Communicating Finite-State Machines, First-Order Logic, and Star-Free Propositional Dynamic Logic

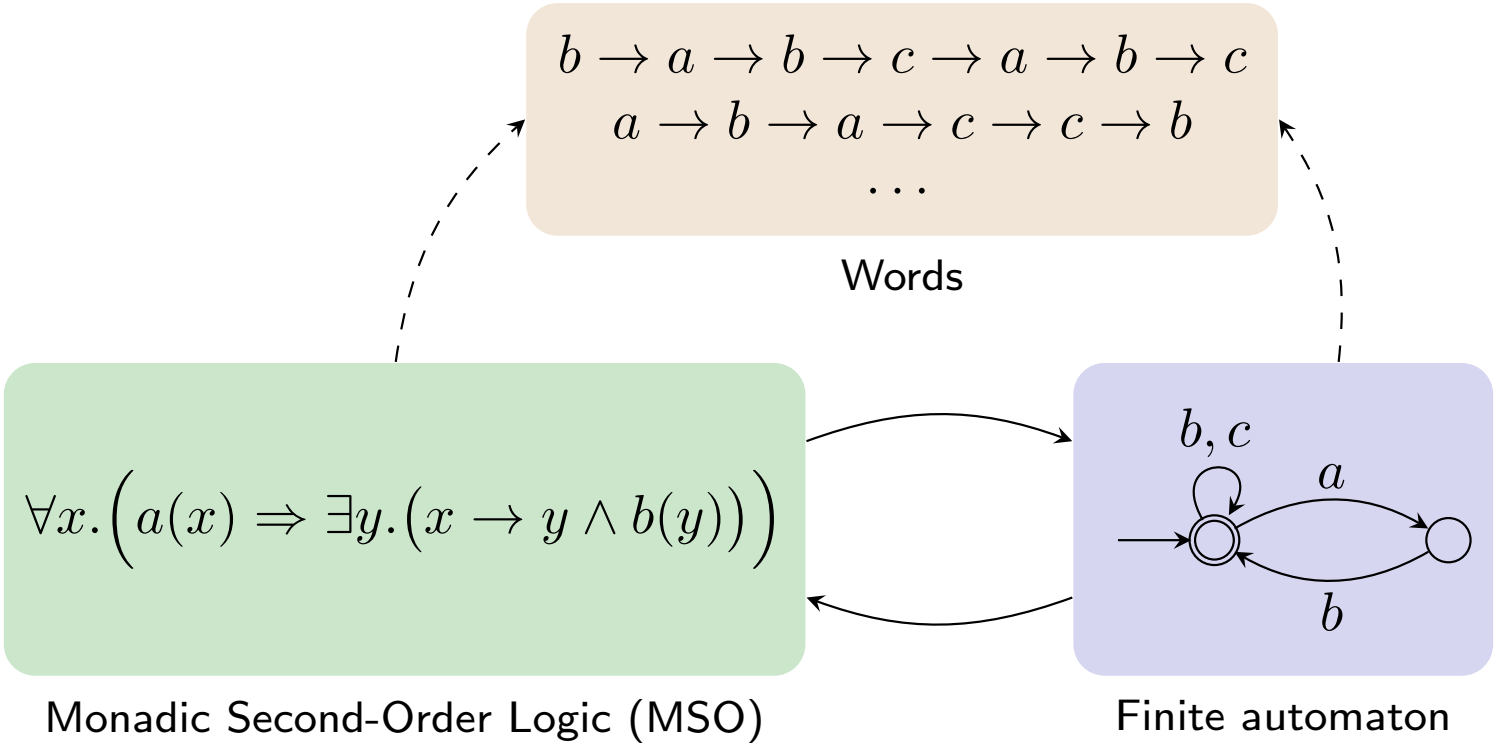
Benedikt Bollig, Marie Fortin, Paul Gastin

CNRS, LSV, ENS Paris-Saclay

CAALM
Chennai Mathematical Institute
January 21, 2019

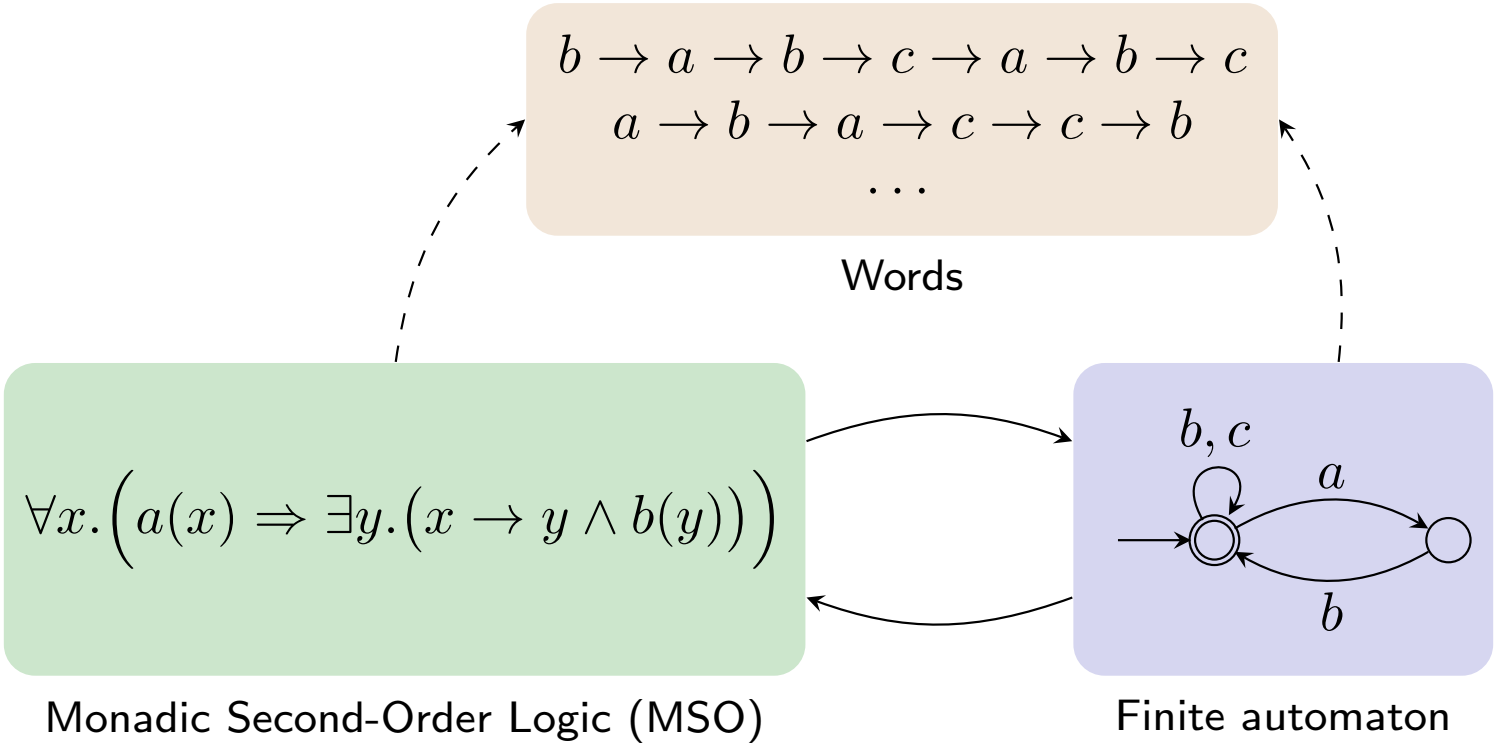
Slides courtesy of Marie Fortin.

Büchi-Elgot-Trakhtenbrot Theorem ('60s)



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

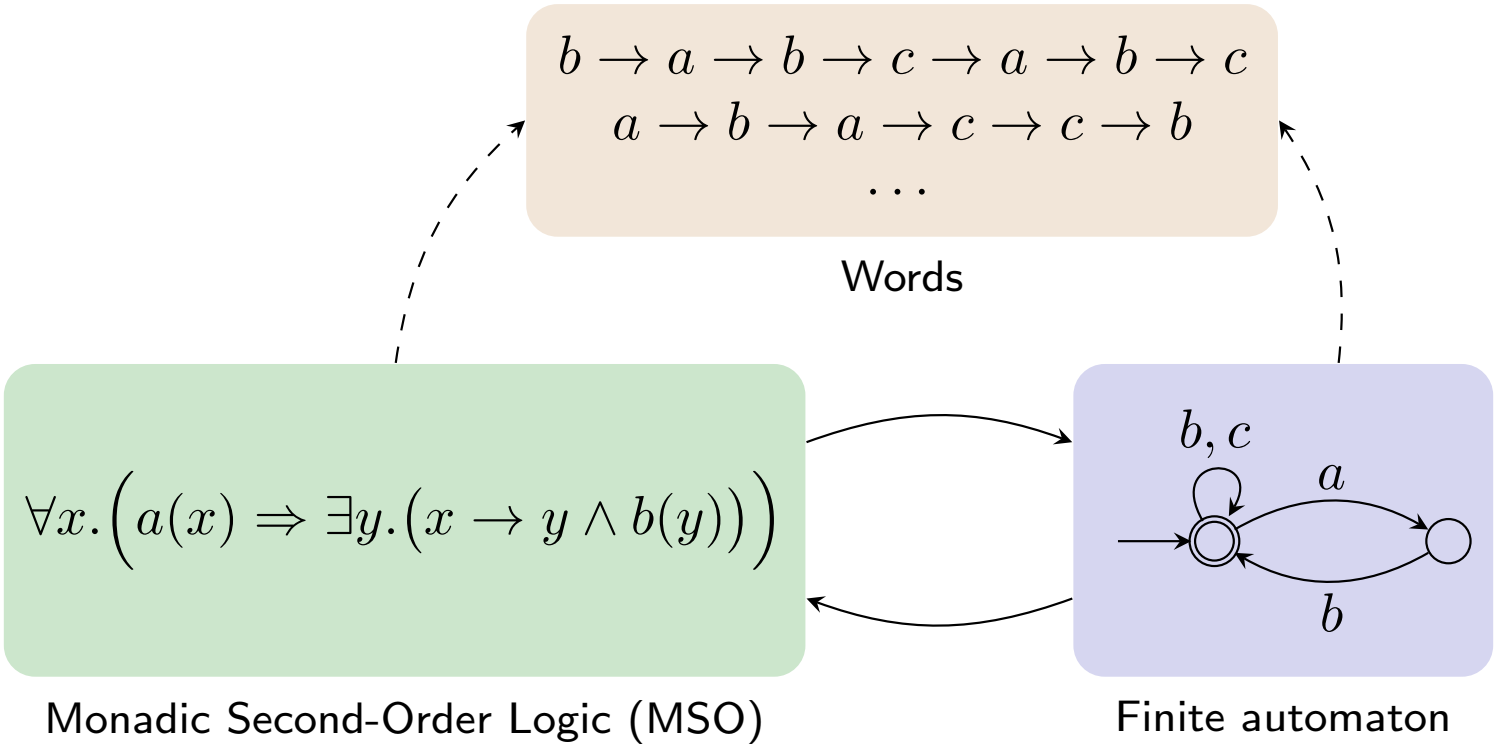
Büchi-Elgot-Trakhtenbrot Theorem ('60s)



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

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

Büchi-Elgot-Trakhtenbrot Theorem ('60s)

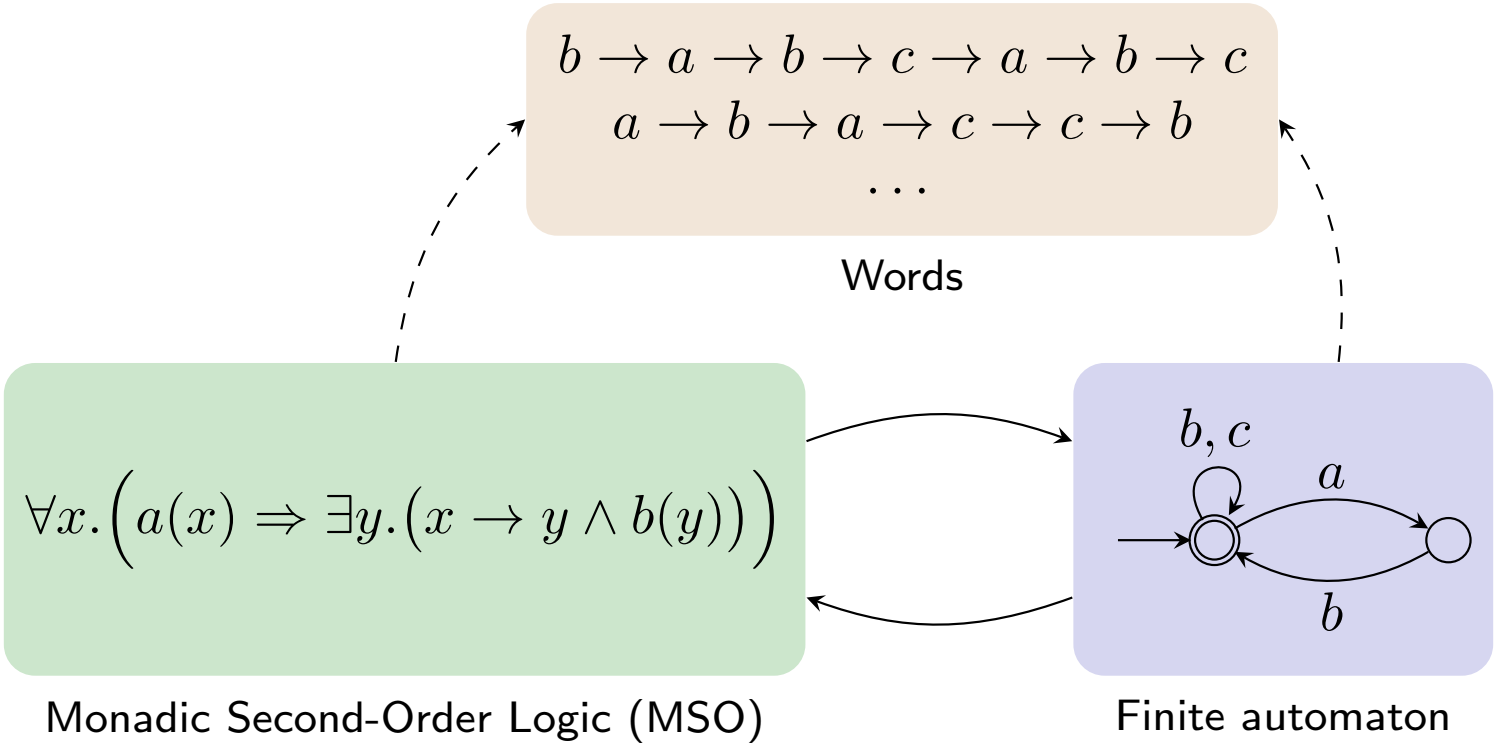


$$\mathsf{MSO}[\to] = \text{Finite Automata} = \mathsf{EMSO}[\to]$$

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

Extended to trees [Thatcher-Wright '68], Mazurkiewicz traces [Thomas '90], nested words [Alur-Madhusudan '04], data words [Bojańczyk et al. '06], weighted automata [Droste-Gastin '05], ...

Büchi-Elgot-Trakhtenbrot Theorem ('60s)



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

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

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

Büchi-Elgot-Trakhtenbrot Theorem ('60s)

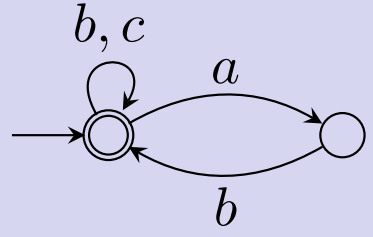
$$b \to a \to b \to c \to a \to b \to c$$
$$a \to b \to a \to c \to c \to b$$
$$\dots$$

Words

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

Monadic Second-Order Logic (MSO)

Synthesis



Finite automaton

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

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

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

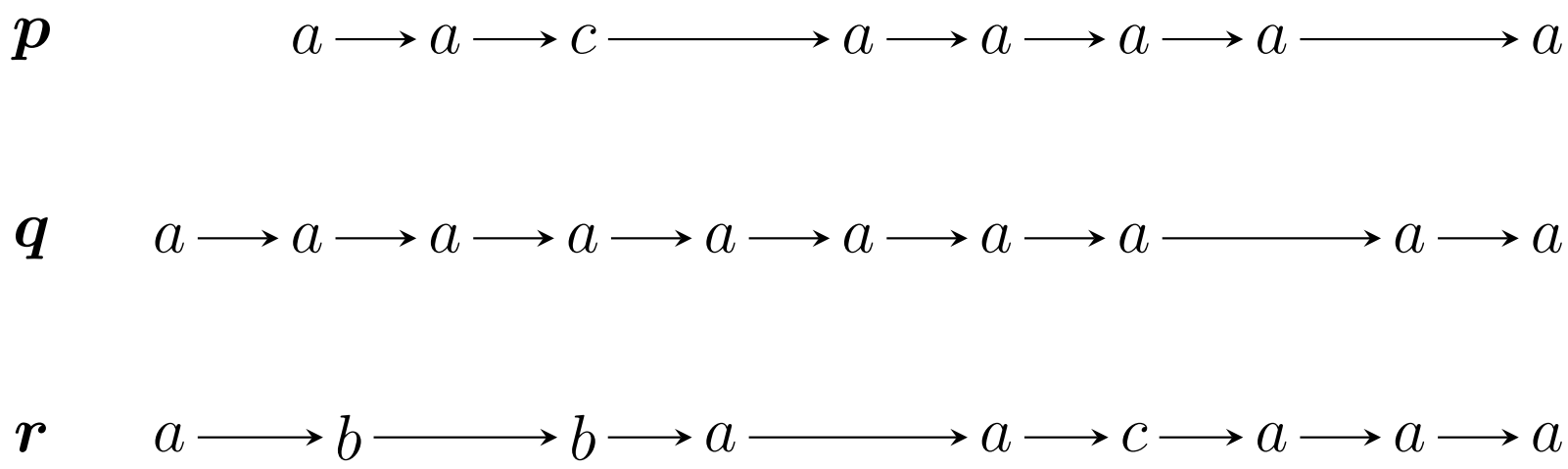
The model

The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)

The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)



The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)
- Reliable unbounded point-to-point FIFO channels

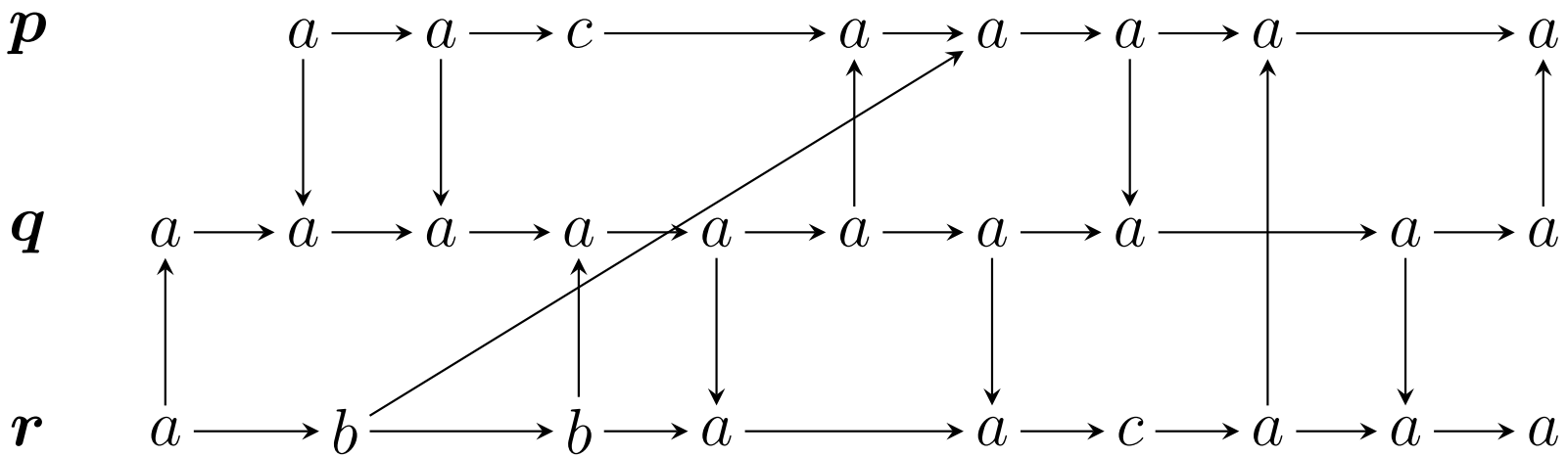
$\boldsymbol{p}$ $\quad a \longrightarrow a \longrightarrow c \longrightarrow\!\!\!\!\longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow\!\!\!\!\longrightarrow a$

$\boldsymbol{q}$ $\quad a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow a \longrightarrow\!\!\!\!\longrightarrow a \longrightarrow a$

$\boldsymbol{r}$ $\quad a \longrightarrow b \longrightarrow\!\!\!\!\longrightarrow b \longrightarrow a \longrightarrow\!\!\!\!\longrightarrow a \longrightarrow c \longrightarrow a \longrightarrow a \longrightarrow a$

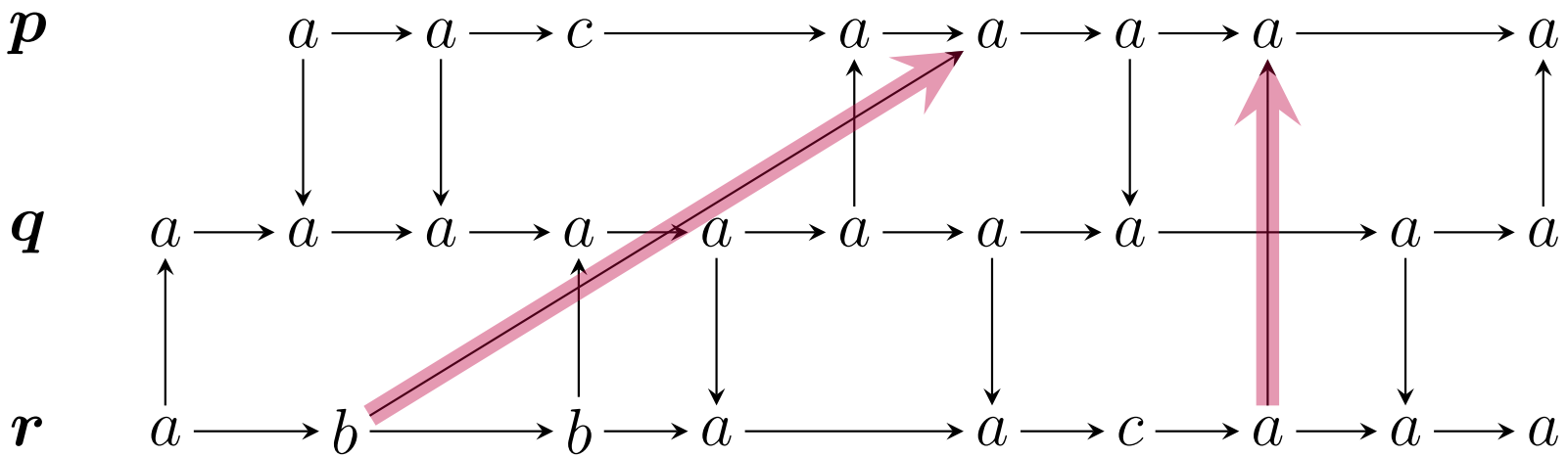
The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)
- Reliable unbounded point-to-point FIFO channels



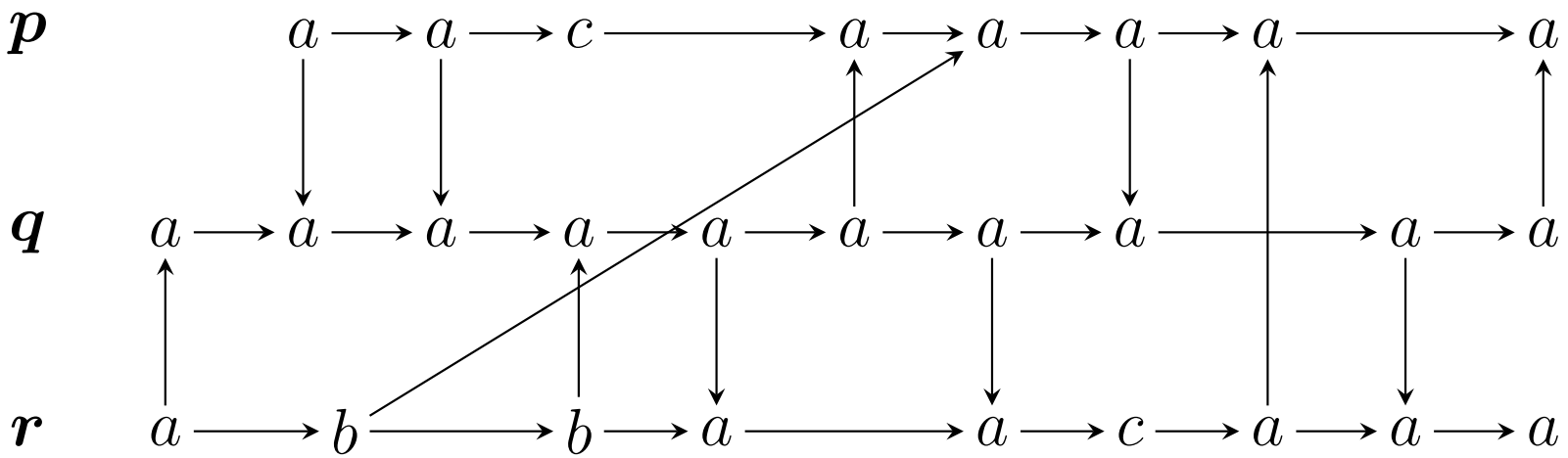
The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)
- Reliable unbounded point-to-point FIFO channels



The model

- Fix finite set of processes and finite alphabet (e.g., $P = \{p, q, r\}$ and $\Sigma = \{a, b, c\}$)
- Reliable unbounded point-to-point FIFO channels
- Partial order semantics: Message Sequence Charts (MSC)



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

Communicating finite-state machines (CFMs)

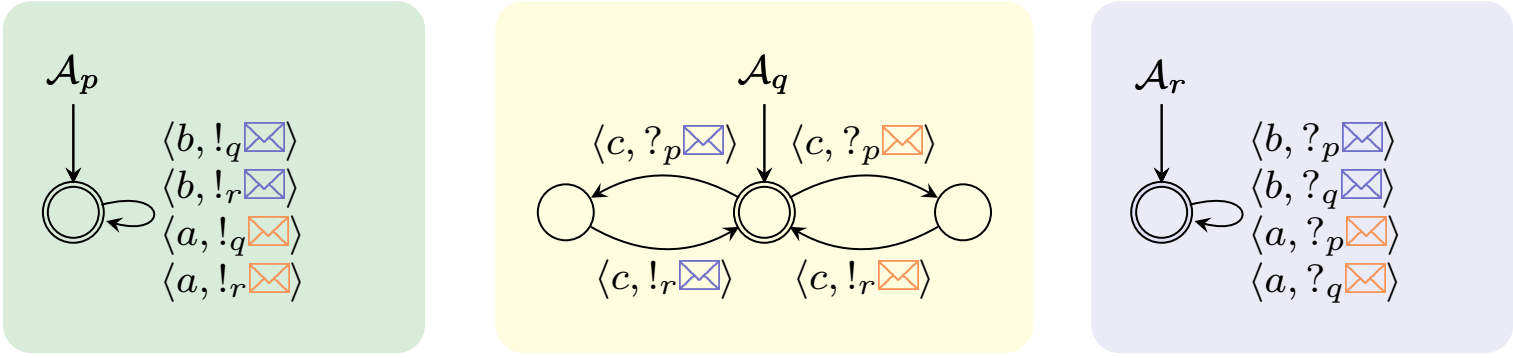
[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$

Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

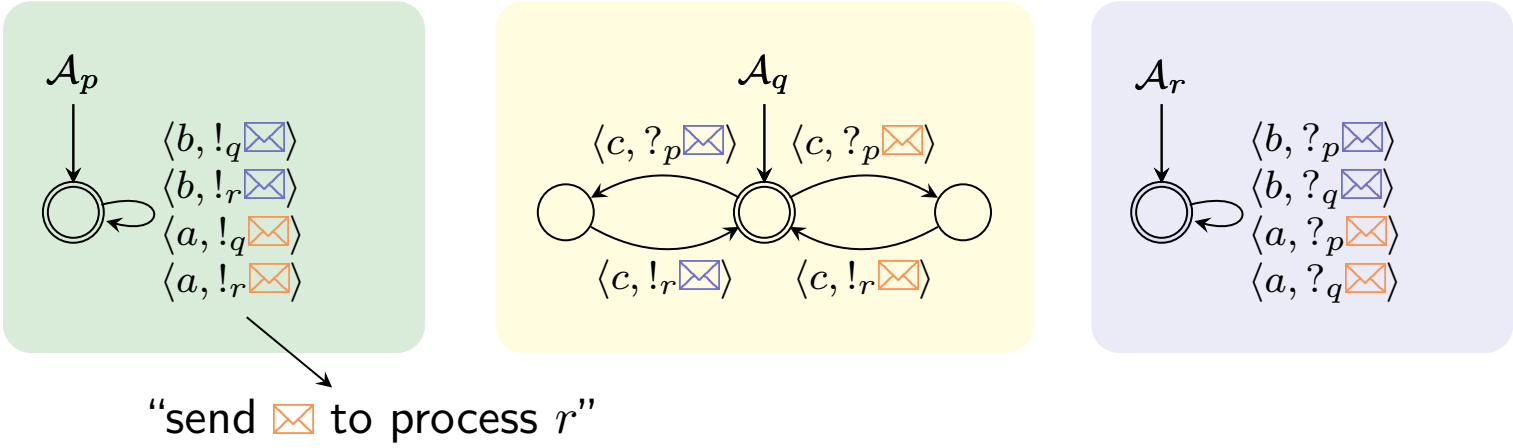
$$P = \{p, q, r\},\ \Sigma = \{a, b, c\}$$



