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

Benedikt Bollig
LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay, Cachan, France
bollig@lsv.fr

Marie Fortin
LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay, Cachan, France
fortin@lsv.fr

Paul Gastin
LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay, Cachan, France
gastin@lsv.fr

Abstract

Message sequence charts (MSCs) naturally arise as executions of communicating finite-state machines (CFMs), in which finite-state processes exchange messages through unbounded FIFO channels. We study the first-order logic of MSCs, featuring Lamport's happened-before relation. We introduce a star-free version of propositional dynamic logic (PDL) with loop and converse. Our main results state that (i) every first-order sentence can be transformed into an equivalent star-free PDL sentence (and conversely), and (ii) every star-free PDL sentence can be translated into an equivalent CFM. This answers an open question and settles the exact relation between CFMs and fragments of monadic second-order logic. As a byproduct, we show that first-order logic over MSCs has the three-variable property.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases communicating finite-state machines, first-order logic, happened-before relation

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2018.7

Related Version A full version of the paper is available at [2], `https://arxiv.org/abs/1804.10076`.

Funding Partly supported by ANR FREDDA (ANR-17-CE40-0013) and UMI RELAX.

1 Introduction

First-order (FO) logic can be considered, in many ways, a reference specification language. It plays a key role in automated theorem proving and formal verification. In particular, FO logic over finite or infinite words is central in the verification of reactive systems. When a word is understood as a total order that reflects a chronological succession of events, it represents an execution of a sequential system. Apart from being a natural concept in itself, FO logic over words enjoys manifold characterizations. It defines exactly the star-free languages and coincides with recognizability by aperiodic monoids or natural subclasses of finite (Büchi, respectively) automata (cf. [8, 31] for overviews). Moreover, linear-time temporal logics are

29th International Conference on Concurrency Theory (CONCUR 2018).
Editors: Sven Schewe and Lijun Zhang; Article No. 7; pp. 7:1–7:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

usually measured against their expressive power with respect to FO logic. For example, LTL is considered the yardstick temporal logic not least due to Kamp's famous theorem, stating that LTL and FO logic are expressively equivalent [21].

While FO logic on words is well understood, a lot remains to be said once concurrency enters into the picture. When several processes communicate through, say, unbounded first-in first-out (FIFO) channels, events are only partially ordered and a behavior, which is referred to as a *message sequence chart (MSC)*, reflects Lamport's happened-before relation: an event e happens before an event f if, and only if, there is a "message flow" path from e to f [23]. *Communicating finite-state machines* (CFMs) [5] are to MSCs what finite automata are to words: a canonical model of finite-state processes that communicate through unbounded FIFO channels. Therefore, the FO logic of MSCs can be considered a canonical specification language for such systems. Unfortunately, its study turned out to be difficult, since algebraic and automata-theoretic approaches that work for words, trees, or Mazurkiewicz traces do not carry over. In particular, until now, the following central problem remained open:

> *Can every first-order sentence be transformed into an equivalent communicating finite-state machine, without any channel bounds?*

Partial answers were given for CFMs with bounded channel capacity [14, 20, 22] and for fragments of FO that restrict the logic to bounded-degree predicates [4] or to two variables [1].

In this paper, we answer the general question positively. To do so, we make a detour through a variant of propositional dynamic logic (PDL) with loop and converse [11, 29]. Actually, we introduce *star-free* PDL, which serves as an interface between FO logic and CFMs. That is, there are two main tasks to accomplish:

(i) Translate every FO sentence into a star-free PDL sentence.
(ii) Translate every star-free PDL sentence into a CFM.

Both parts constitute results of own interest. In particular, step (i) implies that, over MSCs, FO logic has the three-variable property, i.e., every FO sentence over MSCs can be rewritten into one that uses only three different variable names. Note that this is already interesting in the special case of words, where it follows from Kamp's theorem [21]. It is also noteworthy that star-free PDL is a *two-dimensional* temporal logic in the sense of Gabbay et al. [12, 13]. Since every star-free PDL sentence is equivalent to some FO sentence, we actually provide a (higher-dimensional) temporal logic over MSCs that is expressively complete for FO logic.[1] While step (i) is based on purely logical considerations, step (ii) builds on new automata constructions that allow us to cope with the loop operator of PDL.

Combining (i) and (ii) yields the translation from FO logic to CFMs. It follows that CFMs are expressively equivalent to *existential* MSO logic. Moreover, we can derive self-contained proofs of several results on channel-bounded CFMs whose original proofs refer to involved constructions for Mazurkiewicz traces (cf. Section 5).

Related Work. Let us give a brief account of what was already known on the relation between logic and CFMs. In the 60s, Büchi, Elgot, and Trakhtenbrot proved that finite automata over words are expressively equivalent to monadic second-order logic [6, 10, 32]. Note that finite automata correspond to the special case of CFMs with a single process.

This classical result has been generalized to CFMs with bounded channels: Over *universally* bounded MSCs (where all possible linear extensions meet a given channel bound), deterministic CFMs are expressively equivalent to MSO logic [20, 22]. Over *existentially*

[1] It is open whether there is an equivalent one-dimensional one.

bounded MSCs (some linear extension meets the channel bound), CFMs are still expressively equivalent to MSO logic [14], but inherently nondeterministic [15]. The proofs of these characterizations reduce message-passing systems to finite-state shared-memory systems so that deep results from Mazurkiewicz trace theory [9] can be applied.

This generic approach is no longer applicable when the restriction on the channel capacity is dropped. Actually, in general, CFMs do not capture MSO logic [4]. On the other hand, they are expressively equivalent to existential MSO logic when we discard the happened-before relation [4] or when restricting to two first-order variables [1]. Both results rely on normal forms of FO logic, due to Hanf [19] and Scott [17], respectively. However, MSCs with the happened-before relation are structures of *unbounded* degree (while Hanf's normal form requires structures of bounded degree), and we consider FO logic with *arbitrarily* many variables (while Scott's normal form only applies to two-variable logic). That is, neither approach is applicable in our case.

Finally, there exists a translation of a loop-free PDL into CFMs [3]. As our star-free PDL has a loop operator, we cannot exploit [3] either.

Outline. In Section 2, we recall basic notions such as MSCs, FO logic, and CFMs. Moreover, we state one of our main results: the translation of FO formulas into CFMs. Section 3 presents star-free PDL and proves that it captures FO logic. In Section 4, we establish the translation of star-free PDL into CFMs. We conclude in Section 5 mentioning applications of our results. Some proof details can be found in the long version [2].

2 Preliminaries

We consider message-passing systems in which processes communicate through unbounded FIFO channels. We fix a nonempty finite set of *processes* P and a nonempty finite set of *labels* Σ. For all $p, q \in P$ such that $p \neq q$, there is a channel (p, q) that allows p to send messages to q. The set of channels is denoted Ch.

In the following, we define message sequence charts, which represent executions of a message-passing system, and logics to reason about them. Then, we recall the definition of communicating finite-state machines and state one of our main results.

2.1 Message Sequence Charts

A *message sequence chart (MSC)* (over P and Σ) is a graph $M = (E, \rightarrow, \lhd, \mathit{loc}, \lambda)$ with nonempty finite set of nodes E, edge relations $\rightarrow, \lhd \subseteq E \times E$, and node-labeling functions $\mathit{loc} \colon E \to P$ and $\lambda \colon E \to \Sigma$. An example MSC is depicted in Figure 1. A node $e \in E$ is an *event* that is executed by process $\mathit{loc}(e) \in P$. In particular, $E_p := \{e \in E \mid \mathit{loc}(e) = p\}$ is the set of events located on p. The label $\lambda(e) \in \Sigma$ may provide more information about e such as the message that is sent/received at e or "enter critical section" or "output some value".

Edges describe causal dependencies between events:

- The relation $\rightarrow$ contains *process edges*. They connect successive events executed by the same process. That is, we actually have $\rightarrow \subseteq \bigcup_{p \in P}(E_p \times E_p)$. Every process p is sequential so that $\rightarrow \cap (E_p \times E_p)$ must be the direct-successor relation of some total order on E_p. We let $\leq_{\mathsf{proc}} := \rightarrow^*$ and $<_{\mathsf{proc}} := \rightarrow^+$.
- The relation $\lhd$ contains *message edges*. If $e \lhd f$, then e is a *send event* and f is the corresponding *receive event*. In particular, $(\mathit{loc}(e), \mathit{loc}(f)) \in \mathit{Ch}$. Each event is part of at most one message edge. An event that is neither a send nor a receive event is called *internal*. Moreover, for all $(p, q) \in \mathit{Ch}$ and $(e, f), (e', f') \in \lhd \cap (E_p \times E_q)$, we have $e \leq_{\mathsf{proc}} e'$ iff $f \leq_{\mathsf{proc}} f'$ (which guarantees a FIFO behavior).

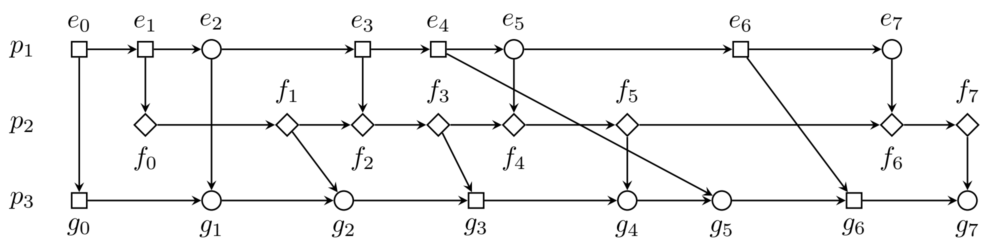


Figure 1 A message sequence chart (MSC).

