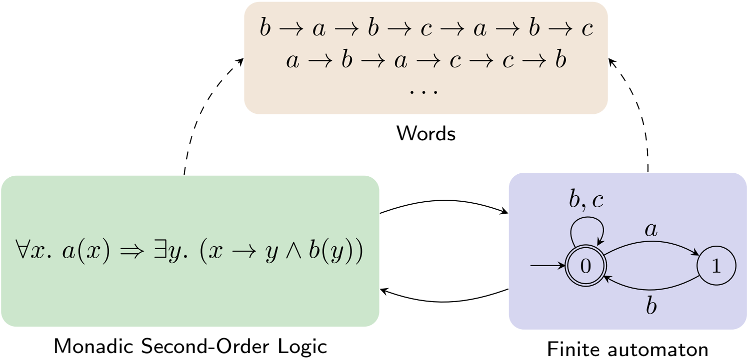
It Is Easy to Be Wise After the Event: Communicating Finite-State Machines Capture First-Order Logic with “Happened Before”

Benedikt Bollig, Marie Fortin, Paul Gastin

LSV, ENS Paris-Saclay

October 22, 2018
IRIF – Séminaire Vérification

Büchi-Elgot-Trakhtenbrot theorem ('60s)



MSO$[\to]$ = Finite Automata

Büchi-Elgot-Trakhtenbrot theorem ('60s)

$b \to a \to b \to c \to a \to b \to c$

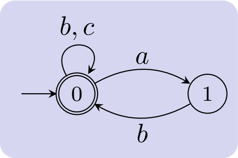
$a \to b \to a \to c \to c \to b$

$\dots$

Words

$\forall x.\ a(x) \Rightarrow \exists y.\ (x \to y \wedge b(y))$

Monadic Second-Order Logic

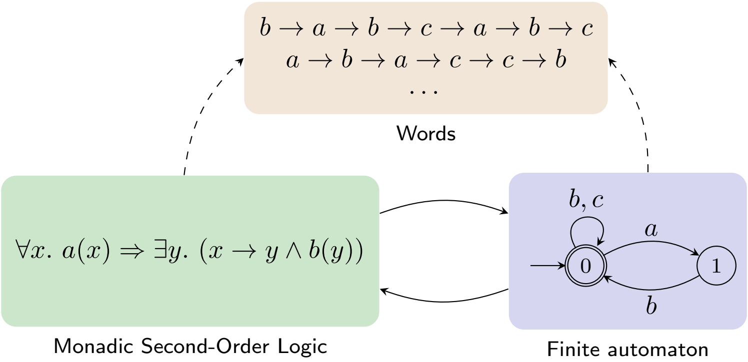


Finite automaton

$$\mathsf{MSO}[\to] = \text{Finite Automata} = \mathsf{EMSO}[\to]$$

$\exists X_0 \dots \exists X_n . \varphi$ with $\varphi \in \mathsf{FO}[\to]$

Büchi-Elgot-Trakhtenbrot theorem ('60s)

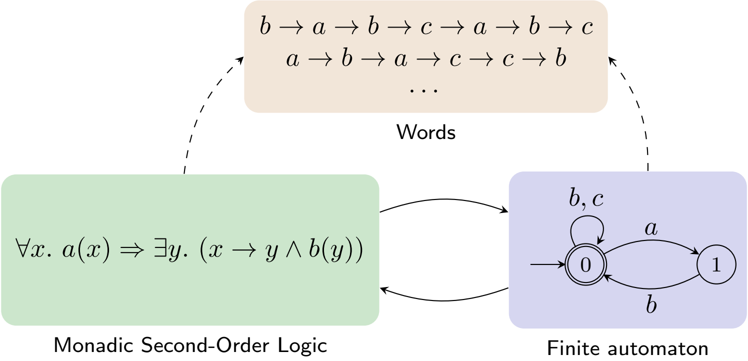


$$\mathsf{MSO}[\to] = \text{Finite Automata} = \mathsf{EMSO}[\to]$$

$\exists X_0 \dots \exists X_n . \varphi$ with $\varphi \in \mathsf{FO}[\to]$

Extended to trees [Thatcher-Wright '68], Mazurkiewicz traces [Thomas '90], ...

Büchi-Elgot-Trakhtenbrot theorem ('60s)



$$\text{MSO}[\to] = \text{Finite Automata} = \text{EMSO}[\to]$$

$\exists X_0 \dots \exists X_n . \varphi$ with $\varphi \in \text{FO}[\to]$

Goal: A Büchi-like theorem for message-passing systems

Büchi-Elgot-Trakhtenbrot theorem ('60s)

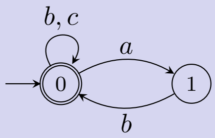
$$b \to a \to b \to c \to a \to b \to c$$
$$a \to b \to a \to c \to c \to b$$
$$\ldots$$

Words

$$\forall x.\ a(x) \Rightarrow \exists y.\ (x \to y \wedge b(y))$$

Monadic Second-Order Logic

Synthesis



Finite automaton

$$\text{MSO}[\to] = \text{Finite Automata} = \text{EMSO}[\to]$$

$\exists X_0 \ldots \exists X_n.\varphi$ with $\varphi \in$ FO$[\to]$

Goal: A Büchi-like theorem for message-passing systems

The model

The model

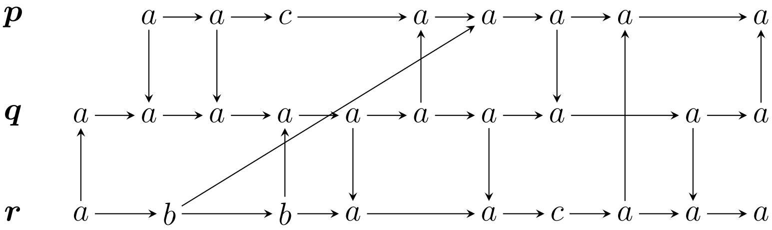
- Fixed, finite set of processes
- (Reliable) unbounded point-to-point FIFO channels

The model

- Fixed, finite set of processes
- (Reliable) unbounded point-to-point FIFO channels
- Partial order semantics: Message Sequence Charts (MSC)

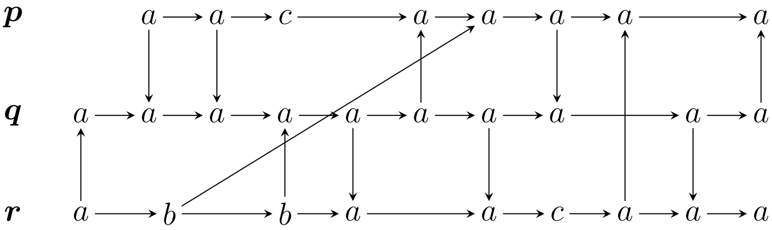
The model

- Fixed, finite set of processes
- (Reliable) unbounded point-to-point FIFO channels
- Partial order semantics: Message Sequence Charts (MSC)



The model

- Fixed, finite set of processes
- (Reliable) unbounded point-to-point FIFO channels
- Partial order semantics: Message Sequence Charts (MSC)



- Specifications based on the partial order, independent from the choice of a particular interleaving

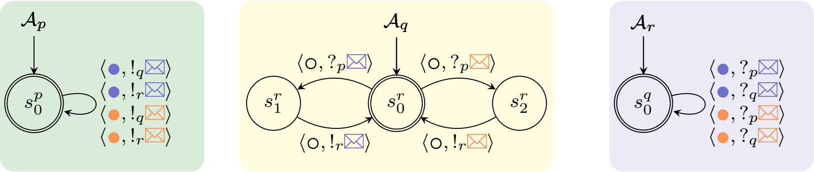
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

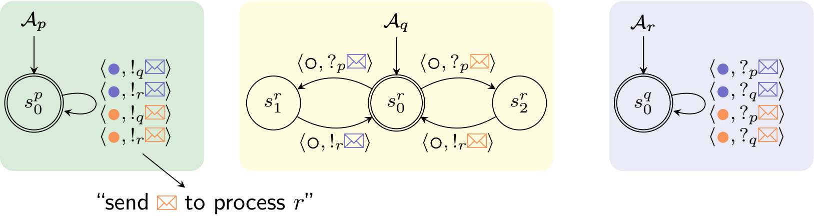
$P = \{p, q, r\}$, $\Sigma = \{\bullet, \bullet, \circ\}$, $Msg = \{✉, ✉\}$



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

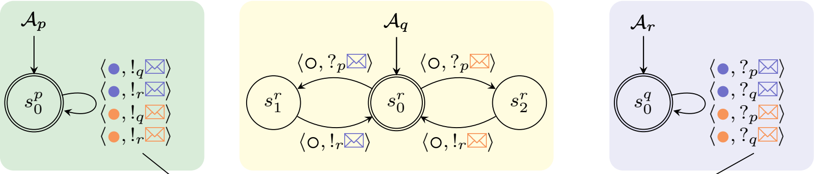
$$P = \{p, q, r\},\ \Sigma = \{\bullet, \bullet, \circ\},\ Msg = \{\boxtimes, \boxtimes\}$$



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$, $Msg = \{$✉, ✉$\}$



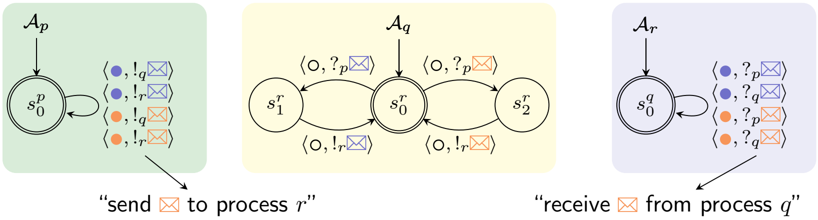
"send ✉ to process r"

"receive ✉ from process q"

Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{●, ●, ○\}$, $Msg = \{✉, ✉\}$



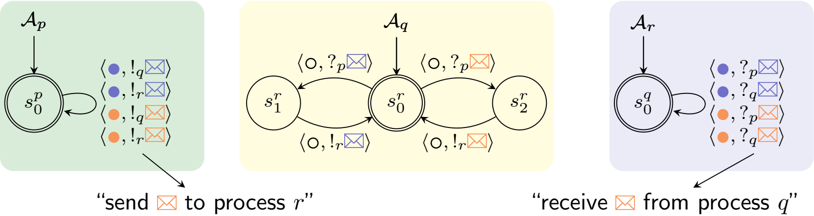
Communicating finite-state machine:

- One finite-state transition system for each process, using a **finite** set of messages
- Global acceptance condition

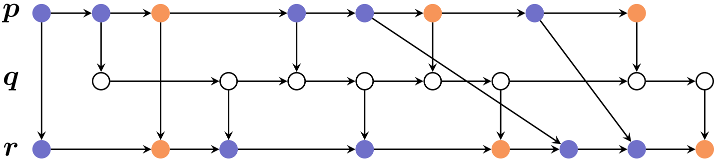
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{\bullet, \bullet, \circ\}$, $Msg = \{\boxtimes, \boxtimes\}$



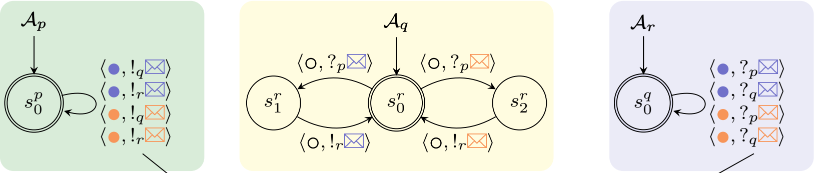
CFMs recognize languages of MSCs:



Communicating finite-state machines (CFMs)

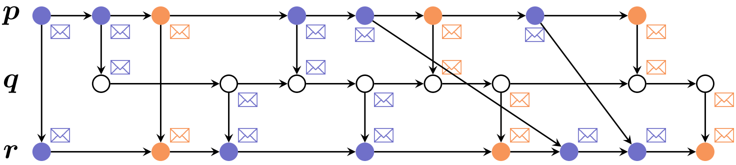
[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{\bullet, \bullet, \circ\}$, $Msg = \{✉, ✉\}$



"send ✉ to process r" "receive ✉ from process q"

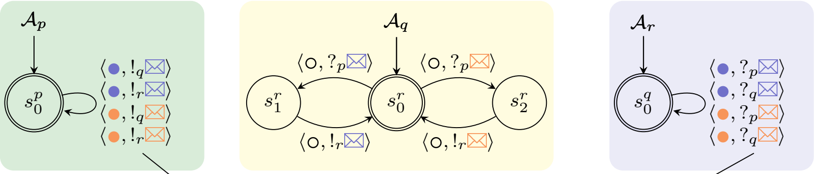
CFMs recognize languages of MSCs:



Communicating finite-state machines (CFMs)

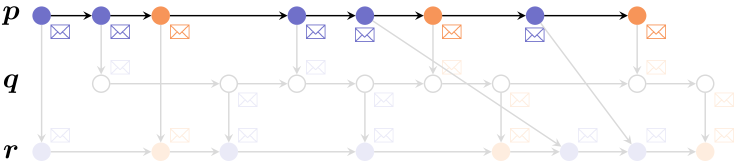
[Brand–Zafiropulo '83]

$$P = \{p, q, r\},\ \Sigma = \{\bullet, \bullet, \circ\},\ Msg = \{\boxtimes, \boxtimes\}$$



"send $\boxtimes$ to process r"   "receive $\boxtimes$ from process q"

CFMs recognize languages of MSCs:

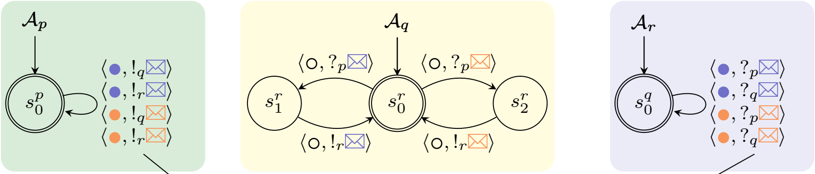


$\in L(\mathcal{A}_p)$

Communicating finite-state machines (CFMs)

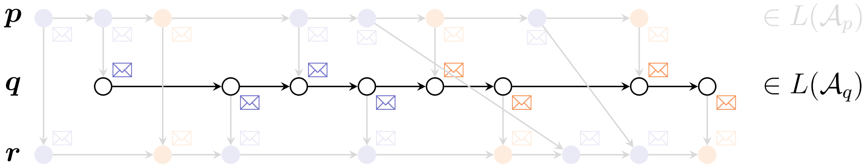
[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{\bullet, \bullet, \circ\}$, $Msg = \{\boxtimes, \boxtimes\}$



"send $\boxtimes$ to process r" "receive $\boxtimes$ from process q"

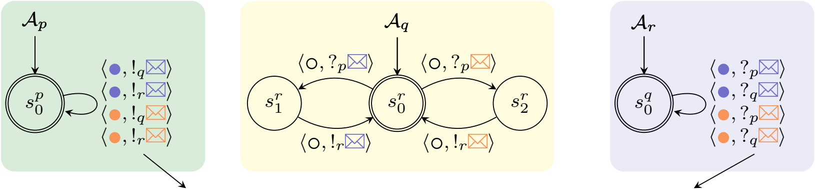
CFMs recognize languages of MSCs:



Communicating finite-state machines (CFMs)

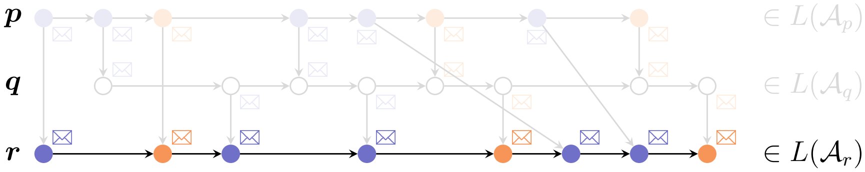
[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$, $Msg = \{$✉, ✉$\}$



"send ✉ to process r" "receive ✉ from process q"

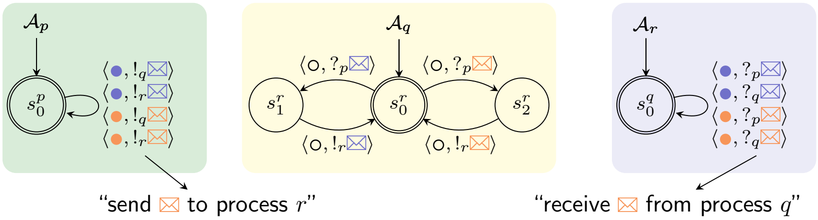
CFMs recognize languages of MSCs:



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$, $Msg = \{$✉, ✉$\}$



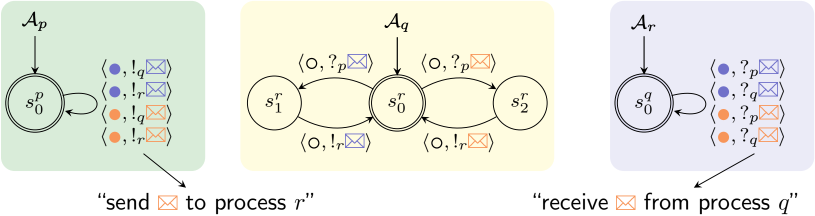
Remarks

- The emptiness problem for CFMs is **undecidable**.

Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$, $Msg = \{$✉, ✉$\}$



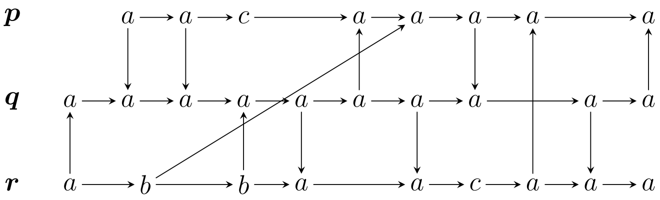
Remarks

- The emptiness problem for CFMs is **undecidable**.
- CFMs are inherently **non-deterministic**.

[Genest-Kuske-Muscholl '07]

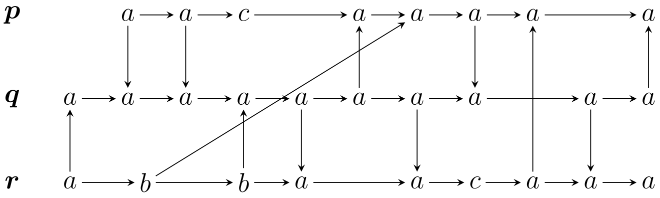
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::=$



Monadic Second-Order logic (MSO) over MSCs

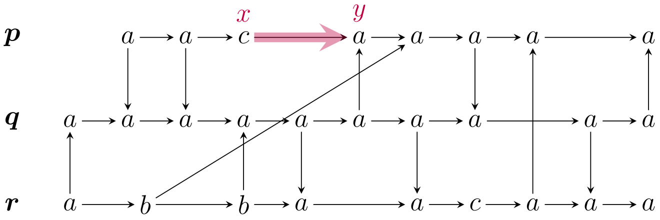
$\varphi ::= \quad a(x) \mid p(x)$ label/process of event x



Monadic Second-Order logic (MSO) over MSCs

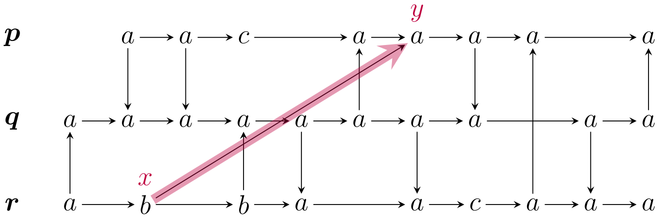
$\varphi ::= \quad a(x) \mid p(x)$ label/process of event x