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$

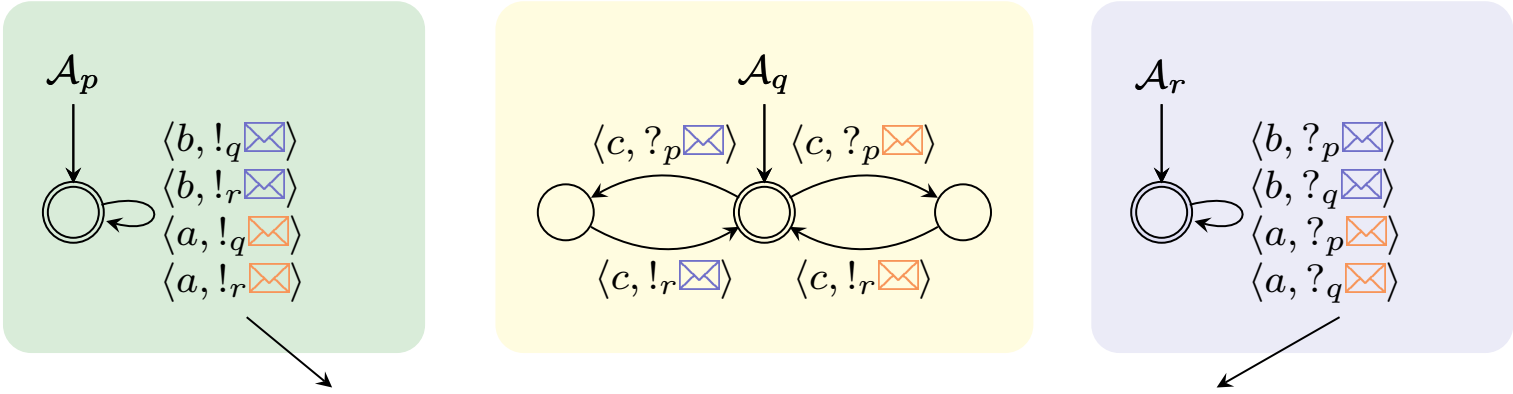


"send ✉ to process r"

Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$

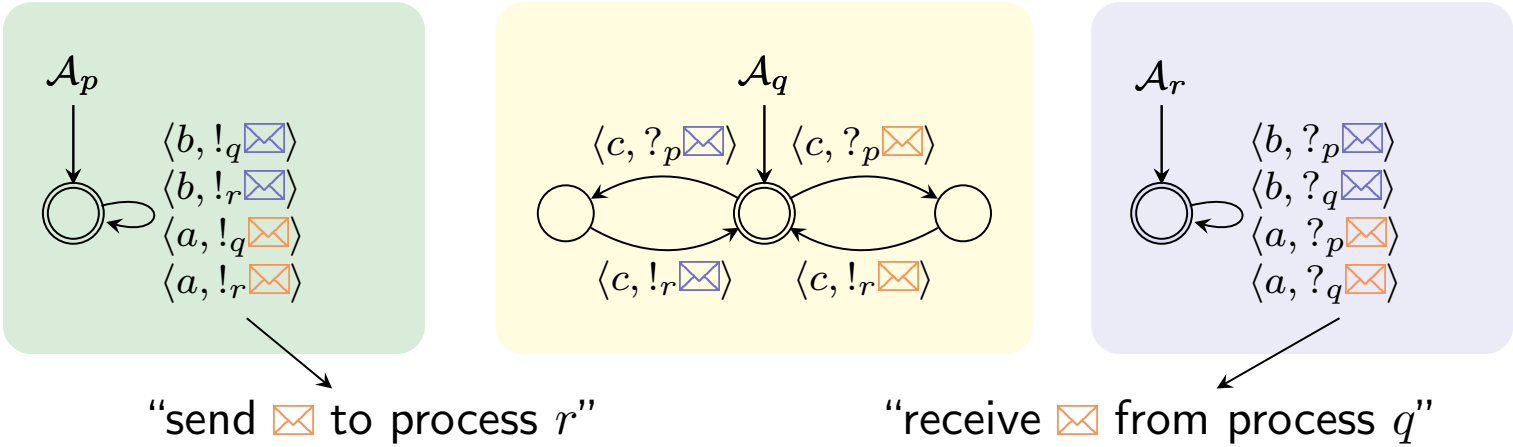


"send ✉ to process r" $\qquad$ "receive ✉ from process q"

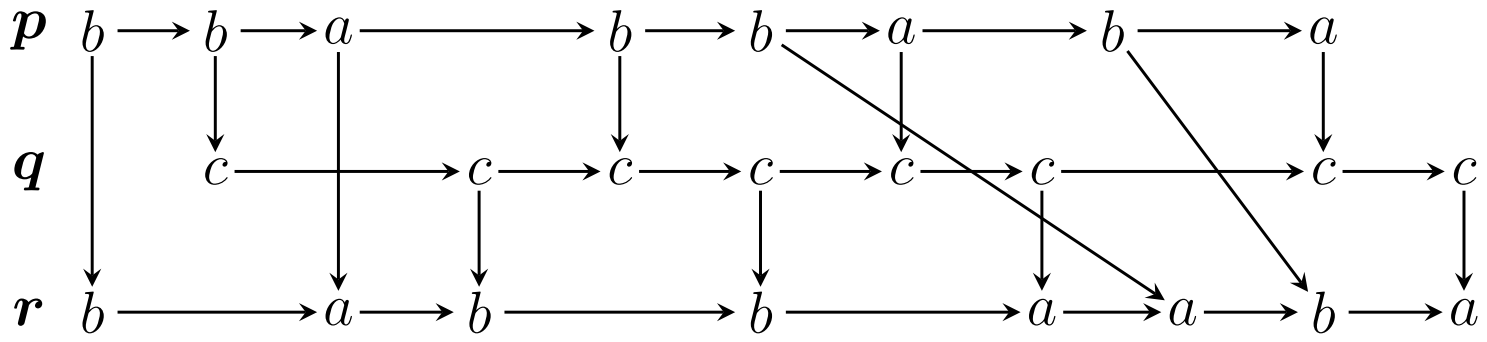
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



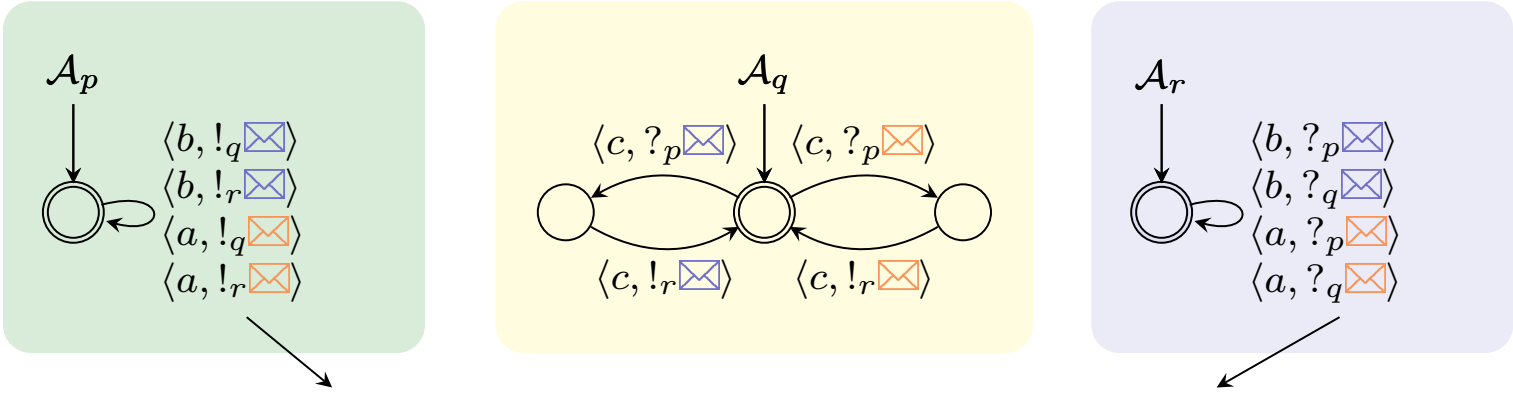
Run of CFMs on MSCs:



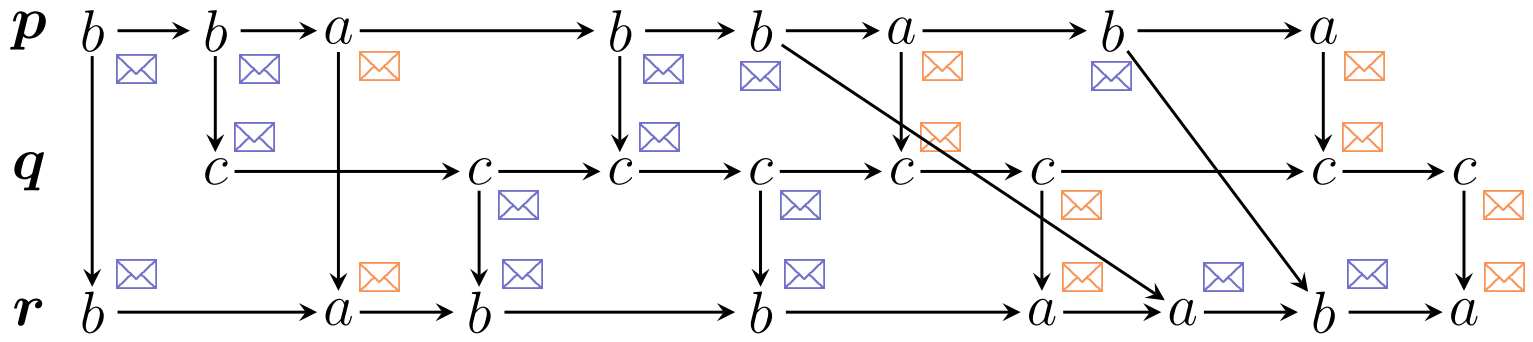
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



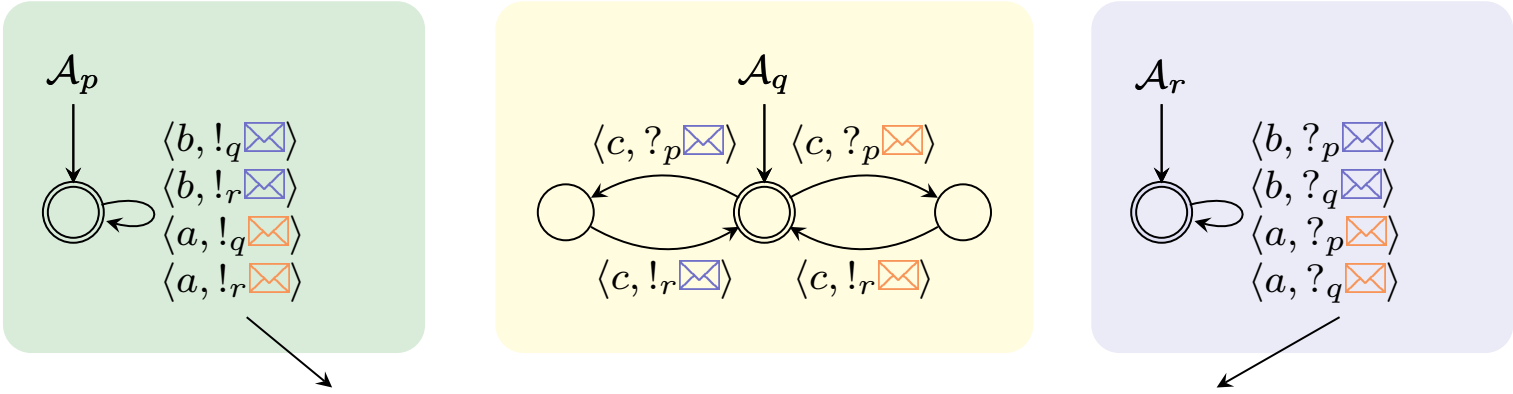
Run of CFMs on MSCs:



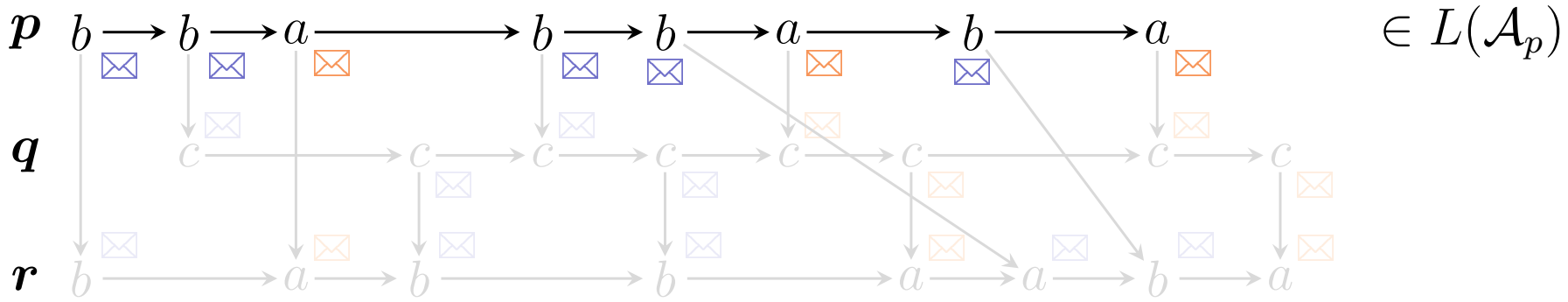
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



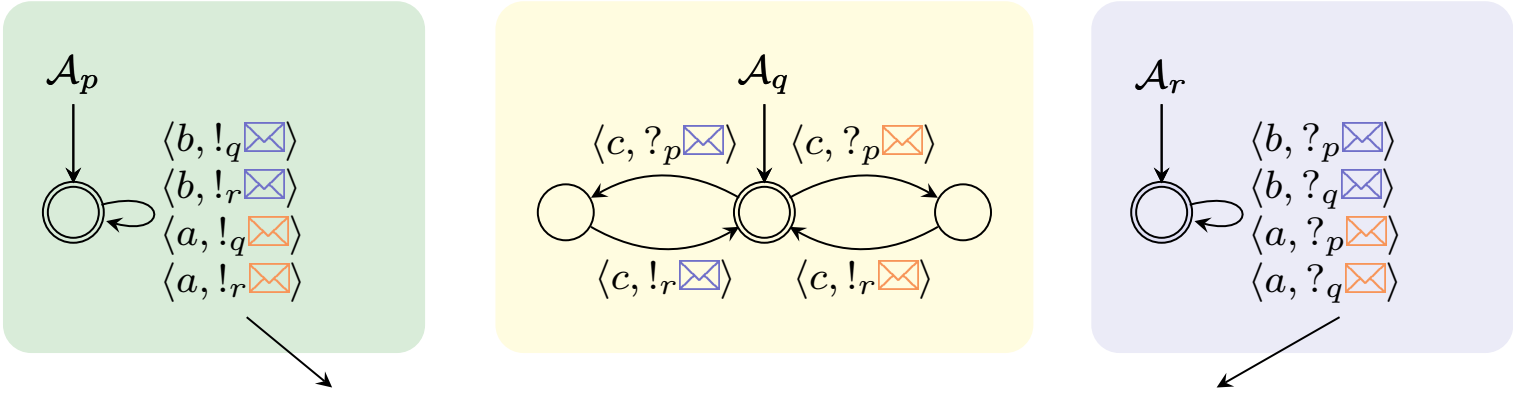
Run of CFMs on MSCs:



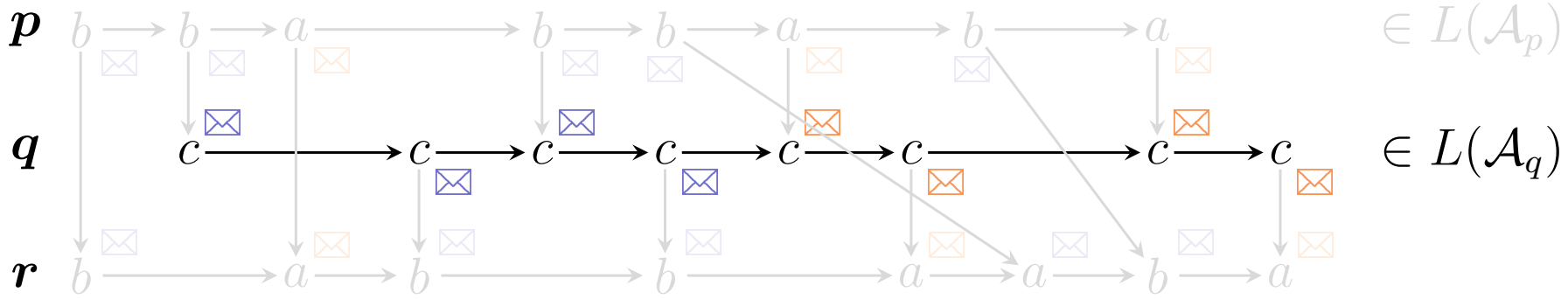
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



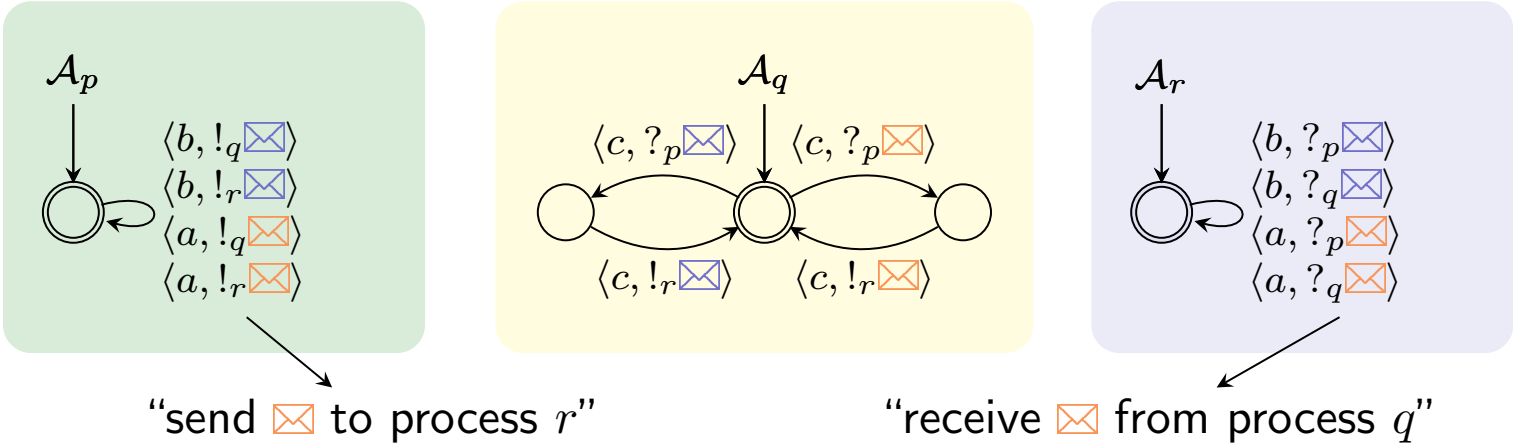
Run of CFMs on MSCs:



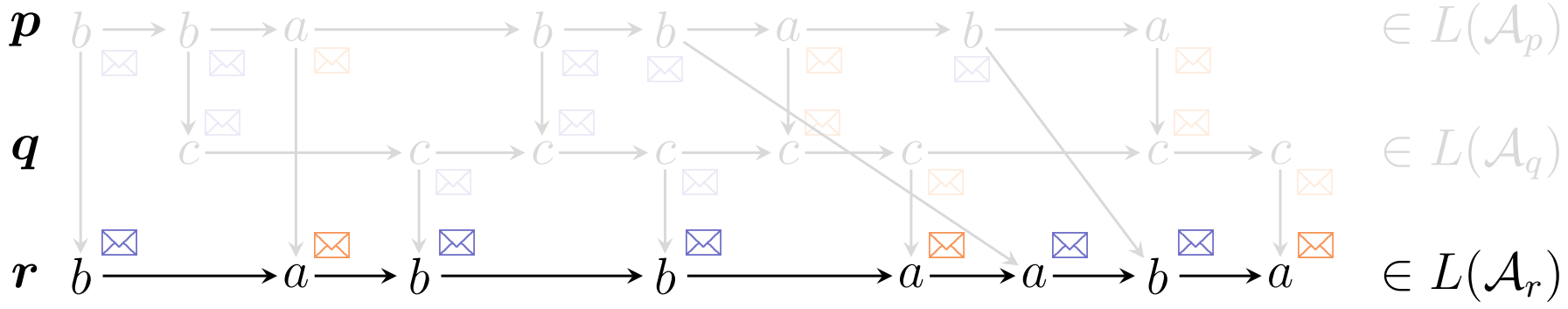
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



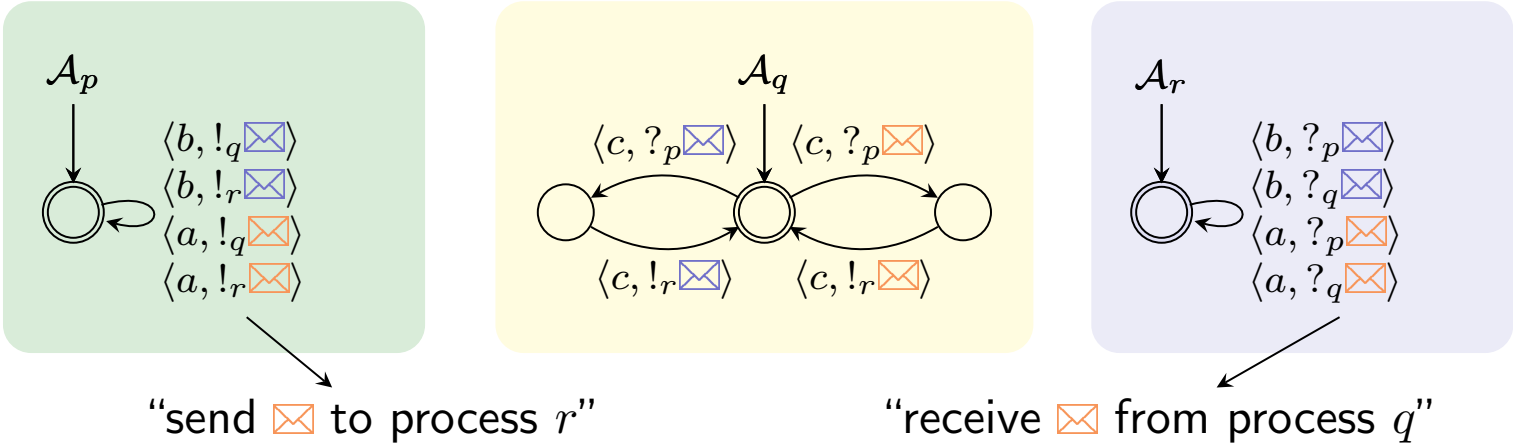
Run of CFMs on MSCs:



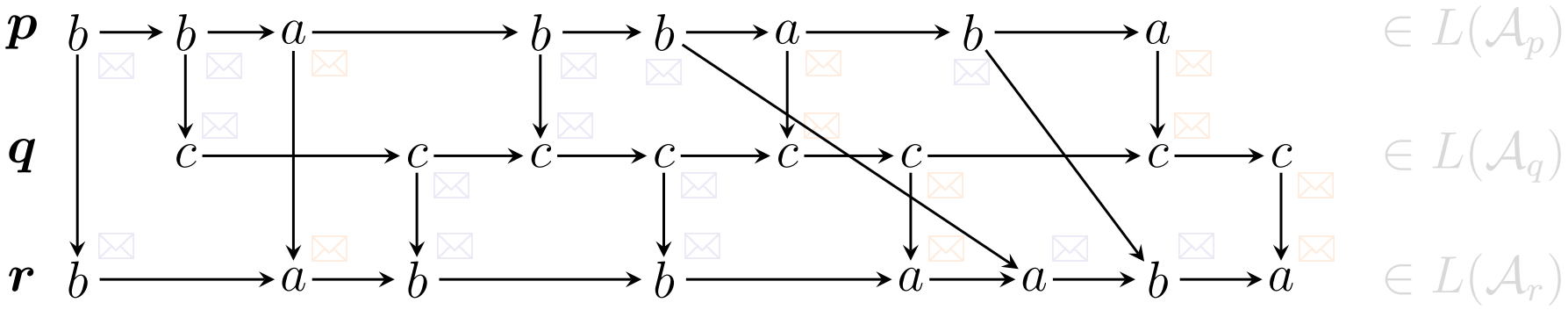
Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



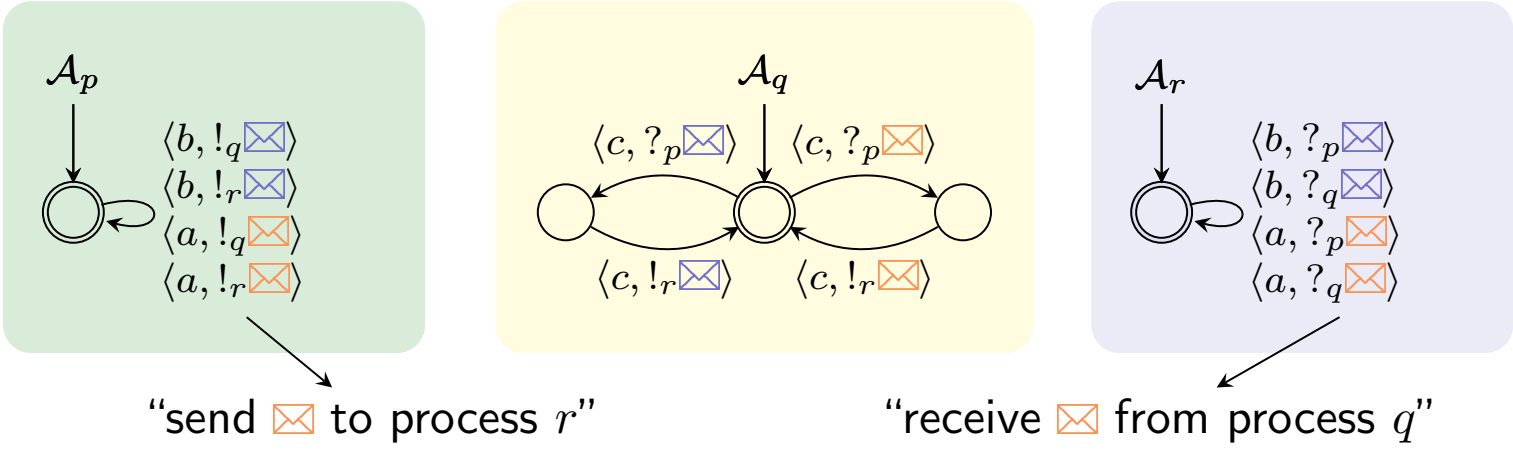
Run of CFMs on MSCs:



Communicating finite-state machines (CFMs)

[Brand–Zafiropulo '83]

$P = \{p, q, r\}$, $\Sigma = \{a, b, c\}$ $\qquad$ $Msg = \{✉, ✉\}$



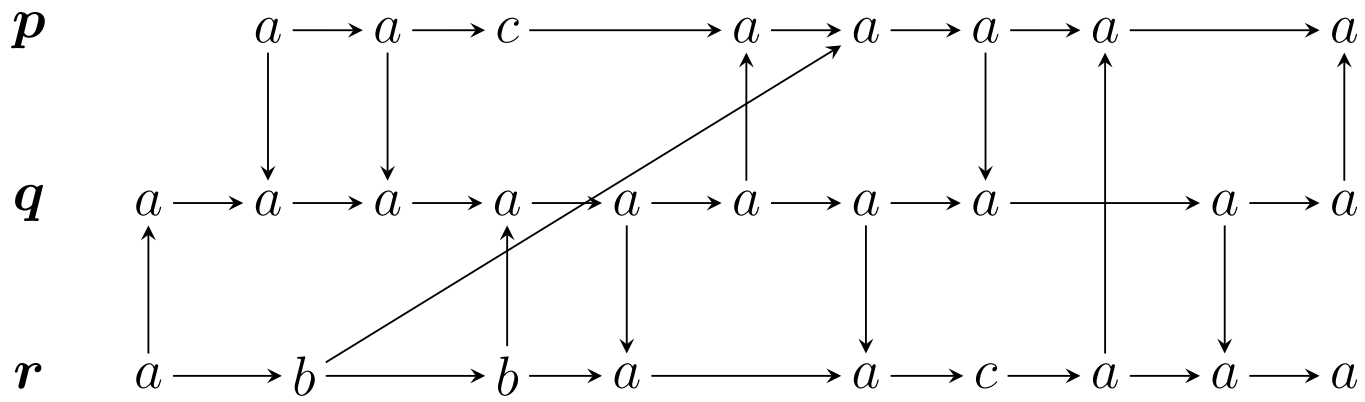
Remarks

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

[Genest-Kuske-Muscholl '07]