We require that $\rightarrow \cup \lhd$ be acyclic (intuitively, messages cannot travel backwards in time). The associated partial order is denoted $\leq := (\rightarrow \cup \lhd)^*$ with strict part $< = (\rightarrow \cup \lhd)^+$. We do not distinguish isomorphic MSCs. Let $\mathbb{MSC}(P, \Sigma)$ denote the set of MSCs over P and Σ.

Actually, MSCs are very similar to the space-time diagrams from Lamport's seminal paper [23], and $\leq$ is commonly referred to as the *happened-before relation*.

It is worth noting that, when P is a singleton, an MSC with events $e_1 \rightarrow e_2 \rightarrow \ldots \rightarrow e_n$ can be identified with the word $\lambda(e_1)\lambda(e_2)\ldots\lambda(e_n) \in \Sigma^*$.

▶ **Example 1.** Consider the MSC from Figure 1 over $P = \{p_1, p_2, p_3\}$ and $\Sigma = \{\square, \bigcirc, \diamond\}$. We have, for instance, $E_{p_1} = \{e_0, \ldots, e_7\}$. The process relation is given by $e_i \rightarrow e_{i+1}$, $f_i \rightarrow f_{i+1}$, and $g_i \rightarrow g_{i+1}$ for all $i \in \{0, \ldots, 6\}$. Concerning the message relation, we have $e_1 \lhd f_0$, $e_4 \lhd g_5$, etc. Moreover, $e_2 \leq f_3$, but neither $e_2 \leq f_1$ nor $f_1 \leq e_2$.

2.2 MSO Logic and Its Fragments

Next, we give an account of *monadic second-order* (MSO) logic and its fragments. Note that we restrict our attention to MSO logic interpreted *over MSCs*. We fix an infinite supply $\mathcal{V}_{\mathsf{event}} = \{x, y, \ldots\}$ of first-order variables, which range over events of an MSC, and an infinite supply $\mathcal{V}_{\mathsf{set}} = \{X, Y, \ldots\}$ of second-order variables, ranging over sets of events. The syntax of MSO (we consider that P and Σ are fixed) is given as follows:

$$\Phi \ ::= \ p(x) \mid a(x) \mid x = y \mid x \rightarrow y \mid x \lhd y \mid x \leq y \mid x \in X \mid \Phi \vee \Phi \mid \neg\Phi \mid \exists x.\Phi \mid \exists X.\Phi$$

where $p \in P$, $a \in \Sigma$, $x, y \in \mathcal{V}_{\mathsf{event}}$, and $X \in \mathcal{V}_{\mathsf{set}}$. We use the usual abbreviations to also include implication $\Longrightarrow$, conjunction $\wedge$, and universal quantification $\forall$. Moreover, the relation $x \leq_{\mathsf{proc}} y$ can be defined by $x \leq y \wedge \bigvee_{p \in P} p(x) \wedge p(y)$. We write $\mathsf{Free}(\Phi)$ the set of free variables of Φ.

Let $M = (E, \rightarrow, \lhd, loc, \lambda)$ be an MSC. An *interpretation* (for M) is a mapping $\nu \colon \mathcal{V}_{\mathsf{event}} \cup \mathcal{V}_{\mathsf{set}} \rightarrow E \cup 2^E$ assigning to each $x \in \mathcal{V}_{\mathsf{event}}$ an event $\nu(x) \in E$, and to each $X \in \mathcal{V}_{\mathsf{set}}$ a set of events $\nu(X) \subseteq E$. We write $M, \nu \models \Phi$ if M satisfies Φ when the free variables of Φ are interpreted according to ν. Hereby, satisfaction is defined in the usual manner. In fact, whether $M, \nu \models \Phi$ holds or not only depends on the interpretation of variables that occur free in Φ. Thus, we may restrict ν to any set of variables that contains at least all free variables. For example, for $\Phi(x, y) = (x \lhd y)$, we have $M, [x \mapsto e, y \mapsto f] \models \Phi(x, y)$ iff $e \lhd f$. For a *sentence* $\Phi \in$ MSO (without free variables), we define $L(\Phi) := \{M \in \mathbb{MSC}(P, \Sigma) \mid M \models \Phi\}$.

We say that two formulas Φ and Φ' are *equivalent*, written $\Phi \equiv \Phi'$, if, for all MSCs $M = (E, \rightarrow, \lhd, loc, \lambda)$ and interpretations $\nu \colon \mathcal{V}_{\mathsf{event}} \cup \mathcal{V}_{\mathsf{set}} \rightarrow E \cup 2^E$, we have $M, \nu \models \Phi$ iff $M, \nu \models \Phi'$.

Let us identify two important fragments of MSO logic: *First-order* (FO) formulas do not make use of second-order quantification (however, they may contain formulas $x \in X$). Moreover, *existential* MSO (EMSO) formulas are of the form $\exists X_1 \ldots \exists X_n.\Phi$ with $\Phi \in$ FO.

Let $\mathcal{F}$ be MSO or EMSO or FO and let $R \subseteq \{\rightarrow, \lhd, \leq\}$. We obtain the logic $\mathcal{F}[R]$ by restricting $\mathcal{F}$ to formulas that do not make use of $\{\rightarrow, \lhd, \leq\} \setminus R$. Note that $\mathcal{F} = \mathcal{F}[\rightarrow, \lhd, \leq]$. Moreover, we let $\mathcal{L}(\mathcal{F}[R]) := \{L(\Phi) \mid \Phi \in \mathcal{F}[R]$ is a sentence$\}$.

Since the reflexive transitive closure of an MSO-definable binary relation is MSO-definable, MSO and MSO$[\rightarrow, \lhd]$ have the same expressive power: $\mathcal{L}(\text{MSO}[\rightarrow, \lhd, \leq]) = \mathcal{L}(\text{MSO}[\rightarrow, \lhd])$. However, MSO$[\leq]$ (without the message relation) is strictly weaker than MSO [4].

▶ **Example 2.** We give an FO formula that allows us to recover, at some event f, the most recent event e that happened in the past on, say, process p. More precisely, we define the predicate $\mathit{latest}_p(x, y)$ as $x \leq y \wedge p(x) \wedge \forall z\big((z \leq y \wedge p(z)) \implies z \leq x\big)$. The "gossip language" says that process q always maintains the latest information that it can have about p. Thus, it is defined by $\Phi^{\mathsf{gossip}}_{p,q} = \forall x \forall y.\big((\mathit{latest}_p(x, y) \wedge q(y)) \implies \bigvee_{a \in \Sigma}(a(x) \wedge a(y))\big) \in \text{FO}^3[\leq]$. For example, for $P = \{p_1, p_2, p_3\}$ and $\Sigma = \{\square, \bigcirc, \diamond\}$, the MSC M from Figure 1 is contained in $L(\Phi^{\mathsf{gossip}}_{p_1,p_3})$. In particular, $M, [x \mapsto e_5, y \mapsto g_5] \models \mathit{latest}_{p_1}(x, y)$ and $\lambda(e_5) = \lambda(g_5) = \bigcirc$.

2.3 Communicating Finite-State Machines

In a communicating finite-state machine, each process $p \in P$ can perform internal actions of the form $\langle a \rangle$, where $a \in \Sigma$, or send/receive messages from a finite set of messages Msg. A send action $\langle a, !_q m \rangle$ of process p writes message $m \in \mathit{Msg}$ to channel (p, q), and performs $a \in \Sigma$. A receive action $\langle a, ?_q m \rangle$ reads message m from channel (q, p). Accordingly, we let $\mathit{Act}_p(\mathit{Msg}) := \{\langle a \rangle \mid a \in \Sigma\} \cup \{\langle a, !_q m \rangle \mid a \in \Sigma,\ m \in \mathit{Msg},\ q \in P \setminus \{p\}\} \cup \{\langle a, ?_q m \rangle \mid a \in \Sigma,\ m \in \mathit{Msg},\ q \in P \setminus \{p\}\}$ denote the set of possible actions of process p.

A *communicating finite-state machine (CFM)* over P and Σ is a tuple $((\mathcal{A}_p)_{p \in P}, \mathit{Msg}, \mathit{Acc})$ consisting of a finite set of messages Msg and a finite-state transition system $\mathcal{A}_p = (S_p, \iota_p, \Delta_p)$ for each process p, with finite set of states S_p, initial state $\iota_p \in S_p$, and transition relation $\Delta_p \subseteq S_p \times \mathit{Act}_p(\mathit{Msg}) \times S_p$. Moreover, we have an acceptance condition $\mathit{Acc} \subseteq \prod_{p \in P} S_p$.

Given a transition $t = (s, \alpha, s') \in \Delta_p$, we let $\mathit{source}(t) = s$ and $\mathit{target}(t) = s'$ denote the source and target states of t. In addition, if $\alpha = \langle a \rangle$, then t is an *internal transition* and we let $\mathit{label}(t) = a$. If $\alpha = \langle a, !_q m \rangle$, then t is a *send transition* and we let $\mathit{label}(t) = a$, $\mathit{msg}(t) = m$, and $\mathit{receiver}(t) = q$. Finally, if $\alpha = \langle a, ?_q m \rangle$, then t is a *receive transition* with $\mathit{label}(t) = a$, $\mathit{msg}(t) = m$, and $\mathit{sender}(t) = q$.

