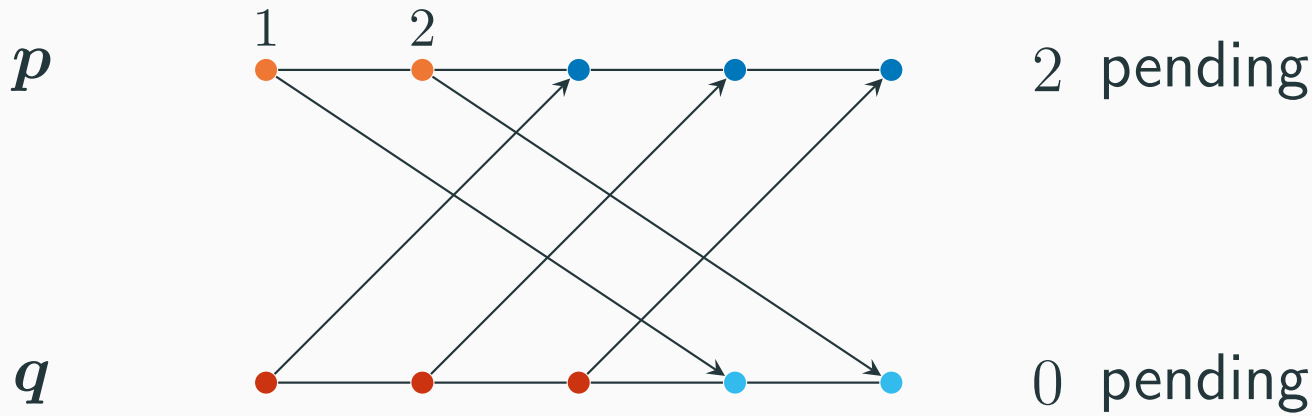
Linearizations and bounded MSCs



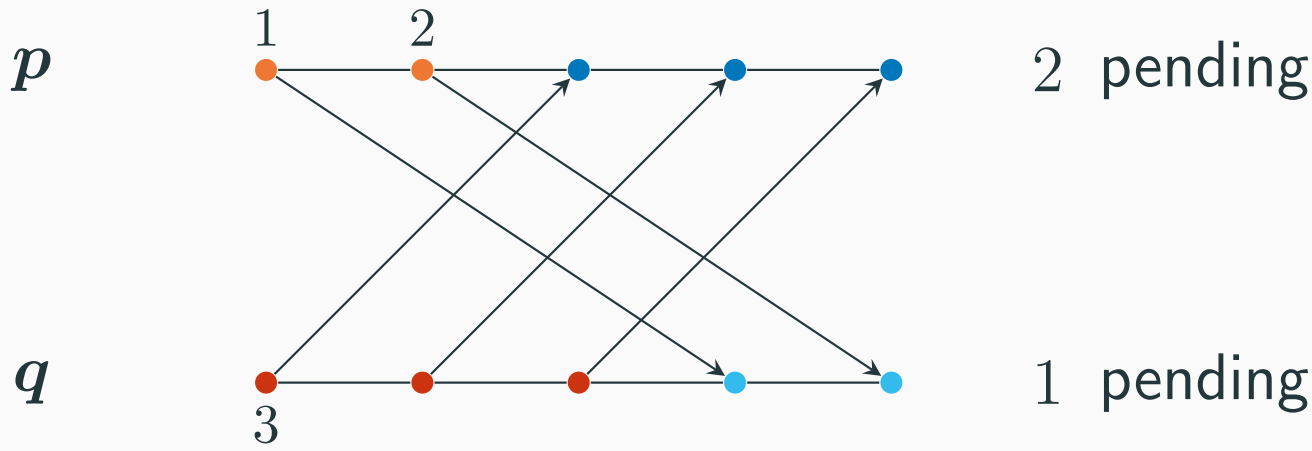
Linearizations and bounded MSCs



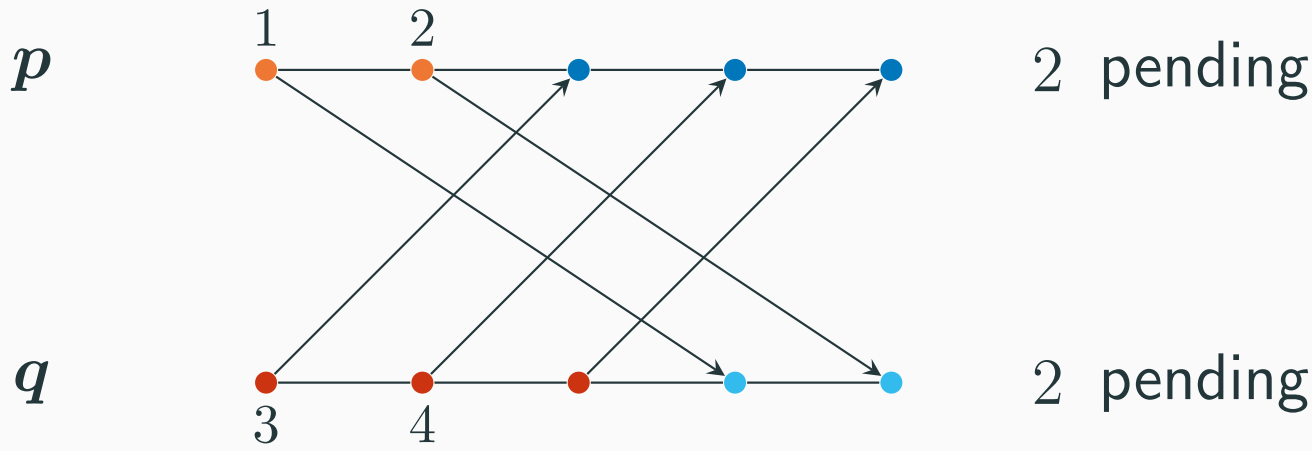
Linearizations and bounded MSCs



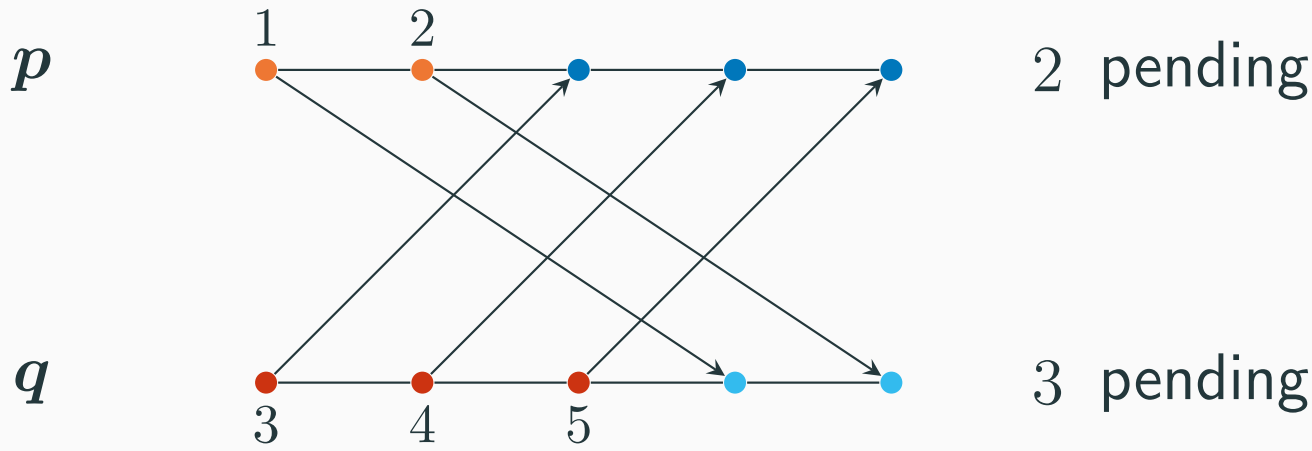
Linearizations and bounded MSCs



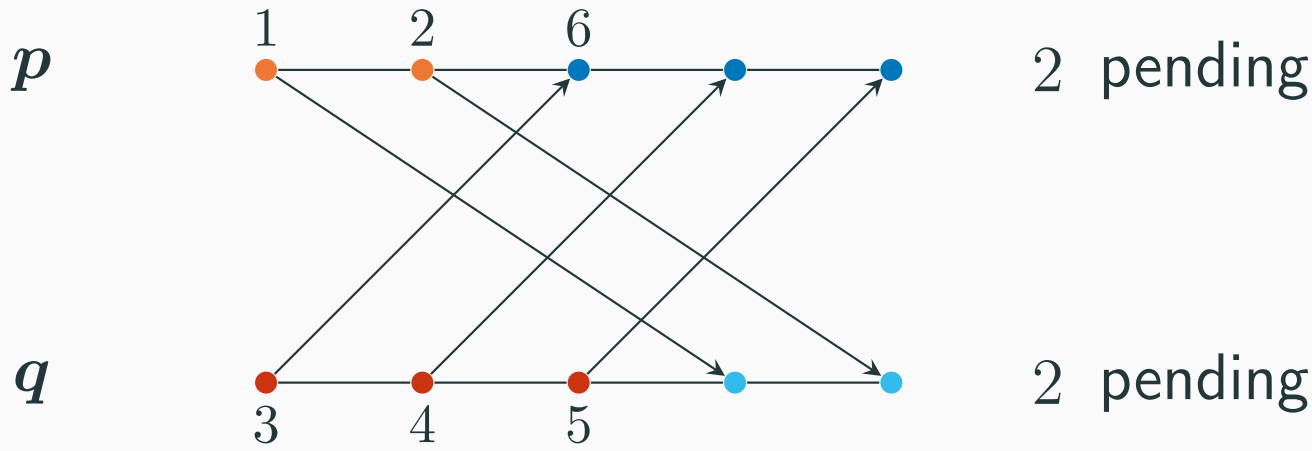
Linearizations and bounded MSCs



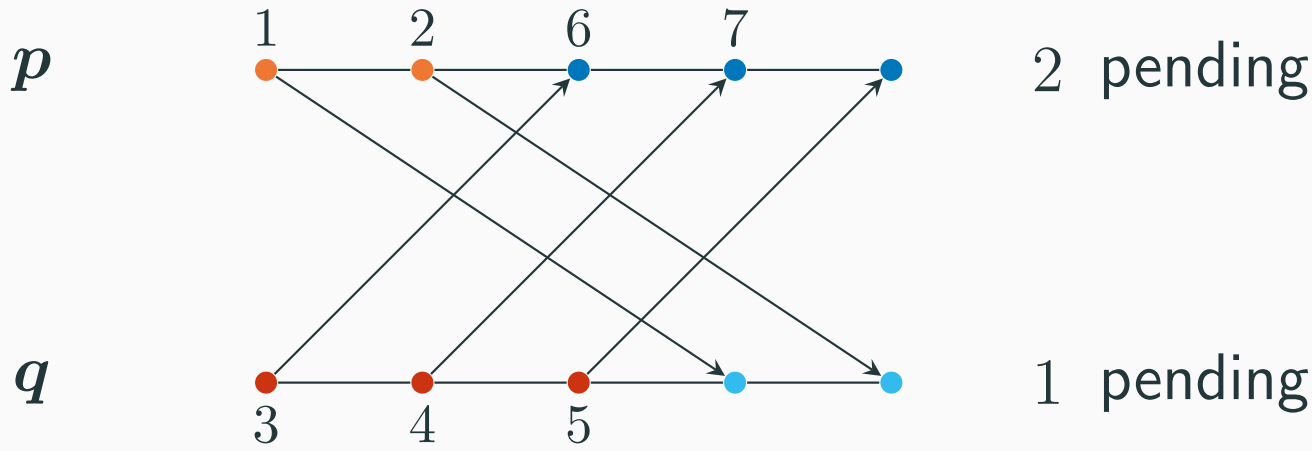
Linearizations and bounded MSCs



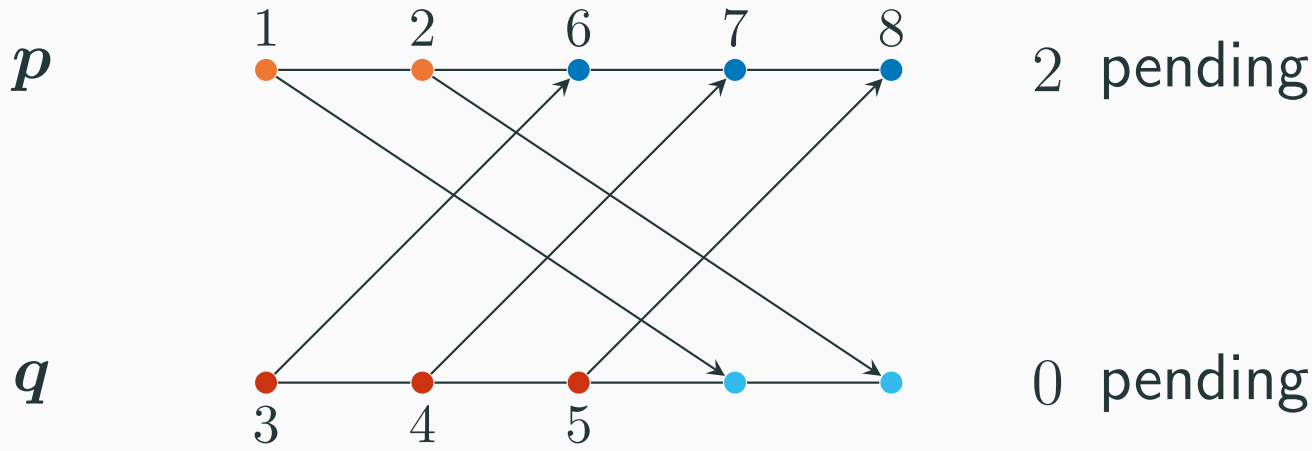
Linearizations and bounded MSCs



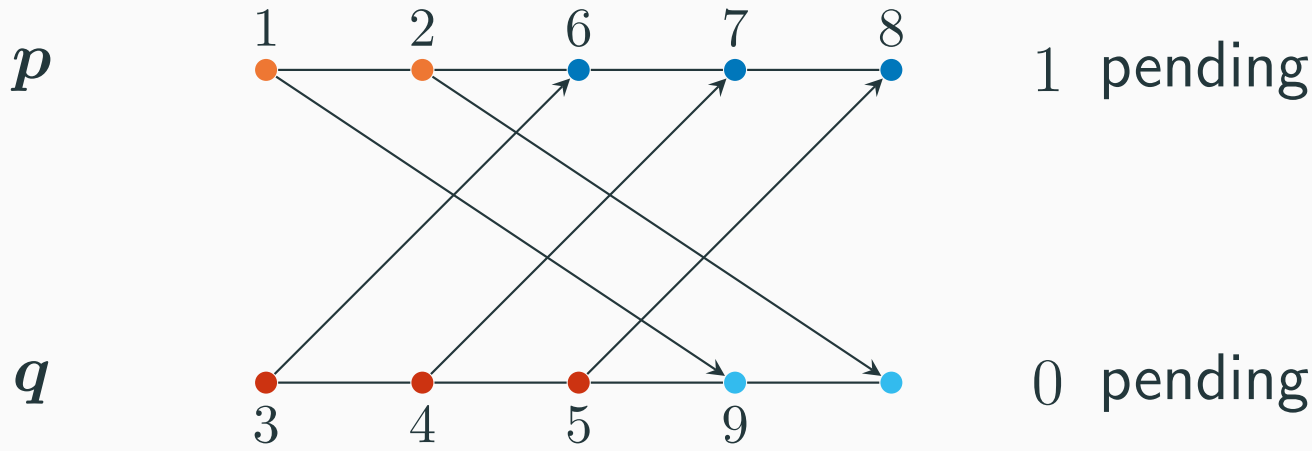
Linearizations and bounded MSCs



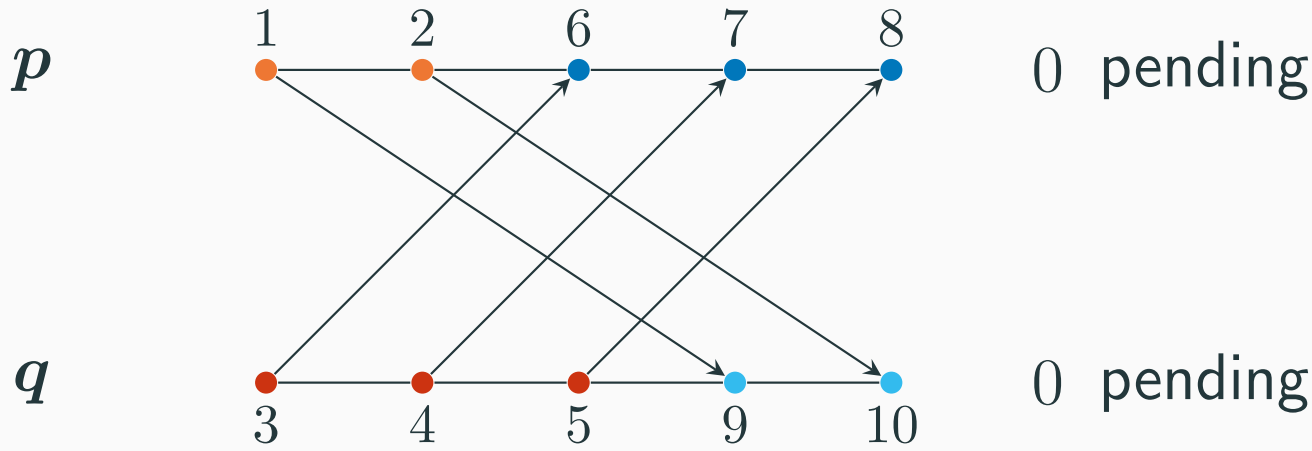
Linearizations and bounded MSCs



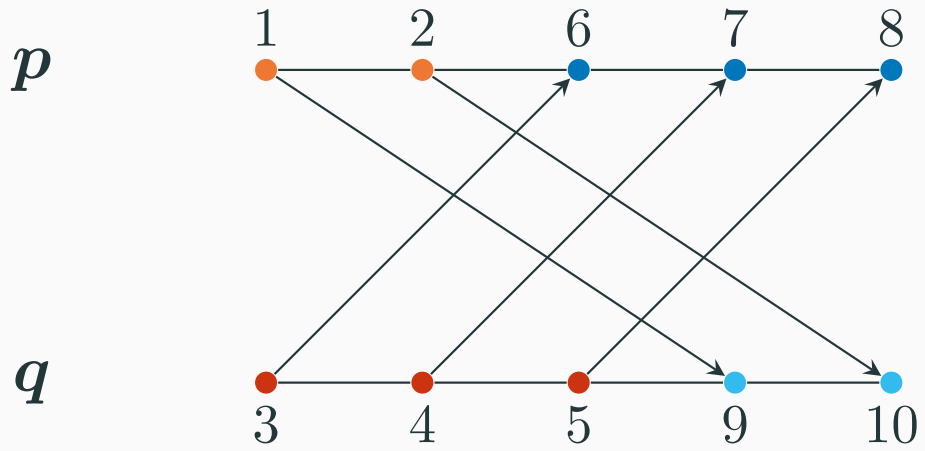
Linearizations and bounded MSCs