$\mid x \rightarrow y$ process successor



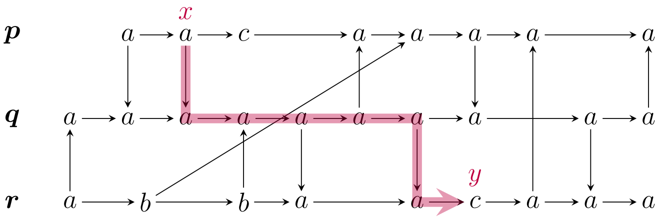
Monadic Second-Order logic (MSO) over MSCs

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



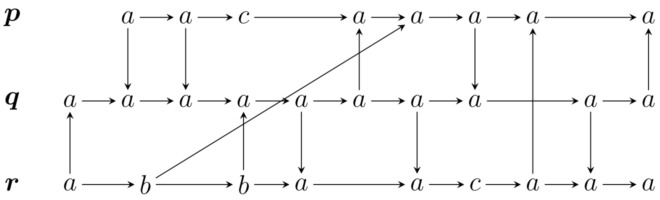
Monadic Second-Order logic (MSO) over MSCs

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



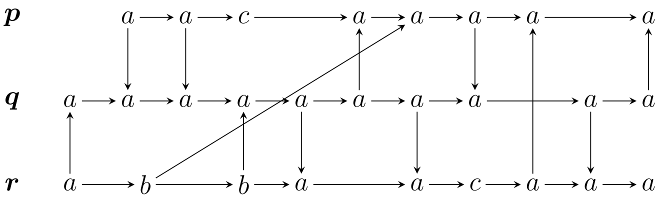
Monadic Second-Order logic (MSO) over MSCs

$\varphi ::= \quad a(x) \mid p(x)$	label/process of event x
$\mid x \to y$	process successor
$\mid x \lhd y$	message relation
$\mid x \leq y$	happened-before
$\mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi \mid \exists X.\ \varphi \mid x \in X$	



Monadic Second-Order logic (MSO) over MSCs

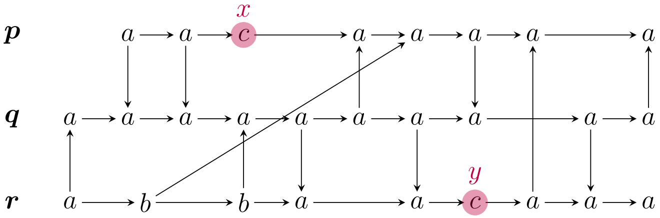
$$
\begin{array}{lll}
\varphi ::= & a(x) \mid p(x) & \text{label/process of event } x \\
& \mid x \rightarrow y & \text{process successor} \\
& \mid x \lhd y & \text{message relation} \\
& \mid x \leq y & \text{happened-before} \\
& \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi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



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

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
	$\mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\ \varphi \mid \exists X.\ \varphi \mid x \in X$	



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

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

Büchi-like theorems for CFMs

Büchi-like theorems for CFMs

When channels are **bounded**:

Büchi-like theorems for CFMs

When channels are **bounded**:

Theorem (Henriksen-Mukund-Narayan Kumar-Sohoni-Thiagarajan '05)

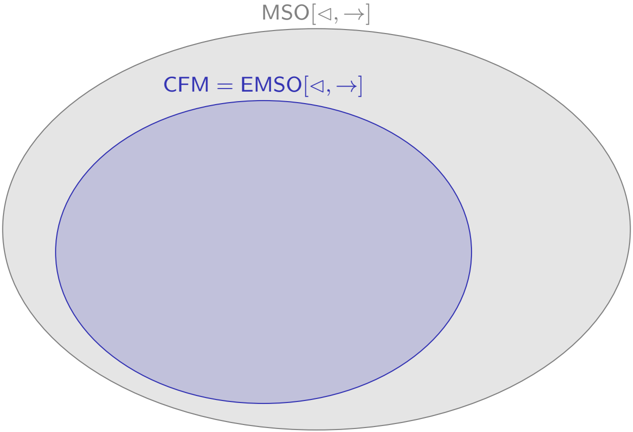
Over universally bounded MSCs, CFM $=$ MSO$[\lhd, \rightarrow, \leq]$.

Theorem (Genest-Kuske-Muscholl '06)

Over existentially bounded MSCs, CFM $=$ MSO$[\lhd, \rightarrow, \leq]$.

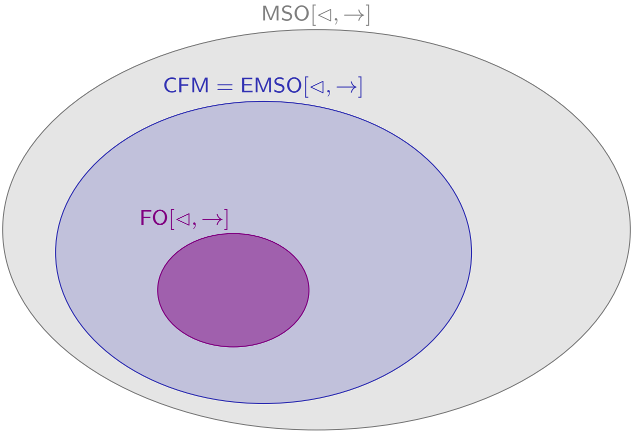
General case

General case



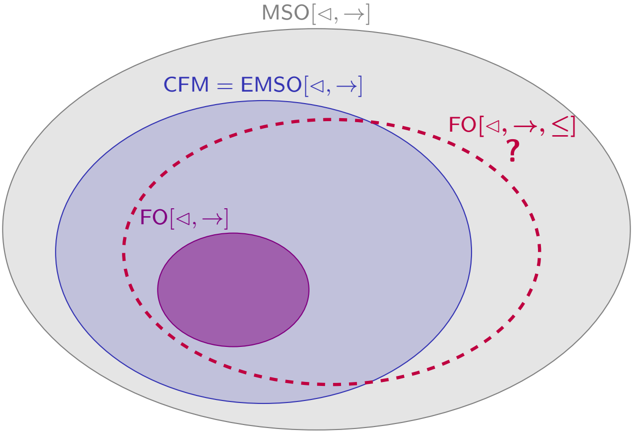
[Bollig-Leucker '06] $\mathrm{CFM} = \mathrm{EMSO}[\lhd, \rightarrow] \subsetneq \mathrm{MSO}[\lhd, \rightarrow]$

General case



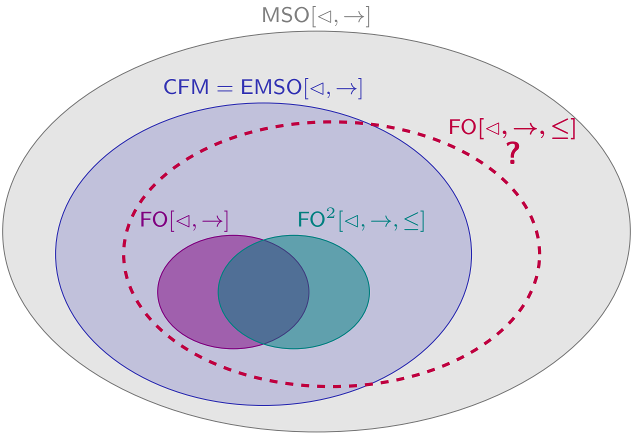
[Bollig-Leucker '06] $CFM = EMSO[\lhd, \rightarrow] \subsetneq MSO[\lhd, \rightarrow]$

General case



[Bollig-Leucker '06] $CFM = EMSO[\lhd, \rightarrow] \subsetneq MSO[\lhd, \rightarrow]$

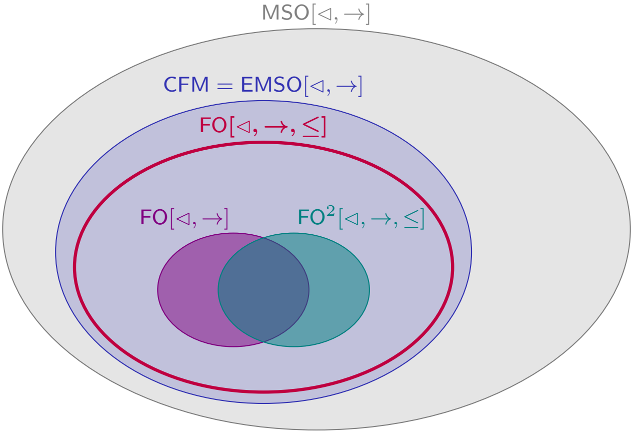
General case



[Bollig-Leucker '06] $\mathsf{CFM} = \mathsf{EMSO}[\lhd, \rightarrow] \subsetneq \mathsf{MSO}[\lhd, \rightarrow]$

[Bollig-F.-Gastin '18] $\mathsf{CFM} = \mathsf{EMSO}^2[\lhd, \rightarrow, \leq]$

General case



[Bollig-Leucker '06] $\mathrm{CFM} = \mathrm{EMSO}[\lhd, \rightarrow] \subsetneq \mathrm{MSO}[\lhd, \rightarrow]$

[Bollig-F.-Gastin '18] $\mathrm{CFM} = \mathrm{EMSO}^2[\lhd, \rightarrow, \leq]$

Main result: $\mathrm{CFM} = \mathrm{EMSO}[\lhd, \rightarrow, \leq]$

Translation from FO to CFMs

Goal

FO$[\lhd, \rightarrow, \leq]$ sentence φ → CFM $\mathcal{A}$ such that $L(A) = L(\varphi)$