A *run* ρ of $\mathcal{A}$ on an MSC $M = (E, \rightarrow, \lhd, \mathit{loc}, \lambda) \in \mathbb{MSC}(P, \Sigma)$ is a mapping associating with each event $e \in E_p$ a transition $\rho(e) \in \Delta_p$, and satisfying the following conditions:

1. for all events $e \in E$, we have $\mathit{label}(\rho(e)) = \lambda(e)$,
2. for all $\rightarrow$-minimal events $e \in E$, we have $\mathit{source}(\rho(e)) = \iota_p$, where $p = \mathit{loc}(e)$,
3. for all process edges $(e, f) \in \rightarrow$, we have $\mathit{target}(\rho(e)) = \mathit{source}(\rho(f))$,
4. for all internal events $e \in E$, $\rho(e)$ is an internal transition, and
5. for all message edges $e \lhd f$, $\rho(e)$ and $\rho(f)$ are respectively send and receive transitions such that $\mathit{msg}(\rho(e)) = \mathit{msg}(\rho(f))$, $\mathit{receiver}(\rho(e)) = \mathit{loc}(f)$, and $\mathit{sender}(\rho(f)) = \mathit{loc}(e)$.

To determine whether ρ is accepting, we collect the last state s_p of every process p. If $E_p \neq \emptyset$, we let $s_p = \mathit{target}(\rho(e))$, where e is the last event of E_p. Otherwise, $s_p = \iota_p$. We say that ρ is *accepting* if $(s_p)_{p \in P} \in \mathit{Acc}$.

The *language* $L(\mathcal{A})$ of $\mathcal{A}$ is the set of MSCs M such that there exists an accepting run of $\mathcal{A}$ on M. Moreover, $\mathcal{L}(\text{CFM}) := \{L(\mathcal{A}) \mid \mathcal{A}$ is a CFM$\}$. Recall that, for these definitions, we have fixed P and Σ.

One of our main results states that CFMs and EMSO logic are expressively equivalent. This solves a problem that was stated as open in [15]:

▶ **Theorem 3.** $\mathcal{L}(\mathrm{EMSO}[\rightarrow, \lhd, \leq]) = \mathcal{L}(\mathrm{CFM})$.

It is standard to prove $\mathcal{L}(\mathrm{CFM}) \subseteq \mathcal{L}(\mathrm{EMSO}[\rightarrow, \lhd])$: The formula guesses an assignment of transitions to events in terms of existentially quantified second-order variables (one for each transition) and then checks, in its first-order kernel, that the assignment is indeed an (accepting) run. As, moreover, the class $\mathcal{L}(\mathrm{CFM})$ is closed under projection, the proof of Theorem 3 comes down to the proposition below (whose proof is spread over Sections 3 and 4). Note that the translation from $\mathrm{FO}[\rightarrow, \lhd, \leq]$ to CFMs is inherently non-elementary, already when $|P| = 1$ [28].

▶ **Proposition 4.** $\mathcal{L}(\mathrm{FO}[\rightarrow, \lhd, \leq]) \subseteq \mathcal{L}(\mathrm{CFM})$.

3 Star-Free Propositional Dynamic Logic

In this section, we introduce a star-free version of propositional dynamic logic and show that it is expressively equivalent to $\mathrm{FO}[\rightarrow, \lhd, \leq]$. This is the second main result of the paper. Then, in Section 4, we show how to translate star-free PDL formulas into CFMs.

3.1 Syntax and Semantics

Originally, propositional dynamic logic (PDL) has been used to reason about program schemas and transition systems [11]. Since then, PDL and its extension with intersection and converse have developed a rich theory with applications in artificial intelligence and verification [7, 16, 18, 24, 25]. It has also been applied in the context of MSCs [3, 27].

Here, we introduce a *star-free* version of PDL, denoted $\mathrm{PDL}_{\mathsf{sf}}$. It will serve as an "interface" between FO logic and CFMs. The syntax of $\mathrm{PDL}_{\mathsf{sf}}$ and its fragment $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ is given by the following grammar:

$$\mathrm{PDL}_{\mathsf{sf}} = \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}, \cup, \cap, \mathsf{c}]$$

$$\begin{array}{lll} \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}] & \xi ::= \mathsf{E}\,\varphi \mid \xi \vee \xi \mid \neg\xi & \\ & \varphi ::= p \mid a \mid \varphi \vee \varphi \mid \neg\varphi \mid \langle\pi\rangle\,\varphi \mid \mathsf{Loop}(\pi) & \\ & \pi ::= \rightarrow \mid \leftarrow \mid \lhd_{p,q} \mid \lhd_{p,q}^{-1} \mid \xrightarrow{\varphi} \mid \xleftarrow{\varphi} \mid \mathsf{jump}_{p,r} \mid \{\varphi\}? \mid \pi \cdot \pi & \mid \pi \cup \pi \mid \pi \cap \pi \mid \pi^{\mathsf{c}} \end{array}$$

where $p, r \in P$, $q \in P \setminus \{p\}$, and $a \in \Sigma$. We refer to ξ as a *sentence*, to φ as an *event formula*, and to π as a *path formula*. We name the logic star-free because we use the operators $(\cup, \cap, \mathsf{c}, \cdot)$ of star-free regular expressions instead of the regular-expression operators $(\cup, \cdot, *)$ of classical PDL. However, the formula $\xrightarrow{\varphi}$, whose semantics is explained below, can be seen as a restricted use of the construct π^*.

A sentence ξ is evaluated wrt. an MSC $M = (E, \rightarrow, \lhd, \mathit{loc}, \lambda)$. An event formula φ is evaluated wrt. M and an event $e \in E$. Finally, a path formula π is evaluated over *two* events. In other words, it defines a binary relation $[\![\pi]\!]_M \subseteq E \times E$. We often write $M, e, f \models \pi$ to denote $(e, f) \in [\![\pi]\!]_M$. Moreover, for $e \in E$, we let $[\![\pi]\!]_M(e) := \{f \in E \mid (e, f) \in [\![\pi]\!]_M\}$. When M is clear from the context, we may write $[\![\pi]\!]$ instead of $[\![\pi]\!]_M$. The semantics of sentences, event formulas, and path formulas is given in Table 1.

Table 1 The semantics of $\mathrm{PDL}_{\mathsf{sf}}$.

$M \models \mathsf{E}\,\varphi$ if $M, e \models \varphi$ for some event $e \in E$	
$M \models \neg\xi$ if $M \not\models \xi$	$M \models \xi_1 \vee \xi_2$ if $M \models \xi_1$ or $M \models \xi_2$
$M, e \models p$ if $loc(e) = p$	$M, e \models \langle\pi\rangle\,\varphi$ if $\exists f \in [\![\pi]\!]_M(e) : M, f \models \varphi$
$M, e \models a$ if $\lambda(e) = a$	$M, e \models \mathsf{Loop}(\pi)$ if $(e, e) \in [\![\pi]\!]_M$
$M, e \models \neg\varphi$ if $M, e \not\models \varphi$	$M, e \models \varphi_1 \vee \varphi_2$ if $M, e \models \varphi_1$ or $M, e \models \varphi_2$
$[\![\rightarrow]\!]_M := \{(e, f) \in E \times E \mid e \rightarrow f\}$	$[\![\lhd_{p,q}]\!]_M := \{(e, f) \in E_p \times E_q \mid e \lhd f\}$
$[\![\leftarrow]\!]_M := \{(f, e) \in E \times E \mid e \rightarrow f\}$	$[\![\lhd_{p,q}^{-1}]\!]_M := \{(f, e) \in E_q \times E_p \mid e \lhd f\}$
$[\![\mathsf{jump}_{p,r}]\!]_M := E_p \times E_r$	$[\![\{\varphi\}?]\!]_M := \{(e, e) \mid e \in E : M, e \models \varphi\}$
$[\![\xrightarrow{\varphi}]\!]_M := \{(e, f) \in E \times E \mid e <_{\mathsf{proc}} f$ and $\forall g \in E$: $e <_{\mathsf{proc}} g <_{\mathsf{proc}} f \implies M, g \models \varphi\}$	
$[\![\xleftarrow{\varphi}]\!]_M := \{(e, f) \in E \times E \mid f <_{\mathsf{proc}} e$ and $\forall g \in E$: $f <_{\mathsf{proc}} g <_{\mathsf{proc}} e \implies M, g \models \varphi\}$	
$[\![\pi_1 \cdot \pi_2]\!]_M := \{(e, g) \in E \times E \mid \exists f \in E : (e, f) \in [\![\pi_1]\!]_M \wedge (f, g) \in [\![\pi_2]\!]_M\}$	
$[\![\pi_1 \cup \pi_2]\!]_M := [\![\pi_1]\!]_M \cup [\![\pi_2]\!]_M$	$[\![\pi^{\mathsf{c}}]\!]_M := (E \times E) \setminus [\![\pi]\!]_M$
$[\![\pi_1 \cap \pi_2]\!]_M := [\![\pi_1]\!]_M \cap [\![\pi_2]\!]_M$	

▶ **Example 5.** Consider again the MSC M from Figure 1 and the path formula $\pi = \lhd_{p_1,p_3}^{-1}\rightarrow\lhd_{p_1,p_2}\rightarrow\lhd_{p_2,p_3}\rightarrow$. We have $M, g_5 \models \mathsf{Loop}(\pi)$. Moreover, $(e_2, e_5) \in [\![\xrightarrow{\Box}]\!]_M$ but $(e_2, e_6) \notin [\![\xrightarrow{\Box}]\!]_M$.