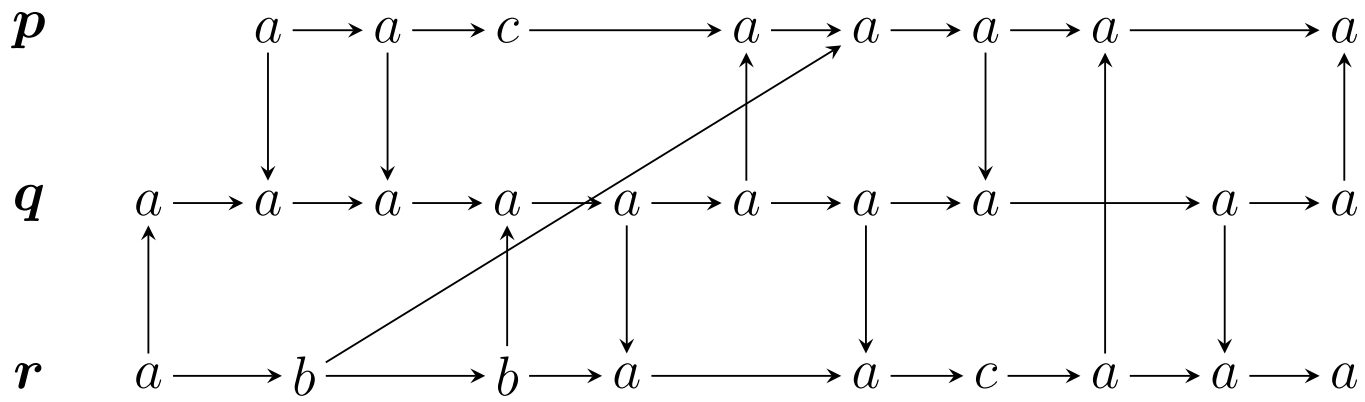
Monadic Second-Order Logic (MSO) over MSCs

$\varphi ::=$



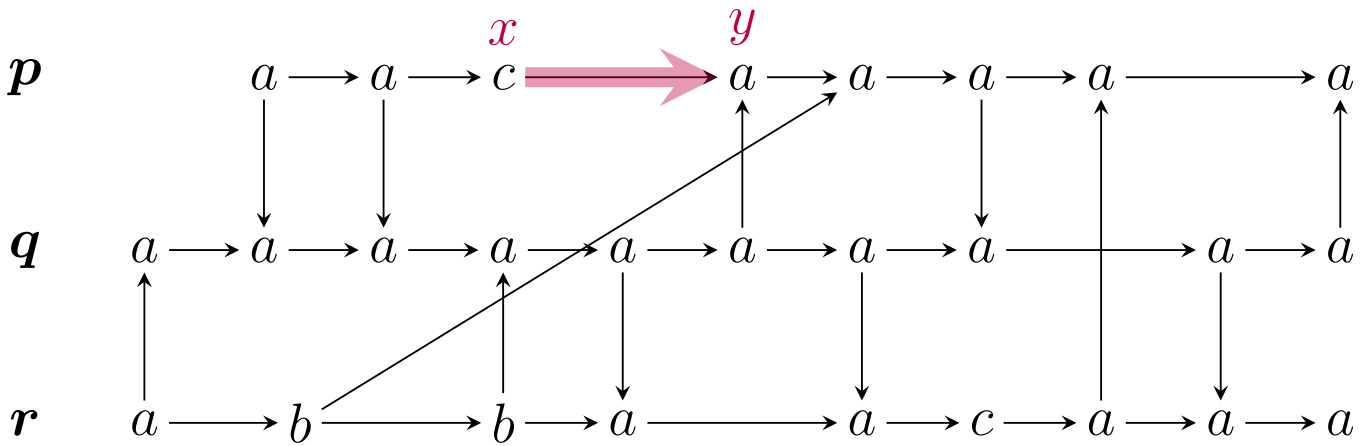
Monadic Second-Order Logic (MSO) over MSCs

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



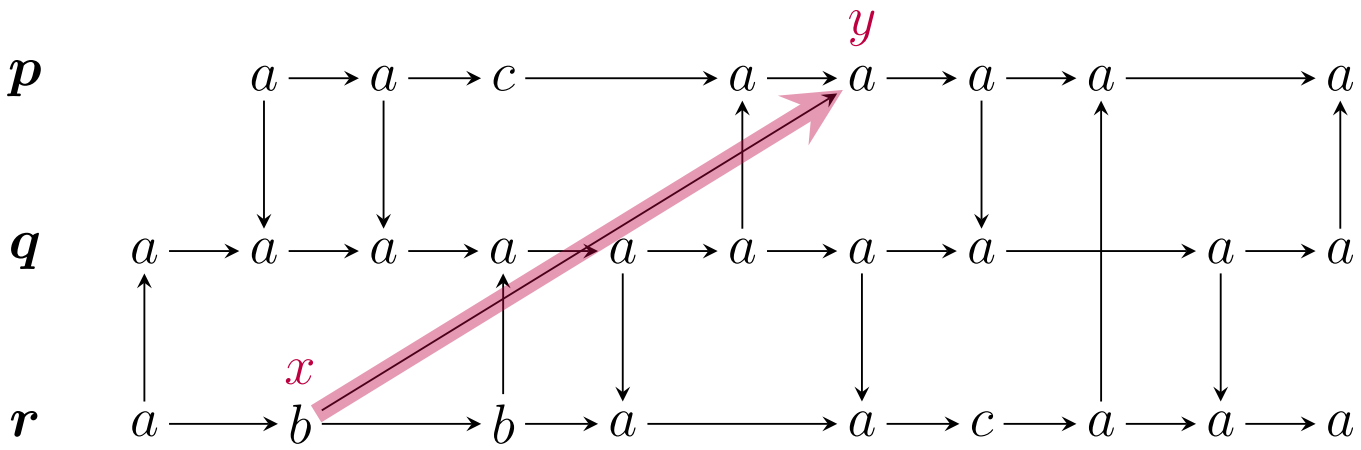
Monadic Second-Order Logic (MSO) over MSCs

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



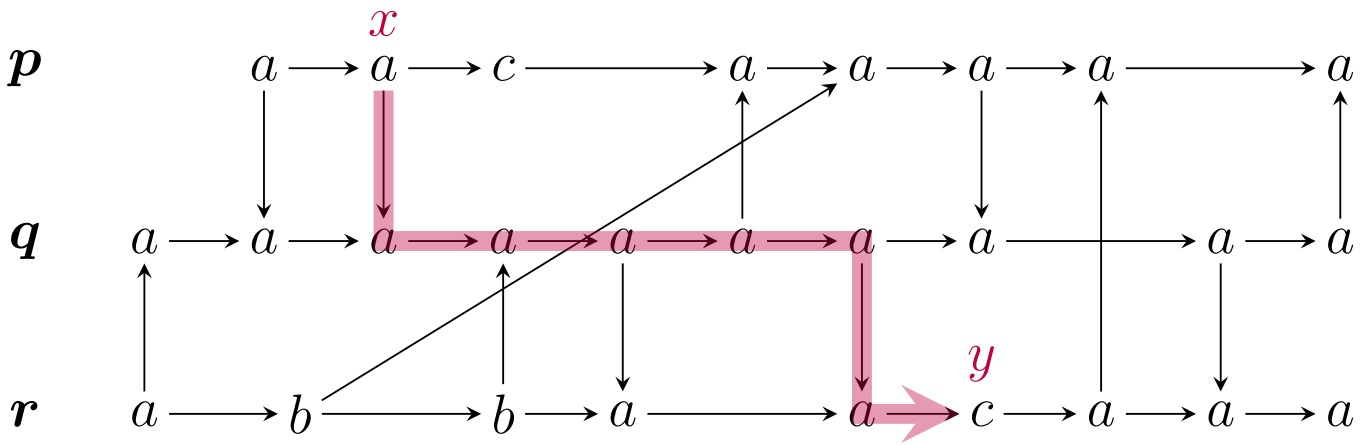
Monadic Second-Order Logic (MSO) over MSCs

$$\begin{aligned} \varphi ::= \quad & a(x) \mid p(x) && \text{label/process of event } x \\ \mid\ & x \to y && \text{process successor} \\ \mid\ & x \searrow y && \text{message relation} \end{aligned}$$



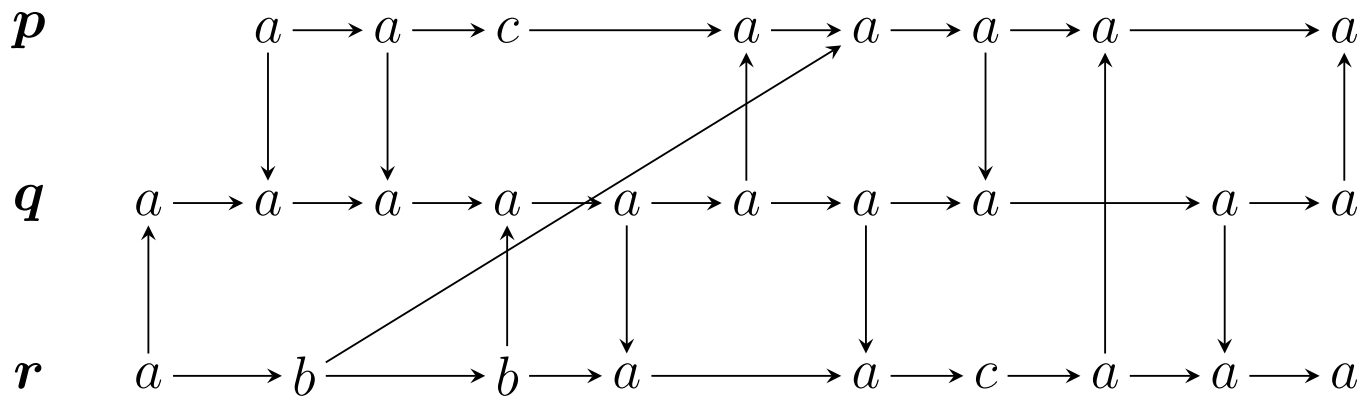
Monadic Second-Order Logic (MSO) over MSCs

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



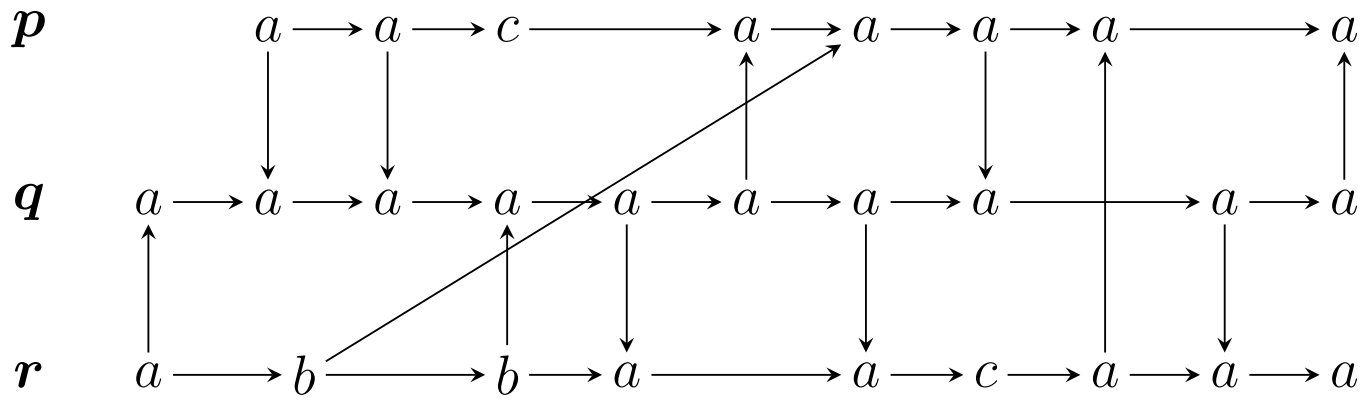
Monadic Second-Order Logic (MSO) over MSCs

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



Monadic Second-Order Logic (MSO) over MSCs

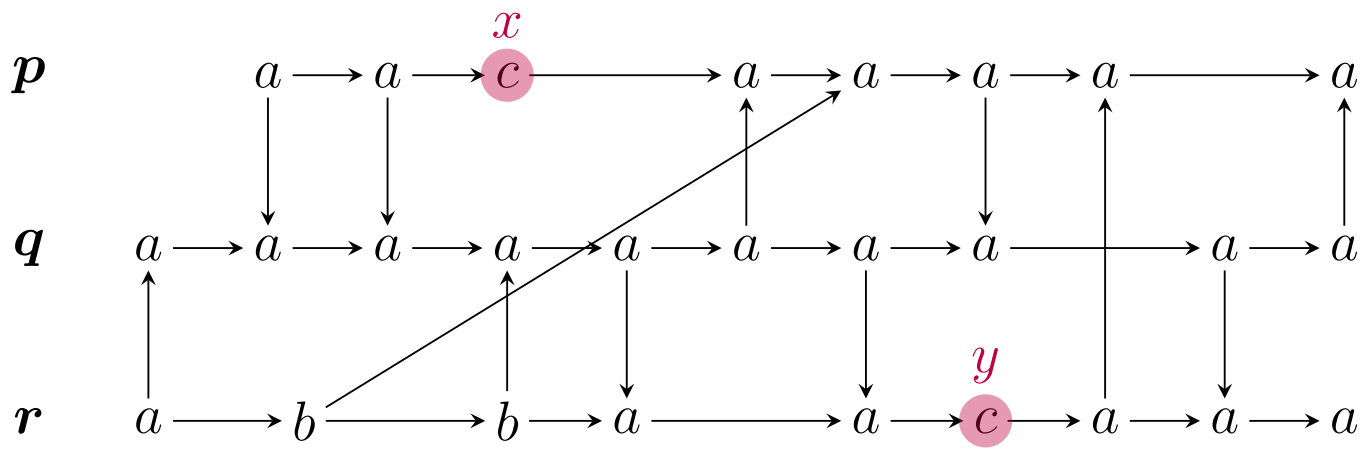
$$
\begin{array}{lll}
\varphi ::= & a(x) \mid p(x) & \text{label/process of event } x \\
\mid & x \rightarrow y & \text{process successor} \\
\mid & x \searrow y & \text{message relation} \\
\mid & x \leq y & \text{happened-before} \\
\mid & \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\varphi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



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

Monadic Second-Order Logic (MSO) over MSCs

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



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

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

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)

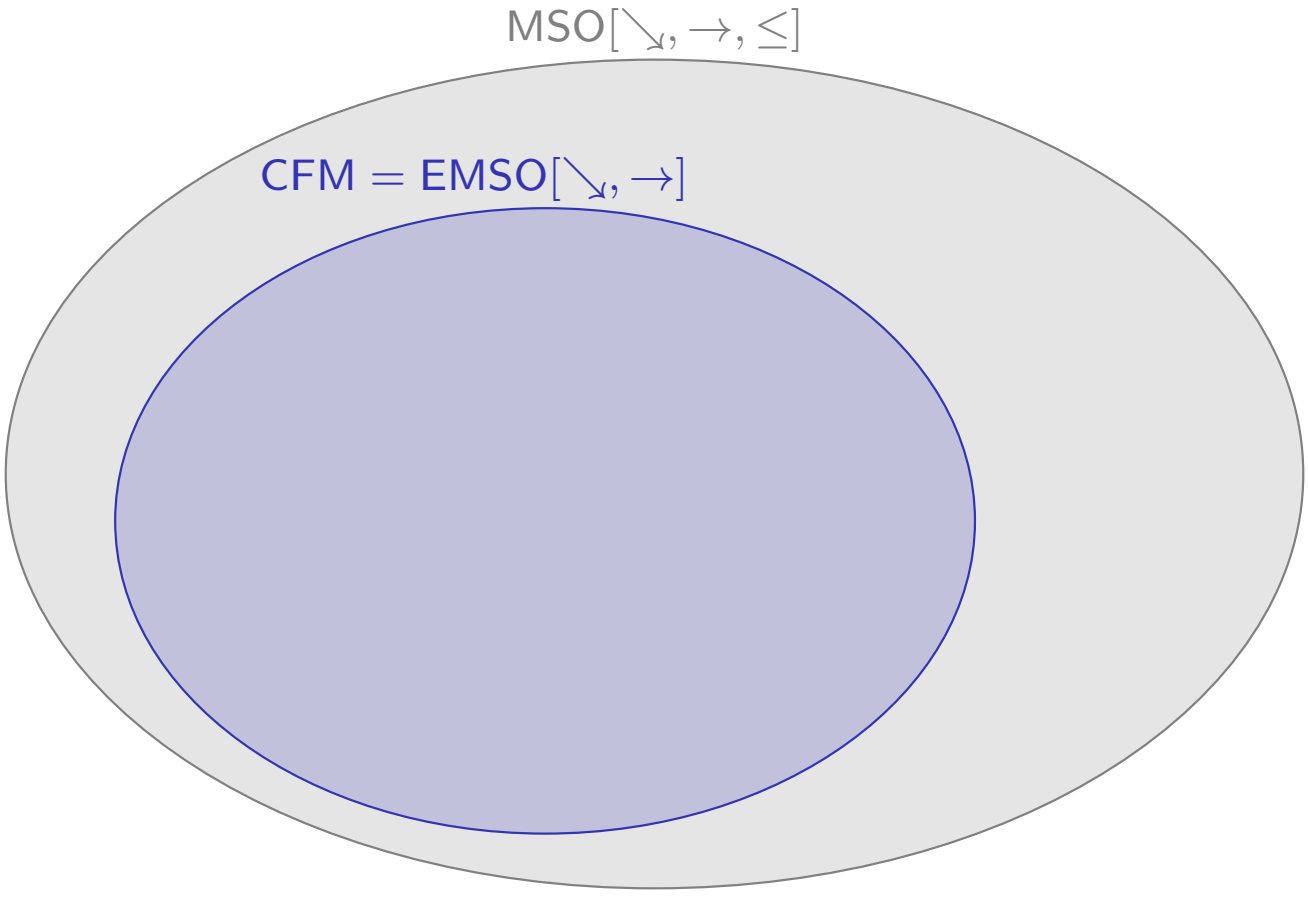
Over universally bounded MSCs, CFM $=$ MSO$[\searrow, \rightarrow, \leq]$.

Theorem (Genest-Kuske-Muscholl '06)

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

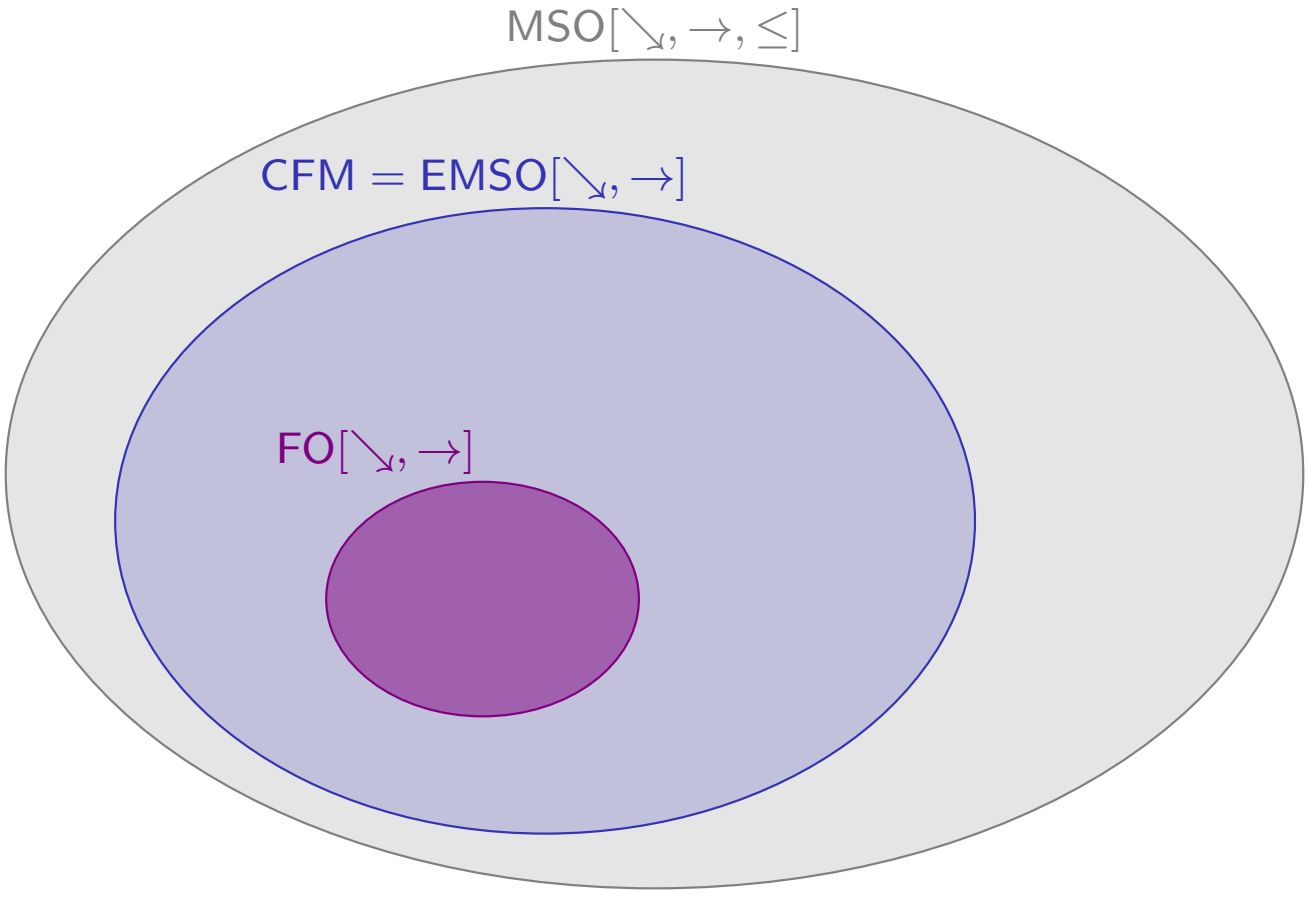
General case

General case



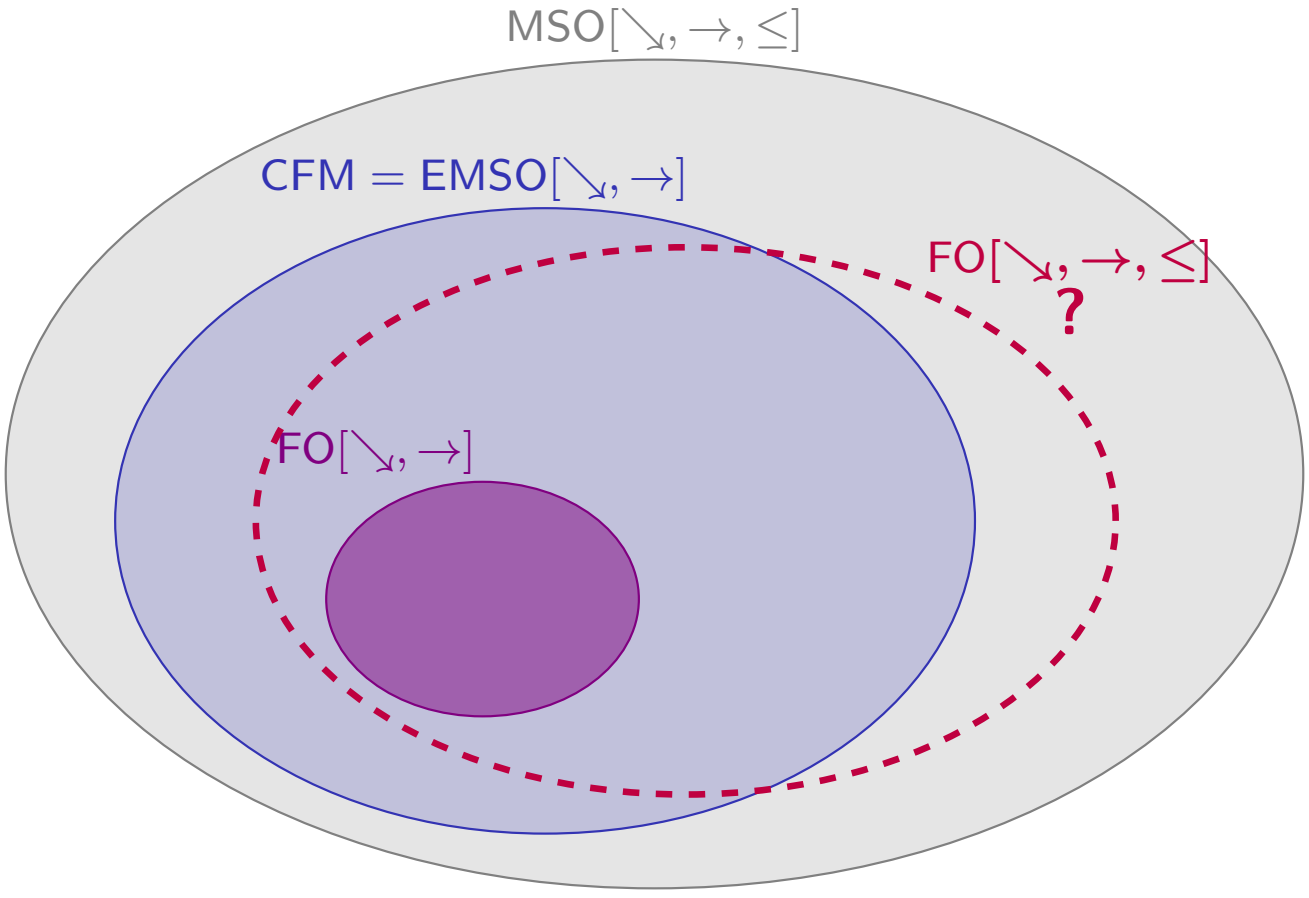
[B.-Leucker '06] $CFM = EMSO[\searrow, \rightarrow] \subsetneq MSO[\searrow, \rightarrow]$

General case



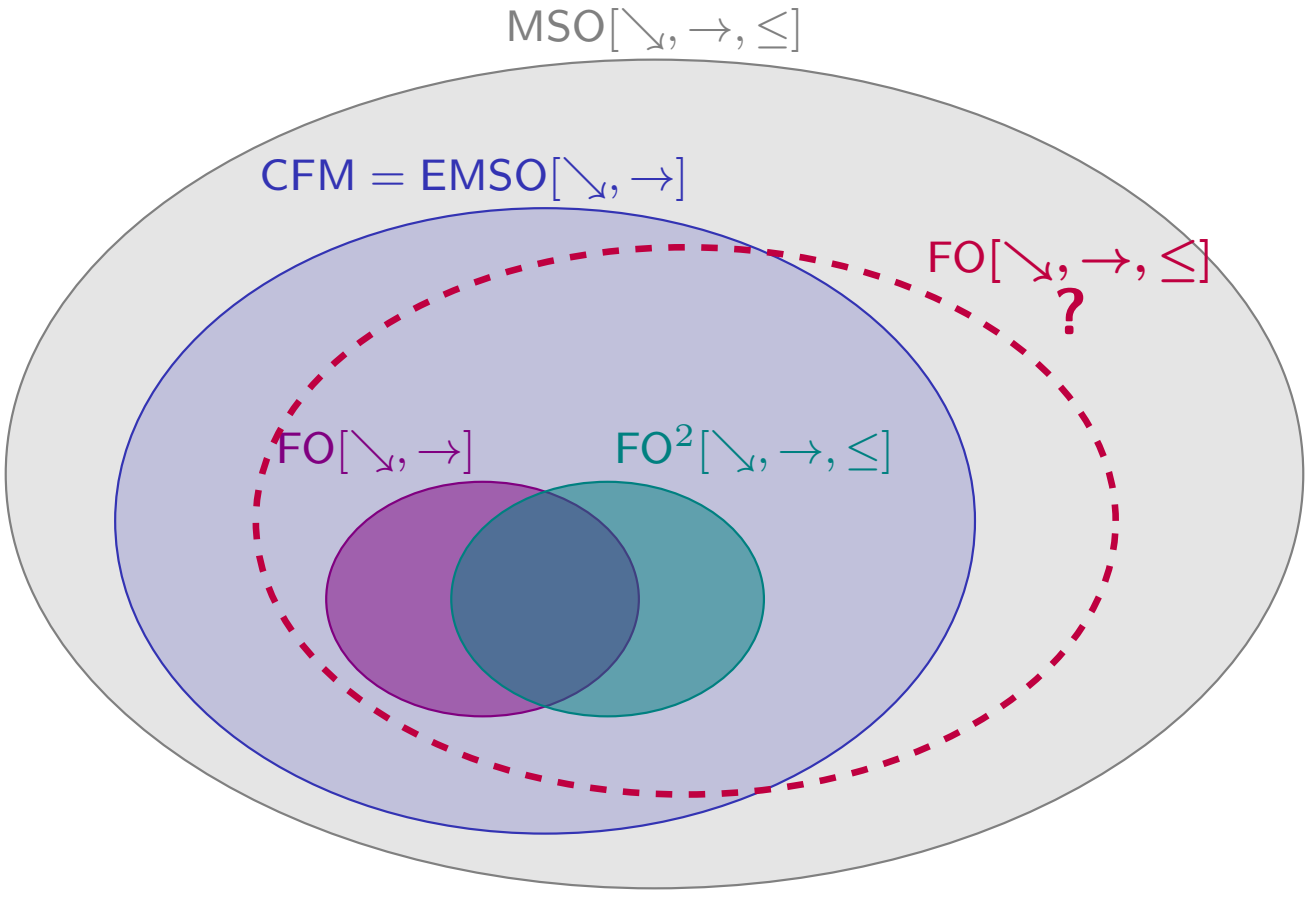
[B.-Leucker '06] $\mathrm{CFM} = \mathrm{EMSO}[\searrow, \rightarrow] \subsetneq \mathrm{MSO}[\searrow, \rightarrow]$