Translation from FO to CFMs

Goal

$\mathrm{FO}[\lhd, \rightarrow, \leq]$ sentence φ $\longrightarrow$ CFM $\mathcal{A}$ such that $L(A) = L(\varphi)$

- CFMs are not closed under complementation
  – no direct induction on φ

Translation from FO to CFMs

Goal

$FO[\lhd, \rightarrow, \leq]$ sentence φ → CFM $\mathcal{A}$ such that $L(A) = L(\varphi)$

- CFMs are not closed under complementation – no direct induction on φ
- Techniques used for previous cases do not apply here

Translation from FO to CFMs

Goal

$\mathsf{FO}[\lhd, \rightarrow, \leq]$ sentence φ $\longrightarrow$ $\mathsf{PDL}_{\mathsf{sf}}$ formula $\widetilde{\varphi}$ $\longrightarrow$ CFM $\mathcal{A}$ such that $L(A) = L(\varphi)$

- CFMs are not closed under complementation
  – no direct induction on φ
- Techniques used for previous cases do not apply here

Solution: go through an intermediate language:
"Star-free" Propositional Dynamic Logic (with Loop and Converse)

A simple modal logic for MSCs: $\mathrm{PDL}_{\mathsf{sf}}^{-}$

$\varphi, \psi ::=$

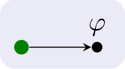
A simple modal logic for MSCs: PDL_{sf}^{-}

$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

A simple modal logic for MSCs: $\mathrm{PDL}_{\mathrm{sf}}^{-}$

$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle \rightarrow \rangle\, \varphi$$

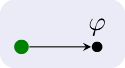


("Next φ")

A simple modal logic for MSCs: $\mathrm{PDL}^{-}_{\mathrm{sf}}$

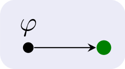
$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle \rightarrow \rangle\, \varphi$$



("Next φ")

$$\mid \langle \leftarrow \rangle\, \varphi$$

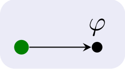


("Yesterday φ")

A simple modal logic for MSCs: $\mathrm{PDL}_{\mathrm{sf}}^{-}$

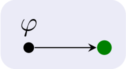
$$\varphi,\psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle\rightarrow\rangle\,\varphi$$



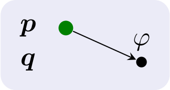
("Next φ")

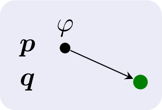
$$\mid \langle\leftarrow\rangle\,\varphi$$



("Yesterday φ")

$$\mid \langle\lhd_{p,q}\rangle\,\varphi \mid \langle\lhd_{q,r}^{-1}\rangle\,\varphi$$

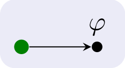




A simple modal logic for MSCs: $\mathrm{PDL}_{\mathrm{sf}}^{-}$

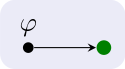
$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle \rightarrow \rangle\, \varphi$$



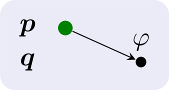
("Next φ")

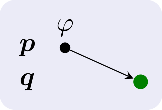
$$\mid \langle \leftarrow \rangle\, \varphi$$



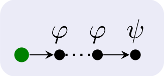
("Yesterday φ")

$$\mid \langle \lhd_{p,q} \rangle\, \varphi \mid \langle \lhd_{q,r}^{-1} \rangle\, \varphi$$





$$\mid \langle \xrightarrow{\varphi} \rangle\, \psi$$

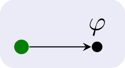


("φ Until ψ")

A simple modal logic for MSCs: $\mathrm{PDL}_{\mathrm{sf}}^{-}$

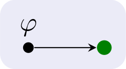
$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle \rightarrow \rangle\, \varphi$$



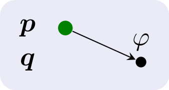
("Next φ")

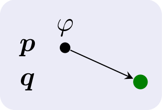
$$\mid \langle \leftarrow \rangle\, \varphi$$



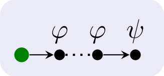
("Yesterday φ")

$$\mid \langle \lhd_{p,q} \rangle\, \varphi \mid \langle \lhd_{q,r}^{-1} \rangle\, \varphi$$



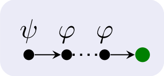


$$\mid \langle \xrightarrow{\varphi} \rangle\, \psi$$



("φ Until ψ")

$$\mid \langle \xleftarrow{\varphi} \rangle\, \psi$$

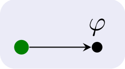


("φ Since ψ")

A simple modal logic for MSCs: $\mathrm{PDL}_{\mathsf{sf}}^{-}$

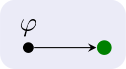
$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$\mid \langle \rightarrow \rangle\, \varphi$



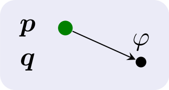
("Next φ")

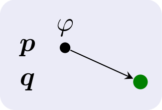
$\mid \langle \leftarrow \rangle\, \varphi$



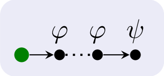
("Yesterday φ")

$\mid \langle \lhd_{p,q} \rangle\, \varphi \mid \langle \lhd_{q,r}^{-1} \rangle\, \varphi$



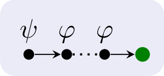


$\mid \langle \xrightarrow{\varphi} \rangle\, \psi$



("φ Until ψ")

$\mid \langle \xleftarrow{\varphi} \rangle\, \psi$

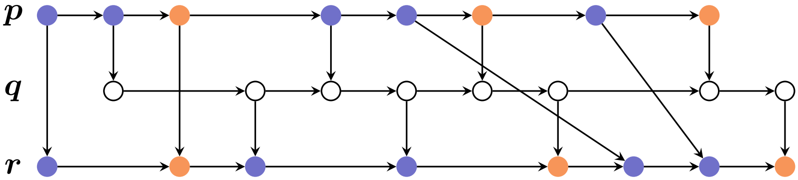


("φ Since ψ")

$\mid \langle \mathsf{jump}_{p,q} \rangle\, \varphi$

Examples

$P = \{p, q, r\}$, $\Sigma = \{\bullet, \bullet, \circ\}$

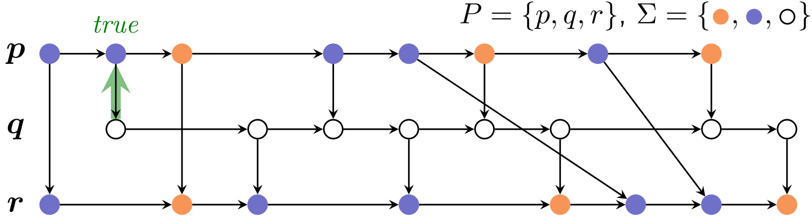


"A receive from p to q is immediately followed by a send to r":

$$\boxed{\langle \lhd_{p,q}^{-1} \rangle\, true} \implies \boxed{\langle \rightarrow \rangle \langle \lhd_{p,q} \rangle\, true}$$

Examples

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$

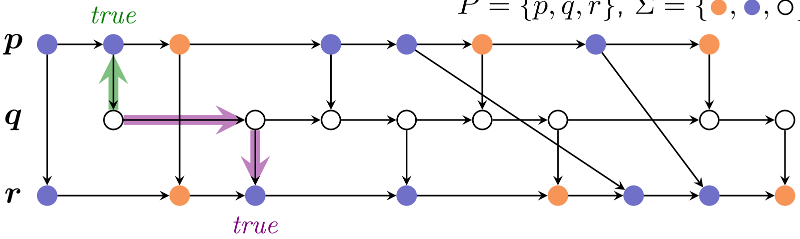


"A receive from p to q is immediately followed by a send to r":

$$\langle \lhd_{p,q}^{-1} \rangle \; true \implies \langle \rightarrow \rangle \langle \lhd_{p,q} \rangle \; true$$

Examples

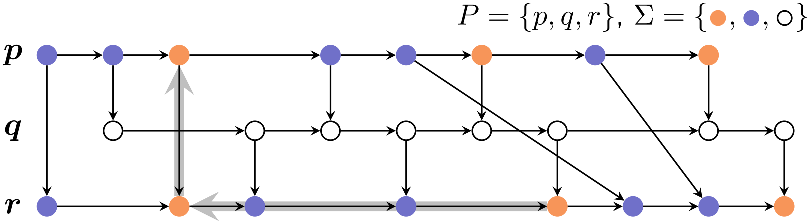
$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$



"A receive from p to q is immediately followed by a send to r":

$$\langle \lhd_{p,q}^{-1} \rangle \, true \implies \langle \rightarrow \rangle \, \langle \lhd_{p,q} \rangle \, true$$

Examples



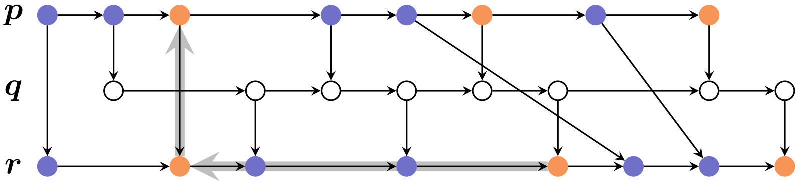
"A receive from p to q is immediately followed by a send to r":

$$\boxed{\langle \lhd_{p,q}^{-1} \rangle\, \mathit{true}} \implies \boxed{\langle \rightarrow \rangle \langle \lhd_{p,q} \rangle\, \mathit{true}}$$

"The latest send from p to r is labeled $\bullet$":

Examples

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$



"A receive from p to q is immediately followed by a send to r":

$$\langle \lhd^{-1}_{p,q} \rangle\, true \implies \langle \rightarrow \rangle \langle \lhd_{p,q} \rangle\, true$$