Linearizations and bounded MSCs



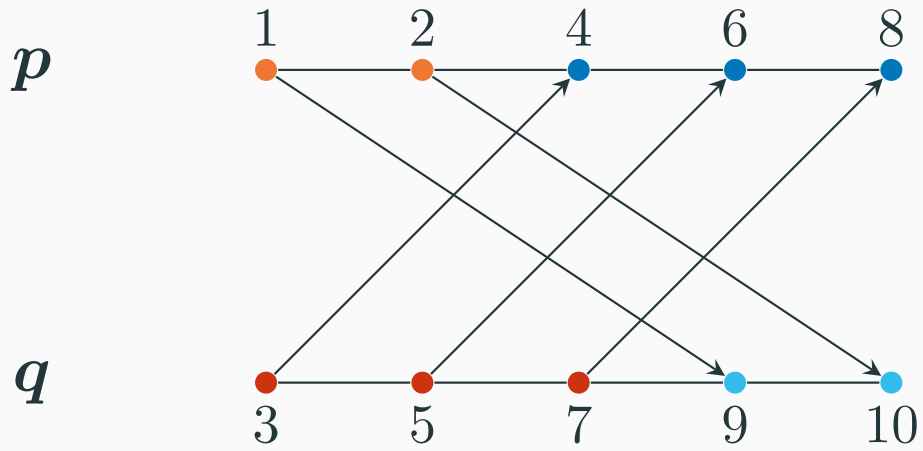
Linearizations and bounded MSCs



$(p!q)$ $(p!q)$ $(q!p)$ $(q!p)$ $(q!p)$ $(q?p)$ $(q?p)$ $(p?q)$ $(p?q)$ $(p?q)$ **3-bounded**

B-bounded = at most B pending messages in each channel

Linearizations and bounded MSCs



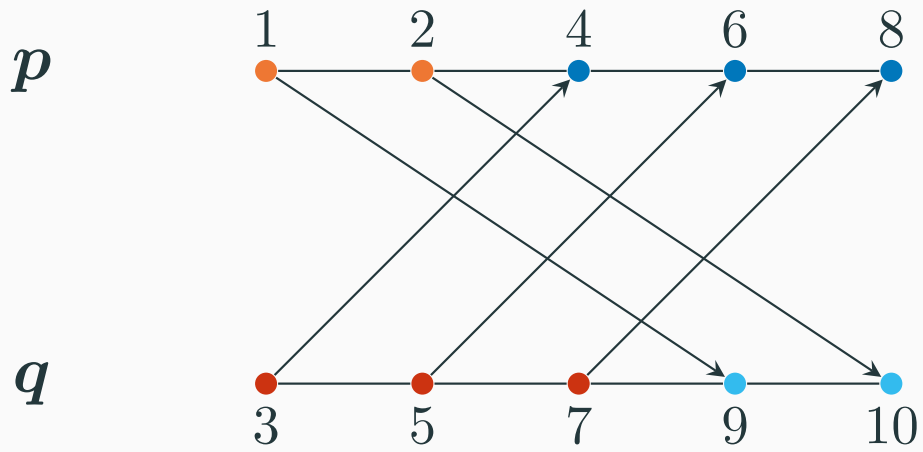
$(p!q)\ (p!q)\ (q!p)\ (q!p)\ (q!p)\ (q?p)\ (q?p)\ (p?q)\ (p?q)\ (p?q)$ **3-bounded**

$(p!q)\ (p!q)\ (q!p)\ (p?q)\ (q!p)\ (p?q)\ (q!p)\ (p?q)\ (q?p)\ (q?p)$ **2-bounded**

$\ldots$

B-bounded $=$ at most B pending messages in each channel

Linearizations and bounded MSCs



$(p!q)\ (p!q)\ (q!p)\ (q!p)\ (q!p)\ (q?p)\ (q?p)\ (p?q)\ (p?q)\ (p?q)$ **3-bounded**

$(p!q)\ (p!q)\ (q!p)\ (p?q)\ (q!p)\ (p?q)\ (q!p)\ (p?q)\ (q?p)\ (q?p)$ **2-bounded**