General case



[B.-Leucker '06] $\mathrm{CFM} = \mathrm{EMSO}[\searrow, \rightarrow] \subsetneq \mathrm{MSO}[\searrow, \rightarrow]$

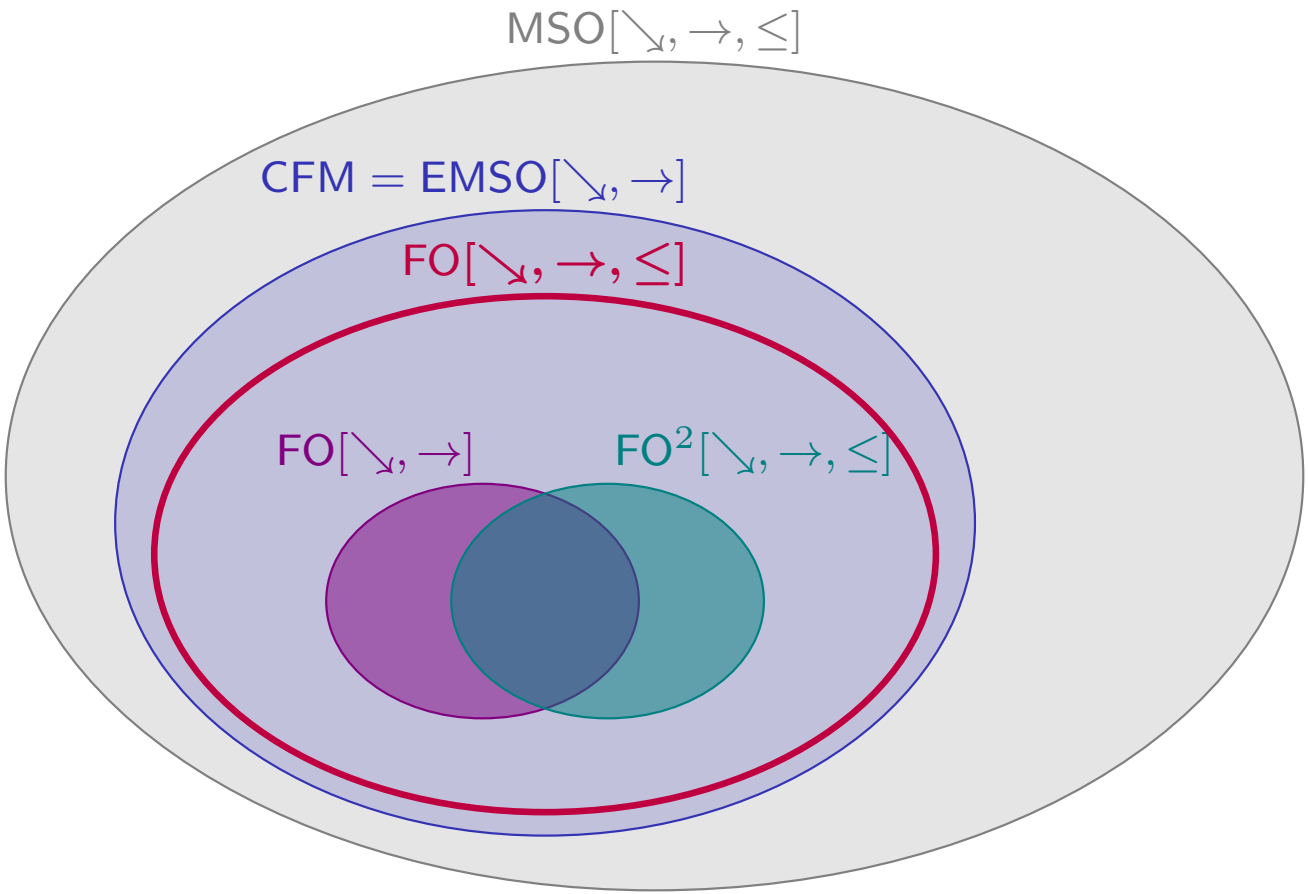
General case



[B.-Leucker '06] $\mathsf{CFM} = \mathsf{EMSO}[\searrow, \rightarrow] \subsetneq \mathsf{MSO}[\searrow, \rightarrow]$

[B.-Fortin-Gastin '18] $\mathsf{CFM} = \mathsf{EMSO}^2[\searrow, \rightarrow, \leq]$

General case

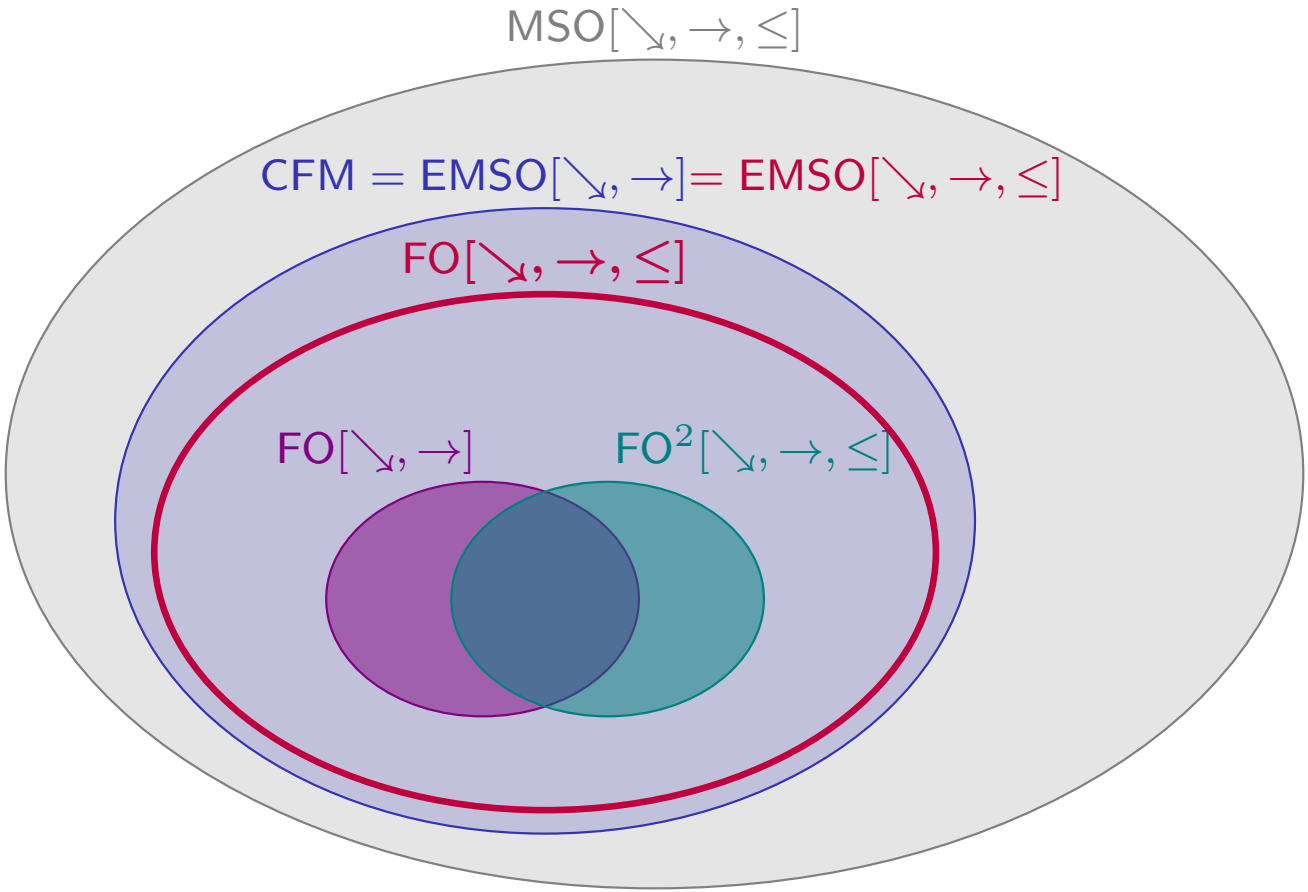


[B.-Leucker '06] $\mathsf{CFM} = \mathsf{EMSO}[\searrow, \rightarrow] \subsetneq \mathsf{MSO}[\searrow, \rightarrow]$

[B.-Fortin-Gastin '18] $\mathsf{CFM} = \mathsf{EMSO}^2[\searrow, \rightarrow, \leq]$

Main result: $\mathsf{CFM} = \mathsf{EMSO}[\searrow, \rightarrow, \leq]$

General case



[B.-Leucker '06] $CFM = EMSO[\searrow, \rightarrow] \subsetneq MSO[\searrow, \rightarrow]$

[B.-Fortin-Gastin '18] $CFM = EMSO^2[\searrow, \rightarrow, \leq]$

Main result: $CFM = EMSO[\searrow, \rightarrow, \leq]$

Translation from FO to CFMs

Goal

$FO[\searrow, \rightarrow, \leq]$ sentence φ

CFM $\mathcal{A}$ such that $L(\mathcal{A}) = L(\varphi)$

Translation from FO to CFMs

Goal

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

Translation from FO to CFMs

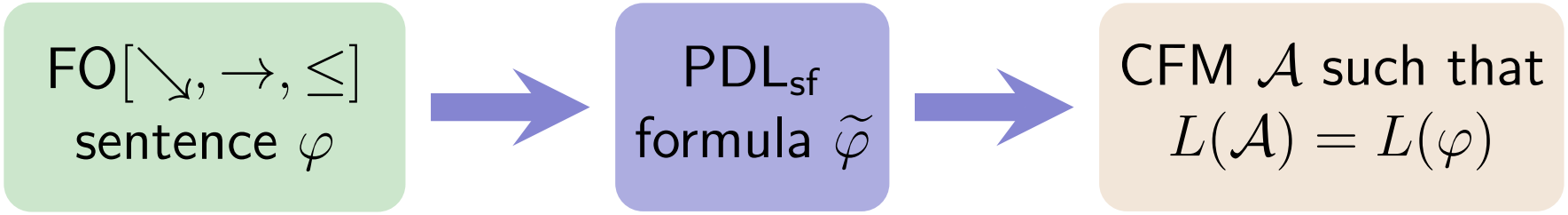
Goal

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

- 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



- 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: PDL_{sf}^{-}

$\varphi, \psi ::=$

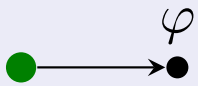
A simple modal logic for MSCs: PDL_{sf}^{-}

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

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

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

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

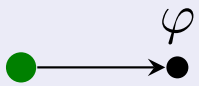


("Next φ")

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

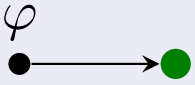
$$\varphi, \psi ::= \quad a \mid p \mid \varphi \vee \varphi \mid \neg\varphi$$

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



("Next φ")

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

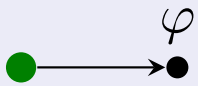


("Yesterday φ")

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

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

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

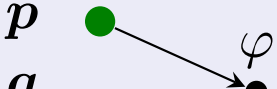


("Next φ")

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

("Yesterday φ")

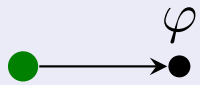
$$\mid \langle {}^{p}\searrow_{q} \rangle \varphi \mid \langle {}^{p}\nwarrow_{q} \rangle \varphi$$



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

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

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

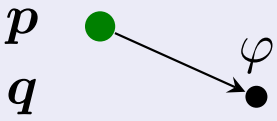


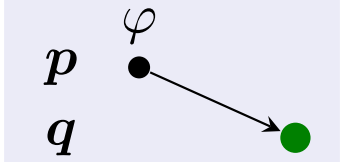
("Next φ")

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

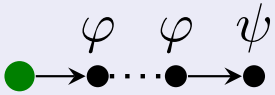
("Yesterday φ")

$$\mid \langle {}^{p}\!\searrow_{q} \rangle \varphi \mid \langle {}^{p}\!\nwarrow_{q} \rangle \varphi$$





$$\mid \langle \stackrel{\varphi}{\rightarrow} \rangle \psi$$

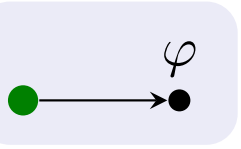


("φ Until ψ")

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

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

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

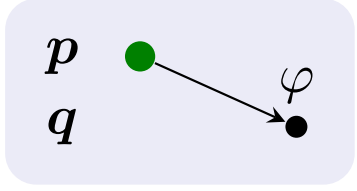


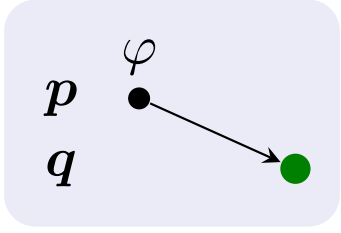
("Next φ")

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

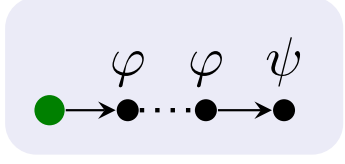
("Yesterday φ")

$$\mid \langle {}^{p}\!\searrow_{q} \rangle \varphi \mid \langle {}^{p}\!\nwarrow_{q} \rangle \varphi$$



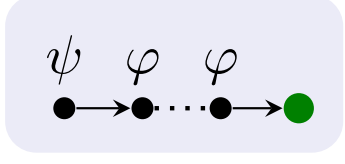


$$\mid \langle \overset{\varphi}{\rightarrow} \rangle \psi$$



("φ Until ψ")

$$\mid \langle \overset{\varphi}{\leftarrow} \rangle \psi$$

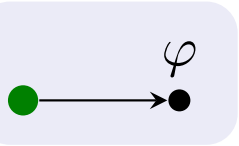


("φ Since ψ")

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

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

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

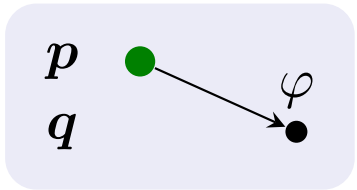


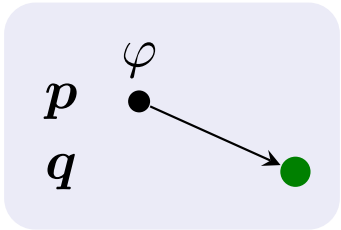
("Next φ")

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

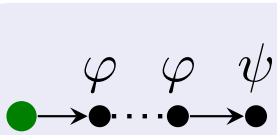
("Yesterday φ")

$$\mid \langle {}^{p}\!\searrow_{q} \rangle \varphi \mid \langle {}^{p}\!\nwarrow_{q} \rangle \varphi$$



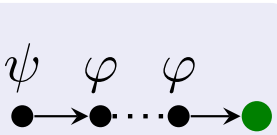


$$\mid \langle \overset{\varphi}{\rightarrow} \rangle \psi$$



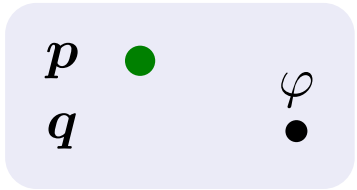
("φ Until ψ")

$$\mid \langle \overset{\varphi}{\leftarrow} \rangle \psi$$



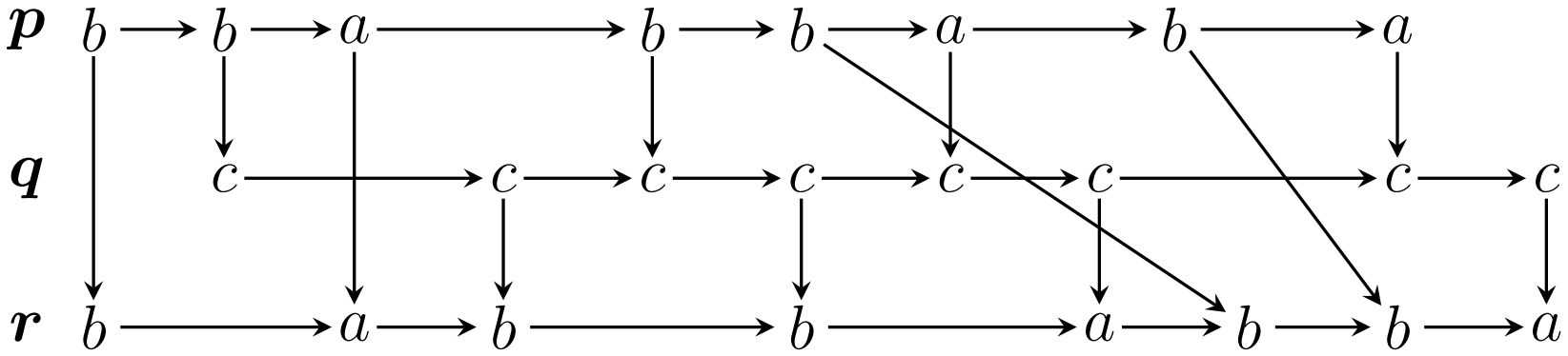
("φ Since ψ")

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



Examples

$$P = \{p, q, r\},\ \Sigma = \{a, b, c\}$$

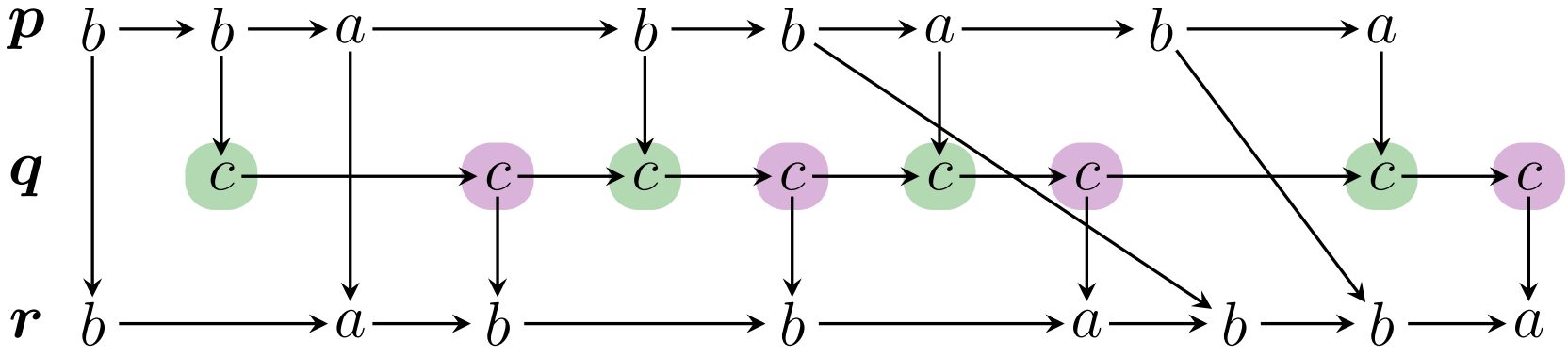


"On q, a receive from p is directly followed by a send to r":

$$\langle {}^{p}\!\nwarrow_{q} \rangle\, true \implies \langle \rightarrow \rangle \langle {}^{q}\!\searrow_{r} \rangle\, true$$

Examples

$$P = \{p, q, r\},\ \Sigma = \{a, b, c\}$$

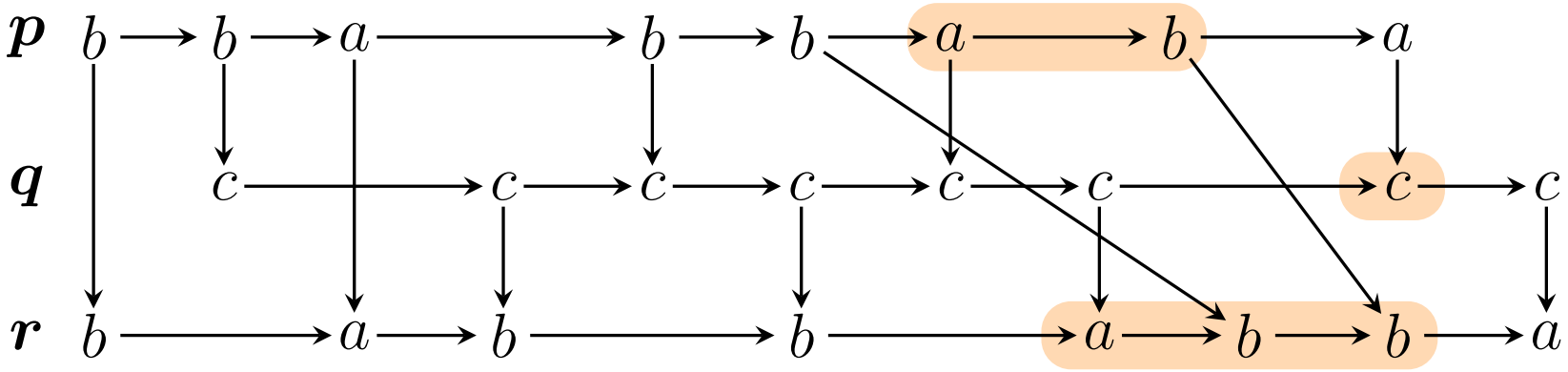


"On q, a receive from p is directly followed by a send to r":

$$\langle {}^{p}\!\nwarrow_{q} \rangle\, true \implies \langle \rightarrow \rangle \langle {}^{q}\!\searrow_{r} \rangle\, true$$

Examples

$$P = \{p, q, r\},\ \Sigma = \{a, b, c\}$$



"On q, a receive from p is directly followed by a send to r":

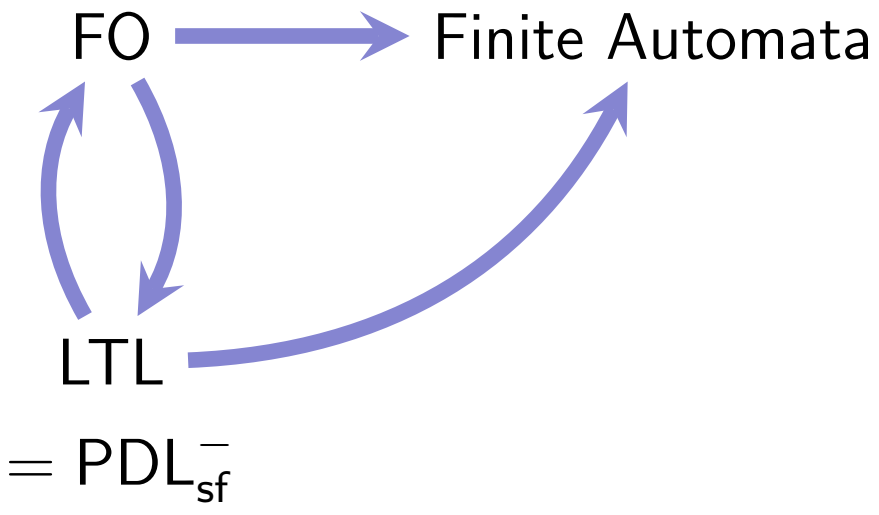
$$\langle {}^{p}\!\nwarrow_{q} \rangle\, true \implies \langle \rightarrow \rangle \langle {}^{q}\!\searrow_{r} \rangle\, true$$

"All events strictly in between the current and the last event on a process are bs":

$$\langle \overset{b}{\rightarrow} \rangle \neg \langle \rightarrow \rangle\, true$$

Expressivity of PDL_{sf}^{-}

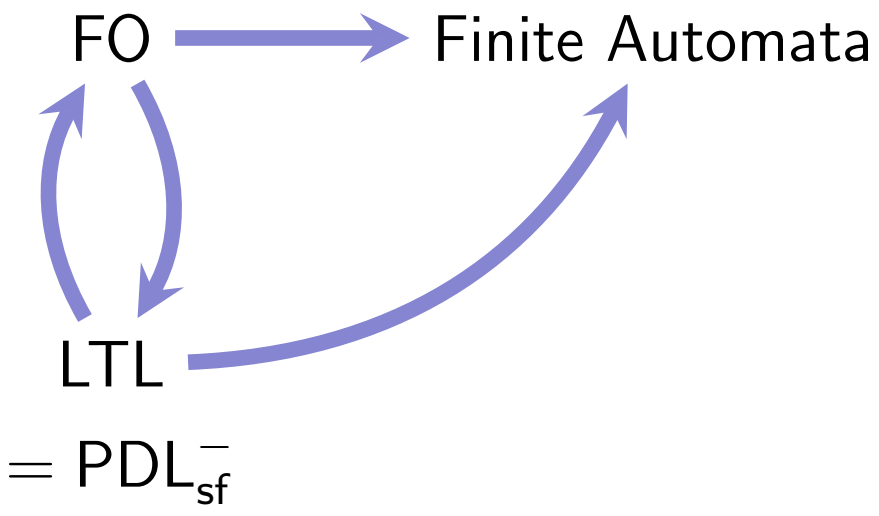
Over words ($|P| = 1$):



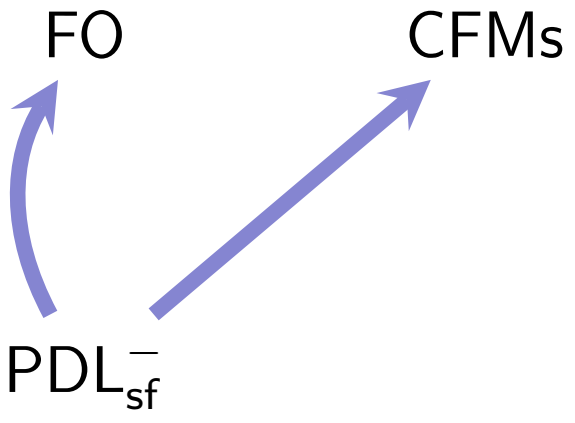
$= PDL_{sf}^{-}$

Expressivity of PDL_{sf}^{-}

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

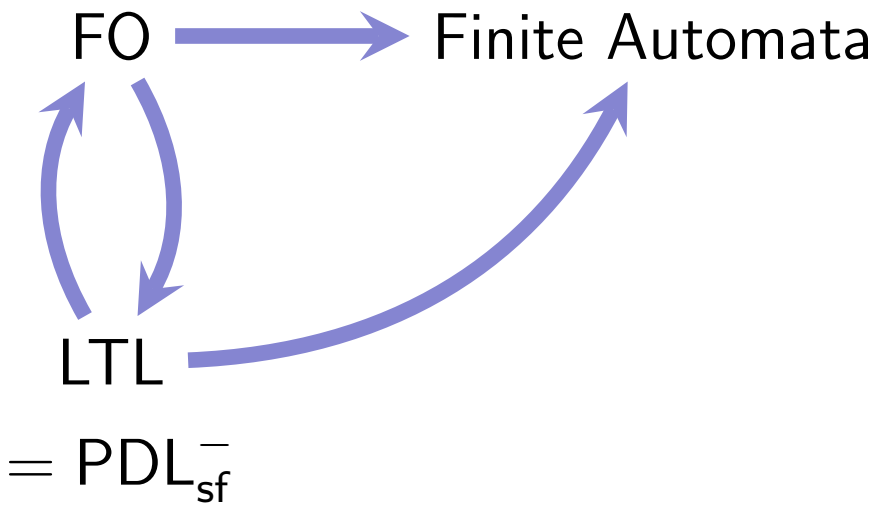


Over general MSCs:

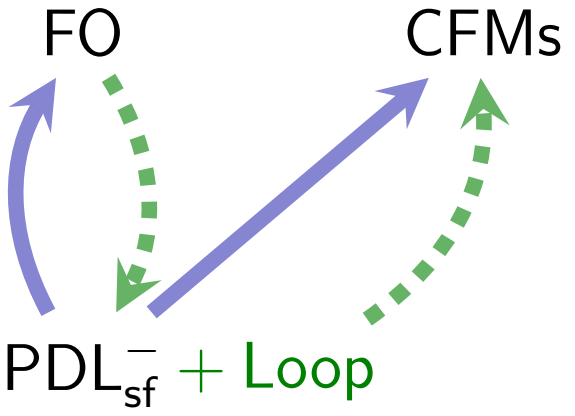


Expressivity of PDL_{sf}^{-}

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



Over general MSCs:



Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

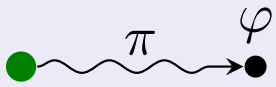
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



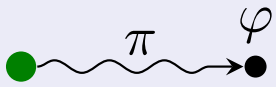
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



$$\pi ::= \rightarrow \mid \leftarrow \mid {}^{p}\searrow_{q} \mid {}^{p}\nwarrow_{q} \mid \mathsf{jump}_{p,q} \mid \overset{\varphi}{\rightarrow} \mid \overset{\varphi}{\leftarrow}$$

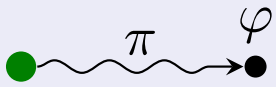
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



$$\pi ::= \rightarrow \mid \leftarrow \mid {}^{p}\searrow_{q} \mid {}^{p}\nwarrow_{q} \mid \mathsf{jump}_{p,q} \mid \overset{\varphi}{\rightarrow} \mid \overset{\varphi}{\leftarrow} \mid \pi \cdot \pi$$

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

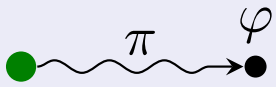
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



$$\pi ::= \rightarrow \mid \leftarrow \mid {}^{p}\searrow_{q} \mid {}^{p}\nwarrow_{q} \mid \mathsf{jump}_{p,q} \mid \overset{\varphi}{\rightarrow} \mid \overset{\varphi}{\leftarrow} \mid \pi \cdot \pi \mid \{\varphi\}?$$

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

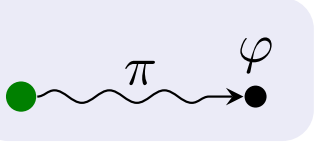
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