We use the usual abbreviations for sentences and event formulas such as implication and conjunction. Moreover, $\mathit{true} := p \vee \neg p$ (for some arbitrary process $p \in P$) and $\mathit{false} := \neg\mathit{true}$. Finally, we define the event formula $\langle\pi\rangle := \langle\pi\rangle\,\mathit{true}$, and the path formulas $\xrightarrow{+} := \xrightarrow{\mathit{true}}$ and $\xrightarrow{*} := \xrightarrow{+} \cup \{\mathit{true}\}?$.

Note that there are some redundancies in the logic. For example (letting $\equiv$ denote logical equivalence), $\rightarrow \equiv \xrightarrow{\mathit{false}}$, $\pi_1 \cap \pi_2 \equiv (\pi_1^{\mathsf{c}} \cup \pi_2^{\mathsf{c}})^{\mathsf{c}}$, and $\mathsf{Loop}(\pi) \equiv \langle\{\mathit{true}\}? \cap \pi\rangle$. Some of them are necessary to define certain subclasses of $\mathrm{PDL}_{\mathsf{sf}}$. For every $R \subseteq \{\mathsf{Loop}, \cup, \cap, \mathsf{c}\}$, we let $\mathrm{PDL}_{\mathsf{sf}}[R]$ denote the fragment of $\mathrm{PDL}_{\mathsf{sf}}$ that does not make use of $\{\mathsf{Loop}, \cup, \cap, \mathsf{c}\} \setminus R$. In particular, $\mathrm{PDL}_{\mathsf{sf}} = \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}, \cup, \cap, \mathsf{c}]$. Note that, syntactically, $\xrightarrow{*}$ is not contained in $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ since union is not permitted.

3.2 Main Results

Let $\mathrm{FO}^3[\rightarrow, \lhd, \leq]$ be the set of formulas from $\mathrm{FO}[\rightarrow, \lhd, \leq]$ that use at most three different first-order variables (however, a variable can be quantified and reused several times in a formula). The main result of this section is that, for formulas with zero or one free variable, the logics $\mathrm{FO}[\rightarrow, \lhd, \leq]$, $\mathrm{FO}^3[\rightarrow, \lhd, \leq]$, $\mathrm{PDL}_{\mathsf{sf}}$, and $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ are expressively equivalent.

Consider $\mathrm{FO}[\rightarrow, \lhd, \leq]$ formulas Φ_0, $\Phi_1(x)$ and $\Phi_2(x, y)$ with respectively zero, one, and two free variables (hence, Φ_0 is a sentence). Consider also some $\mathrm{PDL}_{\mathsf{sf}}$ sentence ξ, event formula φ, and path formula π. The respective formulas are equivalent, written $\Phi_0 \equiv \xi$,

$\Phi_1(x) \equiv \varphi$, and $\Phi_2(x, y) \equiv \pi$, if, for all MSCs M and all events e, f in M, we have

$$M \models \Phi_0 \quad \text{iff} \quad M \models \xi$$
$$M, [x \mapsto e] \models \Phi_1(x) \quad \text{iff} \quad M, e \models \varphi$$
$$M, [x \mapsto e, y \mapsto f] \models \Phi_2(x, y) \quad \text{iff} \quad M, e, f \models \pi$$

We start with a simple observation, which can be shown easily by induction:

▶ **Proposition 6.** *Every* $\mathrm{PDL}_{\mathsf{sf}}$ *formula is equivalent to some* $\mathrm{FO}^3[\rightarrow, \lhd, \leq]$ *formula. More precisely, for every* $\mathrm{PDL}_{\mathsf{sf}}$ *sentence* ξ*, event formula* φ*, and path formula* π*, there exist some* $\mathrm{FO}^3[\rightarrow, \lhd, \leq]$ *sentence* $\widetilde{\xi}$*, formula* $\widetilde{\varphi}(x)$ *with one free variable, and formula* $\widetilde{\pi}(x, y)$ *with two free variables, respectively, such that,* $\xi \equiv \widetilde{\xi}$*,* $\varphi \equiv \widetilde{\varphi}(x)$*, and* $\pi \equiv \widetilde{\pi}(x, y)$*.*

The main result is a *strong* converse of Proposition 6:

▶ **Theorem 7.** *Every* $\mathrm{FO}[\rightarrow, \lhd, \leq]$ *formula with at most two free variables is equivalent to some* $\mathrm{PDL}_{\mathsf{sf}}$ *formula. More precisely, for every* $\mathrm{FO}[\rightarrow, \lhd, \leq]$ *sentence* Φ_0*, formula* $\Phi_1(x)$ *with one free variable, and formula* $\Phi_2(x, y)$ *with two free variables, there exist some* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *sentence* ξ*,* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *event formula* φ*, and* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *path formulas* π_{ij}*, respectively, such that,* $\Phi_0 \equiv \xi$*,* $\Phi_1(x) \equiv \varphi$*, and* $\Phi_2(x, y) \equiv \bigcup_i \bigcap_j \pi_{ij}$*.*

From Theorem 7 and Proposition 6, we deduce that FO has the three variable property:

▶ **Corollary 8.** $\mathcal{L}(\mathrm{FO}[\rightarrow, \lhd, \leq]) = \mathcal{L}(\mathrm{FO}^3[\rightarrow, \lhd, \leq])$.

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

In the remainder of this section, we give the translation from FO to $\mathrm{PDL}_{\mathsf{sf}}$. We start with some basic properties of $\mathrm{PDL}_{\mathsf{sf}}$. First, the converse of a $\mathrm{PDL}_{\mathsf{sf}}$ formula is definable in $\mathrm{PDL}_{\mathsf{sf}}$ (easy induction on π).

▶ **Lemma 9.** *Let* $R \subseteq \{\mathsf{Loop}, \cup, \cap, \mathsf{c}\}$ *and* $\pi \in \mathrm{PDL}_{\mathsf{sf}}[R]$ *be a path formula. There exists* $\pi^{-1} \in \mathrm{PDL}_{\mathsf{sf}}[R]$ *such that, for all MSCs* M*,* $[\![\pi^{-1}]\!]_M = [\![\pi]\!]_M^{-1} = \{(f, e) \mid (e, f) \in [\![\pi]\!]_M\}$*.*

Given a $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ path formula π, we denote by $\mathsf{Comp}(\pi)$ the set of pairs $(p, q) \in P \times P$ such that there may be a π-path from some event on process p to some event on process q. Formally, we let $\mathsf{Comp}(\rightarrow) = \mathsf{Comp}(\leftarrow) = \mathsf{Comp}(\xrightarrow{\varphi}) = \mathsf{Comp}(\xleftarrow{\varphi}) = \mathsf{Comp}(\{\varphi\}?) = \mathsf{id}$, where $\mathsf{id} = \{(p, p) \mid p \in P\}$; $\mathsf{Comp}(\lhd_{p,q}) = \mathsf{Comp}(\lhd_{q,p}^{-1}) = \{(p, q)\}$; $\mathsf{Comp}(\mathsf{jump}_{p,r}) = \{(p, r)\}$; and $\mathsf{Comp}(\pi_1 \cdot \pi_2) = \mathsf{Comp}(\pi_2) \circ \mathsf{Comp}(\pi_1) = \{(p, r) \mid \exists q : (p, q) \in \mathsf{Comp}(\pi_1), (q, r) \in \mathsf{Comp}(\pi_2)\}$.

Notice that, for all path formulas $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$, the relation $\mathsf{Comp}(\pi)$ is either empty or a singleton $\{(p, q)\}$ or the identity id. Moreover, $M, e, f \models \pi$ implies $(loc(e), loc(f)) \in \mathsf{Comp}(\pi)$. Therefore, all events in $[\![\pi]\!](e)$ are on the same process, and if this set is nonempty (i.e., if $M, e \models \langle\pi\rangle$), then $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$ are well-defined.

▶ **Example 10.** Consider the MSC from Figure 1 and $\pi = \xrightarrow{+}\lhd_{p_1,p_2}\rightarrow\lhd_{p_2,p_3}\rightarrow$. We have $\mathsf{Comp}(\pi) = \{(p_1, p_3)\}$. Moreover, $\min[\![\pi]\!](e_2) = g_4$ and $\max[\![\pi]\!](e_2) = g_5$.

We say that $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ is *monotone* if, for all MSCs M and events e, f such that $M, e \models \langle\pi\rangle$, $M, f \models \langle\pi\rangle$, and $e \leq_{\mathsf{proc}} f$, we have $\min[\![\pi]\!](e) \leq_{\mathsf{proc}} \min[\![\pi]\!](f)$ and $\max[\![\pi]\!](e) \leq_{\mathsf{proc}} \max[\![\pi]\!](f)$. Lemmas 11 and 12 are easily shown by simultaneous induction.

▶ **Lemma 11.** *Let* $\pi_1, \pi_2 \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *be path formulas, and* $\pi = \pi_1 \cdot \pi_2$*. For all MSCs* M *and events* e *such that* $M, e \models \langle\pi\rangle$*, we have*

$$\min[\![\pi]\!](e) = \min[\![\pi_2]\!](\min[\![\pi_1 \cdot \{\langle\pi_2\rangle\}?]\!](e)) \text{ and}$$
$$\max[\![\pi]\!](e) = \max[\![\pi_2]\!](\max[\![\pi_1 \cdot \{\langle\pi_2\rangle\}?]\!](e))\,.$$

▶ **Lemma 12.** *All* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *path formulas are monotone.*

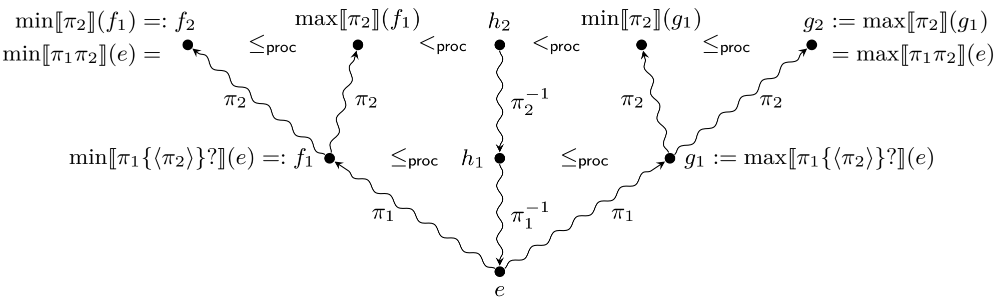
The following crucial lemma states that, for all path formulas $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ and events e in some MSC, $[\![\pi]\!](e)$ contains precisely the events that lie in the interval between $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$ and that satisfy $\langle\pi^{-1}\rangle$.