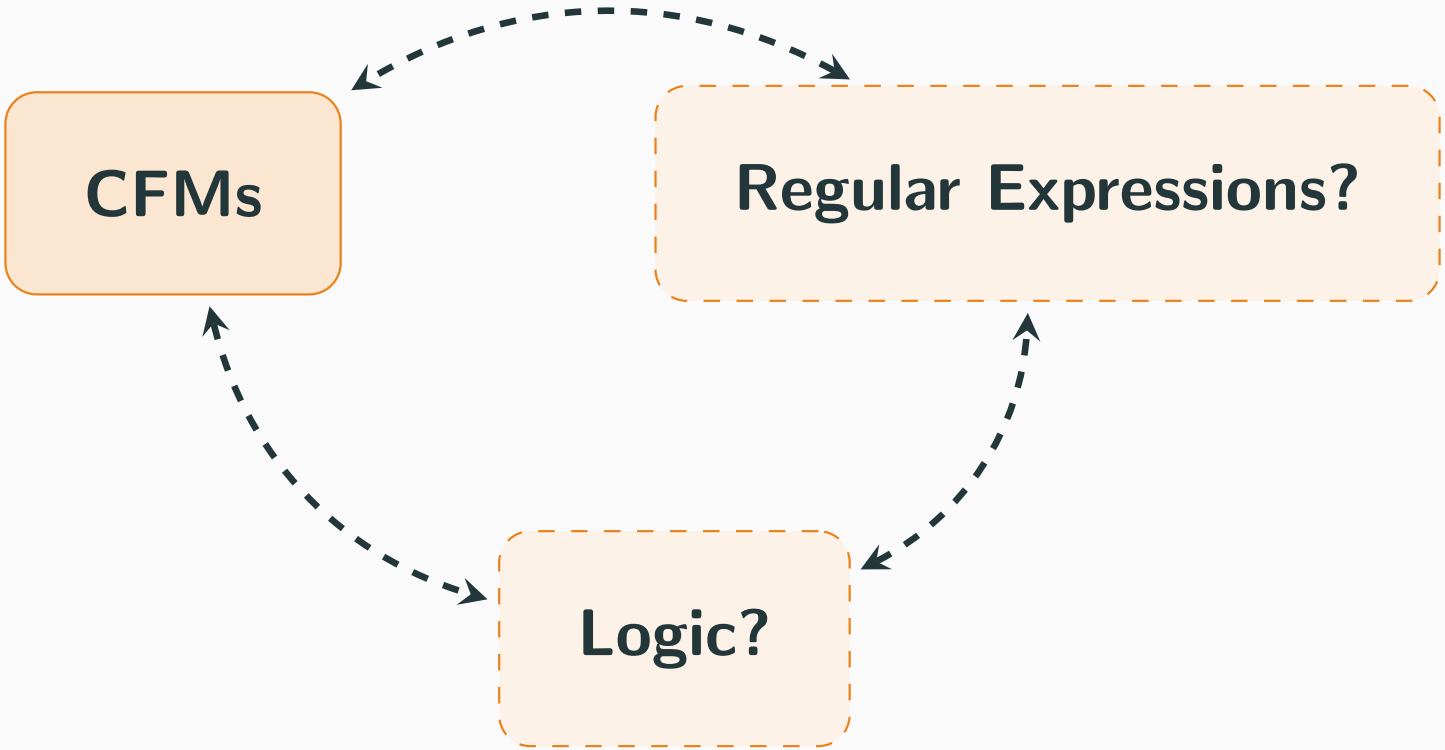
$\ldots$

M is $\exists B$-bounded if at least one linearization is B-bounded

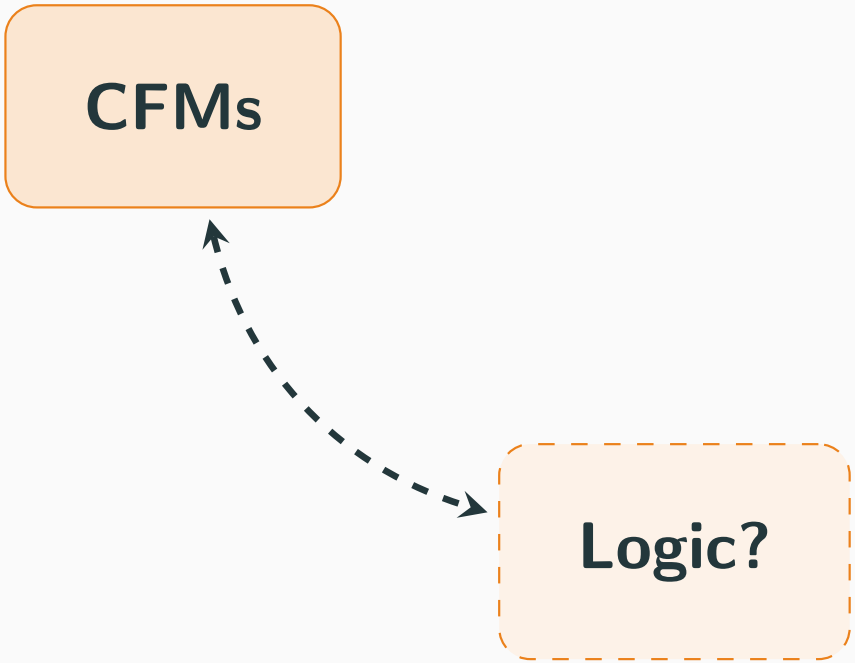
$\forall B$-bounded if all linearizations are B-bounded

What about Kleene and Büchi theorems?

What about Kleene and Büchi theorems?

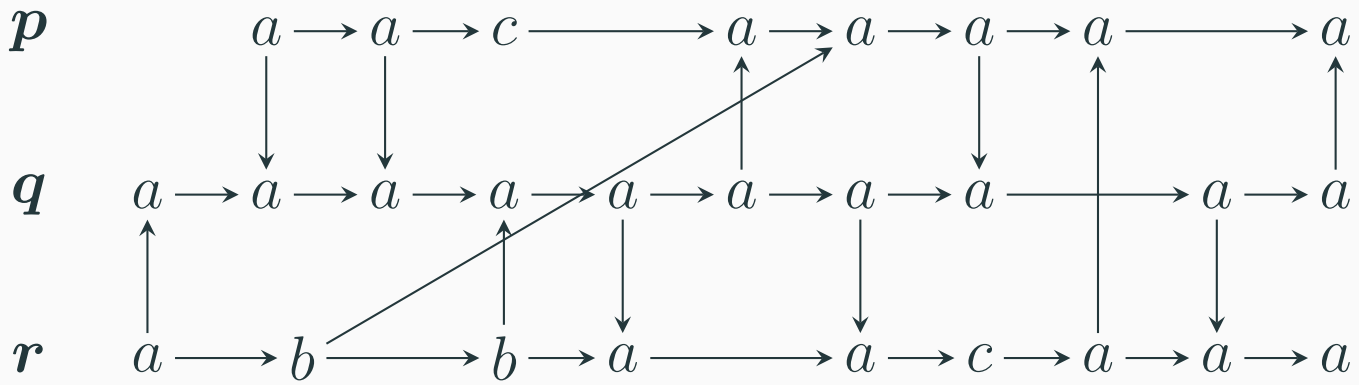


What about Kleene and Büchi theorems?



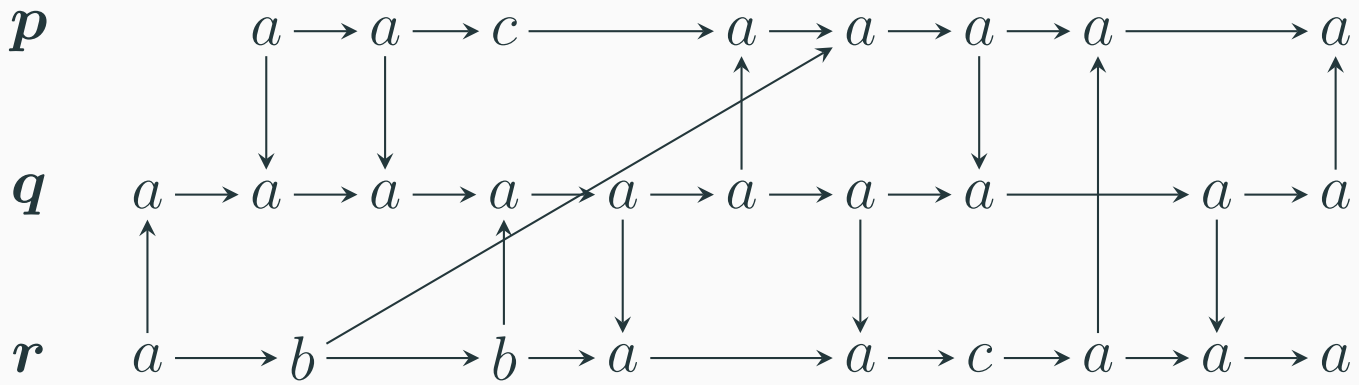
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::=$



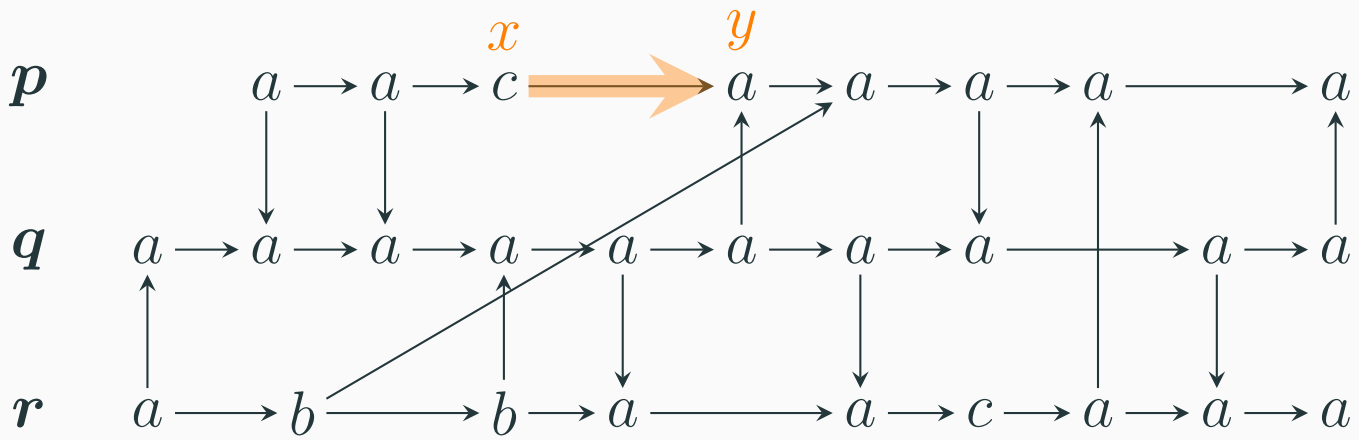
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::= \quad a(x) \mid p(x)$ label/process of event x



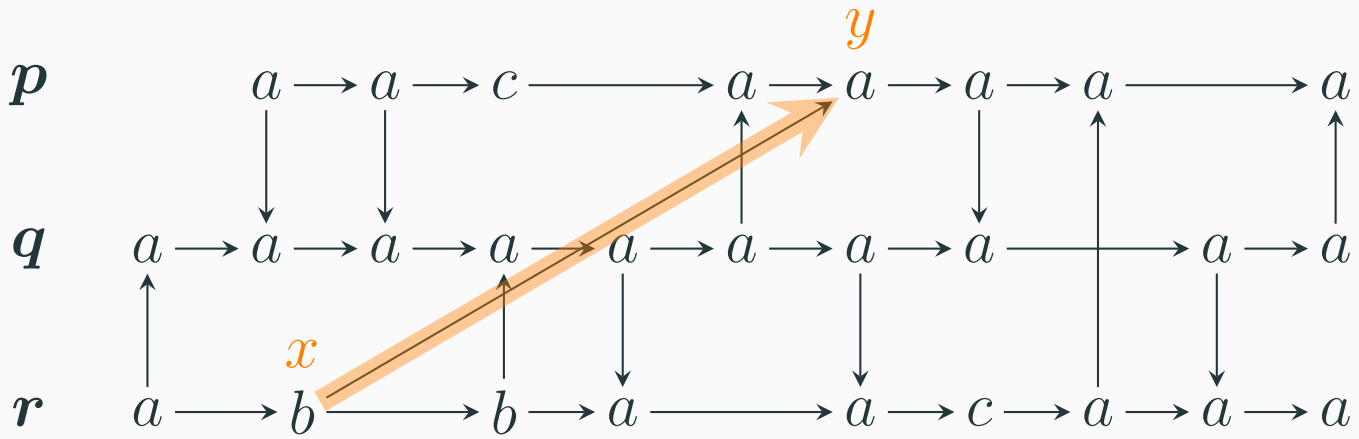
Monadic Second-Order logic (MSO) over MSCs

$$\varphi ::= \quad a(x) \mid p(x) \qquad \text{label/process of event } x$$
$$\mid x \to y \qquad \text{process successor}$$