"The latest send from p to r is labeled ●": (on process r)

$$\langle \lhd^{-1}_{p,r} \rangle\, \bullet \;\vee\; \neg \langle \lhd^{-1}_{p,r} \rangle\, true \;\wedge\; \Big\langle \xleftarrow{\neg \langle \lhd^{-1}_{p,r} \rangle\, true} \Big\rangle \langle \lhd^{-1}_{p,r} \rangle\, \bullet$$

Expressivity of PDL$_{sf}^{-}$

Over words ($|P| = 1$),

$$\begin{tikzcd}
\text{FO} \arrow[r] \arrow[d, bend left] & \text{Finite Automata} \\
\text{LTL} \arrow[u, bend left] \arrow[ur, bend right] &
\end{tikzcd}$$

$= \text{PDL}_{sf}^{-}$

Expressivity of $\mathsf{PDL}^{-}_{\mathsf{sf}}$

Over words ($|P| = 1$),

$$\begin{tikzcd}
\mathsf{FO} \arrow[r] \arrow[d, bend left] & \text{Finite Automata} \\
\mathsf{LTL} \arrow[u, bend left] \arrow[ur, bend right] &
\end{tikzcd}$$

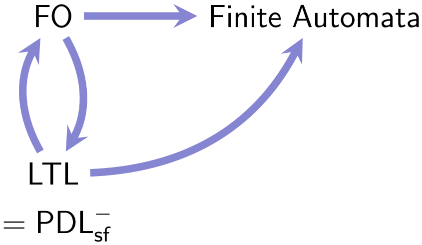
$= \mathsf{PDL}^{-}_{\mathsf{sf}}$

Over general MSCs,

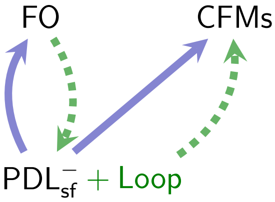
$$\begin{tikzcd}
\mathsf{FO} & \mathsf{CFMs} \\
\mathsf{PDL}^{-}_{\mathsf{sf}} \arrow[u, bend left] \arrow[ur] &
\end{tikzcd}$$

Expressivity of $\mathsf{PDL}_{\mathsf{sf}}^{-}$

Over words ($|P| = 1$),



Over general MSCs,



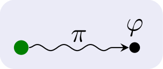
Star-free Propositional Dynamic Logic ($\mathrm{PDL}_{\mathsf{sf}}$)

[Fisher-Ladner 1979] (PDL)

State formulas

$$\varphi ::= a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle\pi\rangle\,\varphi$$



Path formulas

$$\pi ::= \rightarrow \mid \leftarrow \mid \lhd_{p,q} \mid \lhd_{p,q}^{-1} \mid \mathsf{jump}_{p,q} \mid \xrightarrow{\varphi} \mid \xleftarrow{\varphi} \mid \pi \cdot \pi$$

Notation: $\langle\alpha_1 \cdot \alpha_2 \cdots \alpha_k\rangle\,\varphi \equiv \langle\alpha_1\rangle\,(\langle\alpha_2\rangle \cdots (\langle\alpha_k\rangle\,\varphi)\cdots)$

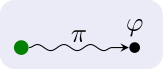
Star-free Propositional Dynamic Logic ($\mathrm{PDL}_{\mathrm{sf}}$)

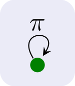
[Fisher-Ladner 1979] (PDL)

State formulas

$$\varphi ::= a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

$$\mid \langle \pi \rangle \, \varphi$$





$$\mid \mathsf{Loop}(\pi)$$

Path formulas

$$\pi ::= \rightarrow \mid \leftarrow \mid \lhd_{p,q} \mid \lhd_{p,q}^{-1} \mid \mathsf{jump}_{p,q} \mid \xrightarrow{\varphi} \mid \xleftarrow{\varphi} \mid \pi \cdot \pi \mid \{\varphi\}?$$

Notation: $\langle \alpha_1 \cdot \alpha_2 \cdots \alpha_k \rangle \, \varphi \equiv \langle \alpha_1 \rangle \, (\langle \alpha_2 \rangle \cdots (\langle \alpha_k \rangle \, \varphi) \cdots)$

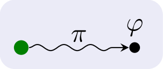
Star-free Propositional Dynamic Logic ($\mathrm{PDL}_{\mathrm{sf}}$)

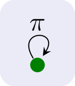
[Fisher-Ladner 1979] (PDL)

State formulas

$\varphi ::= a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$

$\mid \langle\pi\rangle\,\varphi$





$\mid \mathsf{Loop}(\pi)$

Path formulas

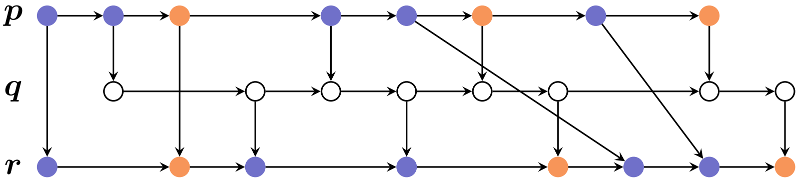
$\pi ::= \rightarrow \mid \leftarrow \mid \lhd_{p,q} \mid \lhd_{p,q}^{-1} \mid \mathsf{jump}_{p,q} \mid \xrightarrow{\varphi} \mid \xleftarrow{\varphi} \mid \pi \cdot \pi \mid \{\varphi\}?$

$\mid \pi \cup \pi \mid \pi \cap \pi \mid \pi^{\mathsf{c}}$

Notation: $\langle\alpha_1 \cdot \alpha_2 \cdots \alpha_k\rangle\,\varphi \equiv \langle\alpha_1\rangle\,(\langle\alpha_2\rangle \cdots (\langle\alpha_k\rangle\,\varphi)\cdots)$

Example

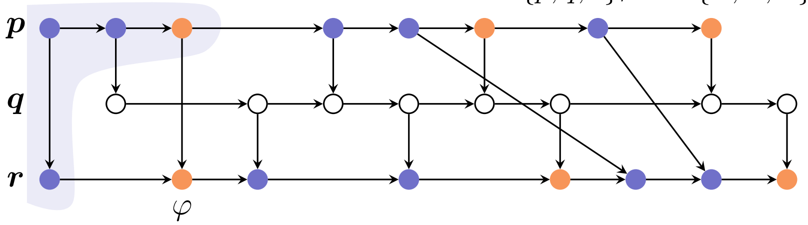
$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$



$\varphi =$ "The latest event on process p is labeled ●"

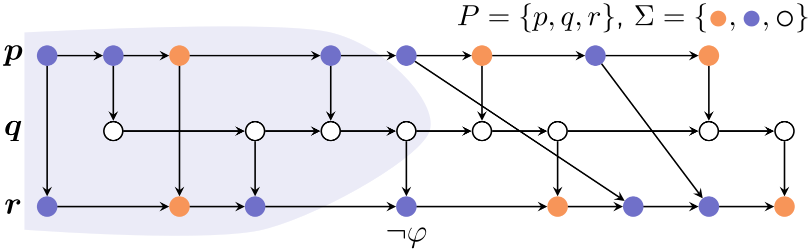
Example

$P = \{p, q, r\}$, $\Sigma = \{●, ●, ○\}$



$\varphi =$ "The latest event on process p is labeled ●"

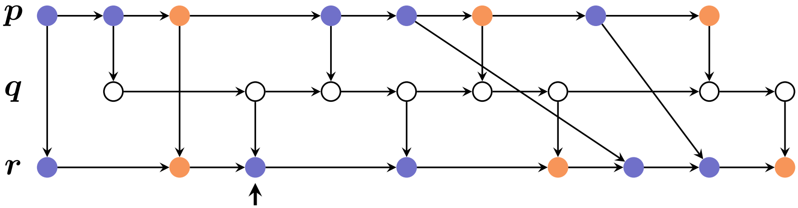
Example



$\varphi =$ "The latest event on process p is labeled ●"

Example

$P = \{p, q, r\}$, $\Sigma = \{●, ●, ○\}$

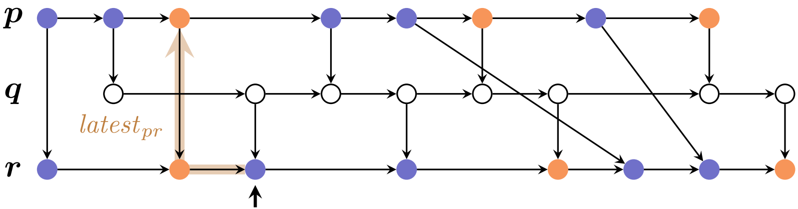


$\varphi =$ “The latest event on process p is labeled ●”

Assume e.g. the current event is a read from channel (q, r).

Example

$P = \{p, q, r\}$, $\Sigma = \{$●, ●, ○$\}$

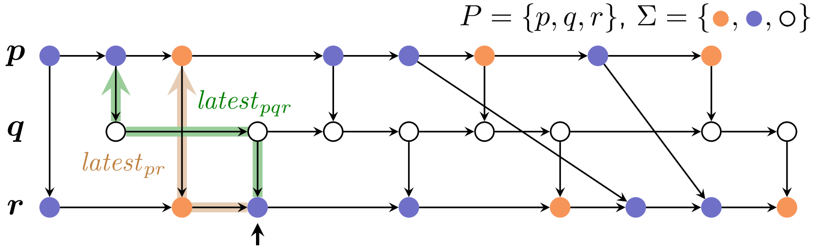


$\varphi =$ "The latest event on process p is labeled ●"

Assume e.g. the current event is a read from channel (q, r).