▶ **Lemma 13.** *Let* π *be a* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *path formula. For all MSCs* M *and events* e *such that* $M, e \models \langle\pi\rangle$*, we have*

$$[\![\pi]\!](e) = \{f \in E \mid \min[\![\pi]\!](e) \leq_{\mathsf{proc}} f \leq_{\mathsf{proc}} \max[\![\pi]\!](e) \wedge M, f \models \langle\pi^{-1}\rangle\}\,.$$

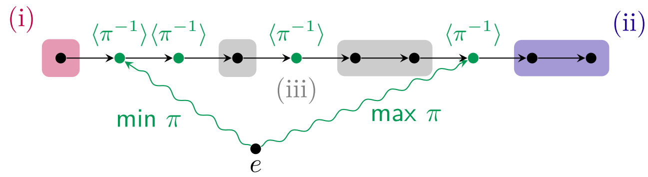
Proof. The left-to-right inclusion is trivial. We prove the right-to-left inclusion by induction on π. The base cases are immediate.

Assume that $\pi = \pi_1 \cdot \pi_2$. For illustration, consider the figure below.



We let $f_1 = \min[\![\pi_1\{\langle\pi_2\rangle\}?]\!](e)$, $f_2 = \min[\![\pi_2]\!](f_1)$, $g_1 = \max[\![\pi_1\{\langle\pi_2\rangle\}?]\!](e)$, and $g_2 = \max[\![\pi_2]\!](g_1)$. By Lemma 11, we have $f_2 = \min[\![\pi_1\pi_2]\!](e)$ and $g_2 = \max[\![\pi_1\pi_2]\!](e)$. Let $h_2 \in E$ such that $f_2 \leq_{\mathsf{proc}} h_2 \leq_{\mathsf{proc}} g_2$ and $M, h_2 \models \langle(\pi_1\pi_2)^{-1}\rangle$. If $h_2 \leq_{\mathsf{proc}} \max[\![\pi_2]\!](f_1)$, then by induction hypothesis, $M, f_1, h_2 \models \pi_2$, and we obtain $M, e, h_2 \models \pi_1\pi_2$. Similarly, if $\min[\![\pi_2]\!](g_1) \leq_{\mathsf{proc}} h_2$, then $M, g_1, h_2 \models \pi_2$ and $M, e, h_2 \models \pi_1\pi_2$. So assume $\max[\![\pi_2]\!](f_1) <_{\mathsf{proc}} h_2 <_{\mathsf{proc}} \min[\![\pi_2]\!](g_1)$. Since $M, h_2 \models \langle\pi_2^{-1}\pi_1^{-1}\rangle$, there exists h_1 such that $M, h_1, h_2 \models \pi_2$ and $M, h_1 \models \langle\pi_1^{-1}\rangle$. Moreover, $\min[\![\pi_2]\!](h_1) \leq_{\mathsf{proc}} h_2 <_{\mathsf{proc}} \min[\![\pi_2]\!](g_1)$, hence $h_1 \leq_{\mathsf{proc}} g_1$ by Lemma 12 (notice that g_1 and h_1 must be on the same process). Similarly, $\max[\![\pi_2]\!](f_1) <_{\mathsf{proc}} h_2 \leq_{\mathsf{proc}} \max[\![\pi_2]\!](h_1)$, hence $f_1 \leq_{\mathsf{proc}} h_1$. We then have $f_1 \leq_{\mathsf{proc}} h_1 \leq_{\mathsf{proc}} g_1$, and $M, h_1 \models \langle\pi_1^{-1}\rangle$. By induction hypothesis, $M, e, h_1 \models \pi_1$. Hence, $M, e, h_2 \models \pi_1\pi_2$. ◀

Using Lemma 13, we can give a characterization of $[\![\pi^{\mathsf{c}}]\!](e)$ (when $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$) that also relies on intervals delimited by $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$. More precisely, $[\![\pi^{\mathsf{c}}]\!](e)$ is the union of the following sets (see figure below): (i) the interval of all events to the left of $\min[\![\pi]\!](e)$, (ii) the interval of all events to the right of $\max[\![\pi]\!](e)$, (iii) the set of events located between $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$ and satisfying $\neg\langle\pi^{-1}\rangle$, (iv) all events located on other processes than $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$.



This description of $[\![\pi^{\mathsf{c}}]\!](e)$ can be used to rewrite π^{c} as a union of $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ formulas. In a first step, we show that, if π is a $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ formula, then the relation $\{(e, \min[\![\pi]\!](e))\}$ can also be expressed in $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ (and similarly for max).

▶ **Lemma 14.** *Let $R = \emptyset$ or $R = \{\mathsf{Loop}\}$. For every path formula $\pi \in \mathrm{PDL}_{\mathsf{sf}}[R]$, there exist $\mathrm{PDL}_{\mathsf{sf}}[R]$ path formulas $\mathsf{min}\ \pi$ and $\mathsf{max}\ \pi$ such that $M, e, f \models \mathsf{min}\ \pi$ iff $f = \min[\![\pi]\!](e)$, and $M, e, f \models \mathsf{max}\ \pi$ iff $f = \max[\![\pi]\!](e)$.*

Proof. We construct, by induction on π, formulas $\mathsf{min}\ (\pi \cdot \{\psi\}?)$ for all $\mathrm{PDL}_{\mathsf{sf}}[R]$ event formulas ψ. For $\pi \in \{\rightarrow, \leftarrow, \lhd_{p,q}, \lhd_{p,q}^{-1}, \{\varphi\}?\}$, we let $\mathsf{min}\ (\pi \cdot \{\psi\}?) = \pi \cdot \{\psi\}?$. Then,

$$\begin{aligned}
\mathsf{min}\ (\xrightarrow{\varphi} \cdot \{\psi\}?) &= \xrightarrow{\varphi\wedge\neg\psi} \cdot \{\psi\}? \\
\mathsf{min}\ (\xleftarrow{\varphi} \cdot \{\psi\}?) &= \xleftarrow{\varphi} \cdot \{\psi \wedge (\neg\varphi \vee \neg \langle \xleftarrow{\varphi} \rangle\, \psi)\}? \\
\mathsf{min}\ (\mathsf{jump}_{p,q} \cdot \{\psi\}?) &= \mathsf{jump}_{p,q} \cdot \{\psi \wedge \neg \langle \xleftarrow{+} \rangle\, \psi\}? \\
\mathsf{min}\ (\pi_1 \cdot \pi_2 \cdot \{\psi\}?) &= \mathsf{min}\ (\pi_1 \cdot \{\langle \pi_2 \rangle\, \psi\}?) \cdot \mathsf{min}\ (\pi_2 \cdot \{\psi\}?)\,.
\end{aligned}$$

The construction of $\mathsf{max}\ \pi$ is similar. ◀

We are now ready to prove that any boolean combination of $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ formulas is equivalent to a positive one, i.e., one that does not use complement.

▶ **Lemma 15.** *For all path formulas $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$, there exist $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ path formulas $(\pi_i)_{1\le i\le |P|^2+3}$ such that $\pi^{\mathsf{c}} \equiv \bigcup_{1\le i\le |P|^2+3} \pi_i$.*

Proof. We show $\pi^{\mathsf{c}} \equiv \sigma$, where

$$\sigma = (\mathsf{min}\ \pi \cdot \xleftarrow{+}) \cup (\mathsf{max}\ \pi \cdot \xrightarrow{+}) \cup (\pi \cdot \xrightarrow{+} \cdot \{\neg \langle \pi^{-1} \rangle\}?) \cup \bigcup_{(p,q)\in P^2} \{\neg \langle \pi \rangle\, q\}? \cdot \mathsf{jump}_{p,q}\,.$$

Let $M = (E, \rightarrow, \lhd, loc, \lambda)$ be an MSC and $e, f \in E$. We write $p = loc(e)$, $q = loc(f)$. Let us show that $M, e, f \models \pi^{\mathsf{c}}$ iff $M, e, f \models \sigma$. If $M, e \models \neg \langle \pi \rangle\, q$, then both $M, e, f \models \pi^{\mathsf{c}}$ and $M, e, f \models \sigma$ hold. In the following, we assume that $M, e \models \langle \pi \rangle\, q$, and thus that $\min[\![\pi]\!](e)$ and $\max[\![\pi]\!](e)$ are well-defined and on process q. Again, if $f <_{\mathsf{proc}} \min[\![\pi]\!](e)$ or $\max[\![\pi]\!](e) <_{\mathsf{proc}} f$, then both $M, e, f \models \pi^{\mathsf{c}}$ and $M, e, f \models \sigma$ hold. And if $\min[\![\pi]\!](e) \le_{\mathsf{proc}} f \le_{\mathsf{proc}} \max[\![\pi]\!](e)$, then, by Lemma 13, we have $M, e, f \models \pi^c$ iff $M, f \models \neg \langle \pi^{-1} \rangle$, iff $M, e, f \models \sigma$. ◀