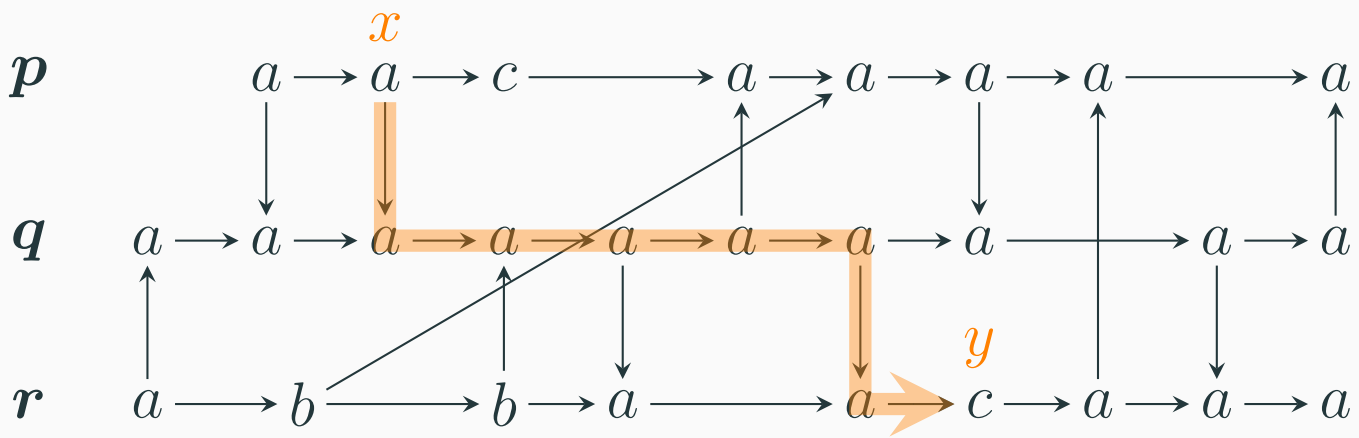
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::=$	$a(x) \mid p(x)$	label/process of event x
	$\mid x \to y$	process successor
	$\mid x \lhd y$	message relation



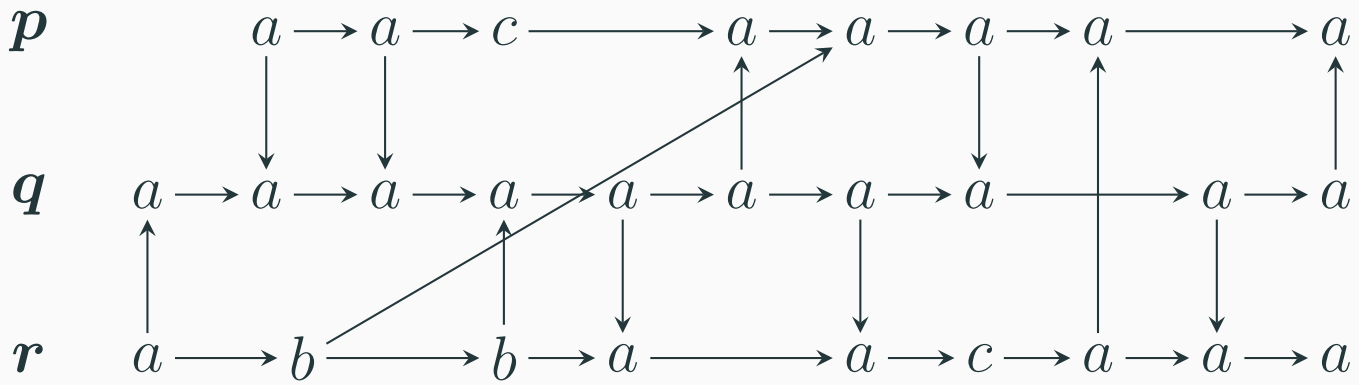
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::=$	$a(x) \mid p(x)$	label/process of event x
	$\mid x \to y$	process successor
	$\mid x \lhd y$	message relation
	$\mid x \leq y$	happened-before



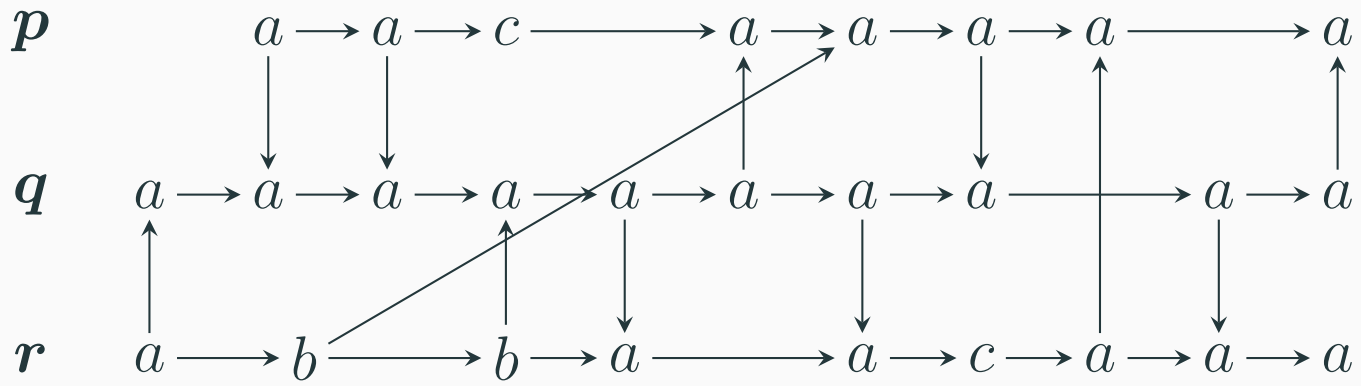
Monadic Second-Order logic (MSO) over MSCs

$$
\begin{array}{lll}
\varphi ::= & a(x) \mid p(x) & \text{label/process of event } x \\
\mid & x \to y & \text{process successor} \\
\mid & x \lhd y & \text{message relation} \\
\mid & x \leq y & \text{happened-before} \\
\mid & \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi \mid \exists X.\ \varphi \mid x \in X &
\end{array}
$$



Monadic Second-Order logic (MSO) over MSCs

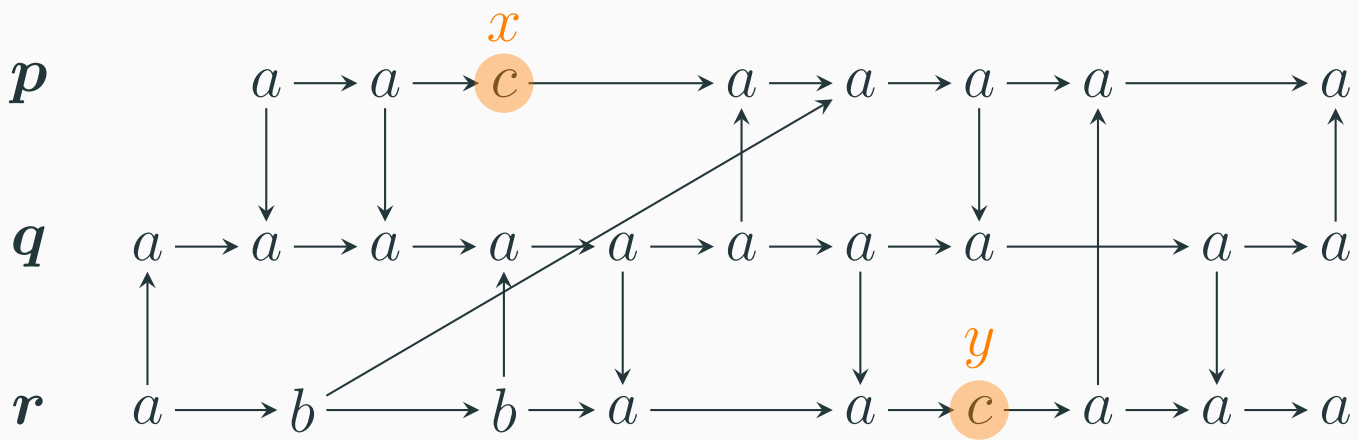
$$
\begin{array}{lll}
\varphi ::= & a(x) \mid p(x) & \text{label/process of event } x \\
& \mid x \to y & \text{process successor} \\
& \mid x \lhd y & \text{message relation} \\
& \mid x \leq y & \text{happened-before} \\
& \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi \mid \exists X.\ \varphi \mid x \in X &
\end{array}
$$



Example: mutual exclusion $\neg(\exists x.\ \exists y.\ c(x) \wedge c(y) \wedge\ x \parallel y\ )$

Monadic Second-Order logic (MSO) over MSCs

$$
\begin{array}{lll}
\varphi ::= & a(x) \mid p(x) & \text{label/process of event } x \\
\mid & x \to y & \text{process successor} \\
\mid & x \lhd y & \text{message relation} \\
\mid & x \leq y & \text{happened-before} \\
\mid & \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi \mid \exists X.\ \varphi \mid x \in X &
\end{array}
$$



Example: mutual exclusion $\neg(\exists x.\ \exists y.\ c(x) \wedge c(y) \wedge\ x \parallel y\ )$

$\neg(x \leq y) \wedge \neg(y \leq x)$ ⟵

Let's get back to words...

MSO $\rightarrow$ Automata

Inductive translation:

- $\ldots$
- Conjunction $\rightarrow$ Product
- Disjunction $\rightarrow$ Union
- Existential quantification $\rightarrow$ Projection
- Negation $\rightarrow$ Complement

Let's get back to words...

MSO $\rightarrow$ Automata

Inductive translation:

- $\ldots$
- Conjunction $\rightarrow$ Product ✓
- Disjunction $\rightarrow$ Union ✓
- Existential quantification $\rightarrow$ Projection ✓
- Negation $\rightarrow$ Complement

What about MSCs?

Let's get back to words...

MSO $\rightarrow$ Automata What about MSCs?

Inductive translation:

- $\ldots$
- Conjunction $\rightarrow$ Product ✓
- Disjunction $\rightarrow$ Union ✓
- Existential quantification $\rightarrow$ Projection ✓
- Negation $\rightarrow$ Complement ✗

Theorem (Bollig, Leucker 2006)
CFMs are **not** closed under complement.
Thus CFM $\neq$ MSO$[\leq, \lhd]$.

Logical characterization of CFMs

Theorem (Bollig, Leucker 2006)
Over finite MSCs, CFM $=$ EMSO$[\lhd, \rightarrow] \subsetneq$ MSO$[\lhd, \rightarrow]$.

Logical characterization of CFMs

Theorem (Bollig, Leucker 2006)
Over finite MSCs, CFM $=$ EMSO$[\lhd, \rightarrow] \subsetneq$ MSO$[\lhd, \rightarrow]$.

Theorem (Bollig, F., Gastin 2018 & 2021)
CFM $=$ EMSO$[\lhd, \leq]$

Logical characterization of CFMs

Theorem (Bollig, Leucker 2006)
Over finite MSCs, CFM $=$ EMSO$[\lhd, \rightarrow] \subsetneq$ MSO$[\lhd, \rightarrow]$.

Theorem (Bollig, F., Gastin 2018 & 2021)
CFM $=$ EMSO$[\lhd, \leq]$

Theorem (Genest, Kuske, Muscholl 2006)
Over **existentially bounded** MSCs, CFM $=$ MSO$[\lhd, \leq]$.