Path formulas

$$\pi ::= \rightarrow \mid \leftarrow \mid {}^{p}\searrow_{q} \mid {}^{p}\nwarrow_{q} \mid \mathsf{jump}_{p,q} \mid \overset{\varphi}{\rightarrow} \mid \overset{\varphi}{\leftarrow} \mid \pi \cdot \pi \mid \{\varphi\}?$$

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

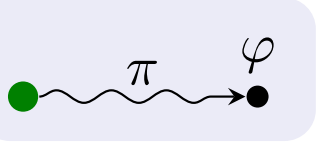
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



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

Path formulas

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

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

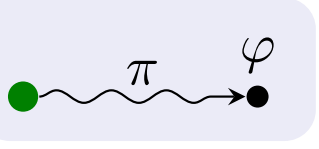
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



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

Path formulas

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

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

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

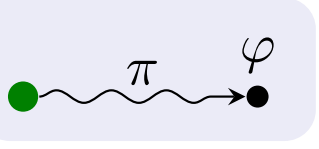
Star-free Propositional Dynamic Logic (PDL_{sf})

[Fischer-Ladner 1979] (PDL)

Event formulas

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

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



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

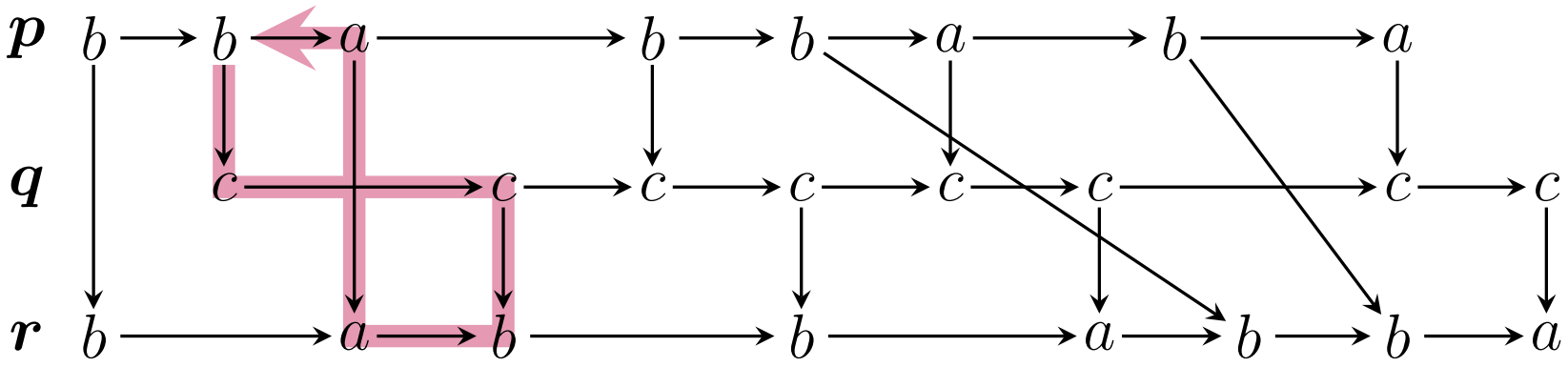
Path formulas

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

Notation: $\langle \pi_1 \cdot \pi_2 \cdots \pi_k \rangle \varphi \equiv \langle \pi_1 \rangle (\langle \pi_2 \rangle \cdots (\langle \pi_k \rangle \varphi) \cdots)$

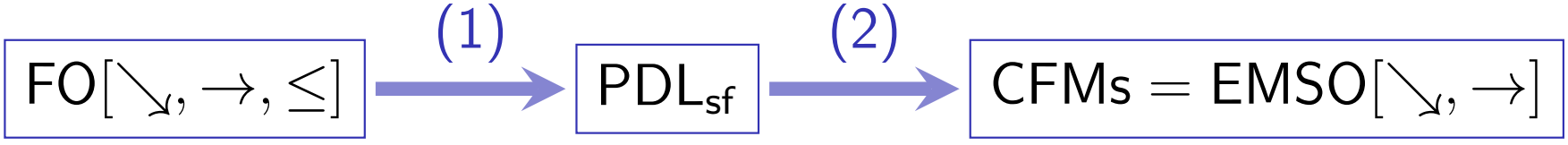
Example

$$P = \{p, q, r\},\ \Sigma = \{a, b, c\}$$

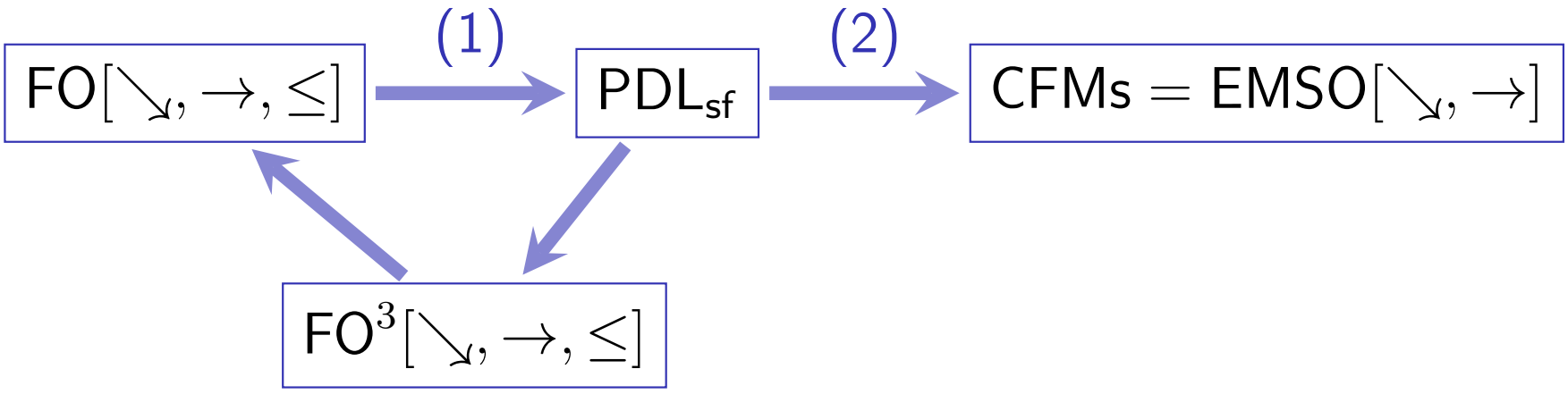


$$\varphi = \mathsf{Loop}({}^{p}\!\searrow_{q} \cdot \rightarrow \cdot {}^{q}\!\searrow_{r} \cdot \{b\}? \cdot \leftarrow \cdot {}^{p}\!\nwarrow_{r} \cdot \leftarrow)$$

Main results



Main results



From PDL_{sf} to FO^3

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

From PDL_{sf} to FO^3

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

$$p \quad \rightsquigarrow \quad p(x)$$

From $\mathsf{PDL}_{\mathsf{sf}}$ to FO^3

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

$$\begin{array}{rcl} p & \rightsquigarrow & p(x) \\ \langle \pi \rangle \, \varphi & \rightsquigarrow & \big(\exists y. \widetilde{\varphi}(y) \wedge \widetilde{\pi}(x,y)\big)(x) \end{array}$$

From $\mathrm{PDL}_{\mathrm{sf}}$ to FO^3

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

$$\begin{array}{rcl} p & \rightsquigarrow & p(x) \\ \langle \pi \rangle \, \varphi & \rightsquigarrow & \big(\exists y.\widetilde{\varphi}(y) \wedge \widetilde{\pi}(x,y)\big)(x) \end{array}$$

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

From $\mathsf{PDL}_{\mathsf{sf}}$ to FO^3

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

$$\begin{array}{rcl} p & \rightsquigarrow & p(x) \\ \langle \pi \rangle \, \varphi & \rightsquigarrow & \big(\exists y.\widetilde{\varphi}(y) \wedge \widetilde{\pi}(x,y)\big)(x) \end{array}$$

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

$$\pi_1 \cdot \pi_2 \quad \rightsquigarrow \quad \big(\exists z.\pi_1(x,z) \wedge \pi_2(z,y)\big)(x,y)$$

From $\mathsf{PDL}_{\mathsf{sf}}$ to FO^3

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

$$\begin{array}{rcl} p & \rightsquigarrow & p(x) \\ \langle \pi \rangle \, \varphi & \rightsquigarrow & \big(\exists y.\widetilde{\varphi}(y) \wedge \widetilde{\pi}(x,y)\big)(x) \end{array}$$

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

$$\pi_1 \cdot \pi_2 \quad \rightsquigarrow \quad \big(\exists z.\pi_1(x,z) \wedge \pi_2(z,y)\big)(x,y)$$

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

From $\mathsf{PDL}_{\mathsf{sf}}$ to FO^3

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

$$\begin{array}{rcl} p & \rightsquigarrow & p(x) \\ \langle \pi \rangle \, \varphi & \rightsquigarrow & \big(\exists y.\widetilde{\varphi}(y) \wedge \widetilde{\pi}(x,y)\big)(x) \end{array}$$

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

$$\pi_1 \cdot \pi_2 \quad \rightsquigarrow \quad \big(\exists z.\pi_1(x,z) \wedge \pi_2(z,y)\big)(x,y)$$

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

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.

From FO to PDL_{sf}

Theorem

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

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

Proof: By induction.

From FO to PDL_{sf}

Theorem

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

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

Proof: By induction.

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.

From FO to PDL_{sf}

Theorem

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

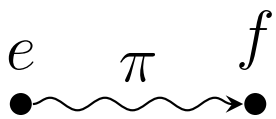
$$\Phi(x_1, \ldots, x_n) \equiv \bigvee \bigwedge \pi(x_i, x_j) \quad \text{where } \pi \in PDL_{sf}$$

Proof: By induction. Two interesting cases:

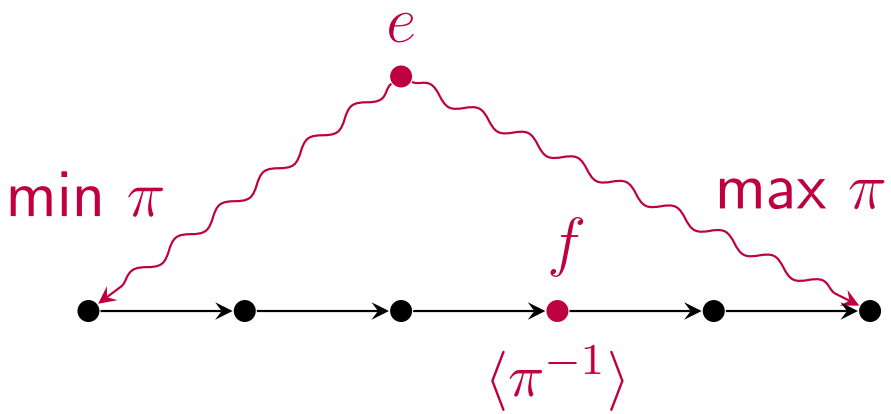
- negation
- existential quantification

Key Lemma

For all $\pi \in \mathsf{PDL}_{\mathsf{sf}}$,

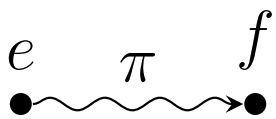


iff

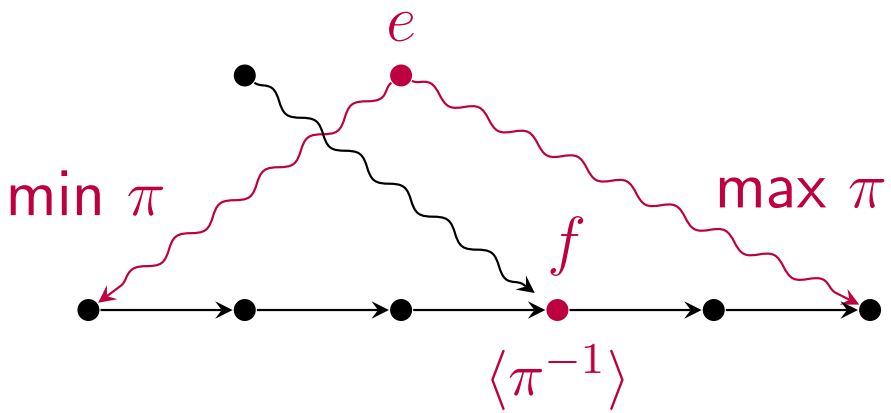


Key Lemma

For all $\pi \in \mathsf{PDL}_{\mathsf{sf}}$,

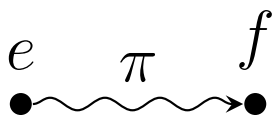


iff

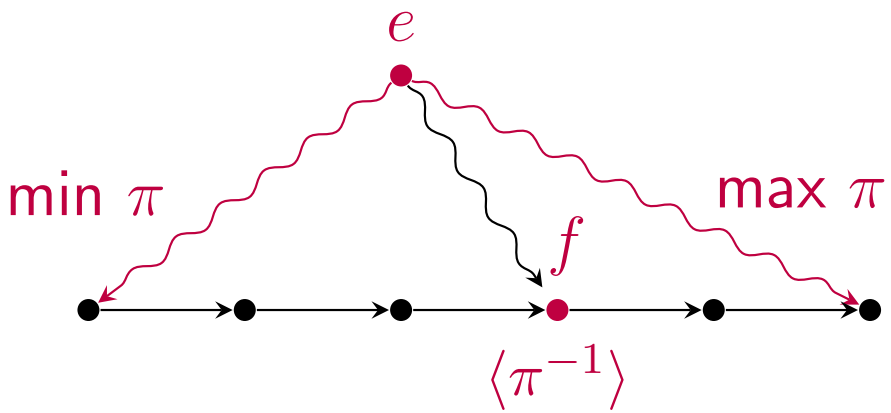


Key Lemma

For all $\pi \in \mathsf{PDL}_{\mathsf{sf}}$,

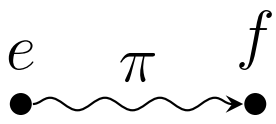


iff

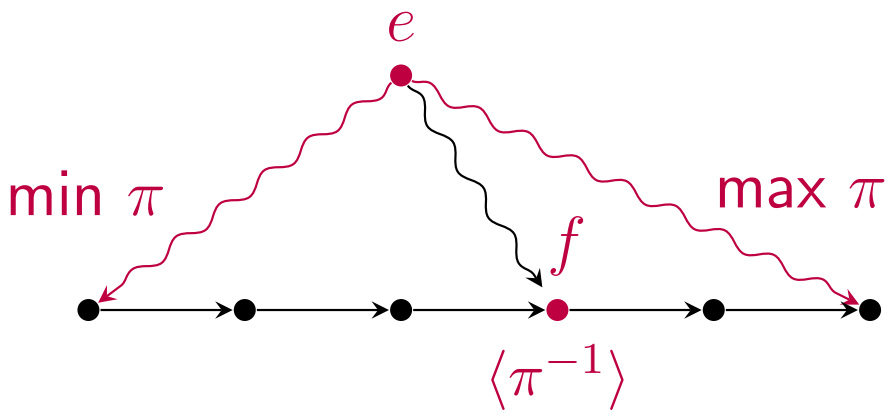


Key Lemma

For all $\pi \in \mathsf{PDL}_{\mathsf{sf}}$,



iff



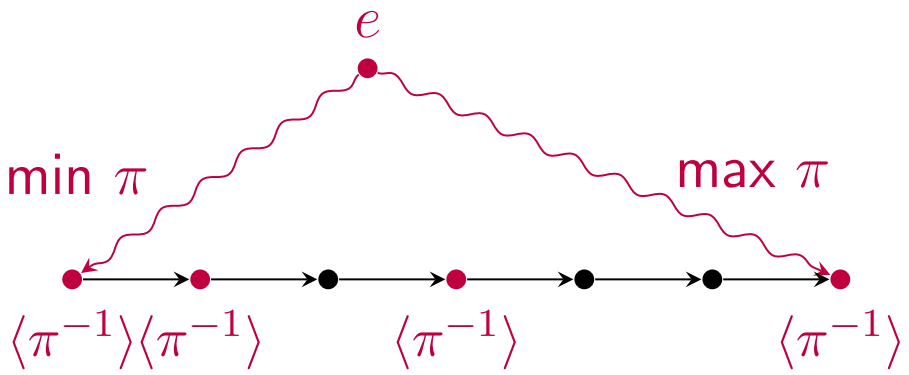
- Monotonicity of path formulas.
- Relies on FIFO behavior.
- Proof is by simple induction.

Negation

$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$

Negation

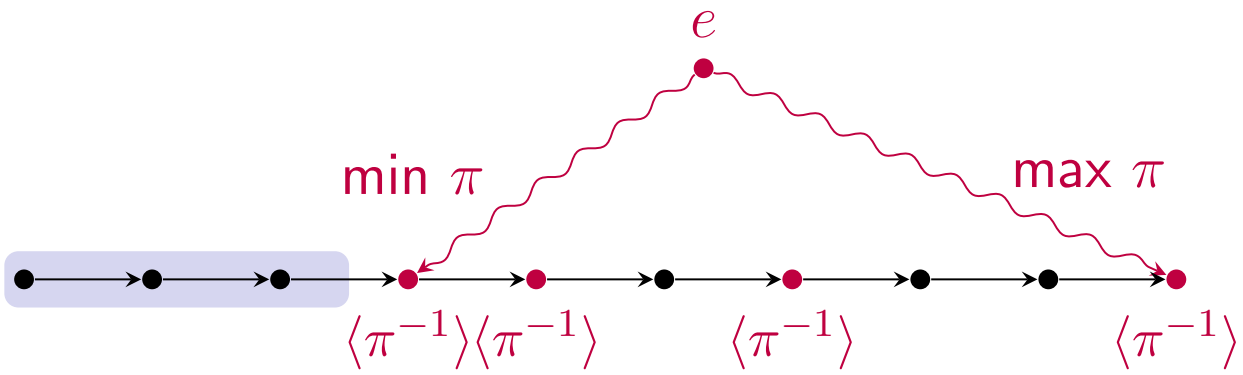
$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$



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

Negation

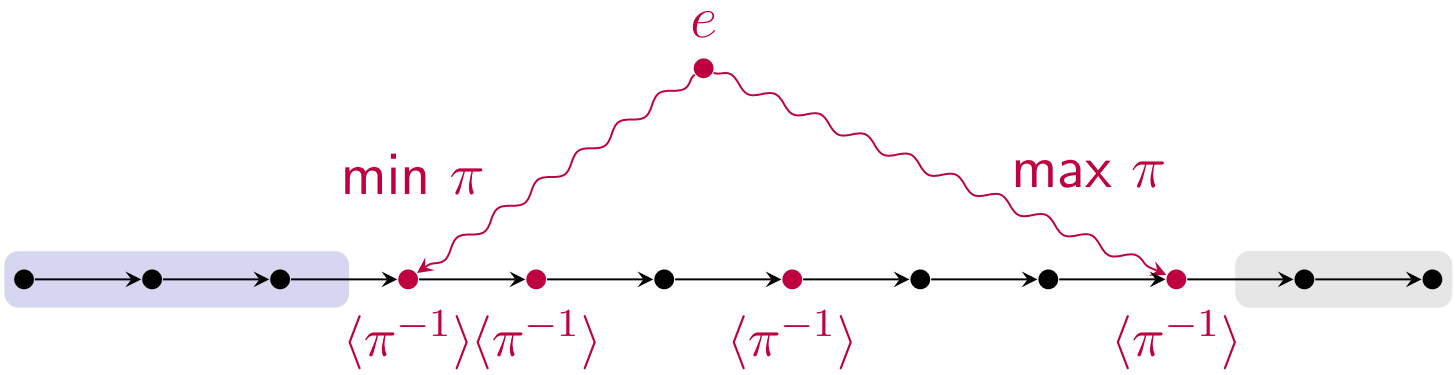
$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$



$\pi^{c} \equiv$ min $\pi \cdot \overleftarrow{+}$

Negation

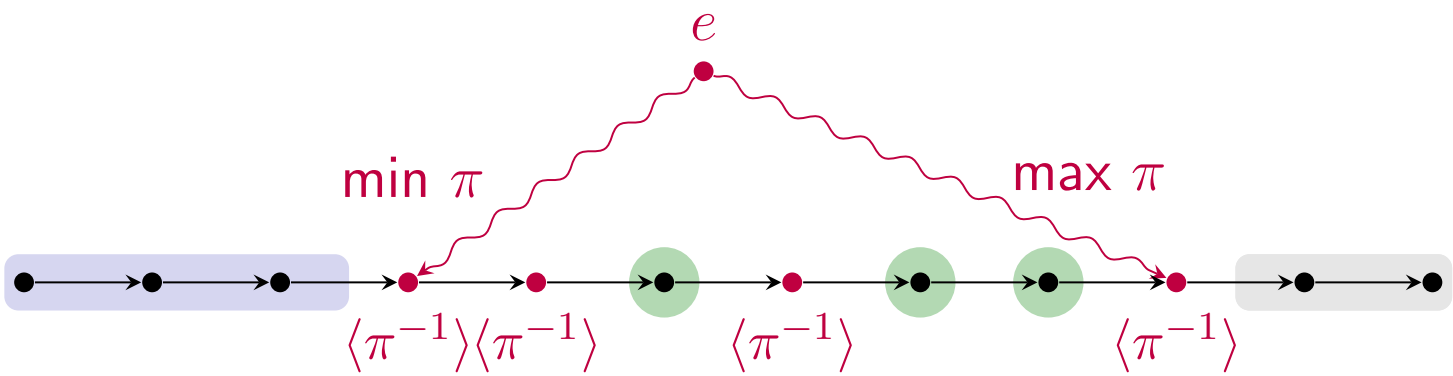
$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$



$$\pi^{c} \equiv \text{min } \pi \cdot \overleftarrow{+} \cup \text{max } \pi \cdot \overrightarrow{+}$$

Negation

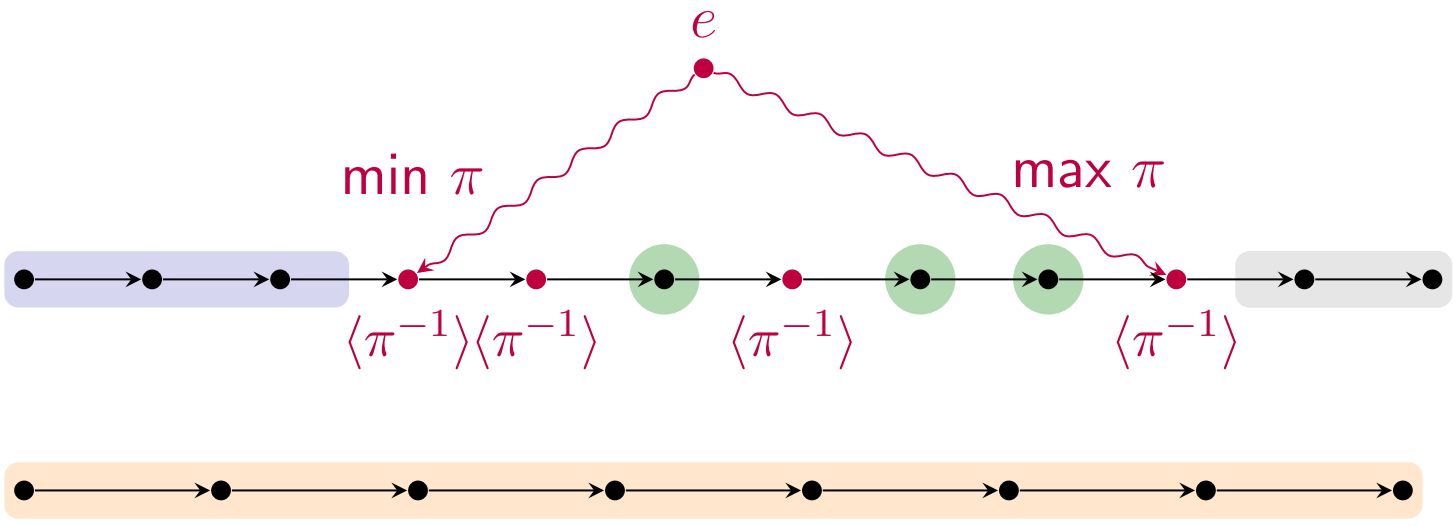
$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$



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

Negation

$$\neg \bigvee \bigwedge \pi(x_i, x_j) \equiv \bigwedge \bigvee \neg\pi(x_i, x_j)$$



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

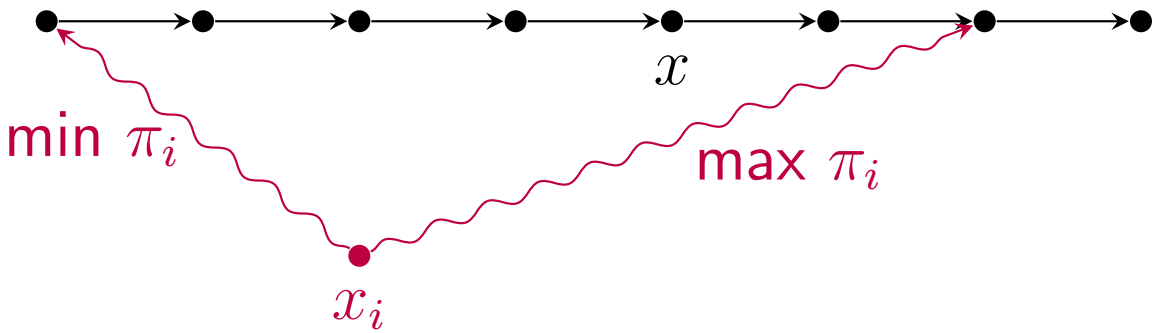
$$\cup \bigcup_{p,q} \{\neg \langle\pi\rangle\, q\}? \cdot \mathsf{jump}_{p,q}$$