The rest of this section is dedicated to the proof of Theorem 7, stating that every $\mathrm{FO}[\rightarrow, \lhd, \le]$ formula with at most two free variables can be translated into an equivalent $\mathrm{PDL}_{\mathsf{sf}}$ formula. As we proceed by induction, we actually need a more general statement, which takes into account arbitrarily many free variables:

▶ **Proposition 16.** *Every formula $\Phi \in \mathrm{FO}[\rightarrow, \lhd, \le]$ with at least one free variable is equivalent to a boolean combination of formulas of the form $\widetilde{\pi}(x, y)$, where $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ and $x, y \in \mathsf{Free}(\Phi)$.*

Proof. In the following, we will simply write $\pi(x, y)$ for $\widetilde{\pi}(x, y)$, where $\widetilde{\pi}(x, y)$ is the FO formula equivalent to π as defined in Proposition 6. The proof is by induction. For convenience, we assume that Φ is in prenex normal form. If Φ is quantifier free, then it is a boolean combination of atomic formulas. For $x, y \in \mathcal{V}_{\mathsf{event}}$, atomic formulas are translated as follows:

$$\begin{array}{lll}
p(x) \equiv \{p\}?(x,x) & x \rightarrow y \equiv \rightarrow(x,y) & x = y \equiv \{true\}?(x,y) \\
a(x) \equiv \{a\}?(x,x) & x \lhd y \equiv \displaystyle\bigvee_{(p,q)\in Ch} \lhd_{p,q}(x,y) &
\end{array}$$

Moreover, $x \le y$ is equivalent to the disjunction of the formulas $\big(\pi \cdot \lhd_{p_1,p_2} \cdot \xrightarrow{+} \cdot \lhd_{p_2,p_3} \cdots \xrightarrow{+} \cdot \lhd_{p_{m-1},p_m} \cdot \pi'\big)(x, y)$, where $1 \le m \le |P|$, $p_1, \ldots, p_m \in P$ are such that $p_i \ne p_{i+1}$ for all $i \in \{1, \ldots, m-1\}$, and $\pi, \pi' \in \{\xrightarrow{+}, \{true\}?\}$.

Universal quantification. We have $\forall x.\Psi \equiv \neg\exists x.\neg\Psi$. Since we allow boolean combinations, dealing with negation is trivial. Hence, this case reduces to existential quantification.

Existential quantification. Suppose that $\Phi = \exists x.\Psi$. If x is not free in Ψ, then $\Phi \equiv \Psi$ and we are done by induction. Otherwise, assume that $\mathsf{Free}(\Psi) = \{x_1, \ldots, x_n\}$ with $n > 1$ and that $x = x_n$. By induction, Ψ is equivalent to a boolean combination of formulas of the form $\pi(y, z)$ with $y, z \in \mathsf{Free}(\Psi)$. We transform it into a finite disjunction of formulas of the form $\bigwedge_j \pi_j(y_j, z_j)$, where $y_j = x_{i_1}$ and $z_j = x_{i_2}$ for some $i_1 \leq i_2$. To do so, we first eliminate negation using Lemma 15. The resulting positive boolean combination is then brought into disjunctive normal form. Note that this latter step may cause an exponential blow-up so that the overall construction is nonelementary (which is unavoidable [28]). Finally, the variable ordering can be guaranteed by replacing π_j with π_j^{-1} whenever needed.

Now, $\Phi = \exists x_n.\Psi$ is equivalent to a finite disjunction of formulas of the form

$$\bigwedge_{j \in I} \pi_j(y_j, z_j) \;\wedge\; \underbrace{\exists x_n.\Big(\bigwedge_{j \in J} \pi_j(y_j, x_n) \wedge \bigwedge_{j \in J'} \pi_j(x_n, x_n)\Big)}_{=:\,\Upsilon}$$

for three finite, pairwise disjoint index sets I, J, J' such that $y_j \in \{x_1, \ldots, x_{n-1}\}$ for all $j \in I \cup J$, and $z_j \in \{x_1, \ldots, x_{n-1}\}$ for all $j \in I$. Notice that $\mathsf{Free}(\Upsilon) \subseteq \{x_1, \ldots, x_{n-1}\}$. If $J = \emptyset$, then[2]

$$\Upsilon \equiv \bigvee_{p,q \in P} \Big(\mathsf{jump}_{p,q} \cdot \{\bigwedge_{j \in J'} \mathsf{Loop}(\pi_j)\}? \cdot \mathsf{jump}_{q,p}\Big)(x_1, x_1)\,.$$

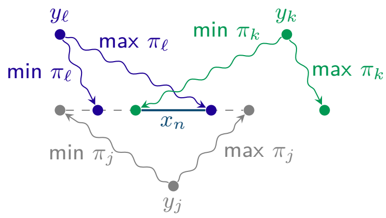
So assume $J \neq \emptyset$. Set

$$\Upsilon' := \bigvee_{k,\ell \in J} \left(\begin{array}{ll} & \bigwedge_{j \in J}((\min\ \pi_j) \cdot \xrightarrow{*} \cdot (\min\ \pi_k)^{-1})(y_j, y_k) \\ \wedge & \bigwedge_{j \in J}((\max\ \pi_\ell) \cdot \xrightarrow{*} \cdot (\max\ \pi_j)^{-1})(y_\ell, y_j) \\ \wedge & (\pi_k \cdot \{\psi\}? \cdot \pi_\ell^{-1})(y_k, y_\ell) \end{array} \right)$$

where $\psi = \bigwedge_{j \in J} \langle \pi_j^{-1} \rangle \wedge \bigwedge_{j \in J'} \mathsf{Loop}(\pi_j)$. We have $\mathsf{Free}(\Upsilon') = \mathsf{Free}(\Upsilon) \subseteq \{x_1, \ldots, x_{n-1}\}$.

▶ **Claim 17.** *We have* $\Upsilon \equiv \Upsilon'$.

Intuitively, by Lemma 13, we know that Υ holds iff the intersection of the intervals $[\min[\![\pi_j]\!](y_j), \max[\![\pi_j]\!](y_j)]$ contains some event satisfying ψ. The formula Υ' identifies some π_k such that $\min[\![\pi_k]\!](y_k)$ is maximal (first line), some π_ℓ such that $\max[\![\pi_\ell]\!](y_\ell)$ is minimal (second line), and tests that there exists an event x_n satisfying ψ between the two (third line). This is illustrated in the figure below.



[2] In this case, Υ is a sentence whereas x_1 is free in the right hand side. Notice that $\equiv$ does not require the two formulas to have the same free variables.

Thus, Υ is equivalent to some positive combination of formulas $\pi(x, y)$ with $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ and $x, y \in \{x_1, \ldots, x_{n-1}\} = \mathsf{Free}(\Phi)$, therefore, so is Φ. Note that the two formulas $\big((\mathsf{min}\ \pi_j) \cdot \xrightarrow{*} \cdot (\mathsf{min}\ \pi_k)^{-1}\big)(y_j, y_k)$ and $\big((\mathsf{max}\ \pi_\ell) \cdot \xrightarrow{*} \cdot (\mathsf{max}\ \pi_j)^{-1}\big)(y_\ell, y_j)$ are not $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ formulas (since $\xrightarrow{*}$ is not). However, they are disjunctions of $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ formulas, for instance, $\big((\mathsf{min}\ \pi_j) \cdot \xrightarrow{*} \cdot (\mathsf{min}\ \pi_k)^{-1}\big)(y_j, y_k) \equiv \big((\mathsf{min}\ \pi_j) \cdot (\mathsf{min}\ \pi_k)^{-1}\big)(y_j, y_k) \vee \big((\mathsf{min}\ \pi_j) \cdot \xrightarrow{+} \cdot (\mathsf{min}\ \pi_k)^{-1}\big)(y_j, y_k)$. ◀

We are now able to prove the main result relating $\mathrm{FO}[\rightarrow, \lhd, \leq]$ and $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$.

Proof of Theorem 7. Let $\Phi_2(x_1, x_2)$ be an $\mathrm{FO}[\rightarrow, \lhd, \leq]$ formula with two free variables. We apply Proposition 16 to $\Phi_2(x_1, x_2)$ and obtain a boolean combination of path formulas $\pi(y, z)$ with $y, z \in \{x_1, x_2\}$. First, we bring it into a positive boolean combination using Lemma 15. Next, we replace formulas $\pi(x_1, x_1)$ with $\bigvee_{p,q}(\{\mathsf{Loop}(\pi)\}? \cdot \mathsf{jump}_{p,q})(x_1, x_2)$. Similarly, $\pi(x_2, x_2)$ is replaced with $\bigvee_{p,q}(\mathsf{jump}_{p,q} \cdot \{\mathsf{Loop}(\pi)\}?)(x_1, x_2)$. Also, $\pi(x_2, x_1)$ is replaced with $\pi^{-1}(x_1, x_2)$. Finally, we transform it into disjunctive normal form: we obtain $\Phi_1(x_1, x_2) \equiv \bigvee_i \bigwedge_j \pi_{ij}(x_1, x_2)$, which concludes the proof in the case of two free variables.

Next, let $\Phi_1(x)$ be an $\mathrm{FO}[\rightarrow, \lhd, \leq]$ formula with one free variable. As above, applying Proposition 16 to $\Phi_1(x)$ and then Lemma 15, we obtain $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ path formulas π_{ij} such that $\Phi_1(x) \equiv \bigvee_i \bigwedge_j \pi_{ij}(x, x)$. Now, $M, [x \mapsto e] \models \pi_{ij}(x, x)$ iff $M, e \models \mathsf{Loop}(\pi_{ij})$. Hence, $\Phi(x) \equiv \bigvee_i \bigwedge_j \mathsf{Loop}(\pi_{ij})$.