- path formula $latest_{pr} = \xleftarrow{\neg \langle \lhd_{p,r}^{-1} \rangle \, true} \cdot \lhd_{pr}^{-1}$

Example



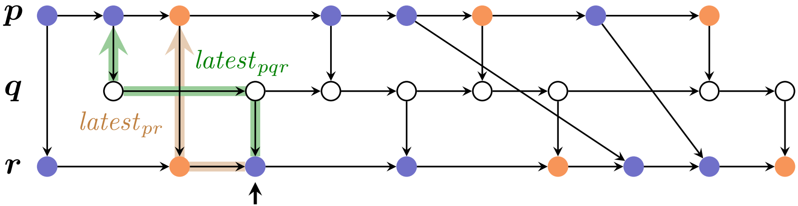
$\varphi =$ "The latest event on process p is labeled ●"

Assume e.g. the current event is a read from channel (q, r).

- path formula $latest_{pr} = \xleftarrow{\neg \langle \lhd_{p,r}^{-1} \rangle \, true} \cdot \lhd_{pr}^{-1}$
- path formula $latest_{pqr} = \lhd_{q,r}^{-1} \cdot \xleftarrow{\neg \langle \lhd_{p,q}^{-1} \rangle \, true} \cdot \lhd_{p,q}^{-1}$

Example

$P = \{p, q, r\}$, $\Sigma = \{\textcolor{orange}{\bullet}, \textcolor{blue}{\bullet}, \circ\}$



$\varphi =$ "The latest event on process p is labeled $\textcolor{orange}{\bullet}$"

Assume e.g. the current event is a read from channel (q, r).

- path formula $latest_{pr} = \xleftarrow{\neg \langle \lhd_{p,r}^{-1} \rangle \, true} \cdot \lhd_{pr}^{-1}$
- path formula $latest_{pqr} = \lhd_{q,r}^{-1} \cdot \xleftarrow{\neg \langle \lhd_{p,q}^{-1} \rangle \, true} \cdot \lhd_{p,q}^{-1}$
- $\varphi = \text{Loop}(latest_{pr} \cdot \{\textcolor{orange}{\bullet}\}? \cdot \xleftarrow{true} \cdot latest_{pqr}^{-1}) \vee$

  $\text{Loop}(latest_{pqr} \cdot \{\textcolor{orange}{\bullet}\}? \cdot \xleftarrow{true} \cdot latest_{pr}^{-1})$

Main results

$$
\begin{tikzcd}
\boxed{\mathsf{FO}[\lhd, \rightarrow, \leq]} \arrow[r, "(1)"] & \boxed{\mathsf{PDL}_{\mathsf{sf}}} \arrow[r, "(2)"] \arrow[dl] & \boxed{\mathsf{CFMs} = \mathsf{EMSO}[\lhd, \rightarrow]} \\
\boxed{\mathsf{FO}^3[\lhd, \rightarrow, \leq]} \arrow[u] & &
\end{tikzcd}
$$

From $\mathrm{PDL}_{\mathrm{sf}}$ to FO

From $\mathrm{PDL}_{\mathrm{sf}}$ to FO

- Any $\mathrm{PDL}_{\mathrm{sf}}$ event formula φ can be transformed into an FO^3 formula $\varphi(x)$ with one free variable.

From PDL$_{sf}$ to FO

- Any PDL$_{sf}$ event formula φ can be transformed into an FO^3 formula $\varphi(x)$ with one free variable.
- Any PDL$_{sf}$ path formula π can be transformed into an FO^3 formula $\pi(x, y)$ with two free variables.

From $\mathrm{PDL}_{\mathrm{sf}}$ to FO

- Any $\mathrm{PDL}_{\mathrm{sf}}$ event formula φ can be transformed into an FO^3 formula $\varphi(x)$ with one free variable.
- Any $\mathrm{PDL}_{\mathrm{sf}}$ path formula π can be transformed into an FO^3 formula $\pi(x, y)$ with two free variables.

$$\mathrm{PDL}_{\mathrm{sf}} \subseteq \mathrm{FO}^3 \subseteq \mathrm{FO}$$

From $\mathrm{PDL}_{\mathsf{sf}}$ to FO

- Any $\mathrm{PDL}_{\mathsf{sf}}$ event formula φ can be transformed into an FO^3 formula $\varphi(x)$ with one free variable.
- Any $\mathrm{PDL}_{\mathsf{sf}}$ path formula π can be transformed into an FO^3 formula $\pi(x, y)$ with two free variables.

$$\mathrm{PDL}_{\mathsf{sf}} \subseteq \mathrm{FO}^3 \subseteq \mathrm{FO} \subseteq \mathrm{PDL}_{\mathsf{sf}}$$

From FO to $\mathsf{PDL}_{\mathsf{sf}}$

Theorem

Any FO formula $\Phi(x_1, \ldots, x_n)$ can be rewritten as

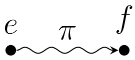
$$\Phi(x_1, \ldots, x_n) \equiv \bigvee \bigwedge \pi(x_i, x_j) \quad \text{where } \pi \in \mathsf{PDL}_{\mathsf{sf}}$$

Proof: by induction.

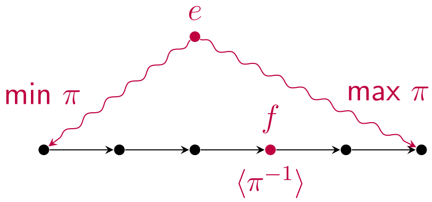
Key Lemma

Key Lemma

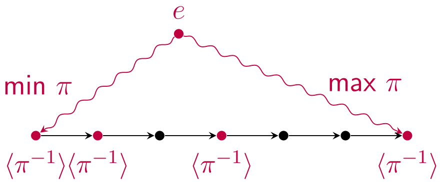
For all $\pi \in \mathrm{PDL_{sf}}$,



iff

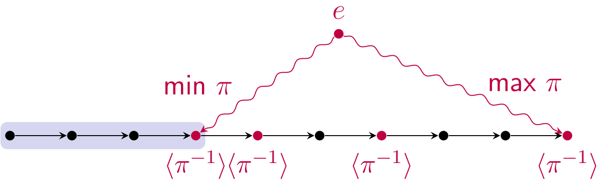


Negation



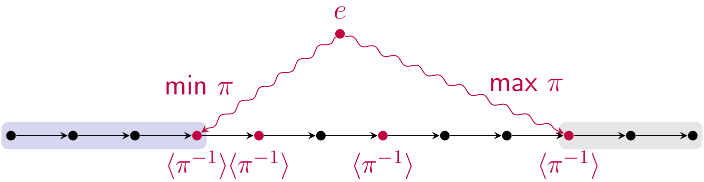
$\pi^{\mathsf{c}} \equiv$

Negation



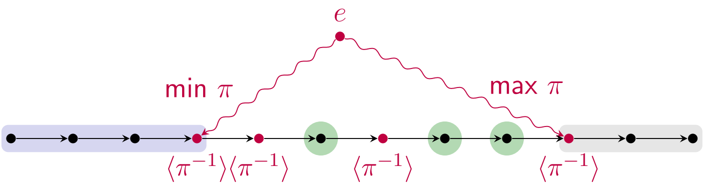
$$\pi^{\mathsf{c}} \equiv \boxed{\mathsf{min}\ \pi \cdot \overset{+}{\longleftarrow}}$$

Negation



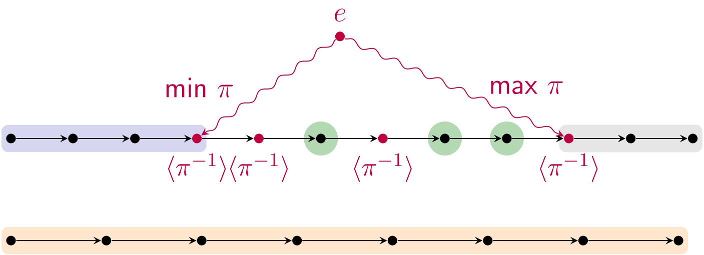
$$\pi^{\mathsf{c}} \equiv \boxed{\text{min } \pi \cdot \xleftarrow{+}} \cup \boxed{\text{max } \pi \cdot \xrightarrow{+}}$$

Negation



$$\pi^{\mathsf{c}} \equiv \boxed{\mathsf{min}\ \pi \cdot \xleftarrow{+}} \cup \boxed{\mathsf{max}\ \pi \cdot \xrightarrow{+}} \cup \boxed{\mathsf{min}\ \pi \cdot \xrightarrow{+} \cdot \{\neg \langle \pi^{-1} \rangle\}?}$$

Negation



$$\pi^{\mathsf{c}} \equiv \boxed{\mathsf{min}\ \pi \cdot \xleftarrow{+}} \cup \boxed{\mathsf{max}\ \pi \cdot \xrightarrow{+}} \cup \boxed{\mathsf{min}\ \pi \cdot \xrightarrow{+} \cdot \{\neg \langle \pi^{-1} \rangle\}?}$$

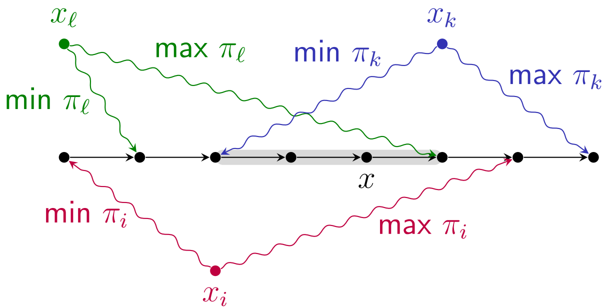
$$\cup \boxed{\bigcup_{p,q} \{\neg \langle \pi \rangle\, q\}? \cdot \mathsf{jump}_{p,q}}$$

Existential quantification

$$\exists x.\ \bigwedge_i \pi_i(x_i, x) \qquad \rightsquigarrow \qquad ?$$

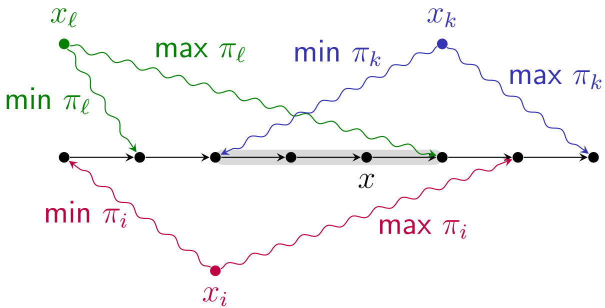
Existential quantification

$\exists x.\ \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



Existential quantification

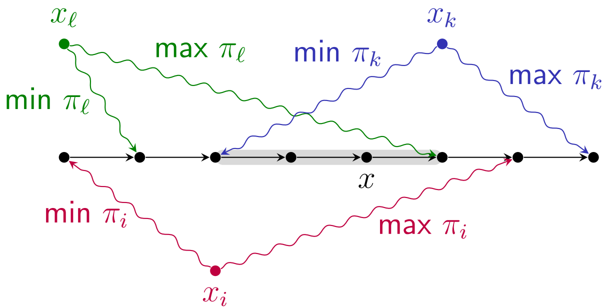
$\exists x.\ \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



Does the intersection of the intervals contain an event satisfying $\psi = \bigwedge_i \langle \pi_i^{-1} \rangle$?