Existential quantification

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

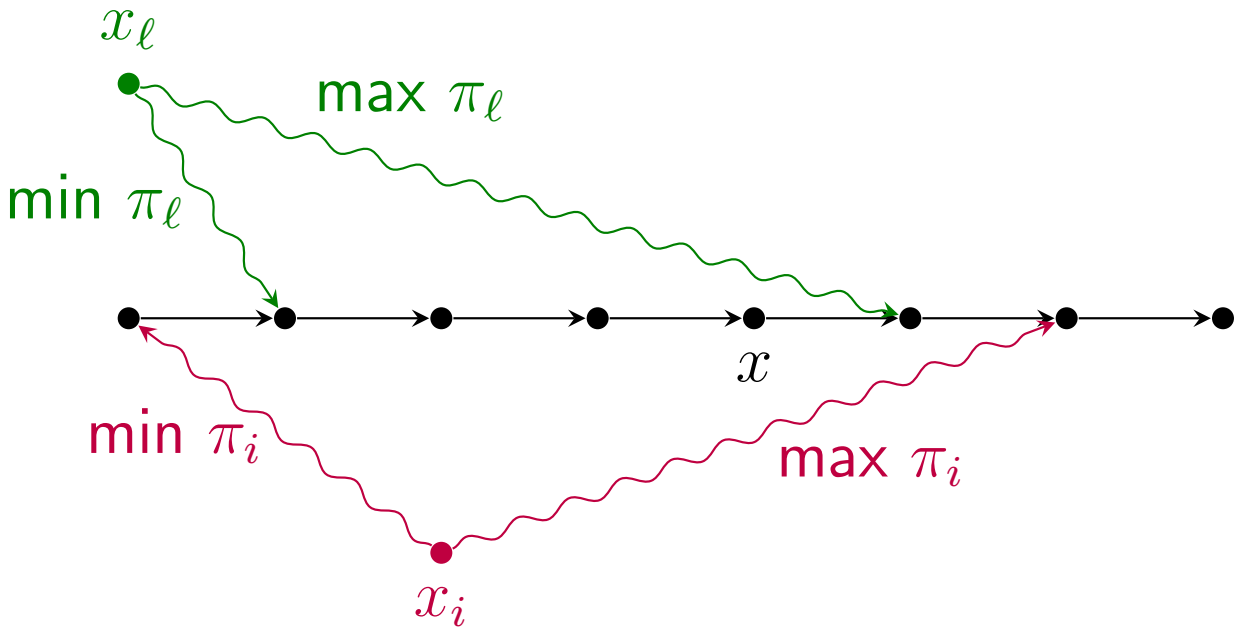
Existential quantification

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



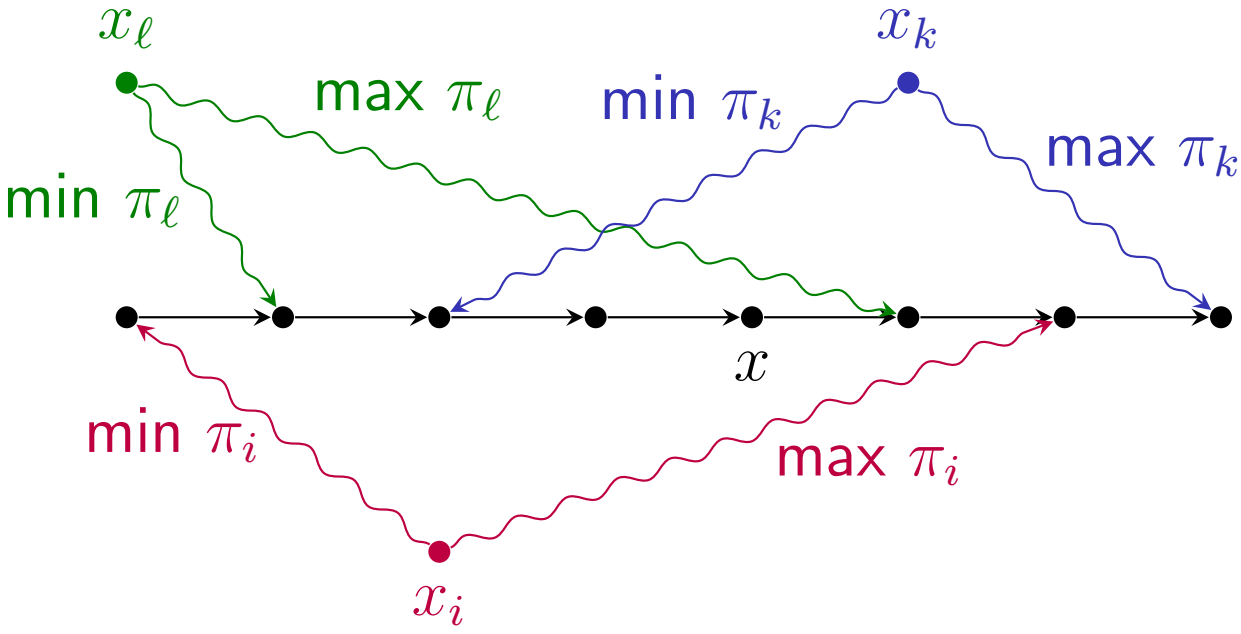
Existential quantification

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



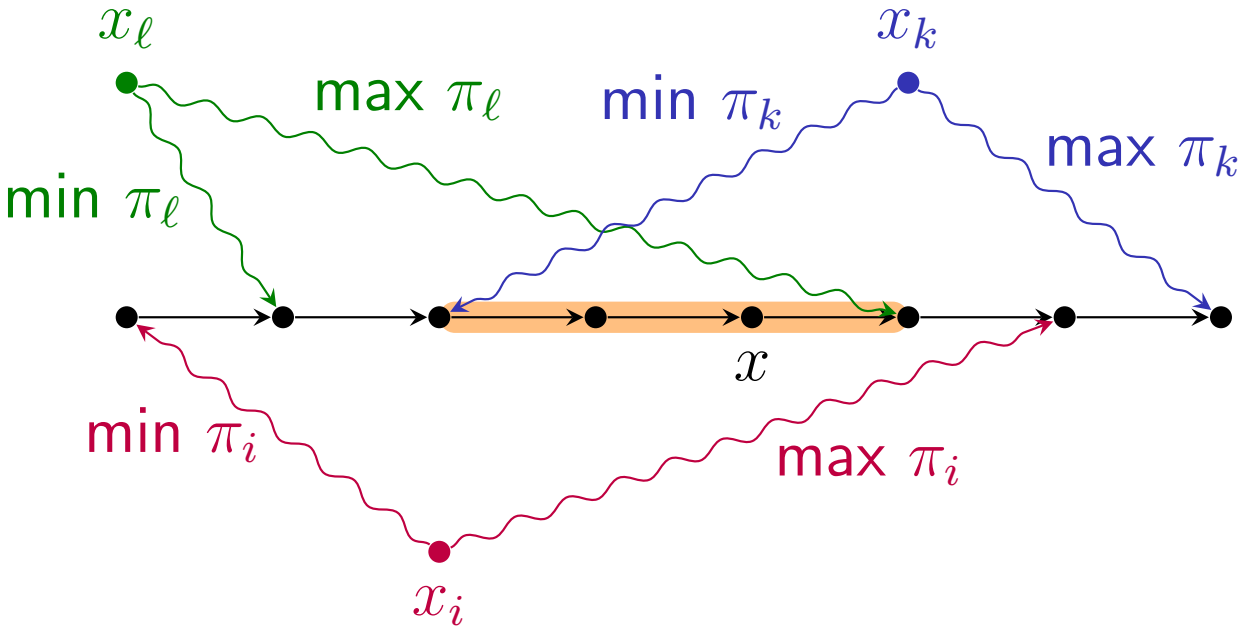
Existential quantification

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



Existential quantification

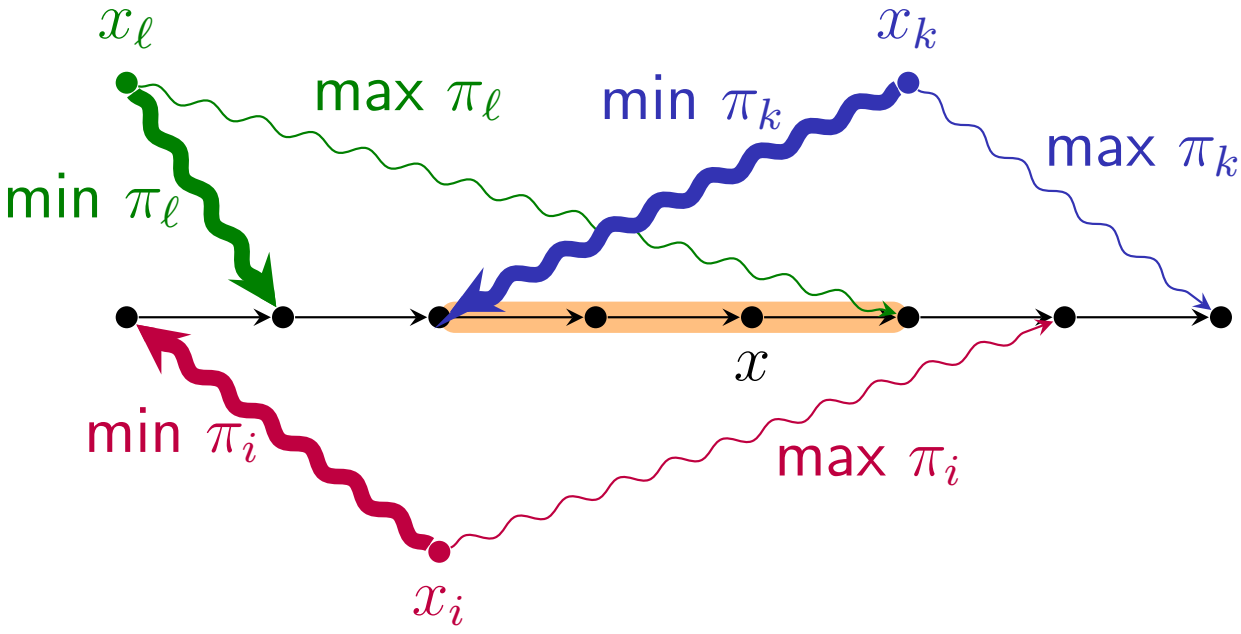
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



Is there an event in the intersection of the intervals that satisfies $\psi = \bigwedge_i \langle \pi_i^{-1} \rangle$?

Existential quantification

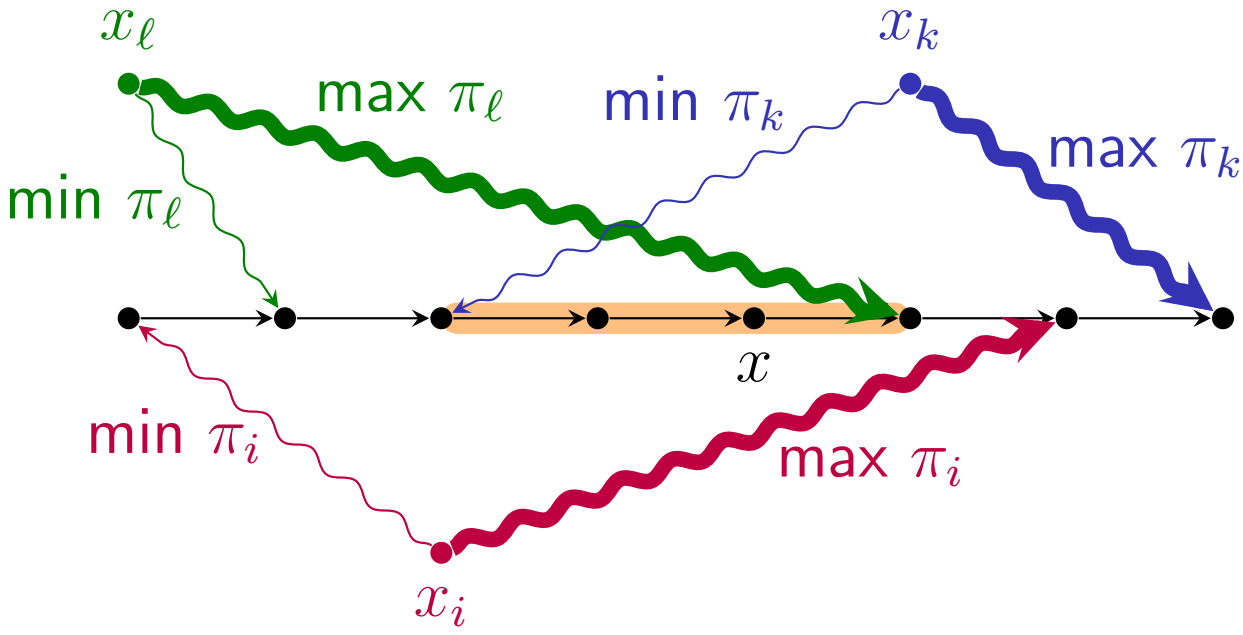
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



Is there an event in the intersection of the intervals that satisfies $\psi = \bigwedge_i \langle \pi_i^{-1} \rangle$?

Existential quantification

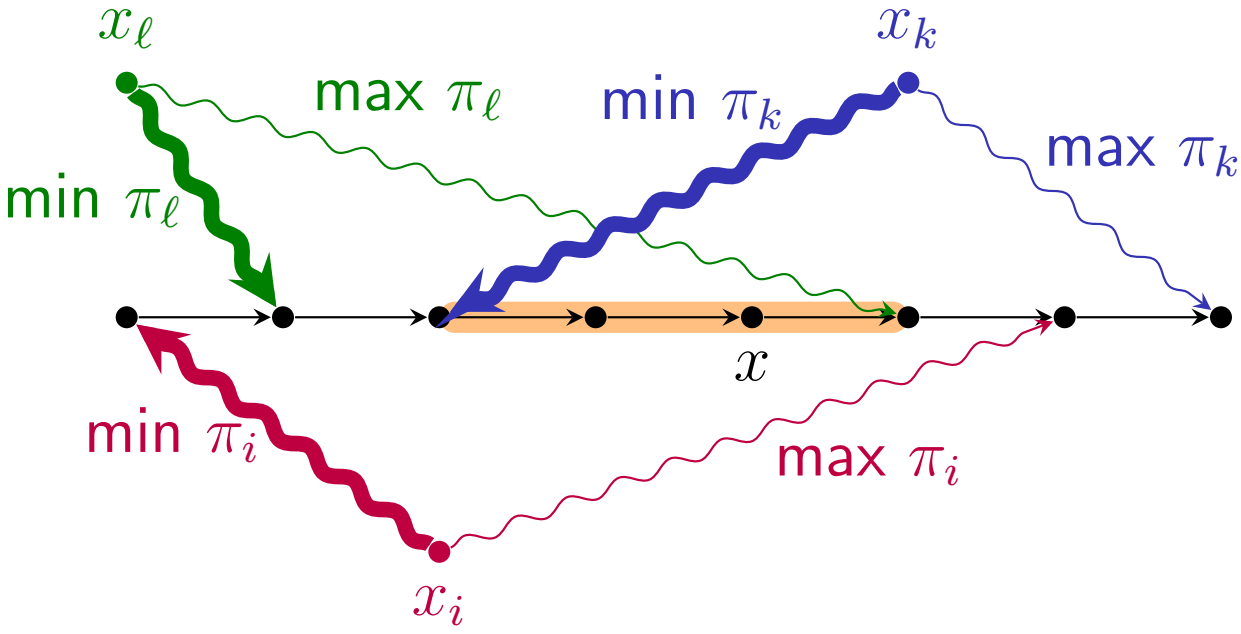
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



Is there an event in the intersection of the intervals that satisfies $\psi = \bigwedge_i \langle \pi_i^{-1} \rangle$?

Existential quantification

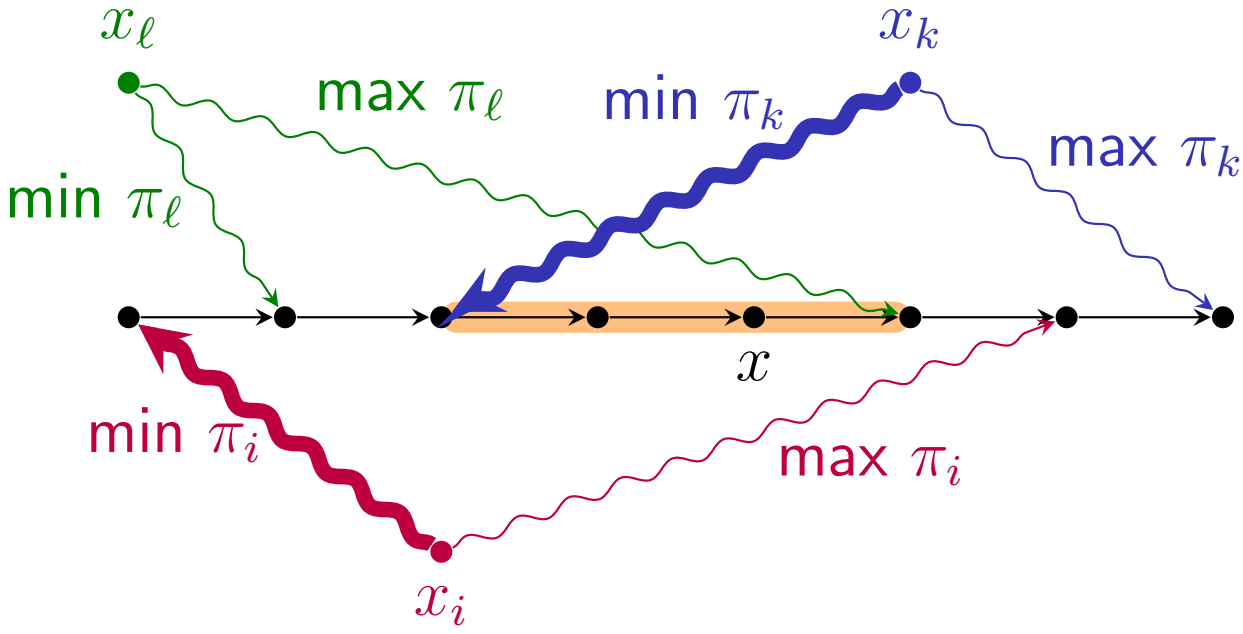
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_k \Big(\qquad \Big)$$

Existential quantification

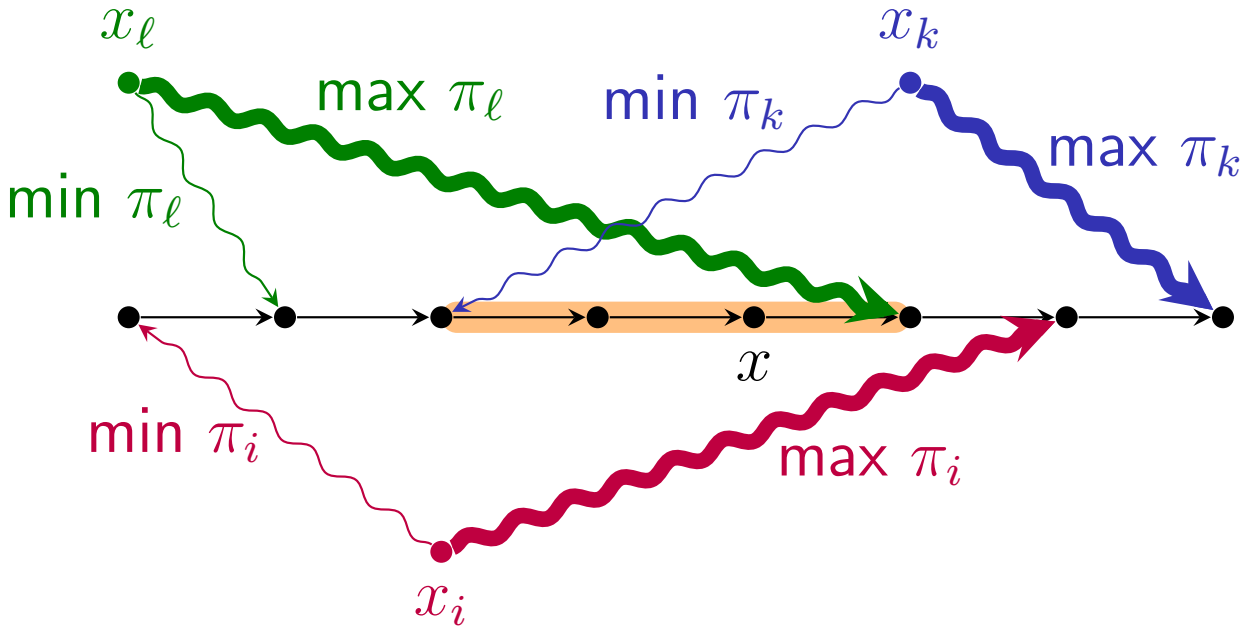
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_k \left(\bigwedge_i (\text{min } \pi_i \cdot \xrightarrow{*} \cdot (\text{min } \pi_k)^{-1})(x_i, x_k) \right)$$

Existential quantification

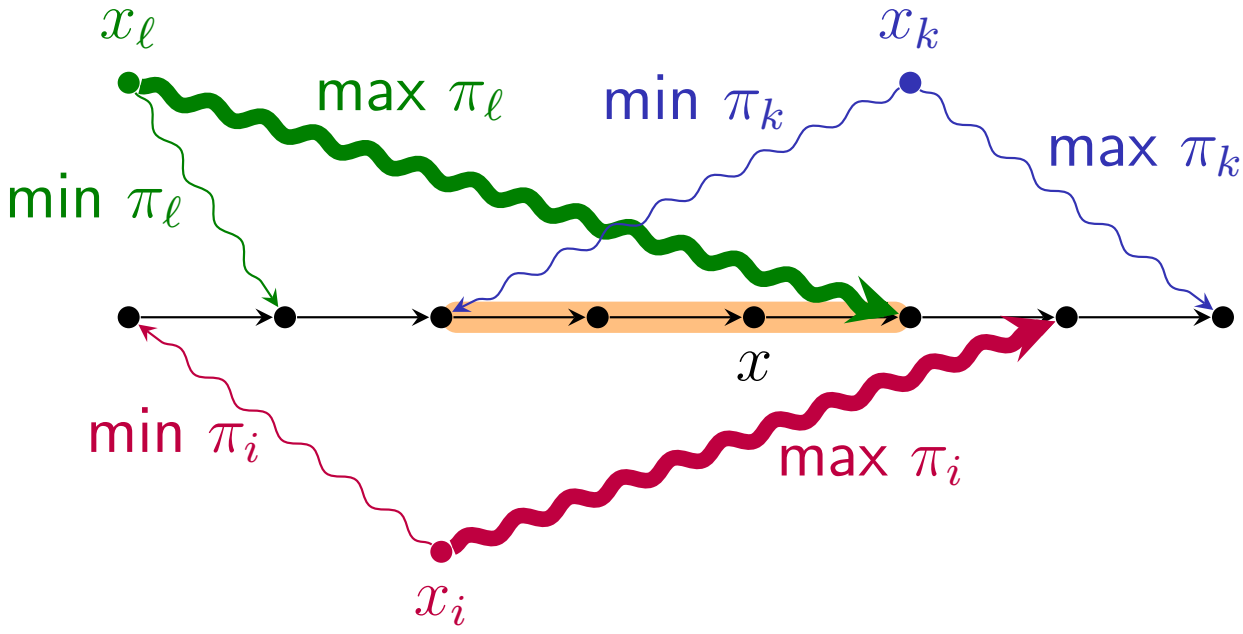
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_{k,\ell} \left(\bigwedge_i (\text{min } \pi_i \cdot \xrightarrow{*} \cdot (\text{min } \pi_k)^{-1})(x_i, x_k) \right)$$

Existential quantification

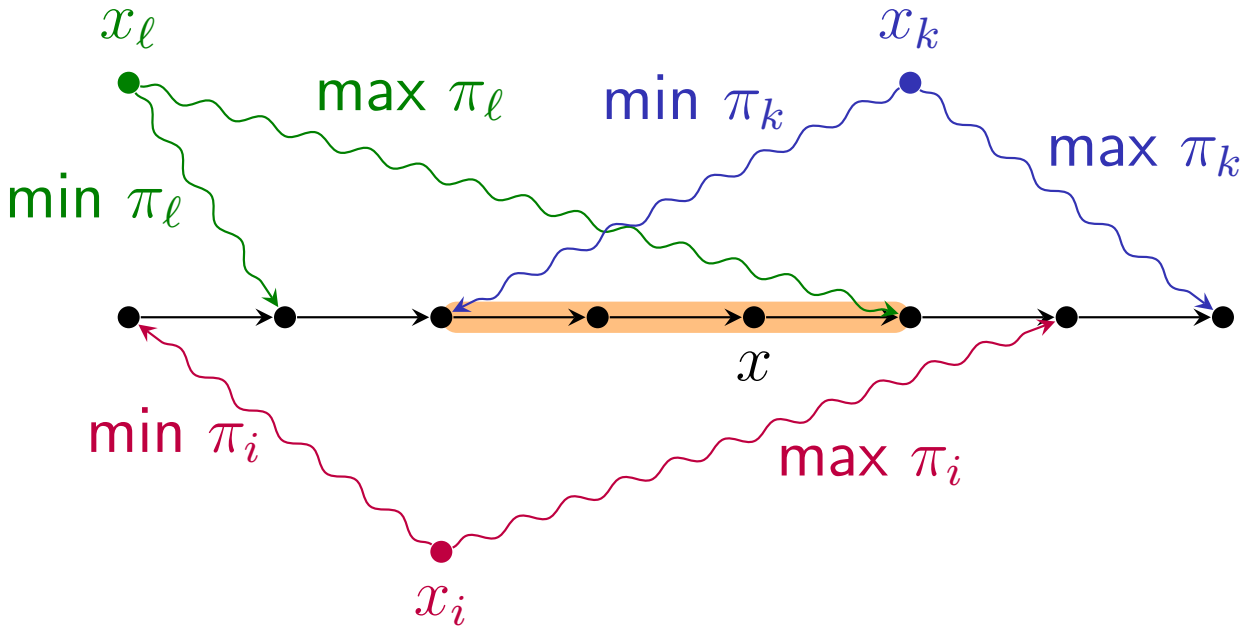
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_{k,\ell} \left(\begin{array}{rl} & \bigwedge_i (\text{min } \pi_i \cdot \xrightarrow{*} \cdot (\text{min } \pi_k)^{-1})(x_i, x_k) \\ \wedge & \bigwedge_i (\text{max } \pi_i \cdot \xleftarrow{*} \cdot (\text{max } \pi_\ell)^{-1})(x_i, x_\ell) \end{array} \right)$$

Existential quantification

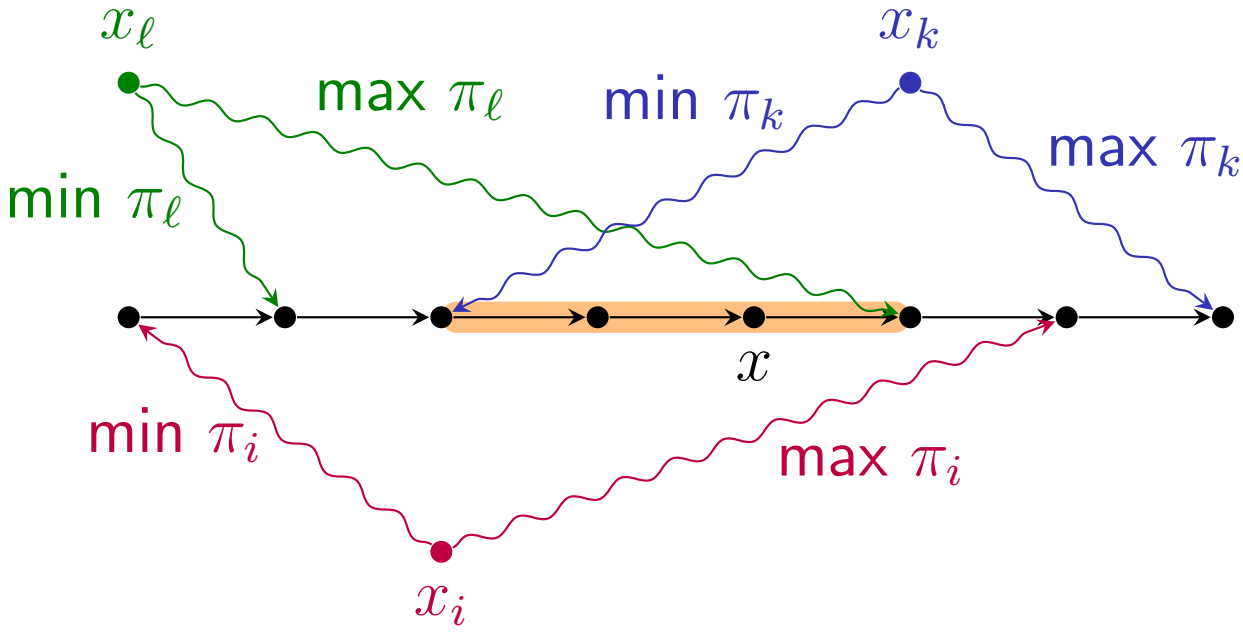
$\exists x.\, \bigwedge_i \pi_i(x_i, x) \quad \rightsquigarrow \quad ?$



$$\bigvee_{k,\ell} \left(\begin{array}{rl} & \bigwedge_i (\text{min } \pi_i \cdot \xrightarrow{*} \cdot (\text{min } \pi_k)^{-1})(x_i, x_k) \\ \wedge & \bigwedge_i (\text{max } \pi_i \cdot \xleftarrow{*} \cdot (\text{max } \pi_\ell)^{-1})(x_i, x_\ell) \end{array} \right)$$

Existential quantification

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



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

From PDL_{sf} to CFMs

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.

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.

- $\varphi = b$

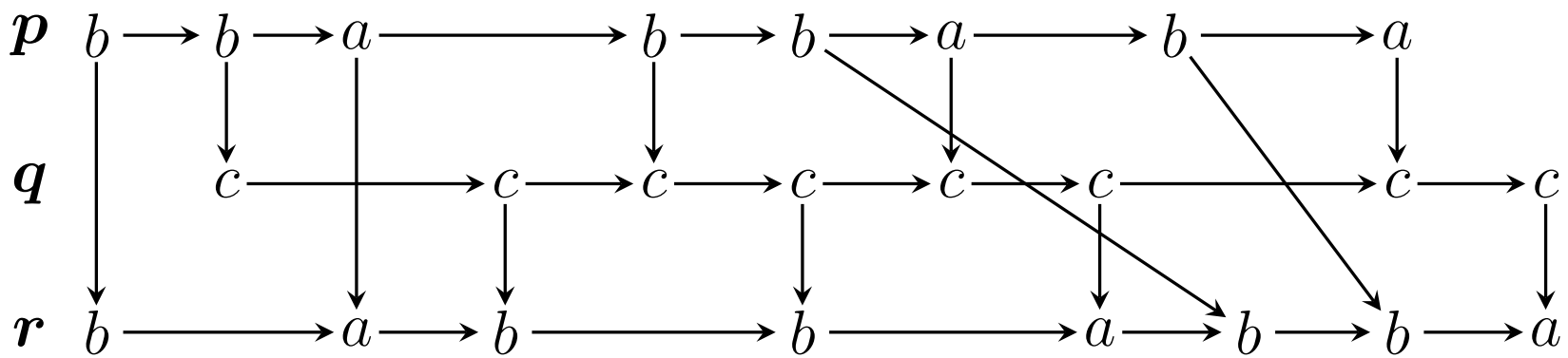
From PDL_{sf} to CFMs

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.

- $\varphi = b$



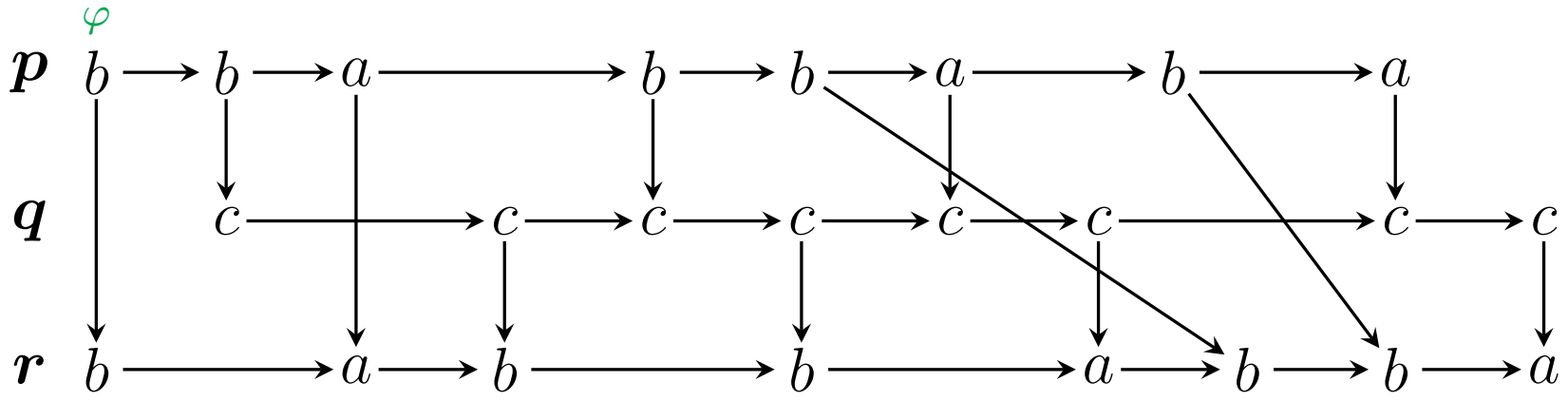
From PDL_{sf} to CFMs

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.