Finally, an $\mathrm{FO}[\rightarrow, \lhd, \leq]$ sentence Φ_0 is a boolean combination of formulas of the form $\exists x.\Phi_1(x)$. Applying the theorem to $\Phi_1(x)$, we obtain an equivalent $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ event formula φ. Then, we take $\xi = \mathsf{E}\,\varphi$, which is trivially equivalent to $\exists x.\Phi_1(x)$. ◀

4 From $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ to CFMs

Letter-to-letter MSC transducers. For the translation of $\mathrm{FO}[\rightarrow, \lhd, \leq]$ sentences into CFMs, we will need to introduce MSC transducers to handle subformulas with one free variable, or, equivalently, $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ event formulas. More precisely, we will associate with an event formula φ a transducer that evaluates φ at all events, and outputs 1 when the formula holds, and 0 otherwise.

Let Γ be a nonempty finite output alphabet. A *(nondeterministic) letter-to-letter MSC transducer* (or simply, *transducer*) $\mathcal{A}$ over P and from Σ to Γ is a CFM over P and $\Sigma \times \Gamma$. The transducer $\mathcal{A}$ accepts the relation $[\![\mathcal{A}]\!] = \{\big((E, \rightarrow, \lhd, \mathit{loc}, \lambda), (E, \rightarrow, \lhd, \mathit{loc}, \gamma)\big) \mid (E, \rightarrow, \lhd, \mathit{loc}, \lambda \times \gamma) \in L(\mathcal{A})\}$. Transducers are closed under product and composition, using standard constructions:

▶ **Lemma 18.** *Let $\mathcal{A}$ be a transducer from Σ to Γ, and $\mathcal{A}'$ a transducer from Σ to Γ'. There exists a transducer $\mathcal{A} \times \mathcal{A}'$ from Σ to $\Gamma \times \Gamma'$ such that*

$$\begin{aligned}[\![\mathcal{A} \times \mathcal{A}']\!] = \big\{\big((E, \rightarrow, \lhd, \mathit{loc}, \lambda), (E, \rightarrow, \lhd, \mathit{loc}, \gamma \times \gamma')\big) \mid \\ \big((E, \rightarrow, \lhd, \mathit{loc}, \lambda), (E, \rightarrow, \lhd, \mathit{loc}, \gamma)\big) \in [\![\mathcal{A}]\!], \\ \big((E, \rightarrow, \lhd, \mathit{loc}, \lambda), (E, \rightarrow, \lhd, \mathit{loc}, \gamma')\big) \in [\![\mathcal{A}']\!]\big\}.\end{aligned}$$

▶ **Lemma 19.** *Let $\mathcal{A}$ be a transducer from Σ to Γ, and $\mathcal{A}'$ a transducer from Γ to Γ'. There exists a transducer $\mathcal{A}' \circ \mathcal{A}$ from Σ to Γ' such that*

$$[\![\mathcal{A}' \circ \mathcal{A}]\!] = [\![\mathcal{A}']\!] \circ [\![\mathcal{A}]\!] = \{(M, M'') \mid \exists M' \in \mathbb{MSC}(P, \Gamma) : (M, M') \in [\![\mathcal{A}]\!], (M', M'') \in [\![\mathcal{A}']\!]\}.$$

Translation of $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ Event Formulas into CFMs. For a $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ event formula φ and an MSC $M = (E, \rightarrow, \lhd, loc, \lambda)$ over P and Σ, we define an MSC $M_\varphi = (E, \rightarrow, \lhd, loc, \gamma)$ over P and $\{0, 1\}$, by setting $\gamma(e) = 1$ if $M, e \models \varphi$, and $\gamma(e) = 0$ otherwise. Our goal is to construct a transducer $\mathcal{A}_\varphi$ such that $[\![\mathcal{A}_\varphi]\!] = \{(M, M_\varphi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$.

We start with the case of formulas from $\mathrm{PDL}_{\mathsf{sf}}[\emptyset]$, i.e., without Loop. A straightforward induction shows:

▶ **Lemma 20.** *Let φ be a $\mathrm{PDL}_{\mathsf{sf}}[\emptyset]$ event formula. There exists a transducer $\mathcal{A}_\varphi$ such that $[\![\mathcal{A}_\varphi]\!] = \{(M, M_\varphi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$.*

Next, we look at a single loop where the path $\pi \in \mathrm{PDL}_{\mathsf{sf}}[\emptyset]$ is of the form $\mathsf{min}\ \pi'$ or $\mathsf{max}\ \pi'$. This case will be simpler than general loop formulas, because of the fact that $[\![\mathsf{min}\ \pi']\!](e)$ is always either empty or a singleton. Recall that, in addition, $\mathsf{min}\ \pi'$ is monotone.

▶ **Lemma 21.** *Let π be a $\mathrm{PDL}_{\mathsf{sf}}[\emptyset]$ path formula of the form $\pi = \mathsf{min}\ \pi'$ or $\pi = \mathsf{max}\ \pi'$, and let $\varphi = \mathsf{Loop}(\pi)$. There exists a transducer $\mathcal{A}_\varphi$ such that $[\![\mathcal{A}_\varphi]\!] = \{(M, M_\varphi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$.*

Proof. We can assume that $\mathsf{Comp}(\pi) \subseteq \mathsf{id}$. We define $\mathcal{A}_\varphi$ as the composition of three transducers that will guess and check the evaluation of φ. More precisely, $\mathcal{A}_\varphi$ will be obtain as an inverse projection α^{-1}, followed by the intersection with an MSC language K, followed by a projection β.

We first enrich the labeling of the MSC with a color from $\Theta = \{\square, \blacksquare, \bigcirc, \bullet\}$. Intuitively, colors $\square$ and $\blacksquare$ will correspond to a guess that the formula φ is satisfied, and colors $\bigcirc$ and $\bullet$ to a guess that the formula is not satisfied. Consider the projection $\alpha\colon \mathbb{MSC}(P, \Sigma \times \Theta) \rightarrow \mathbb{MSC}(P, \Sigma)$ which erases the color from the labeling. The inverse projection α^{-1} can be realized with a transducer $\mathcal{A}$, i.e., $[\![\mathcal{A}]\!] = \{(\alpha(M'), M') \mid M' \in \mathbb{MSC}(P, \Sigma \times \Theta)\}$.

Define the projection $\beta\colon \mathbb{MSC}(P, \Sigma \times \Theta) \rightarrow \mathbb{MSC}(P, \{0, 1\})$ by $\beta((E, \rightarrow, \lhd, loc, \lambda \times \theta)) = (E, \rightarrow, \lhd, loc, \gamma)$, where $\gamma(e) = 1$ if $\theta(e) \in \{\square, \blacksquare\}$, and $\gamma(e) = 0$ otherwise. The projection β can be realized with a transducer $\mathcal{A}''$: we have $[\![\mathcal{A}'']\!] = \{(M', \beta(M')) \mid M' \in \mathbb{MSC}(P, \Sigma \times \Theta)\}$.

Finally, consider the language $K \subseteq \mathbb{MSC}(P, \Sigma \times \Theta)$ of MSCs $M' = (E, \rightarrow, \lhd, loc, \lambda \times \theta)$ satisfying the following two conditions:

1. Colors $\square$ and $\blacksquare$ alternate on each process $p \in P$: if $e_1 < \cdots < e_n$ are the events in $E_p \cap \theta^{-1}(\{\square, \blacksquare\})$, then $\theta(e_i) = \square$ if i is odd, and $\theta(e_i) = \blacksquare$ if i is even.
2. For all $e \in E$, $\theta(e) \in \{\square, \blacksquare\}$ iff there exists $f \in E$ such that $M, e, f \models \pi$ and $\theta(e) = \theta(f)$.

The first property is trivial to check with a CFM. Using Lemma 20, we can easily show that the second property can also be checked with a CFM. We deduce that there is a transducer $\mathcal{A}'$ such that $[\![\mathcal{A}']\!] = \{(M', M') \mid M' \in K\}$. We let $\mathcal{A}_\varphi = \mathcal{A}'' \circ \mathcal{A}' \circ \mathcal{A}$. Notice that $[\![\mathcal{A}_\varphi]\!] = \{(\alpha(M'), \beta(M')) \mid M' \in K\}$. From the following two claims, we deduce immediately that $[\![\mathcal{A}_\varphi]\!] = \{(M, M_\varphi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$.

▶ **Claim 22.** *For all $M \in \mathbb{MSC}(P, \Sigma)$, there exists $M' \in K$ with $\alpha(M') = M$.*

Let $M = (E, \rightarrow, \lhd, loc, \lambda) \in \mathbb{MSC}(P, \Sigma)$. Let $E_1 = \{e \in E \mid M, e \models \varphi\}$ and $E_0 = E \setminus E_1$. Consider the graph $G = (E, \{(e, f) \mid M, e, f \models \pi\})$. Since $\pi = \mathsf{min}\ \pi'$ or $\pi = \mathsf{max}\ \pi'$, every vertex has outdegree at most 1, and, by Lemma 12, there are no cycles except for self-loops. So the restriction of G to E_0 is a forest, and there exists a 2-coloring $\chi\colon E_0 \rightarrow \{\bigcirc, \bullet\}$ such that, for all $e, f \in E_0$ with $M, e, f \models \pi$, we have $\chi(e) \neq \chi(f)$. There exists $\theta\colon E \rightarrow \Theta$ such that $\theta(e) = \chi(e)$ for $e \in E_0$, and $\theta(e) \in \{\square, \blacksquare\}$ for $e \in E_1$ is such that Condition 1 of the definition of K is satisfied. It is easy to see that Condition 2 is also satisfied. Indeed, if $\theta(e) \in \{\square, \blacksquare\}$, then $e \in E_1$ and $M, e, e \models \pi$. Now, if $\theta(e) \notin \{\square, \blacksquare\}$, then $e \in E_0$ and either $M, e \not\models \langle\pi\rangle$ or, by definition of χ, we have $\theta(e) \neq \theta(f)$ for the unique f such that $M, e, f \models \pi$.