Existential quantification

$\exists x.\ \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_{k,\ell} \left(\begin{array}{ll} & \bigwedge_j ((\text{min } \pi_j) \cdot \xrightarrow{*} \cdot (\text{min } \pi_k)^{-1})(x_i, x_k) \\ \wedge & \bigwedge_j ((\text{max } \pi_\ell) \cdot \xrightarrow{*} \cdot (\text{max } \pi_j)^{-1})(x_\ell, x_i) \\ \wedge & (\pi_k \cdot \{\psi\}? \cdot \pi_\ell^{-1})(x_k, x_\ell) \end{array} \right)$$

From $\mathsf{PDL}_{\mathsf{sf}}$ to CFMs

From $\mathsf{PDL}_{\mathsf{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathsf{PDL}_{\mathsf{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$

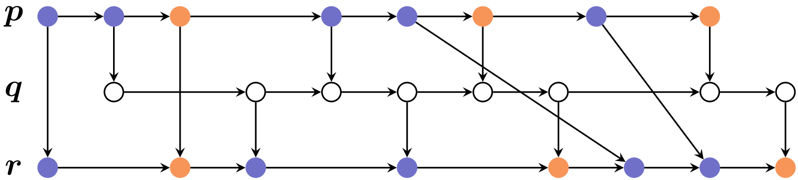
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



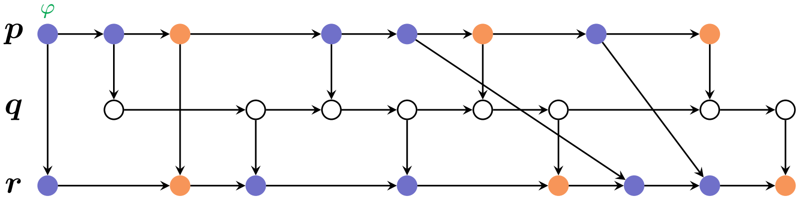
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



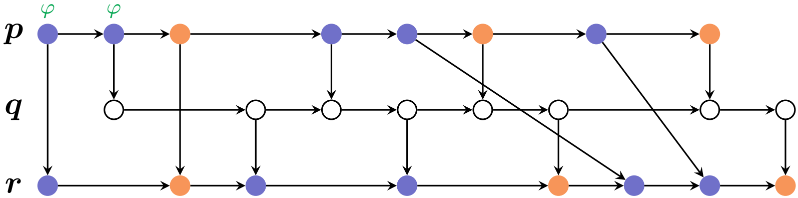
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



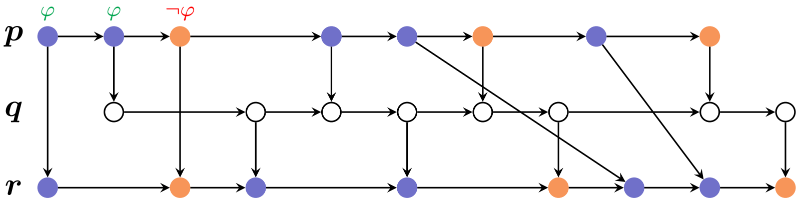
From PDL$_{sf}$ to CFMs

Theorem

Any event formula $\varphi \in \text{PDL}_{sf}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = $ ●



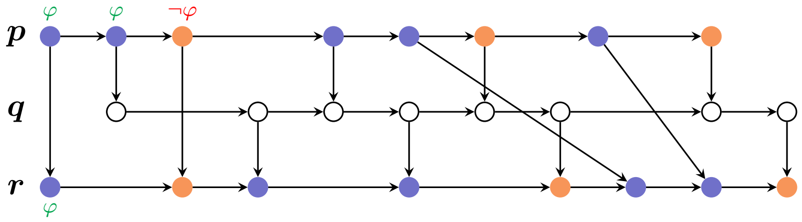
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



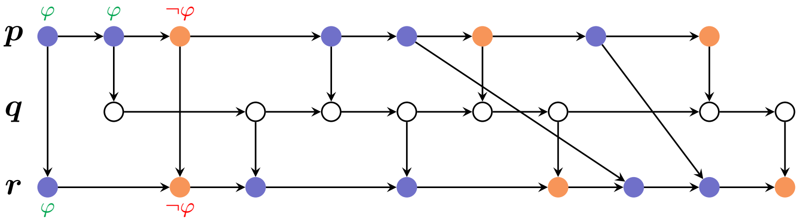
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = $ ●



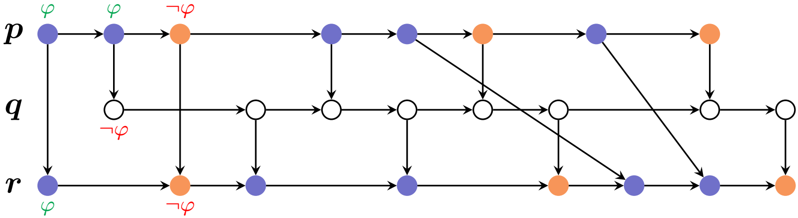
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



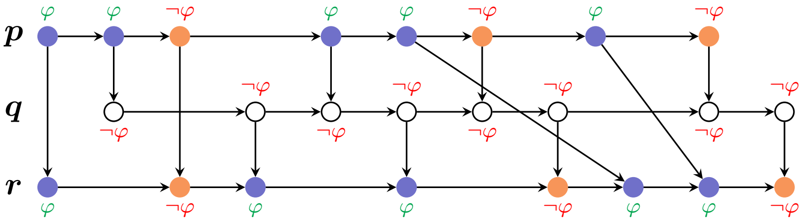
From PDL$_{sf}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{sf}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$



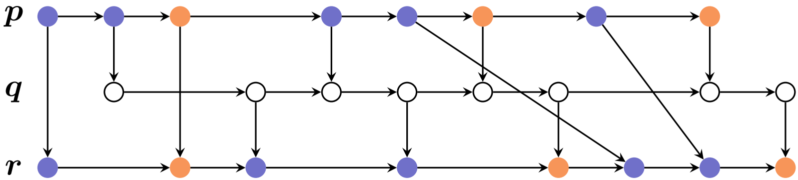
From PDL_{sf} to CFMs

Theorem

Any event formula $\varphi \in PDL_{sf}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$



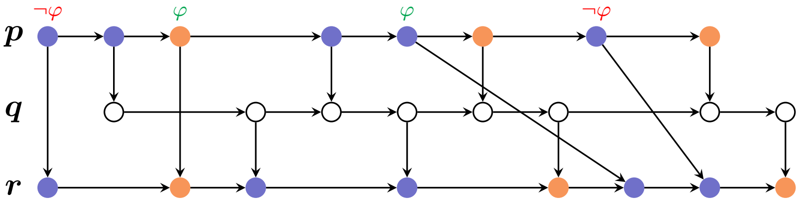
From PDL$_{sf}$ to CFMs

Theorem

Any event formula $\varphi \in$ PDL$_{sf}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$



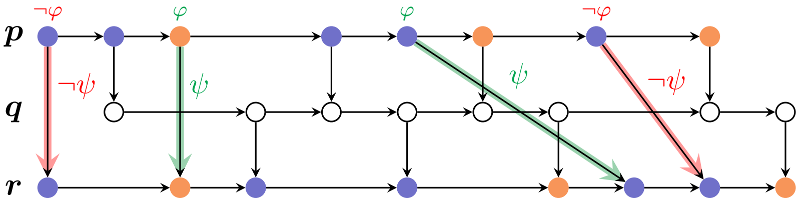
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$



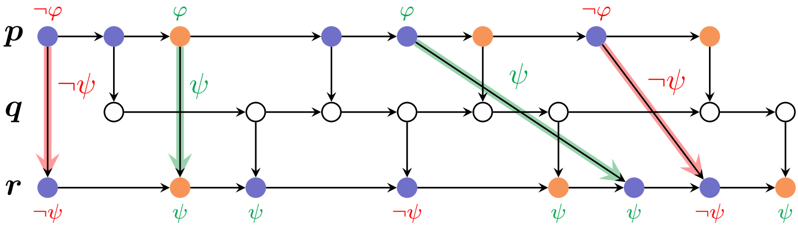
From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$



From $\mathrm{PDL}_{\mathrm{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathrm{PDL}_{\mathrm{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$
- $\cdots$

From $\mathsf{PDL_{sf}}$ to CFMs

Theorem

Any event formula $\varphi \in \mathsf{PDL_{sf}}$ can be translated into a CFM which determines for each event whether φ holds.

Proof: By induction.