- $\varphi = b$



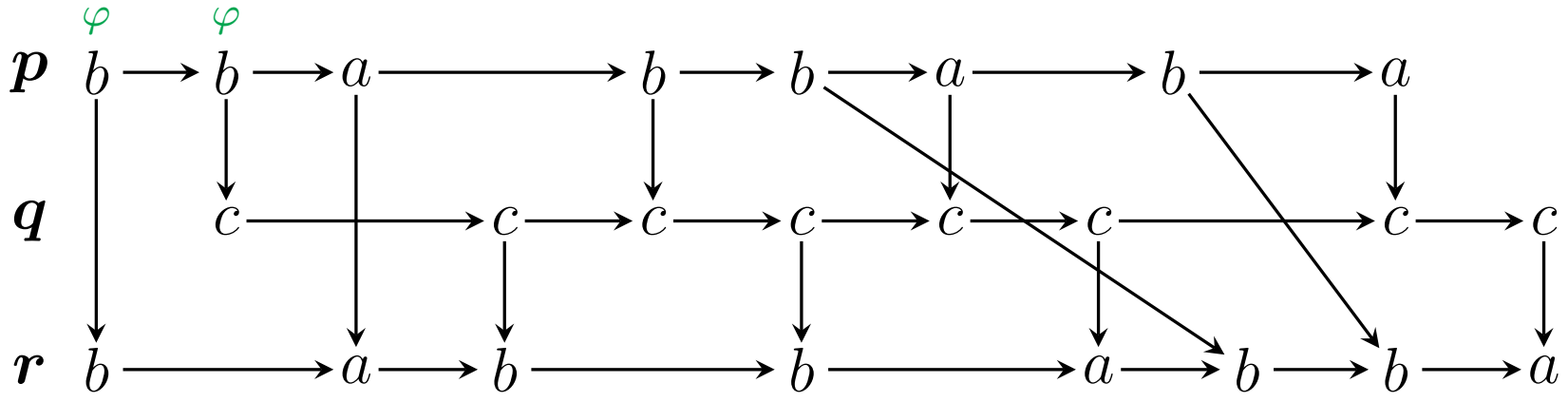
From PDL_{sf} to CFMs

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.

- $\varphi = b$



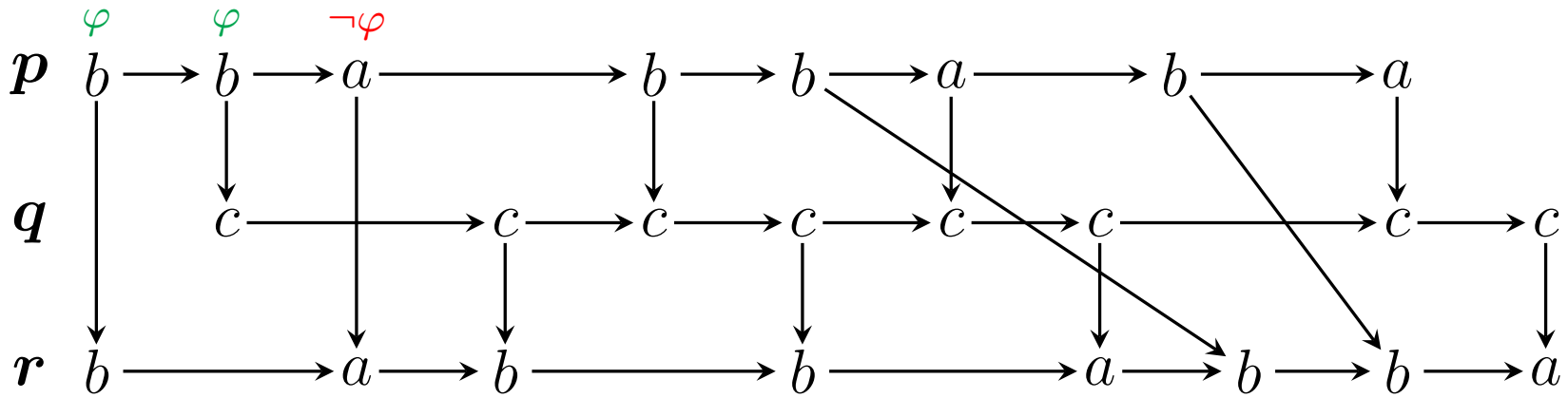
From PDL_{sf} to CFMs

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.

- $\varphi = b$



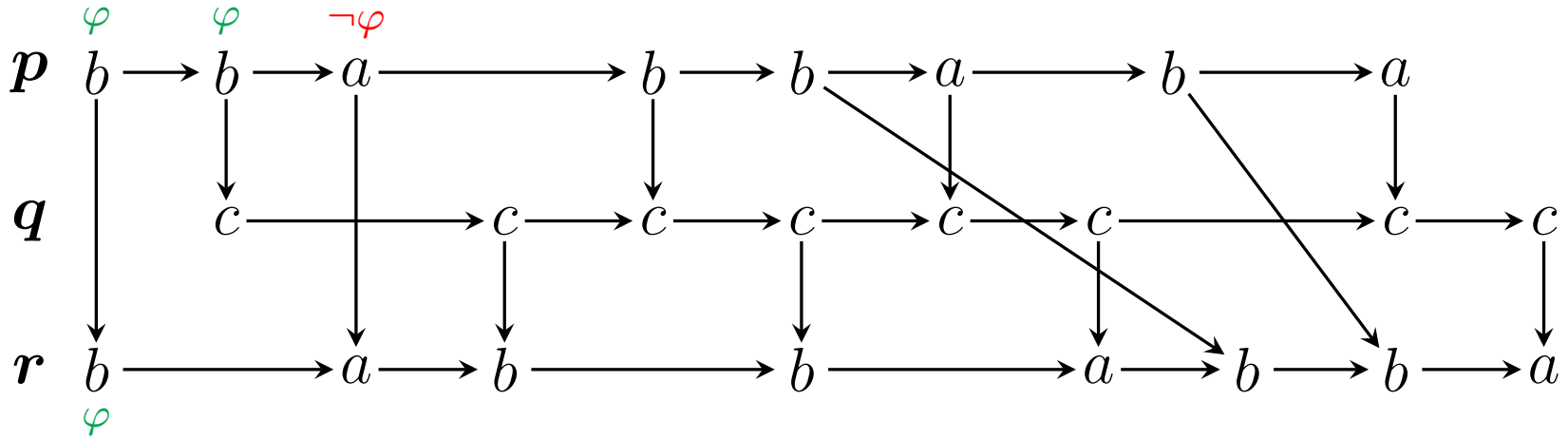
From PDL_{sf} to CFMs

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.

- $\varphi = b$



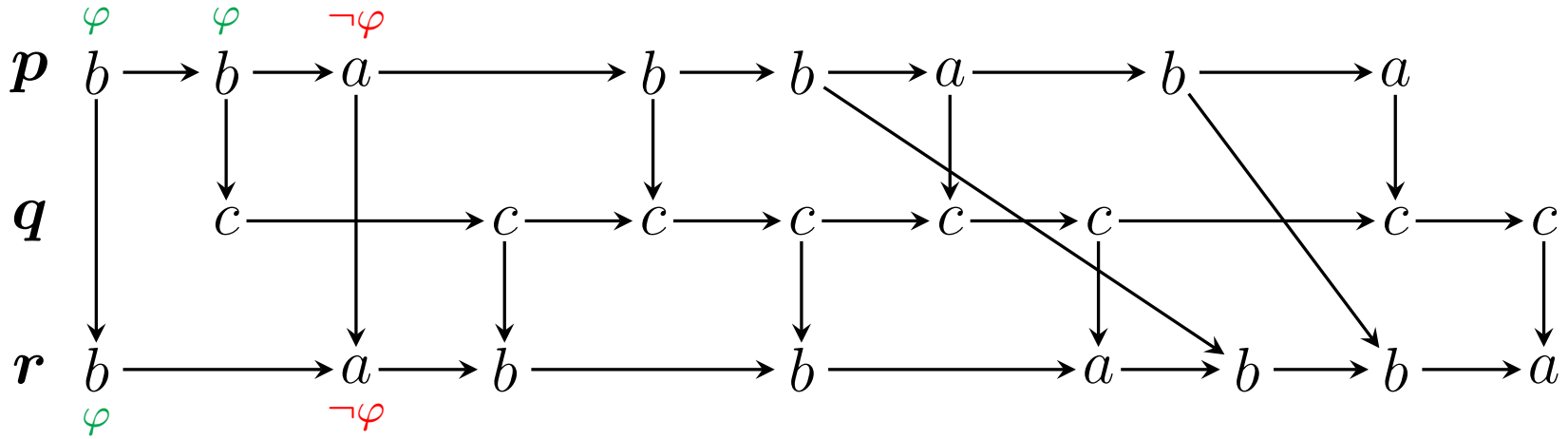
From PDL_{sf} to CFMs

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.

- $\varphi = b$



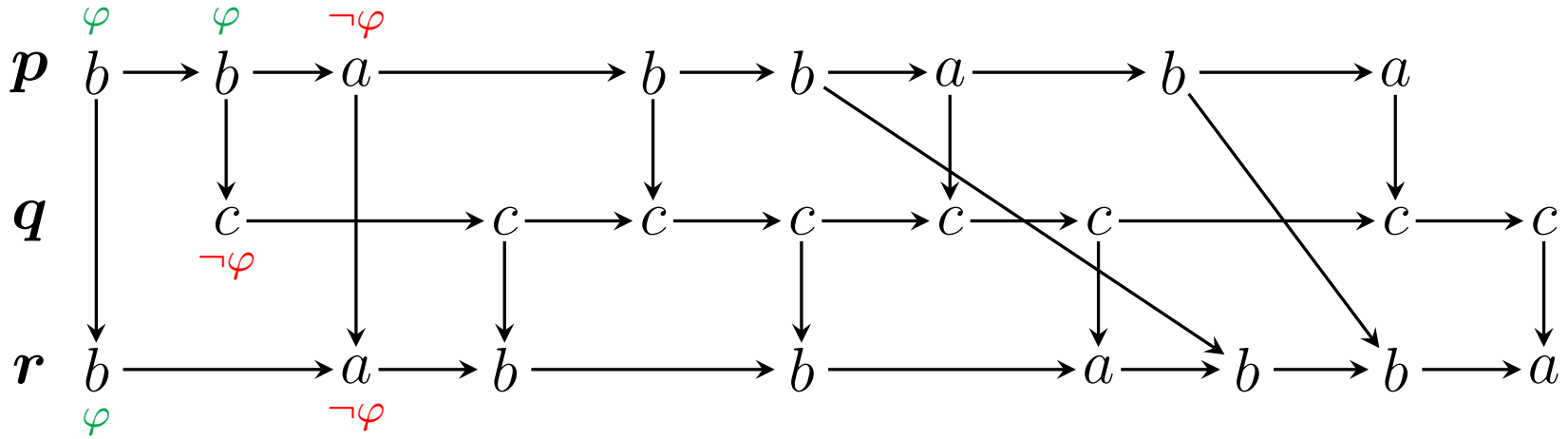
From PDL_{sf} to CFMs

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.

- $\varphi = b$



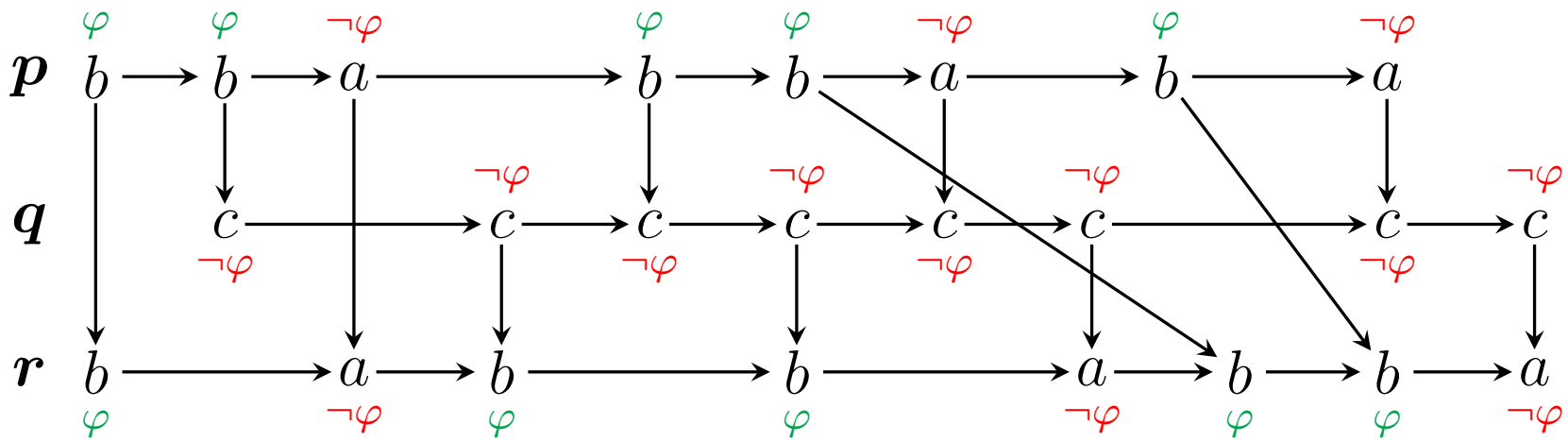
From PDL_{sf} to CFMs

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.

- $\varphi = b$



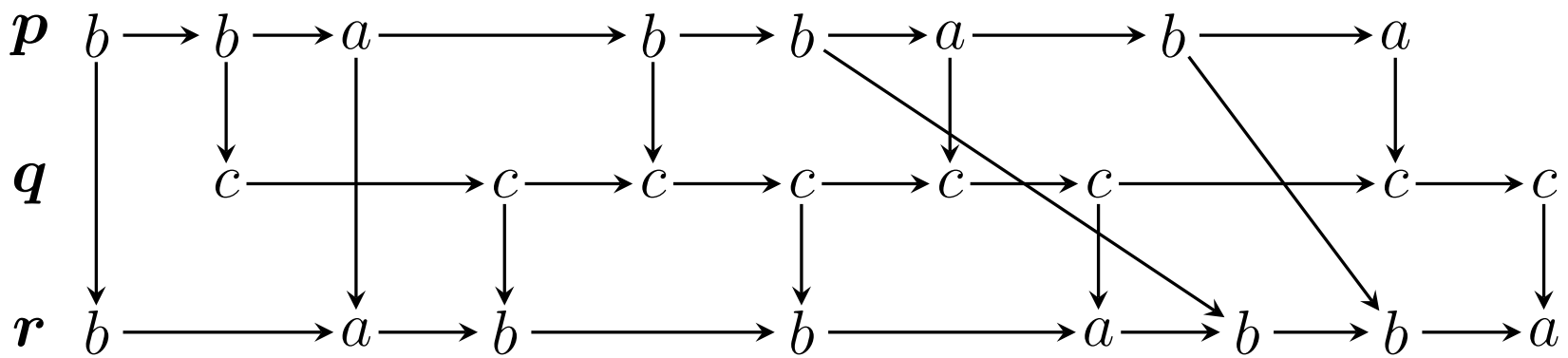
From PDL_{sf} to CFMs

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)



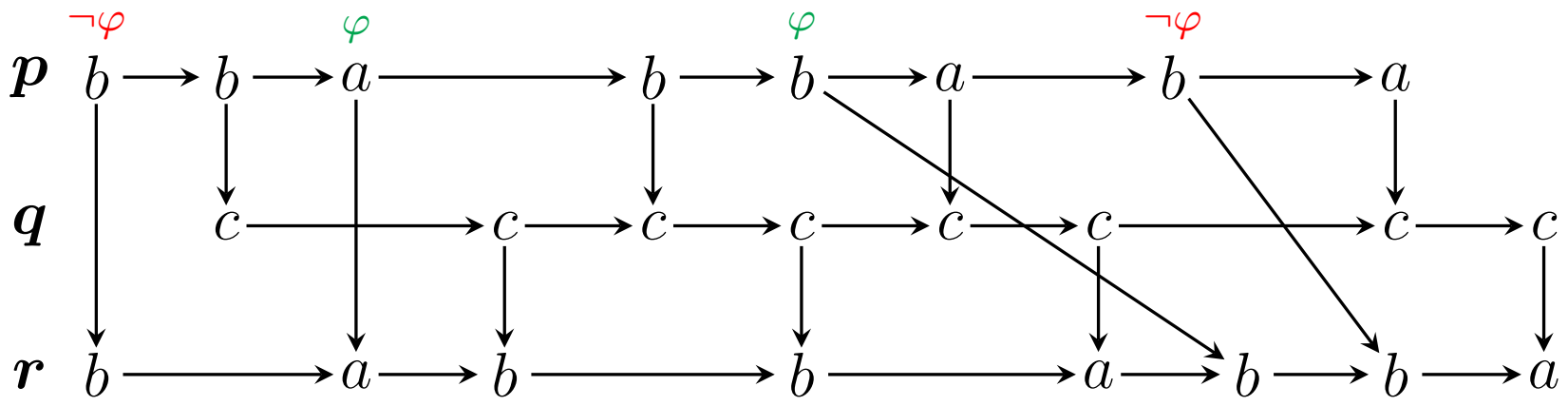
From PDL_{sf} to CFMs

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)



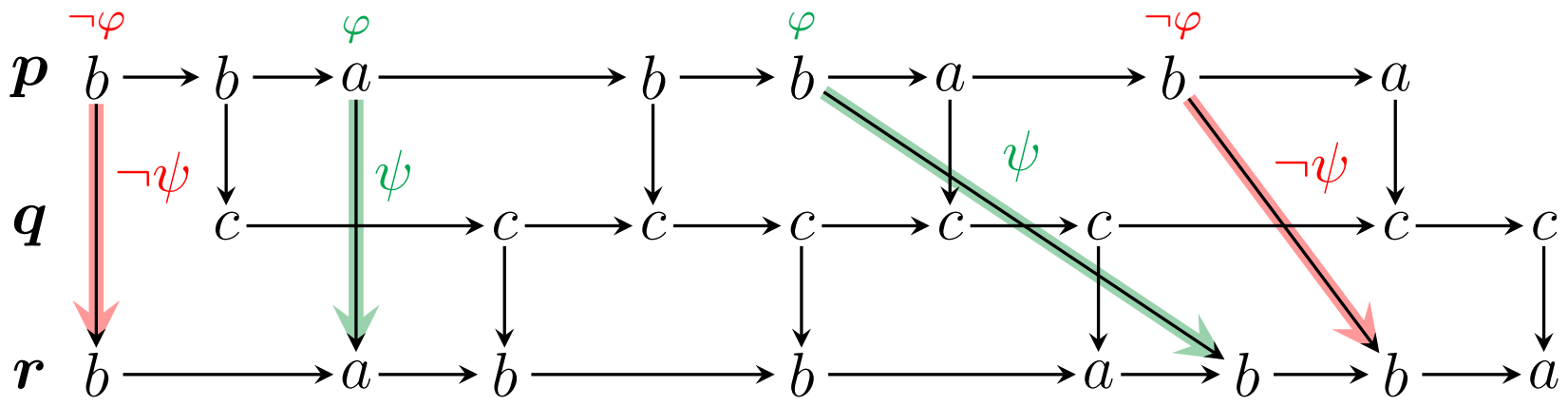
From PDL_{sf} to CFMs

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)



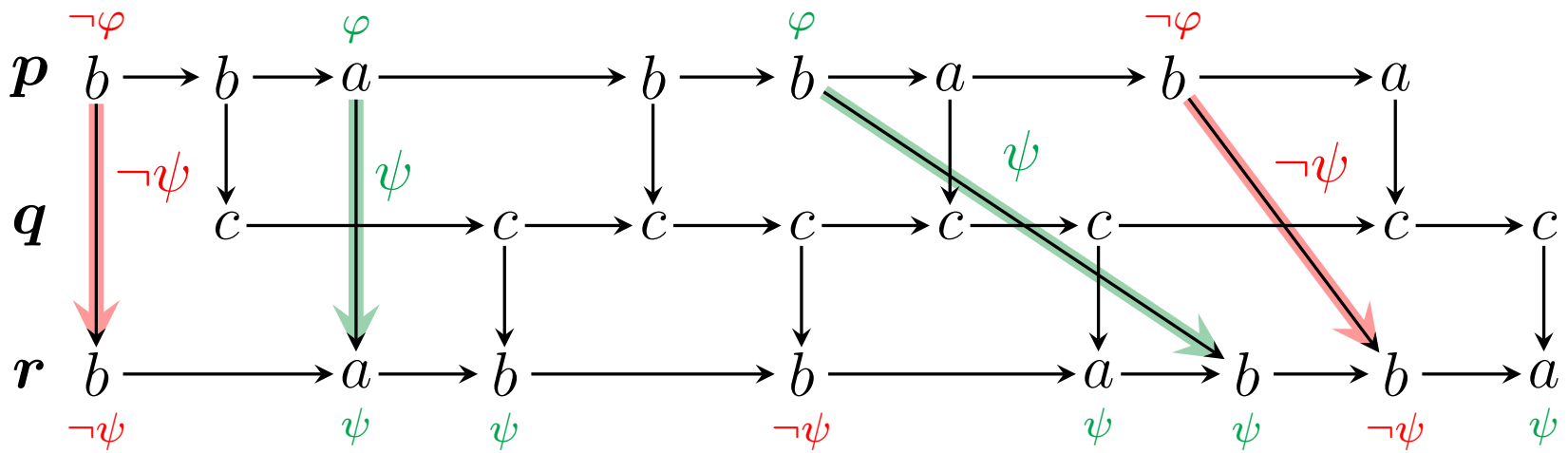
From PDL_{sf} to CFMs

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\!\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)



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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\!\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)
- $\varphi = \neg\psi$

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\!\searrow_{r} \rangle \, \psi$ (CFM for ψ given by induction)
- $\varphi = \neg\psi$
- $\ldots$

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.

- $\varphi = b$
- $\varphi = \langle {}^{p}\searrow_{r} \rangle \psi$ (CFM for ψ given by induction)
- $\varphi = \neg\psi$
- ...
- Only difficult case: $\varphi = \mathsf{Loop}(\pi)$

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathrm{Loop}(\pi)$.

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathsf{Loop}(\pi)$.

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

Translation of Loop formulas

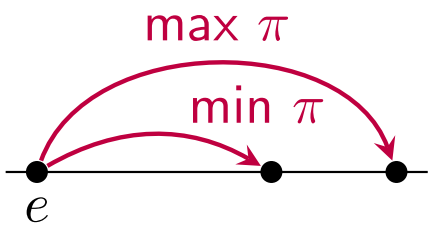
Needed: CFM that, at every event, evaluates $\mathsf{Loop}(\pi)$.

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

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathsf{Loop}(\pi)$.

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

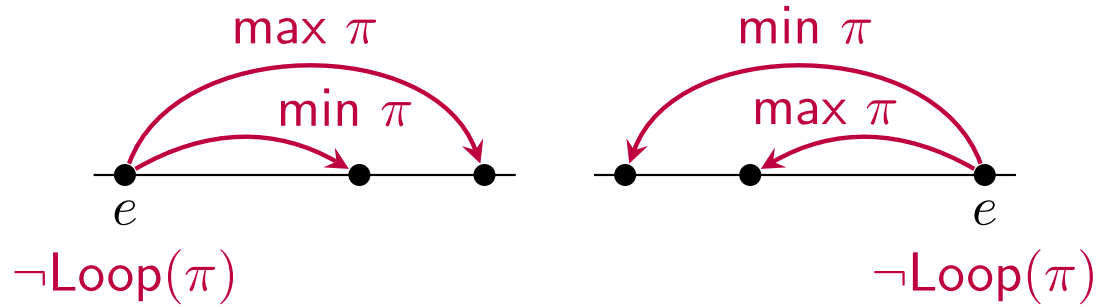


$\neg\mathsf{Loop}(\pi)$

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathsf{Loop}(\pi)$.

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

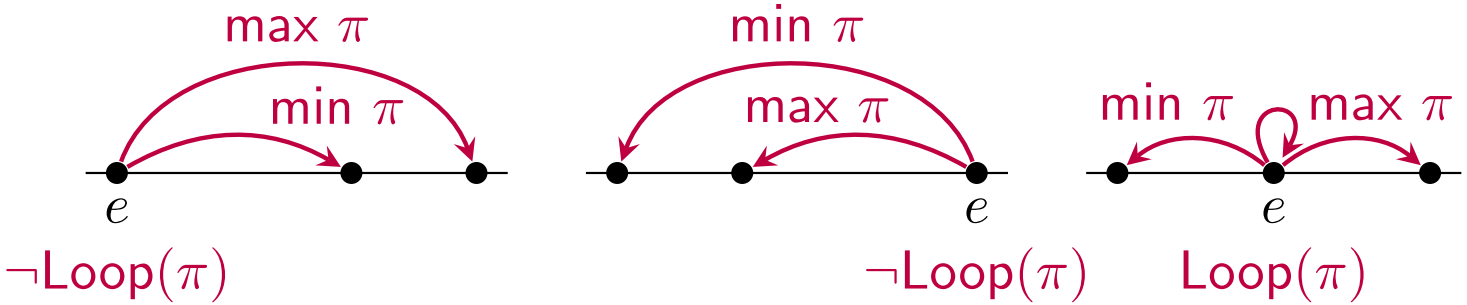


$\neg\mathsf{Loop}(\pi)$ $\neg\mathsf{Loop}(\pi)$

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathrm{Loop}(\pi)$.

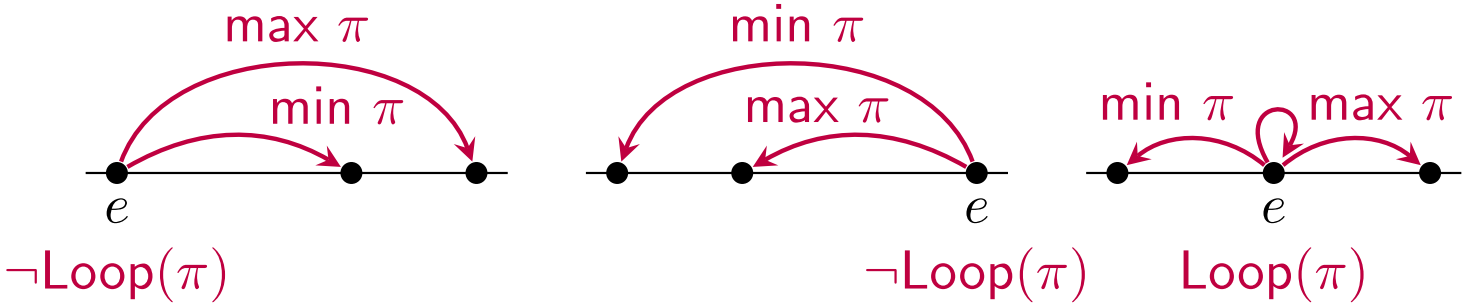
- If $e \not\models \langle\pi\rangle \wedge \langle\pi^{-1}\rangle$, then $e \not\models \mathrm{Loop}(\pi)$.
- If $e \models \langle\pi\rangle \wedge \langle\pi^{-1}\rangle$, three possible cases:



Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathrm{Loop}(\pi)$.

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

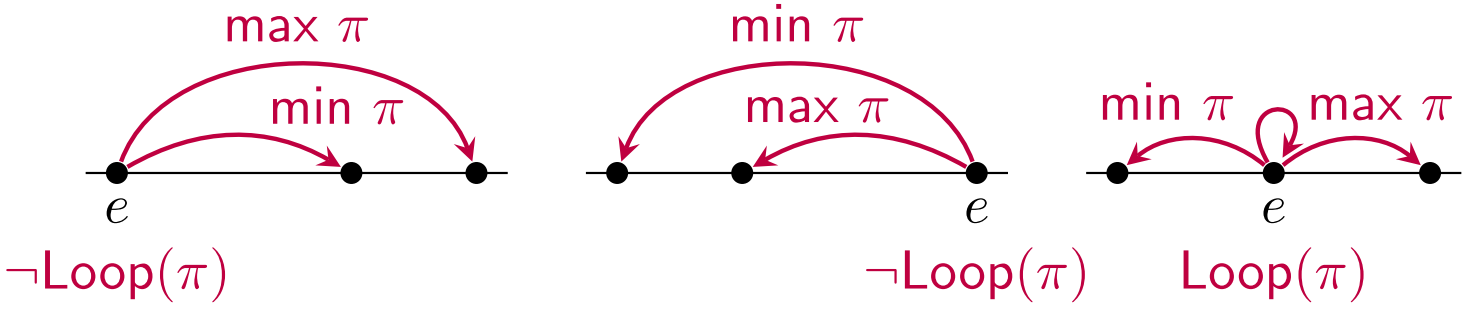


- To know which of these cases applies, it is enough to evaluate formulas $\mathrm{Loop}(\min\ \pi')$ and $\mathrm{Loop}(\max\ \pi')$.

Translation of Loop formulas

Needed: CFM that, at every event, evaluates $\mathrm{Loop}(\pi)$.

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



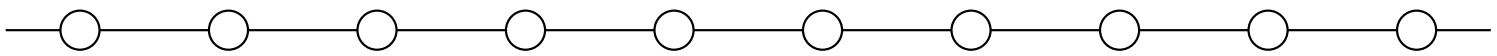
- To know which of these cases applies, it is enough to evaluate formulas $\mathrm{Loop}(\min \pi')$ and $\mathrm{Loop}(\max \pi')$.