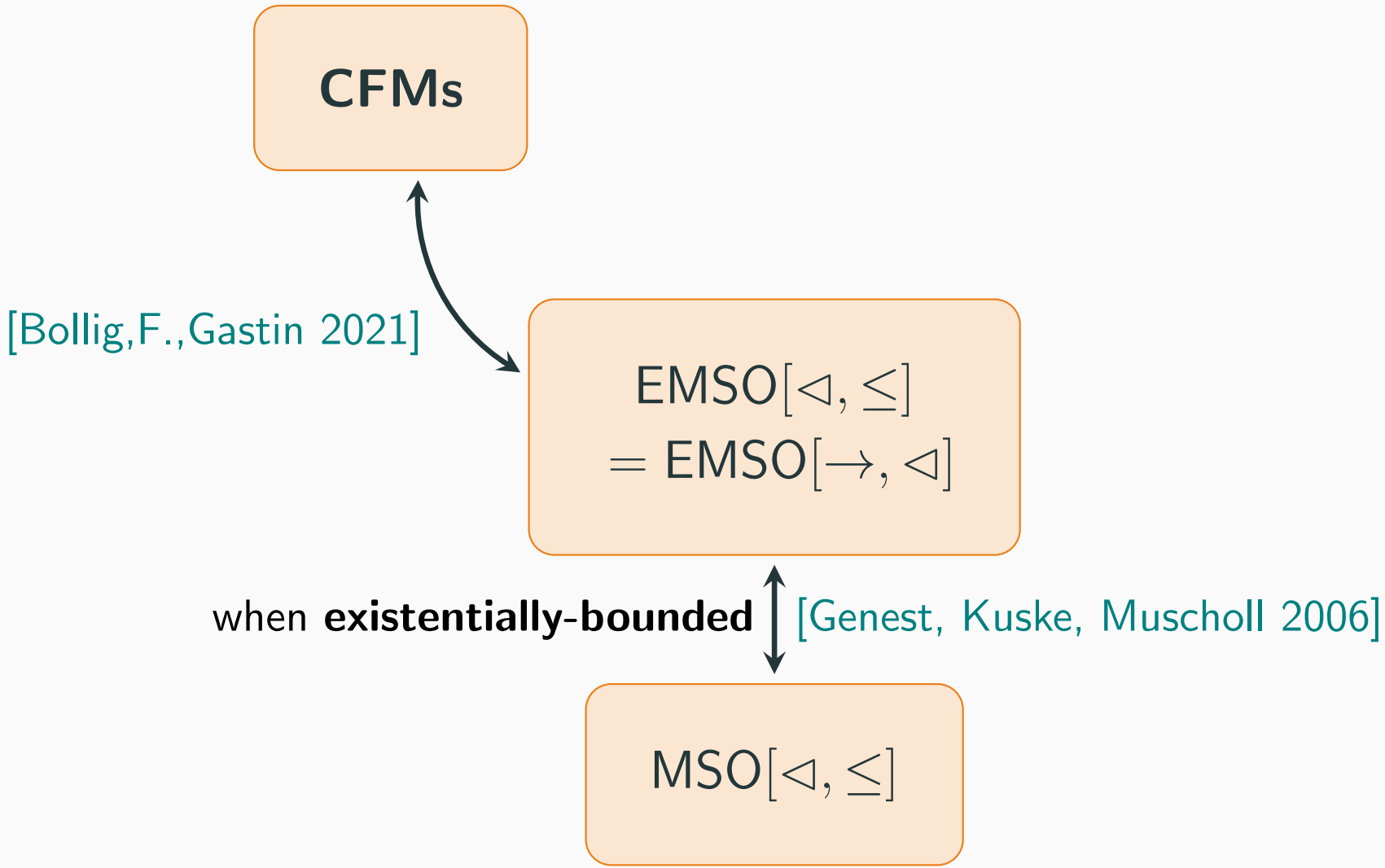
Logical characterization of CFMs

Theorem (Bollig, Leucker 2006)
Over finite MSCs, CFM $=$ EMSO$[\lhd, \rightarrow] \subsetneq$ MSO$[\lhd, \rightarrow]$.

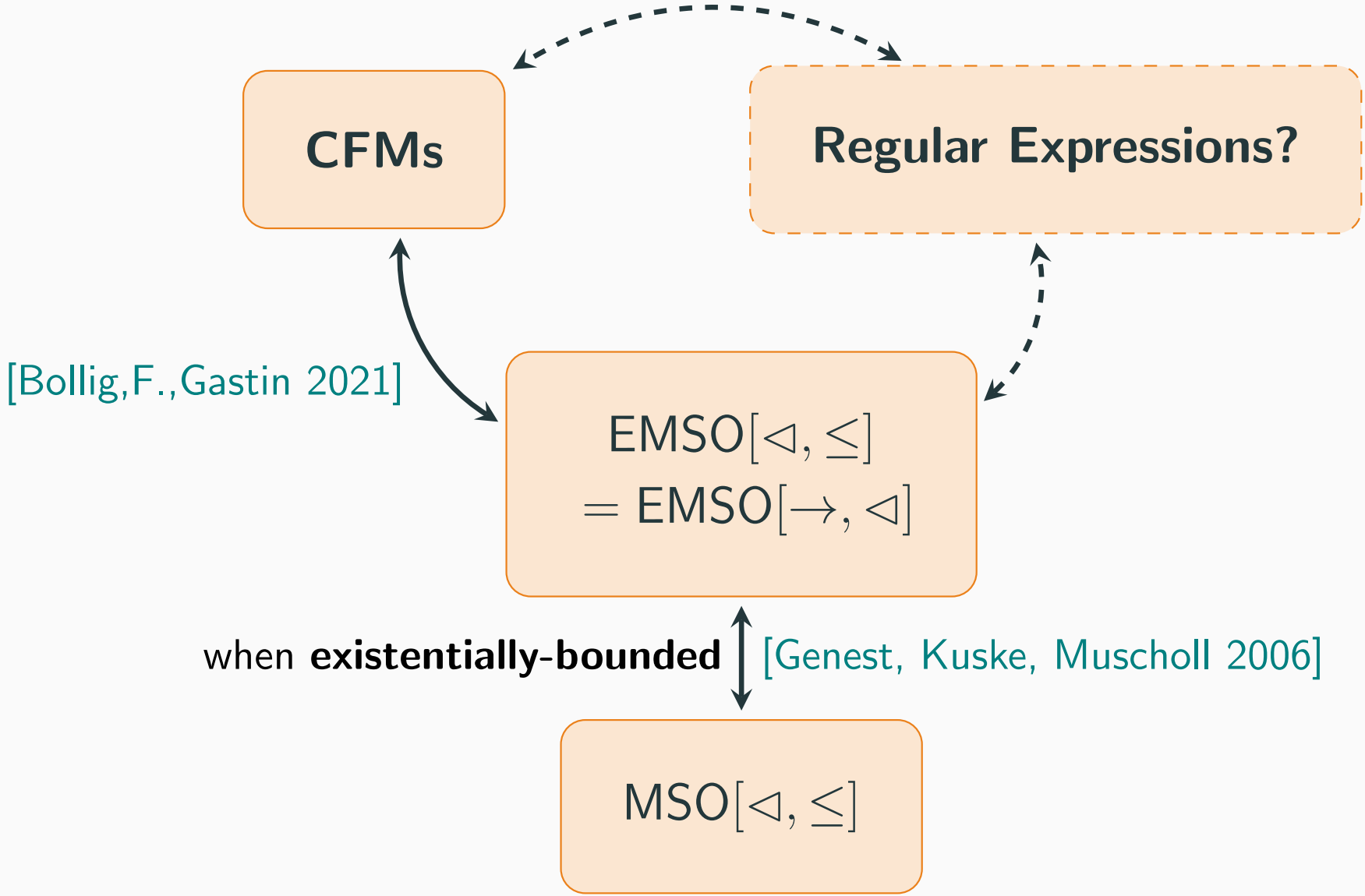
Theorem (Bollig, F., Gastin 2018 & 2021)
CFM $=$ EMSO$[\lhd, \leq]$

Theorem (Genest, Kuske, Muscholl 2006)
Over **existentially bounded** MSCs, CFM $=$ MSO$[\lhd, \leq]$.

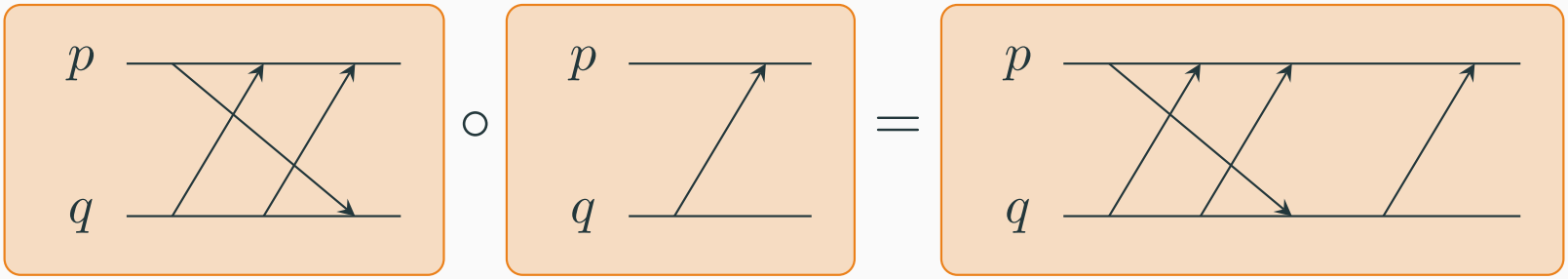
Where we are


CFMs
[Bollig,F.,Gastin 2021]
EMSO[◁, ≤]
= EMSO[→, ◁]
when **existentially-bounded**
[Genest, Kuske, Muscholl 2006]
MSO[◁, ≤]

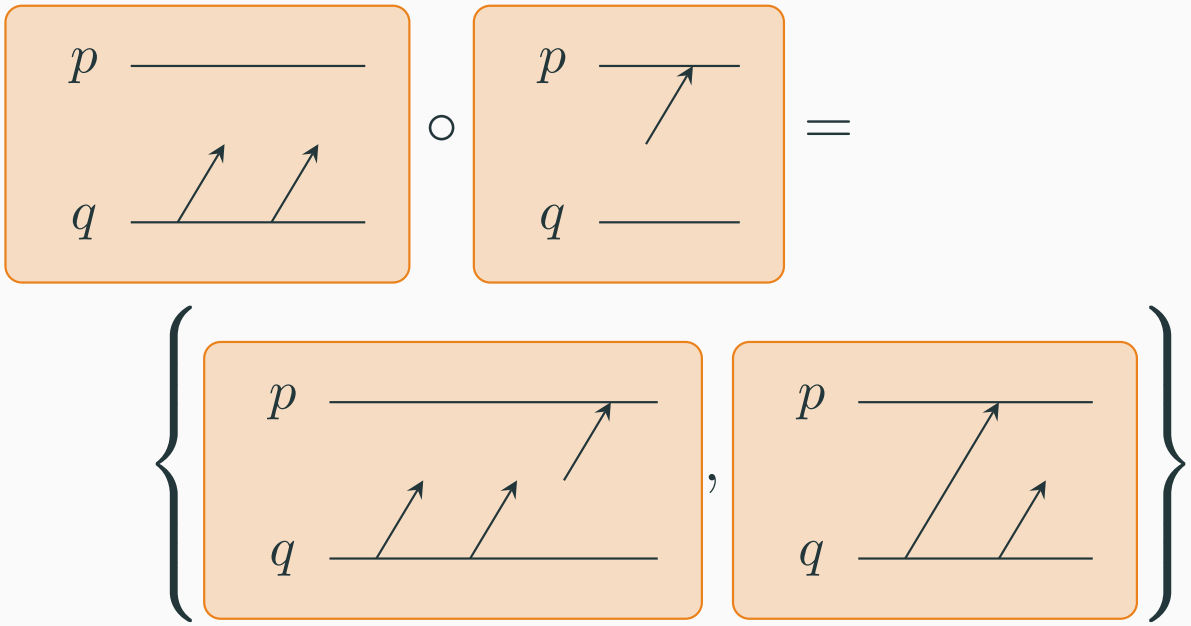
Where we are


CFMs
Regular Expressions?
[Bollig,F.,Gastin 2021]
EMSO[◁, ≤]
= EMSO[→, ◁]
when existentially-bounded
[Genest, Kuske, Muscholl 2006]
MSO[◁, ≤]