- $\varphi = \bullet$
- $\varphi = \langle \lhd_{p,r} \rangle \, \psi$
- $\dots$
- Only difficult case: $\varphi = \mathsf{Loop}(\pi)$

Translation of Loop formulas

We want to determine when $\mathsf{Loop}(\pi)$ hold.

Translation of Loop formulas

We want to determine when $\mathsf{Loop}(\pi)$ hold.

- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models \mathsf{Loop}(\pi)$.

Translation of Loop formulas

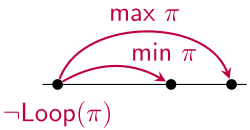
We want to determine when Loop(π) hold.

- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models$ Loop(π).
- Otherwise, three possible cases:

Translation of Loop formulas

We want to determine when $\mathsf{Loop}(\pi)$ hold.

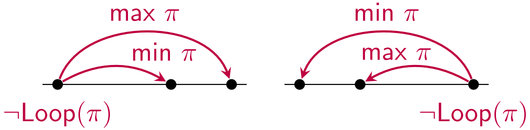
- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models \mathsf{Loop}(\pi)$.
- Otherwise, three possible cases:



Translation of Loop formulas

We want to determine when $\mathsf{Loop}(\pi)$ hold.

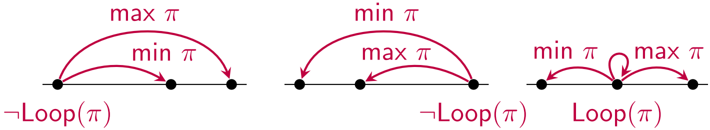
- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models \mathsf{Loop}(\pi)$.
- Otherwise, three possible cases:



Translation of Loop formulas

We want to determine when $\mathsf{Loop}(\pi)$ hold.

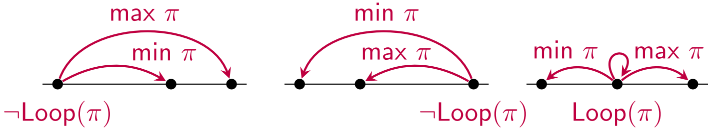
- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models \mathsf{Loop}(\pi)$.
- Otherwise, three possible cases:



Translation of Loop formulas

We want to determine when Loop(π) hold.

- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models$ Loop(π).
- Otherwise, three possible cases:

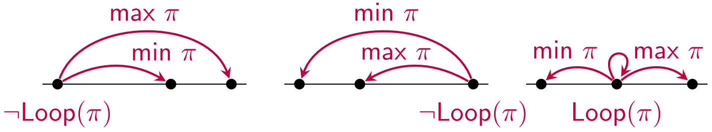


- We can characterize events where a switch occurs using formulas Loop(min $\widetilde{\pi}$) or Loop(max $\widetilde{\pi}$).

Translation of Loop formulas

We want to determine when Loop(π) hold.

- If $e \not\models \langle \pi^{-1} \rangle$, then $e \not\models$ Loop(π).
- Otherwise, three possible cases:



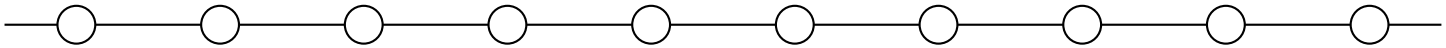
- We can characterize events where a switch occurs using formulas Loop(min $\widetilde{\pi}$) or Loop(max $\widetilde{\pi}$).

First step: translation of formulas Loop(max $\widetilde{\pi}$) into CFMs.
Second step: use this to evaluate Loop(π) from left to right.

CFM for $\varphi = \text{Loop}(\text{max } \pi)$

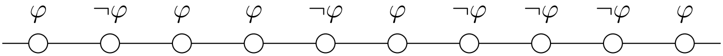
CFM for $\varphi = \text{Loop}(\max \pi)$

- Guess for each event whether φ holds.



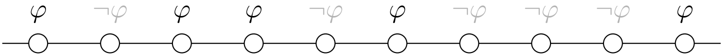
CFM for $\varphi = \mathrm{Loop}(\max \pi)$

- Guess for each event whether φ holds.



CFM for $\varphi = \text{Loop}(\max \pi)$

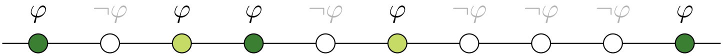
- Guess for each event whether φ holds.



- Check positive guesses:

CFM for $\varphi = \mathrm{Loop}(\max \pi)$

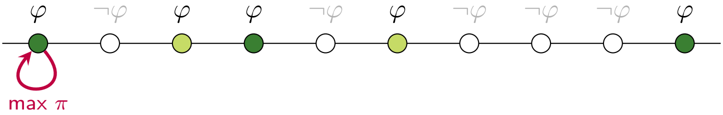
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.

CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

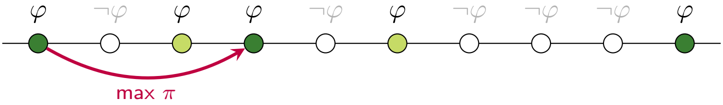
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max\ \pi)$-paths are the same.

CFM for $\varphi = \mathsf{Loop}(\max \pi)$

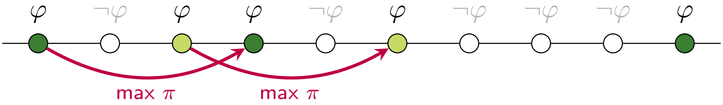
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max \pi)$-paths are the same.

CFM for $\varphi = \text{Loop}(\text{max } \pi)$

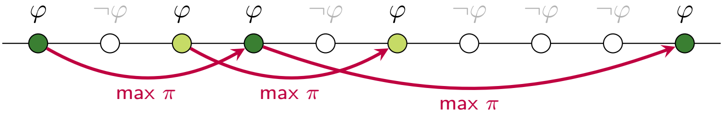
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\text{max } \pi)$-paths are the same.

CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

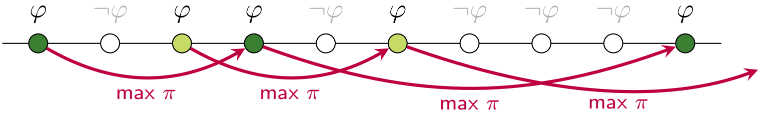
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max\ \pi)$-paths are the same.

CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

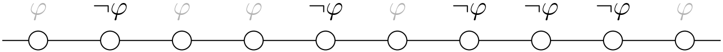
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max\ \pi)$-paths are the same.

CFM for $\varphi = \mathsf{Loop}(\max \pi)$

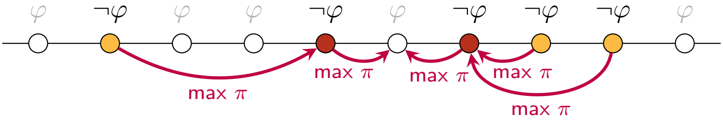
- Guess for each event whether φ holds.



- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max \pi)$-paths are the same.
- Check negative guesses:

CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

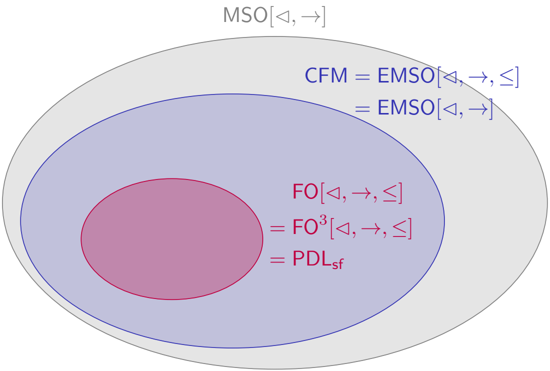
- Guess for each event whether φ holds.



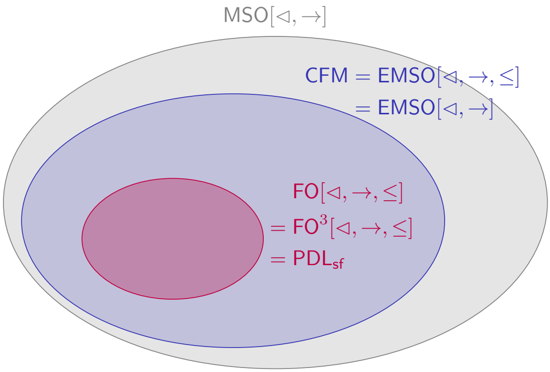
- Check positive guesses:
  - Alternatively assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max\ \pi)$-paths are the same.
- Check negative guesses:
  - Guess a 2-coloring of the $\neg\varphi$-events.
  - Check that the source and target color of $(\max\ \pi)$-paths are distinct.

Conclusion

Conclusion

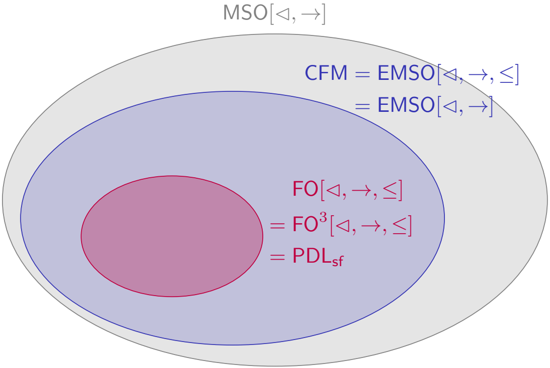

$MSO[\lessdot, \rightarrow]$
$CFM = EMSO[\lessdot, \rightarrow, \leq]$
$= EMSO[\lessdot, \rightarrow]$
$FO[\lessdot, \rightarrow, \leq]$
$= FO^3[\lessdot, \rightarrow, \leq]$
$= PDL_{sf}$

Conclusion



Open question: Is there a temporal logic (with a finite set of modalities) expressively complete for FO over MSCs?

Conclusion



Open question: Is there a temporal logic (with a finite set of modalities) expressively complete for FO over MSCs?

Thank you!