1) Translate $\mathrm{Loop}(\min \pi')$ to $\mathrm{Loop}(\max \pi')$ into CFMs.
2) Use this to evaluate $\mathrm{Loop}(\pi)$ from left to right.

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

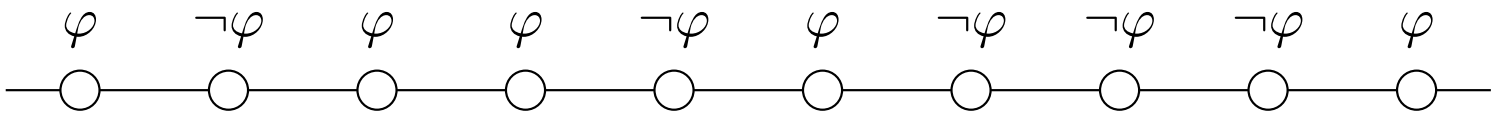
CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

- Guess for each event whether φ holds.



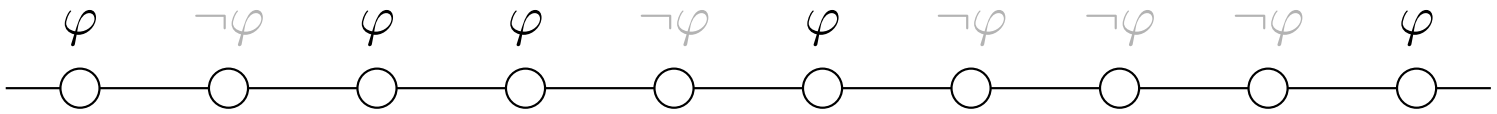
CFM for $\varphi = \mathsf{Loop}(\max\ \pi)$

- Guess for each event whether φ holds.



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

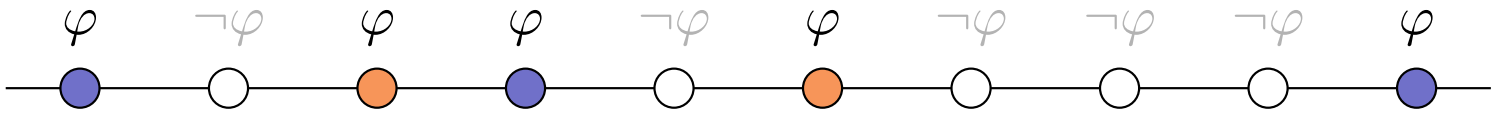
- Guess for each event whether φ holds.



- Check positive guesses:

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

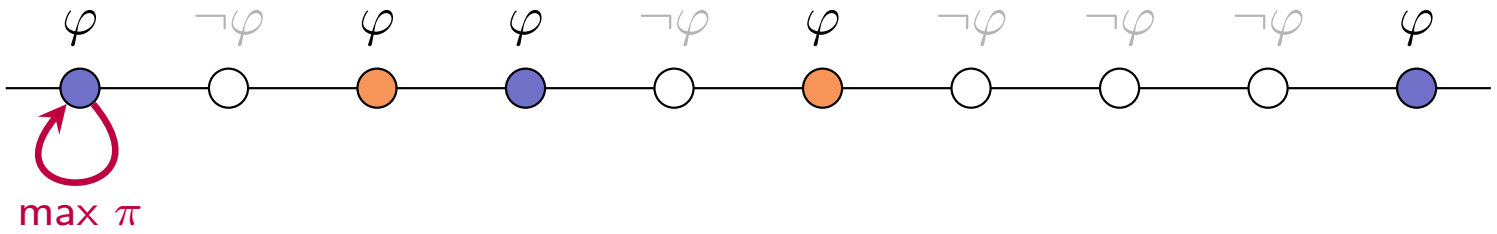
- Guess for each event whether φ holds.



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

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

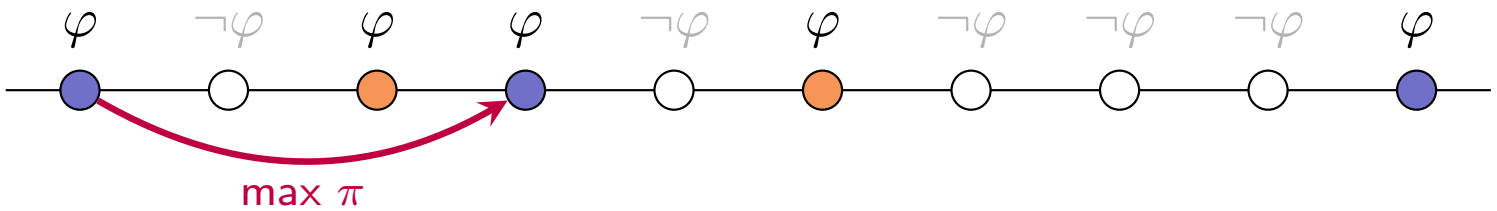
- Guess for each event whether φ holds.



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

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

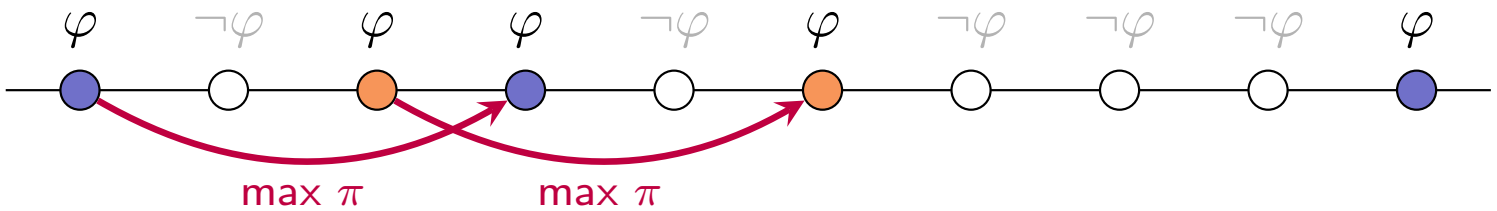
- Guess for each event whether φ holds.



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

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

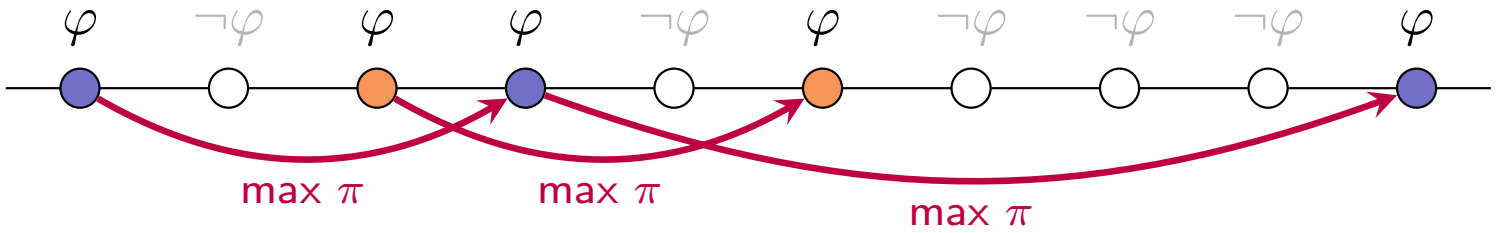
- Guess for each event whether φ holds.



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

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

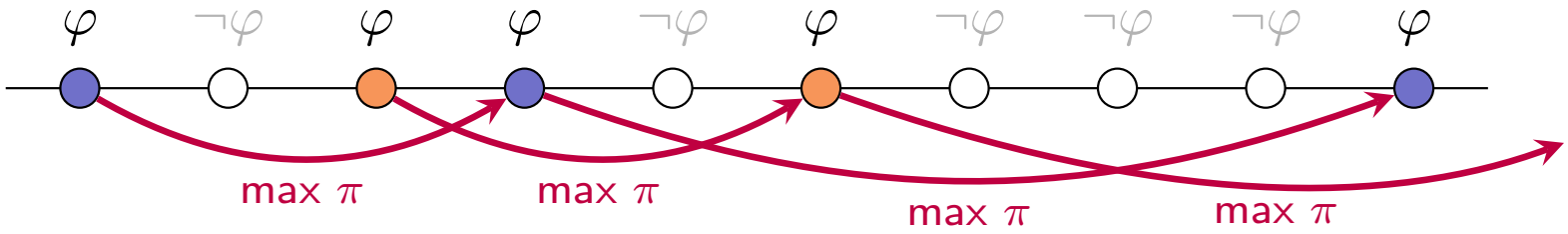
- Guess for each event whether φ holds.



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

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

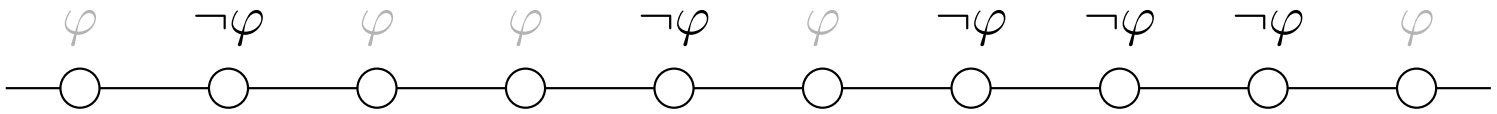
- Guess for each event whether φ holds.



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

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

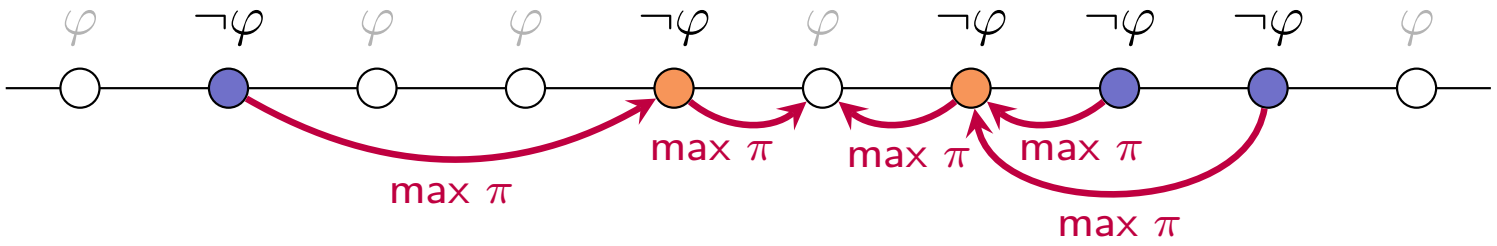
- Guess for each event whether φ holds.



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

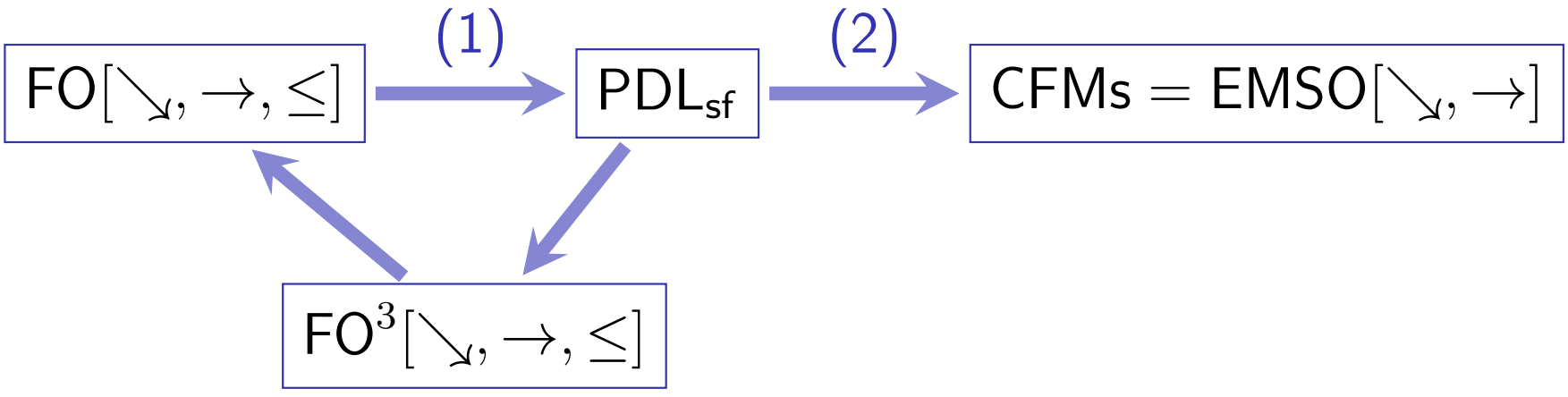
CFM for $\varphi = \mathrm{Loop}(\max\ \pi)$

- Guess for each event whether φ holds.

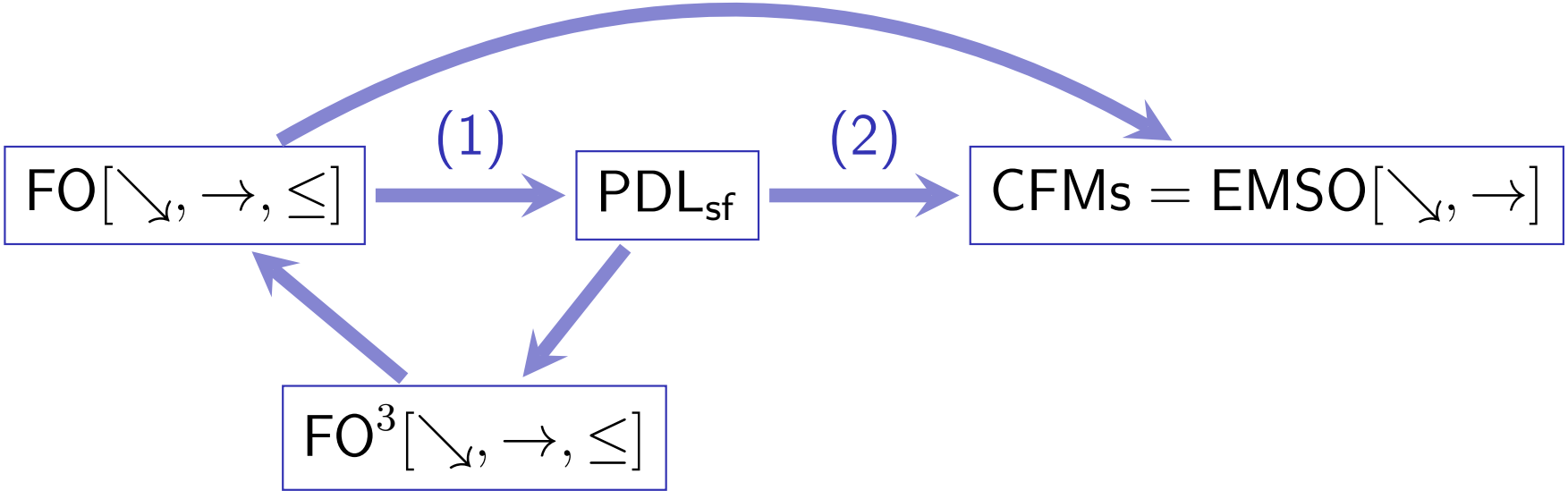


- Check positive guesses:
  - Alternately assign to φ-events colors ● or ●.
  - Check that the source and target color of $(\max\ \pi)$-paths are the same.
- Check negative guesses:
  - Guess a 2-coloring of the $\neg\varphi$-events.
  - Check that the source and target color of $(\max\ \pi)$-paths are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

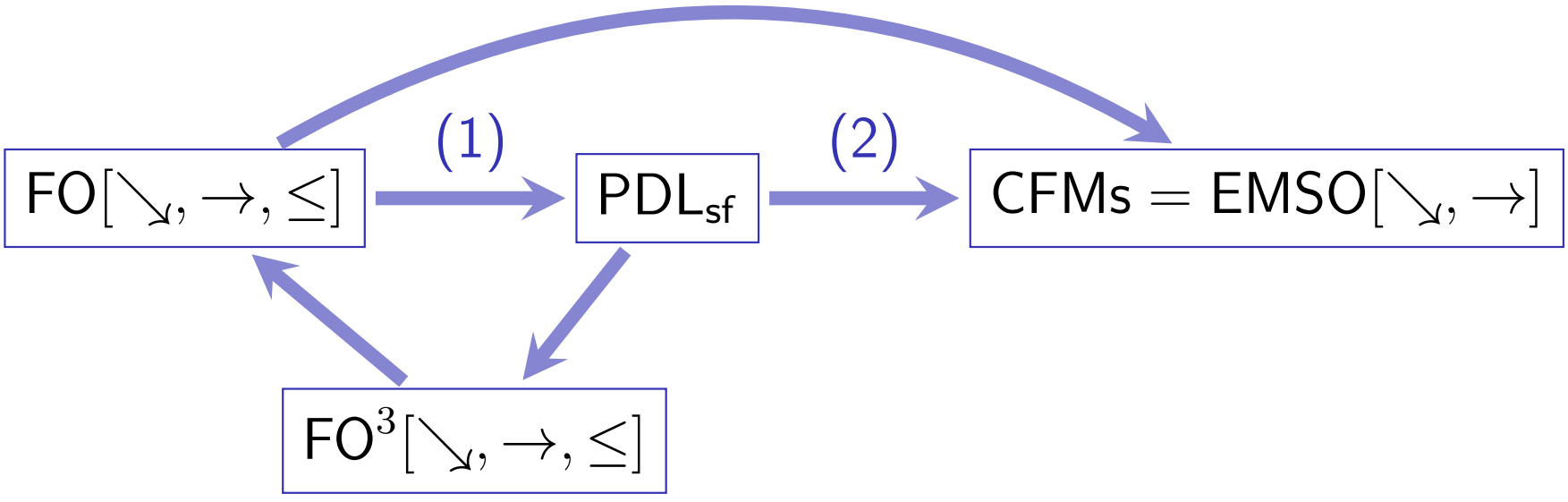
Summary


$FO[\searrow, \rightarrow, \leq]$
(1)
PDL_{sf}
(2)
$CFMs = EMSO[\searrow, \rightarrow]$
$FO^3[\searrow, \rightarrow, \leq]$

Summary

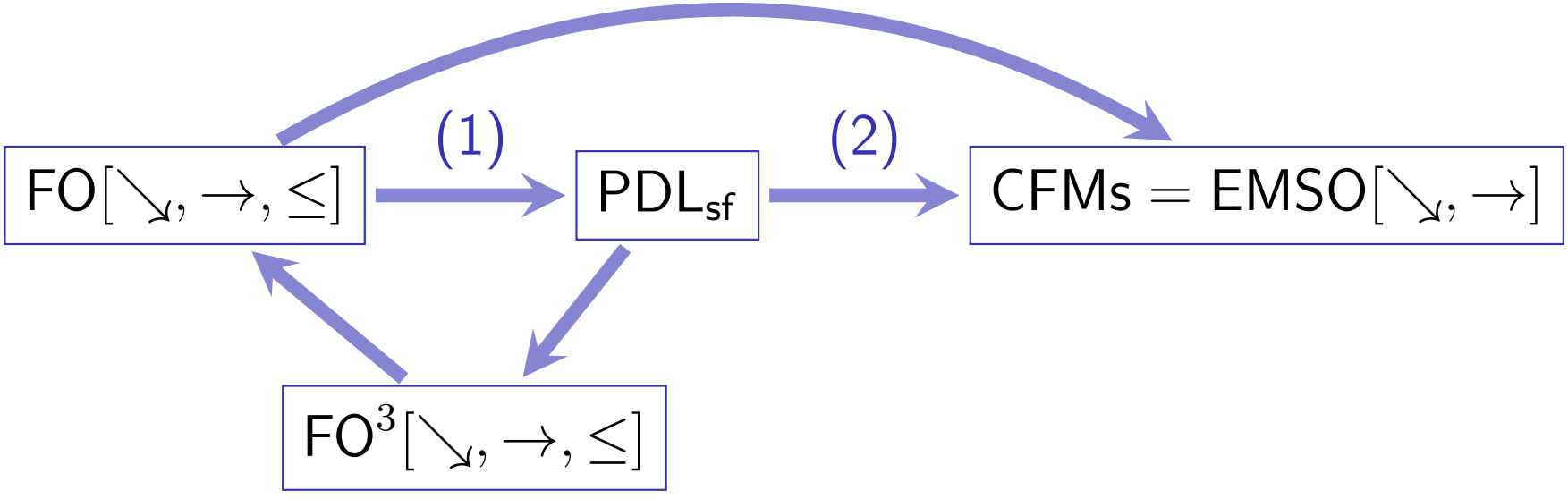

(1)
(2)
$FO[\searrow, \rightarrow, \leq]$
PDL_{sf}
$CFMs = EMSO[\searrow, \rightarrow]$
$FO^3[\searrow, \rightarrow, \leq]$

Summary



- FO sentence is a positive boolean combination of formulas $\exists x.\Phi(x)$ or $\neg\exists x.\Phi(x)$.

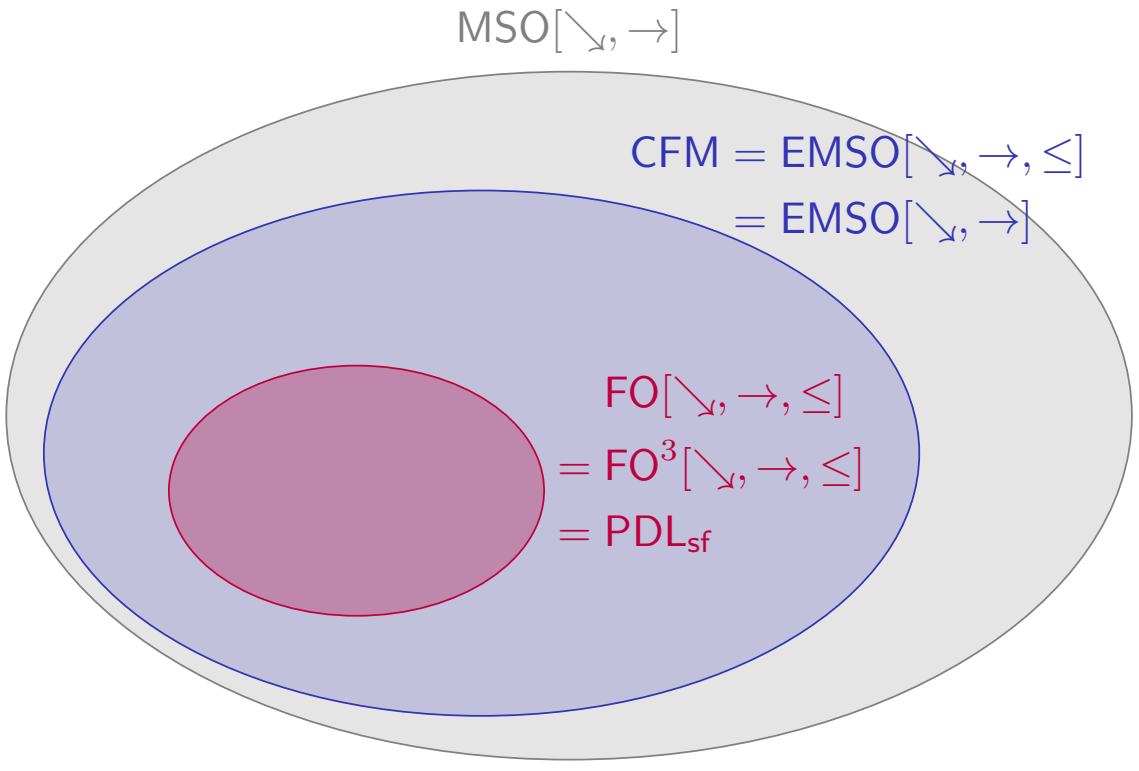
Summary



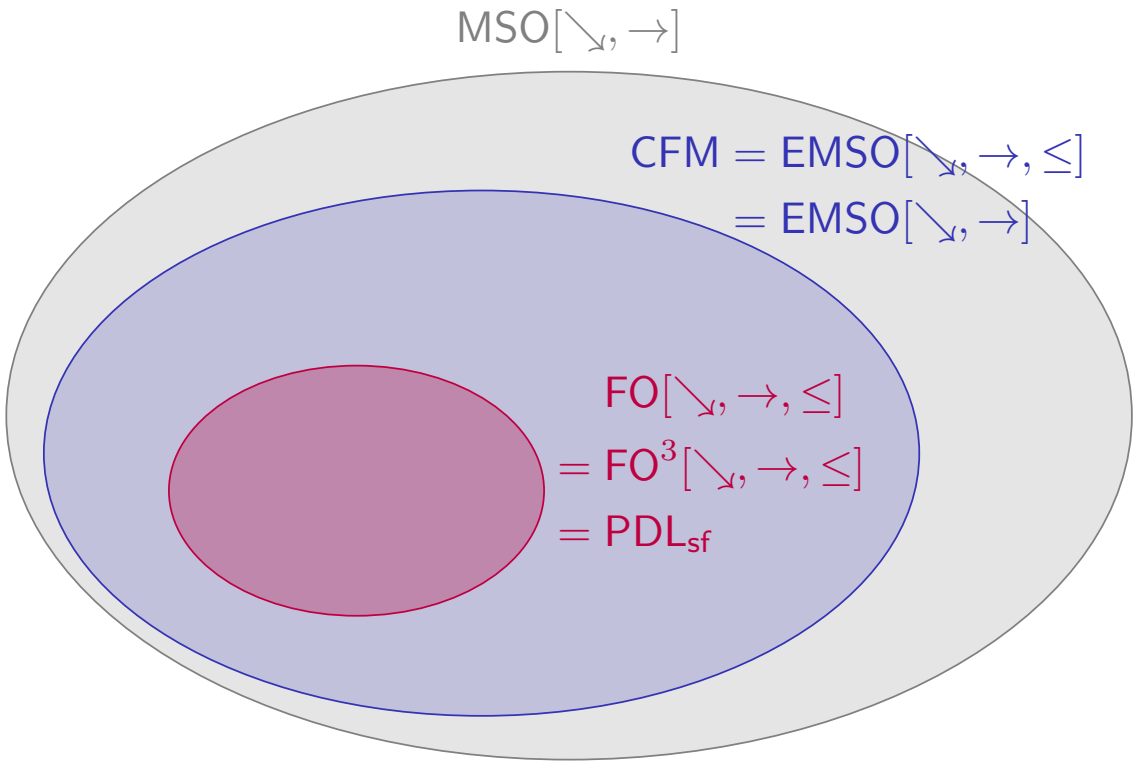
- FO sentence is a positive boolean combination of formulas $\exists x.\Phi(x)$ or $\neg\exists x.\Phi(x)$.
- Both types of formulas can be evaluated using the CFMs for $\Phi(x)$.

Conclusion

Conclusion



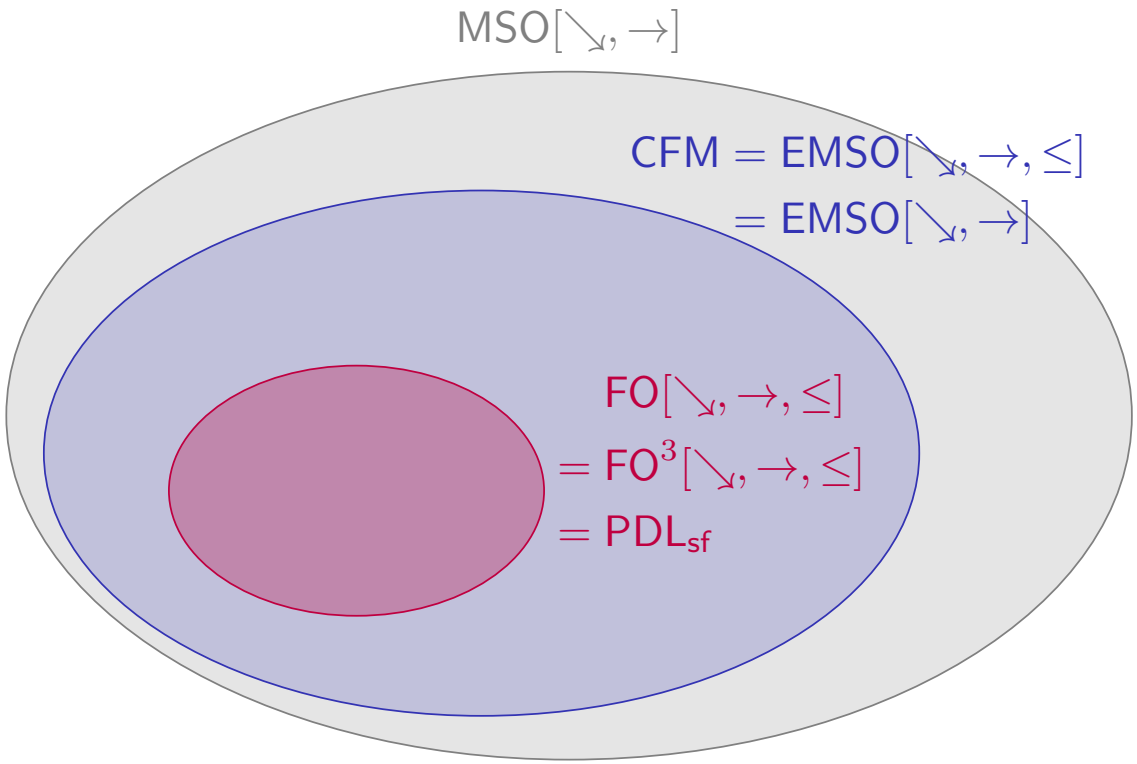
Conclusion



Open questions:

- Temporal logic that is expressively complete for FO?
- PDL (with star but without Loop) vs. CFM?

Conclusion



Open questions:

- Temporal logic that is expressively complete for FO?
- PDL (with star but without Loop) vs. CFM?

Thank you!