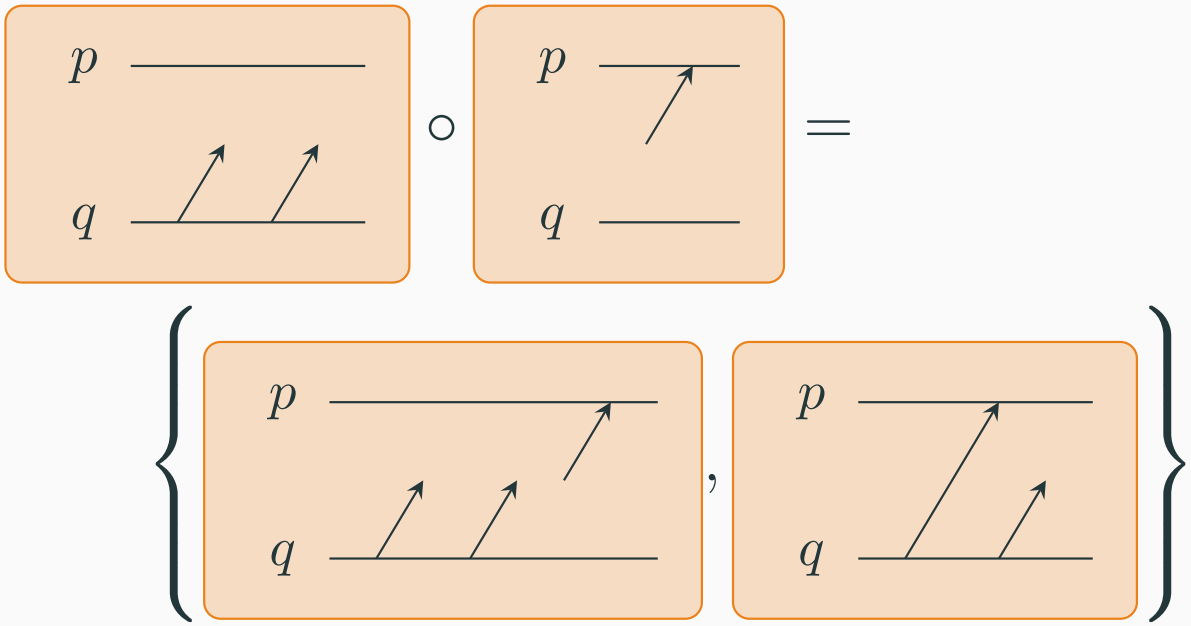
Concatenation and compositional MSCs

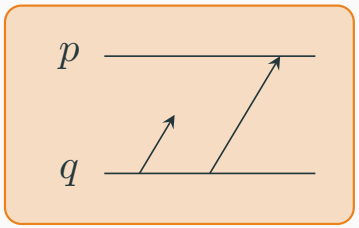


Concatenation and compositional MSCs

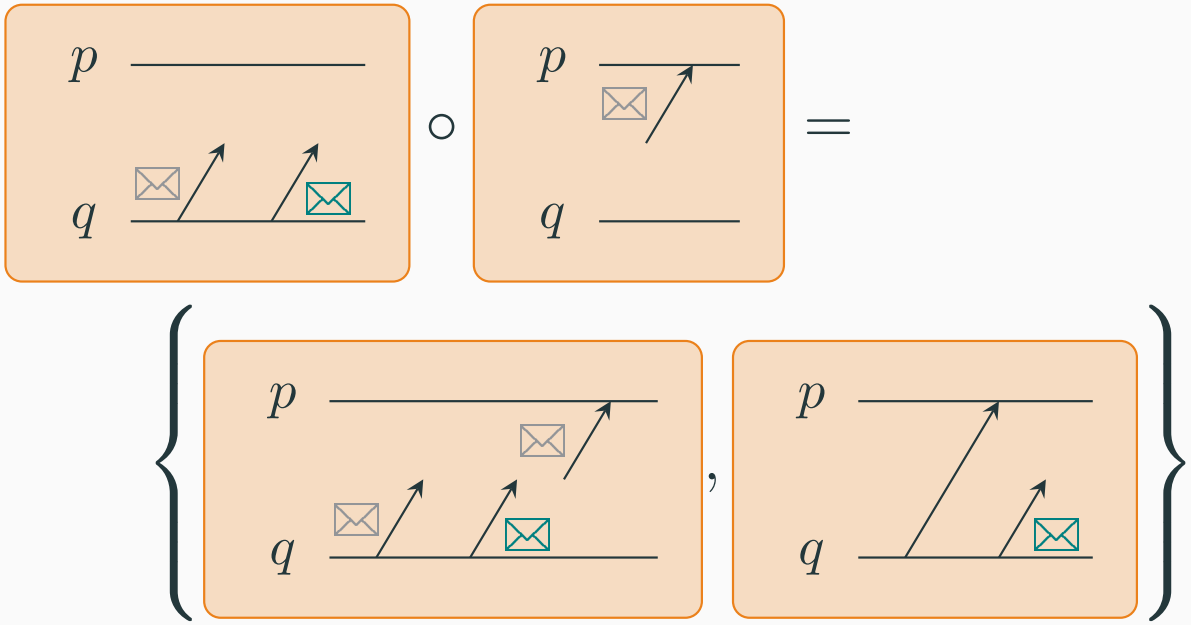


Concatenation and compositional MSCs

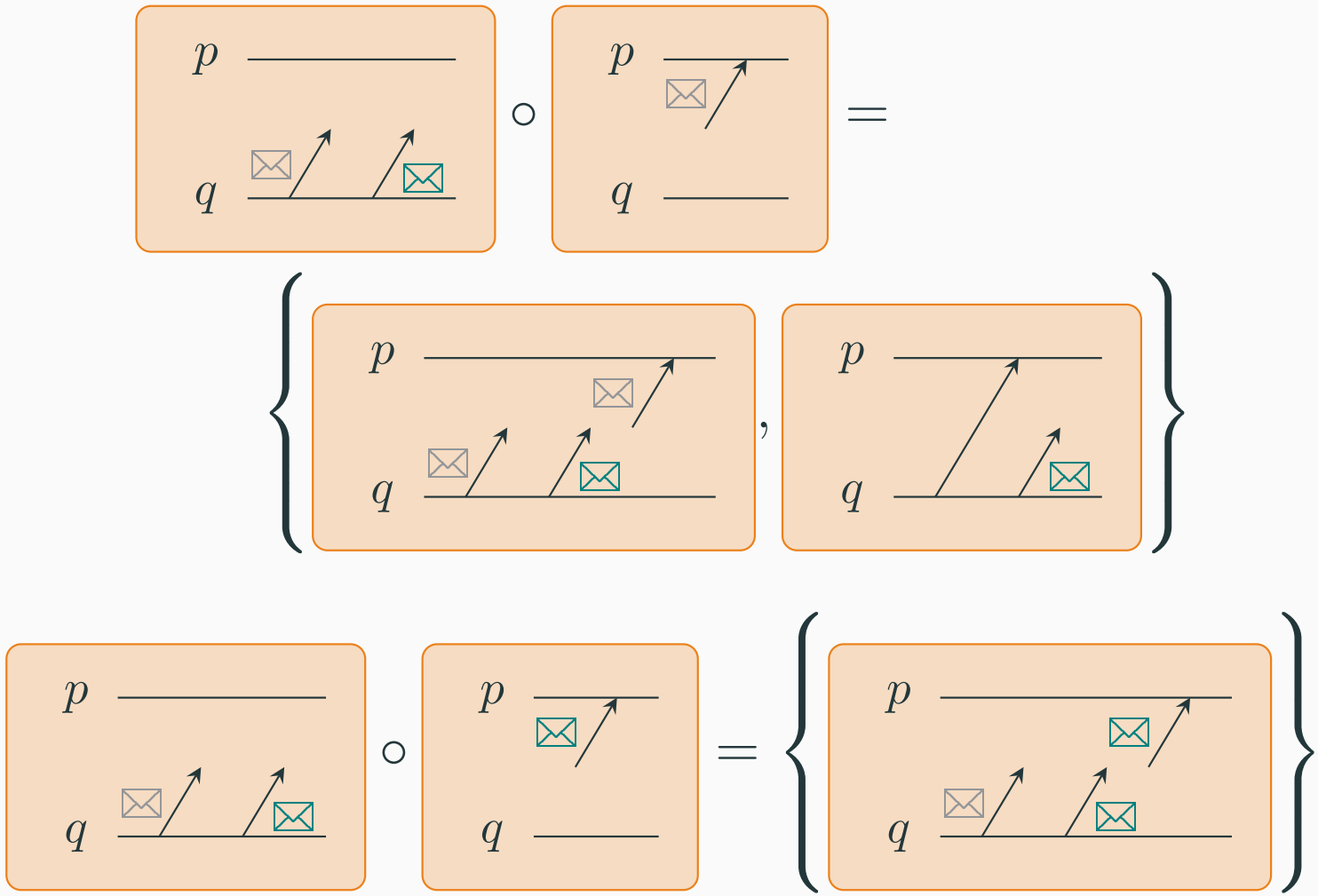


NB:  is not a compositional MSC (not FIFO)

Concatenation and compositional MSCs

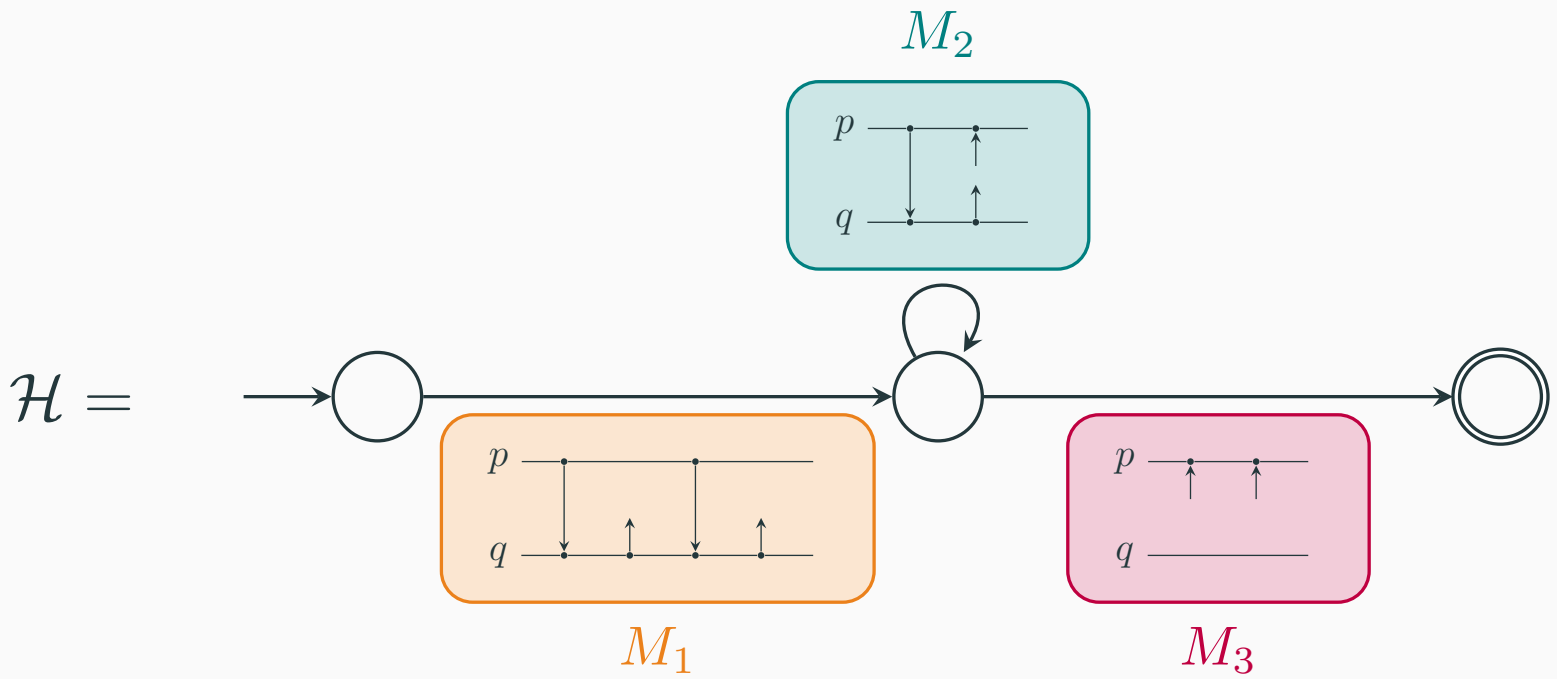


Concatenation and compositional MSCs



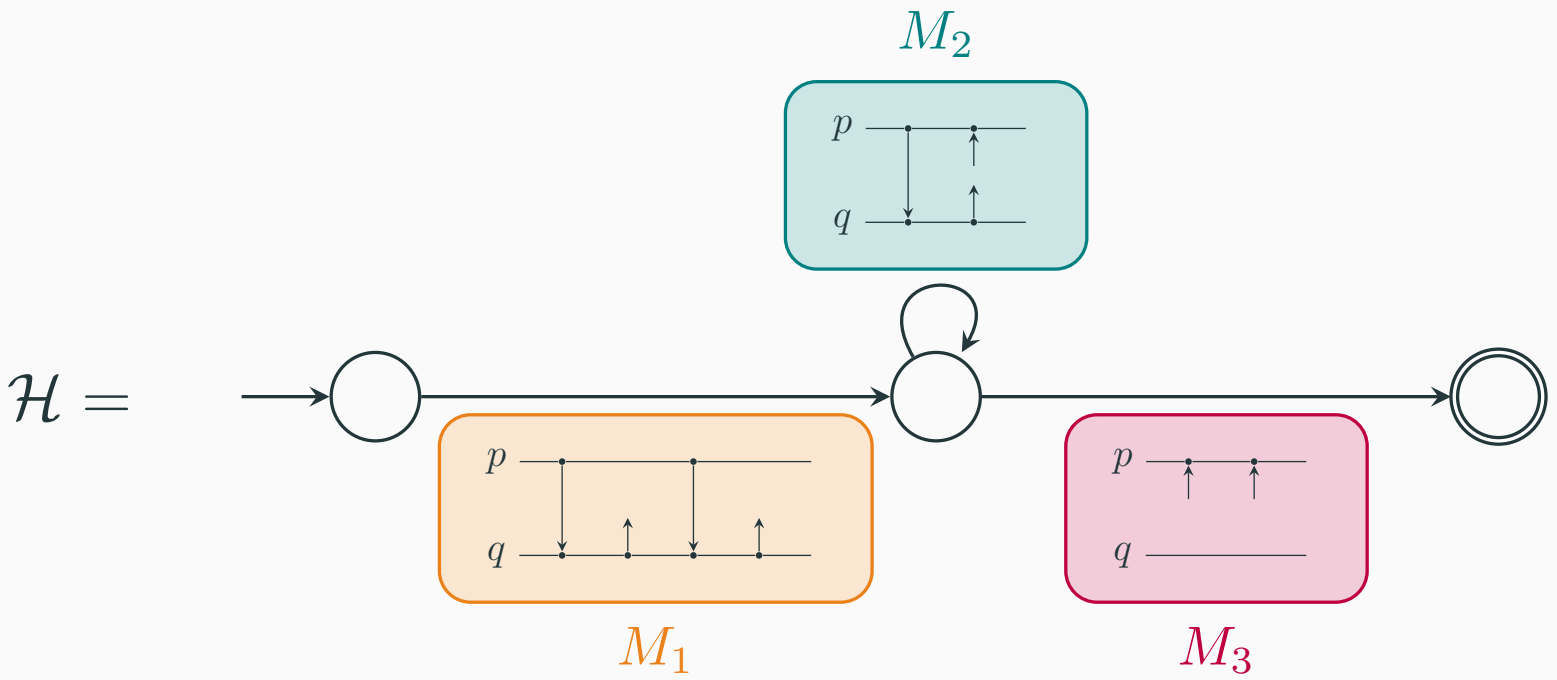
High-level message sequence charts (HMSCs)

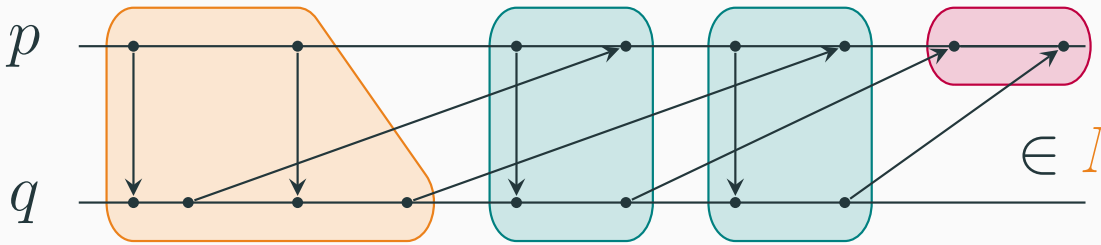
(or message sequence graphs (MSGs))



High-level message sequence charts (HMSCs)

(or message sequence graphs (MSGs))

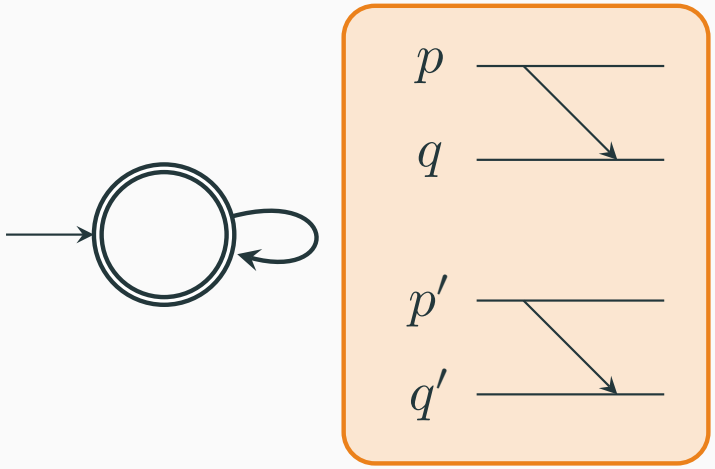




$\in M_1 \cdot M_2 \cdot M_2 \cdot M_3 \subseteq L(\mathcal{H})$

HMSCs are “too” expressive

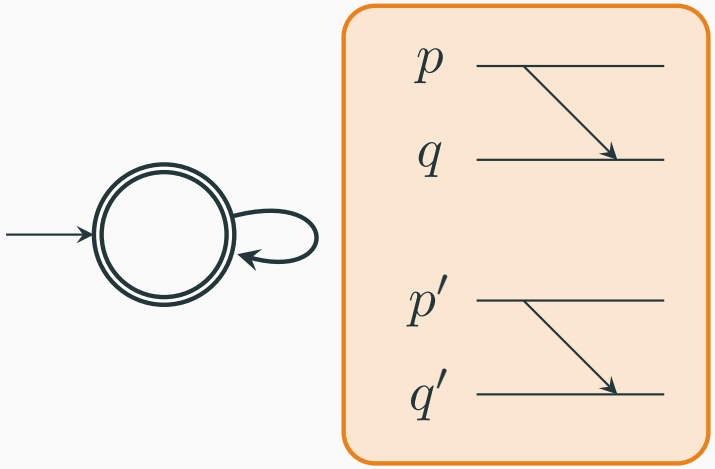
$\mathcal{H} =$



$L(\mathcal{H})$ is not recognisable by a CFM

HMSCs are “too” expressive

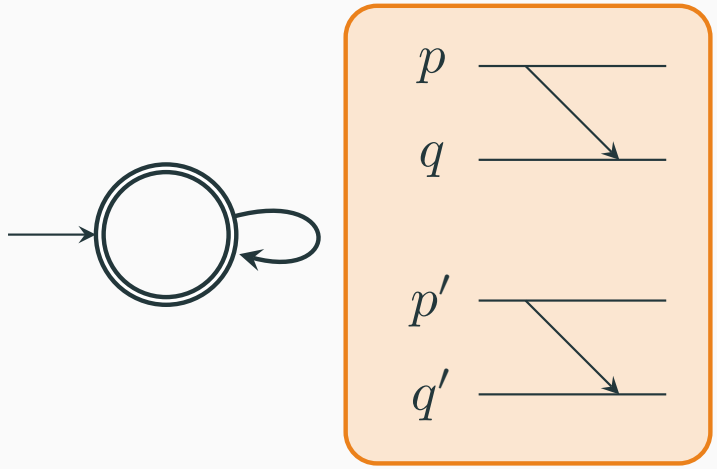
$\mathcal{H} =$



$L(\mathcal{H})$ is not recognisable by a CFM

as many messages between p and q and between p' and q' $\approx a^n b^n$

HMSCs are “too” expressive

$\mathcal{H} =$ 

$L(\mathcal{H})$ is not recognisable by a CFM

as many messages between p and q and between p' and q' $\approx a^n b^n$

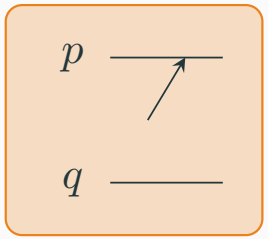
$\rightarrow$ We need restrictions!

Connectivity

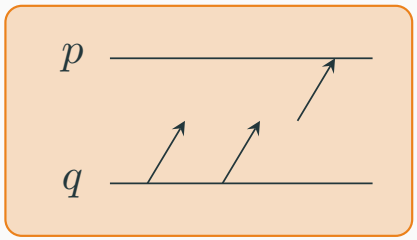
- A cMSC M is connected if the undirected graph $(\text{events}(M), \leq \cup \leq^{-1})$ is connected.
- M is weakly connected if it has a connected undirected communication graph.

Connectivity

- A cMSC M is connected if the undirected graph $(\text{events}(M), \leq \cup \leq^{-1})$ is connected.
- M is weakly connected if it has a connected undirected communication graph.

e.g., 

is both connected and weakly connected, while



is weakly connected but not connected.

Subclasses of HMSCs

- $\mathcal{H}$ is (weakly) loop-connected if for all loops in $\mathcal{H}$ with label $M_1 \cdots M_n$, every cMSC in the product $M_1 \circ \cdots \circ M_n$ is connected.

Subclasses of HMSCs

- $\mathcal{H}$ is (weakly) loop-connected if for all loops in $\mathcal{H}$ with label $M_1 \cdots M_n$, every cMSC in the product $M_1 \circ \cdots \circ M_n$ is connected.
- $\mathcal{H}$ is safe if for all accepting path labeled $M_1 \ldots M_n$ in $\mathcal{H}$, $M_1 \circ \cdots \circ M_n$ contains an MSC ($=$ a cMSC with no unmatched messages).

Subclasses of HMSCs

- $\mathcal{H}$ is (weakly) loop-connected if for all loops in $\mathcal{H}$ with label $M_1 \cdots M_n$, every cMSC in the product $M_1 \circ \cdots \circ M_n$ is connected.
- $\mathcal{H}$ is safe if for all accepting path labeled $M_1 \ldots M_n$ in $\mathcal{H}$, $M_1 \circ \cdots \circ M_n$ contains an MSC ($=$ a cMSC with no unmatched messages).

Theorem (Genest, Kuske, Muscholl 2006)
L is definable by some **safe** weakly connected HMSC if and only if it is **existentially bounded** and recognisable by some CFM.

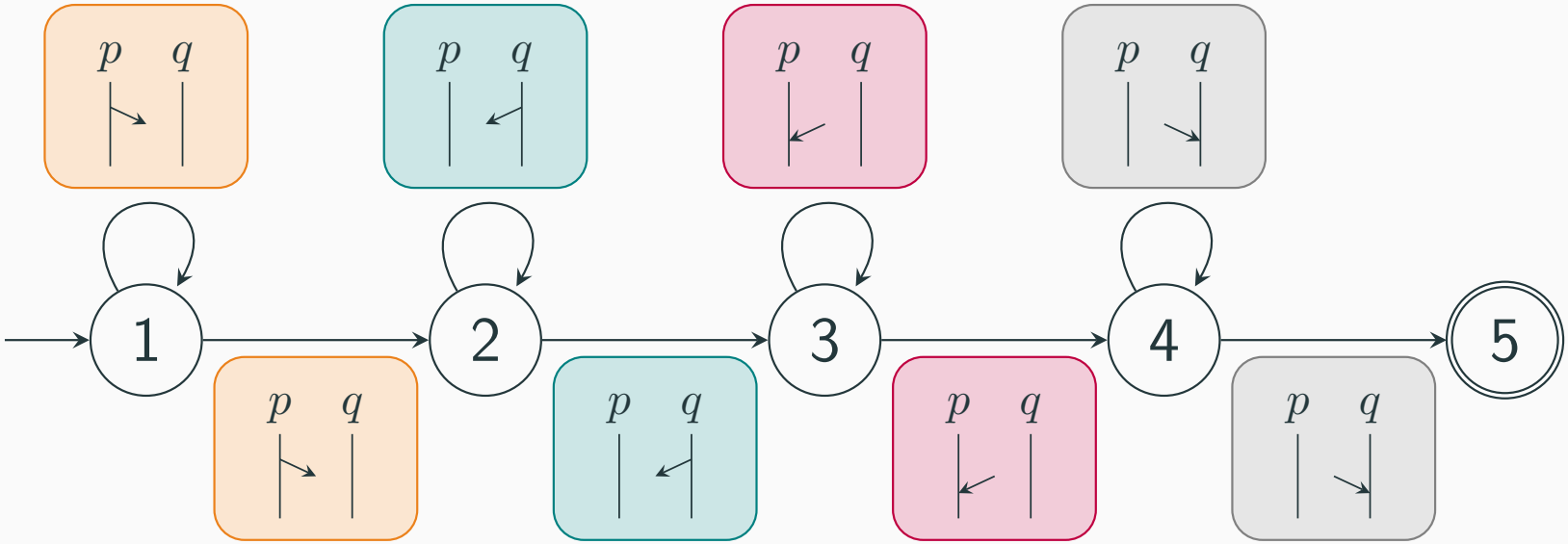
Subclasses of HMSCs

- $\mathcal{H}$ is (weakly) loop-connected if for all loops in $\mathcal{H}$ with label $M_1 \cdots M_n$, every cMSC in the product $M_1 \circ \cdots \circ M_n$ is connected.
- $\mathcal{H}$ is safe if for all accepting path labeled $M_1 \ldots M_n$ in $\mathcal{H}$, $M_1 \circ \cdots \circ M_n$ contains an MSC ($=$ a cMSC with no unmatched messages).

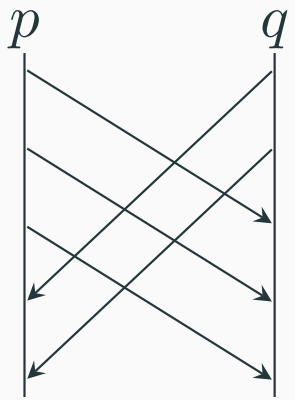
Theorem (Genest, Kuske, Muscholl 2006)
L is definable by some **safe** weakly connected HMSC if and only if it is **existentially bounded** and recognisable by some CFM.

What about HMSCs that are connected but not safe?

Example



$L(\mathcal{H})$ is not $\exists B$-bounded, but is recognisable by a CFM.



Connected HMSCs – Implementability

Theorem (Bollig, F., Gastin 2025)
Every **loop-connected HMSC** can be translated into an equivalent **EMSO** formula (and thus, into an equivalent CFM).

Connected HMSCs – Implementability

Theorem (Bollig, F., Gastin 2025)
Every **loop-connected HMSC** can be translated into an equivalent **EMSO** formula (and thus, into an equivalent CFM).

Step 1: show that EMSO-definable languages of CMSs are closed under

- union
- concatenation
- iteration if all cMSCs in the language are connected.

Connected HMSCs – Implementability

Theorem (Bollig, F., Gastin 2025)
Every **loop-connected HMSC** can be translated into an equivalent **EMSO** formula (and thus, into an equivalent CFM).

Step 1: show that EMSO-definable languages of CMSs are closed under

- union
- concatenation
- iteration if all cMSCs in the language are connected.

Step 2: Apply standard automata-to-expressions translation techniques.

Satisfiability

Satisfiability: Given an HMSC $\mathcal{H}$, is $L(\mathcal{H}) \neq \varnothing$?

Satisfiability

Satisfiability: Given an HMSC $\mathcal{H}$, is $L(\mathcal{H}) \neq \varnothing$?

Theorem (Genest, Kuske, Muscholl 2006)
Satisfiability is **decidable** for **safe weakly connected** HMSCs.

Satisfiability

Satisfiability: Given an HMSC $\mathcal{H}$, is $L(\mathcal{H}) \neq \varnothing$?

Theorem (Genest, Kuske, Muscholl 2006)
Satisfiability is **decidable** for **safe weakly connected** HMSCs.

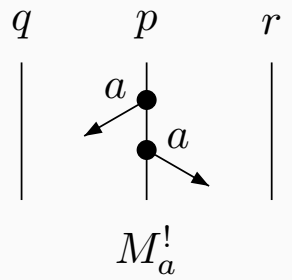
Theorem (Bollig, F., Gastin 2025)

- Emptiness of unrestricted HMSCs is undecidable, even with a message alphabet of size one.
- With a message alphabet of size at least two, emtpiness of HMSCs is **undecidable** even for **loop-connected HMSCs**. This is true even with only two processes, or three processes and flat HSMCs.

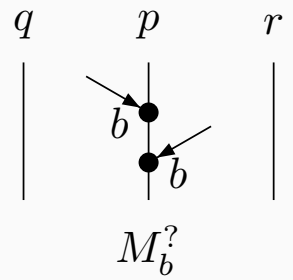
Undecidability for loop-connected HMSCs

Reduction from Post Correspondance Problem:

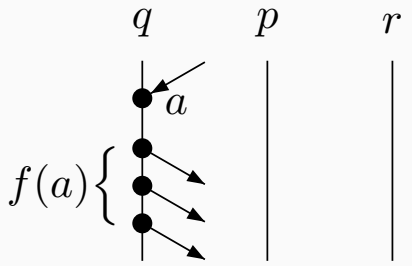
$f, g : A^* \to B^*$



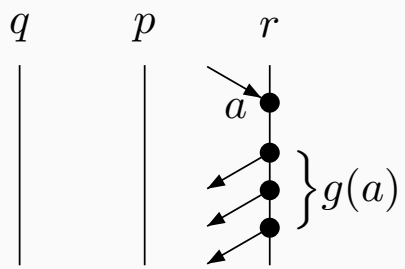
$M_a^!$



$M_b^?$



M_a^f

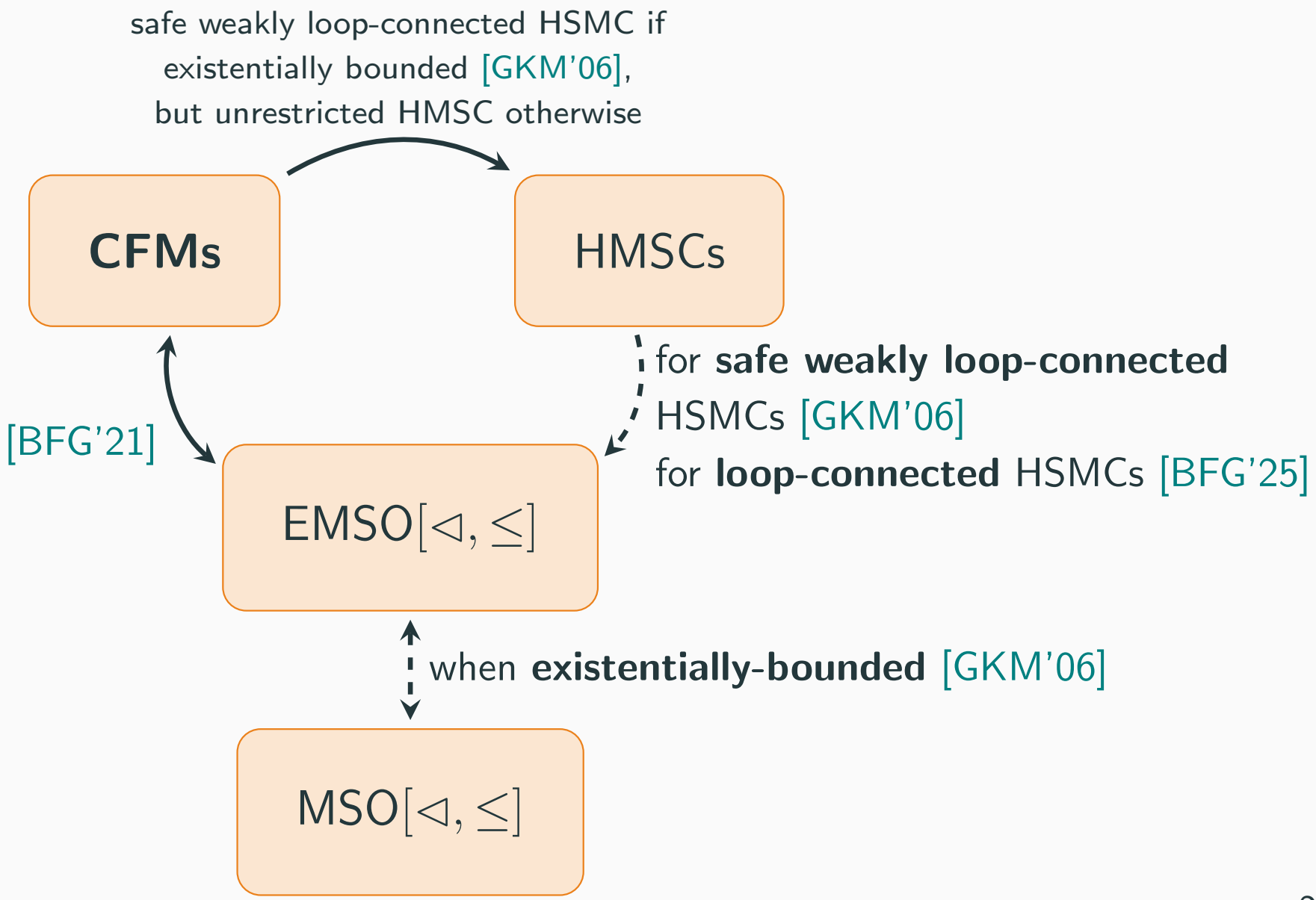


M_a^g

There exists $u \in A^+$ such that $f(u) = g(u)$ if and only if

$$\Big(\sum_{a\in A} M_a^!\Big)^+ \cdot \Big(\sum_{a\in A} M_a^f\Big)^+ \cdot \Big(\sum_{a\in A} M_a^g\Big)^+ \cdot \Big(\sum_{b\in B} M_b^?\Big)^+ \neq \emptyset$$

Conclusion


safe weakly loop-connected HSMC if
existentially bounded [GKM'06],
but unrestricted HMSC otherwise
CFMs
HMSCs
[BFG'21]
for **safe weakly loop-connected**
HSMCs [GKM'06]
for **loop-connected** HSMCs [BFG'25]
EMSO[◁, ≤]
when **existentially-bounded** [GKM'06]
MSO[◁, ≤]

Some open questions

- **Relation between our definition past ones:** Is every safe and weakly loop-connected HMSC equivalent to a loop-connected HMSC?

Some open questions

- **Relation between our definition past ones:** Is every safe and weakly loop-connected HMSC equivalent to a loop-connected HMSC?
- **Refinements of the undecidability results**, e.g., does undecidability holds for loop-connected and flat HMSCs with two processes?

Some open questions

- **Relation between our definition past ones:** Is every safe and weakly loop-connected HMSC equivalent to a loop-connected HMSC?
- **Refinements of the undecidability results**, e.g., does undecidability holds for loop-connected and flat HMSCs with two processes?
- Is there an interesting subclass of (loop-connected or not) **HMSCs beyond existentially bounded MSCs with decidable satisfiability/model checking** problems?

Some open questions

- **Relation between our definition past ones:** Is every safe and weakly loop-connected HMSC equivalent to a loop-connected HMSC?
- **Refinements of the undecidability results**, e.g., does undecidability holds for loop-connected and flat HMSCs with two processes?
- Is there an interesting subclass of (loop-connected or not) **HMSCs beyond existentially bounded MSCs with decidable satisfiability/model checking** problems?
- What are the **CFMs** that correspond to loop-connected HMSCs?

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties
  - even more undecidable

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties
  - even more undecidable
  - comparisons with logic are more subtle

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties
  - even more undecidable
  - comparisons with logic are more subtle
- Enrich HMSCs with unbounded parallel composition or similar operators?

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties
  - even more undecidable
  - comparisons with logic are more subtle
- Enrich HMSCs with unbounded parallel composition or similar operators?

Parameterized systems

Here the set of processes is fixed. What about parameterized systems?

- Parameterized communicating automata [Bollig, 2014]
- Other possible definitions: registers to store process ids, dynamic process creation, etc.
- Difficulties
  - even more undecidable
  - comparisons with logic are more subtle
- Enrich HMSCs with unbounded parallel composition or similar operators?

Thoughts?

Thank you!