▶ **Claim 23.** *For all* $M' \in K$*, we have* $\beta(M') = M_\varphi$*, where* $M = \alpha(M')$*.*

Let $M' = (E, \rightarrow, \lhd, loc, \lambda \times \theta) \in K$ and $M = \alpha(M')$. Suppose towards a contradiction that $M_\varphi \neq \beta(M) = (E, \rightarrow, \lhd, loc, \gamma)$. By Condition 2, for all $e \in E$ such that $\gamma(e) = 0$, we have $M, e \not\models \varphi$. So there exists $f_0 \in E$ such that $\gamma(f_0) = 1$ and $M, f_0 \not\models \varphi$. Notice that $\theta(f_0) \in \{\square, \blacksquare\}$. For all $i \in \mathbb{N}$, let f_{i+1} be the unique event such that $M, f_i, f_{i+1} \models \pi$. Such an event exists by Condition 2, and is unique since $\pi = \mathsf{min}\ \pi'$ or $\pi = \mathsf{max}\ \pi'$. Note that, for all i, $\theta(f_{i+1}) = \theta(f_i) \in \{\square, \blacksquare\}$. Suppose $f_0 <_{\mathsf{proc}} f_1$ (the case $f_1 <_{\mathsf{proc}} f_0$ is similar). By Condition 1, there exists g_0 such that $f_0 <_{\mathsf{proc}} g_0 <_{\mathsf{proc}} f_1$ and $\{\theta(f_0), \theta(g_0)\} = \{\square, \blacksquare\}$. Again, for all $i \in \mathbb{N}$, let g_{i+1} be the unique event such that $M, g_i, g_{i+1} \models \pi$. Note that all $f_0, f_1, \ldots$ have the same color, in $\{\square, \blacksquare\}$, and all $g_0, g_1, \ldots$ carry the complementary color. Thus, $f_i \neq g_j$ for all $i, j \in \mathbb{N}$. But, by Lemma 12, this implies $f_0 <_{\mathsf{proc}} g_0 <_{\mathsf{proc}} f_1 <_{\mathsf{proc}} g_1 <_{\mathsf{proc}} \cdots$, which contradicts the fact that we deal with finite MSCs. ◀

The general case is more complicated. We first show how to rewrite an arbitrary loop formula using loops on paths of the form $\mathsf{max}\ \pi$ or $(\mathsf{max}\ \pi) \cdot \xleftarrow{+}$. Intuitively, this means that loop formulas will only be used to test, given an event e such that $e' = \max[\![\pi]\!](e)$ is well-defined and on the same process as e, whether $e' <_{\mathsf{proc}} e$, $e' = e$, or $e <_{\mathsf{proc}} e'$. Indeed, we have $M, e \models \mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+})$ iff $e <_{\mathsf{proc}} \max[\![\pi]\!](e)$.

▶ **Lemma 24.** *For all* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *path formulas* π*,*

$$\mathsf{Loop}(\pi) \equiv \mathsf{Loop}(\mathsf{max}\ \pi) \vee \Big(\langle \pi^{-1} \rangle \wedge \mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+}) \wedge \neg \mathsf{Loop}((\mathsf{min}\ \pi) \cdot \xleftarrow{+})\Big) .$$

Proof. The result follows from Lemma 13. Indeed, if we have $M, e \models \mathsf{Loop}(\pi)$ and $M, e \not\models \mathsf{Loop}(\mathsf{max}\ \pi)$, then $\min[\![\pi]\!](e) \leq_{\mathsf{proc}} e <_{\mathsf{proc}} \max[\![\pi]\!](e)$ and $M, e \models \langle \pi^{-1} \rangle$, hence $M, e \models \langle \pi^{-1} \rangle \wedge \mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+}) \wedge \neg \mathsf{Loop}((\mathsf{min}\ \pi) \cdot \xleftarrow{+})$. Conversely, if $M, e \models \mathsf{Loop}(\mathsf{max}\ \pi)$, then $M, e \models \mathsf{Loop}(\pi)$, and if $M, e \models (\langle \pi^{-1} \rangle \wedge \mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+}) \wedge \neg \mathsf{Loop}((\mathsf{min}\ \pi) \cdot \xleftarrow{+}))$, then $M, e \models \langle \pi^{-1} \rangle$ and $\min[\![\pi]\!](e) \leq_{\mathsf{proc}} e <_{\mathsf{proc}} \max[\![\pi]\!](e)$, hence $M, e, e \models \pi$, i.e., $M, e \models \mathsf{Loop}(\pi)$. ◀

Notice that, since $\mathsf{min}\ \pi \equiv \mathsf{max}\ (\mathsf{min}\ \pi)$, the formula $\mathsf{Loop}((\mathsf{min}\ \pi) \cdot \xleftarrow{+})$ can also be seen as a special case of a $\mathsf{Loop}((\mathsf{max}\ \pi') \cdot \xleftarrow{+})$ formula.

▶ **Theorem 25.** *For all* $\mathrm{PDL}_{\mathsf{sf}}[\mathsf{Loop}]$ *event formulas* φ*, there exists a transducer* $\mathcal{A}_\varphi$ *such that* $[\![\mathcal{A}_\varphi]\!] = \{(M, M_\varphi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$*.*

Proof. By Lemma 24, we can assume that all loop subformulas in φ are of the form $\mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+})$ or $\mathsf{Loop}(\mathsf{max}\ \pi)$ (notice that $\mathsf{min}\ \pi = \mathsf{max}\ \mathsf{min}\ \pi$). We prove Theorem 25 by induction on the number of loop subformulas in φ. The base case is stated in Lemma 20.

Let $\psi = \mathsf{Loop}(\pi')$ be a subformula of φ such that π' contains no loop subformulas and $\mathsf{Comp}(\pi') \subseteq \mathsf{id}$. Let us show that there exists $\mathcal{A}_\psi$ such that $[\![\mathcal{A}_\psi]\!] = \{(M, M_\psi) \mid M \in \mathbb{MSC}(P, \Sigma)\}$. If $\pi' = \mathsf{max}\ \pi$, then we apply Lemma 21. Otherwise, $\pi' = (\mathsf{max}\ \pi) \cdot \xleftarrow{+}$ for some $\mathrm{PDL}_{\mathsf{sf}}[\emptyset]$ path formula π. So we assume from now on that $\psi = \mathsf{Loop}((\mathsf{max}\ \pi) \cdot \xleftarrow{+})$.

We start with some easy remarks. Let $p \in P$ be some process and $e \in E_p$. A necessary condition for $M, e \models \psi$ is that $M, e \models \langle \pi \rangle \wedge \neg \mathsf{Loop}(\mathsf{max}\ \pi)$. Also, it is easy to see that $M, e \models \mathsf{Loop}(\mathsf{min}\ (\xrightarrow{+} \cdot \pi^{-1}))$ is a sufficient condition for $M, e \models \psi$.

We let E_p^π be the set of events $e \in E_p$ satisfying $\langle \pi \rangle\, p$. For all $e \in E_p^\pi$ we let $e' = [\![\mathsf{max}\ \pi]\!](e) \in E_p$. The transducer $\mathcal{A}_\psi$ will establish, for each $e \in E_p^\pi$, whether $e' <_{\mathsf{proc}} e$, $e' = e$, or $e <_{\mathsf{proc}} e'$, and it will output 1 if $e <_{\mathsf{proc}} e'$, and 0 otherwise. The case $e' = e$ means

$M, e \models \mathsf{Loop}(\mathsf{max}\ \pi)$ and can be checked with the help of Lemma 21. So the difficulty is to distinguish between $e' <_{\mathsf{proc}} e$ and $e <_{\mathsf{proc}} e'$ when $M, e \models \langle\pi\rangle \wedge \neg\mathsf{Loop}(\mathsf{max}\ \pi)$.

The following two claims rely on Lemma 12:

▶ **Claim 26.** *Let f be the minimal event in E_p^π (assuming this set is nonempty). Then, $M, f \models \psi$ iff $M, f \models \mathsf{Loop}(\mathsf{min}\ (\xrightarrow{+} \cdot \pi^{-1}))$.*

▶ **Claim 27.** *Let e, f be consecutive events in E_p^π, i.e., $e, f \in E_p^\pi$ and $M, e, f \models \xrightarrow{\neg\langle\pi\rangle}$.*

1. *If $M, e \not\models \psi$, then $[M, f \models \psi$ iff $M, f \models \mathsf{Loop}(\mathsf{min}\ (\xrightarrow{+} \cdot \pi^{-1}))]$.*
2. *If $M, e \models \psi$, then $[M, f \not\models \psi$ iff $M, f \models \mathsf{Loop}(\mathsf{max}\ \pi) \vee \mathsf{Loop}(\mathsf{max}\ ((\mathsf{max}\ \pi) \cdot \xrightarrow{\neg\langle\pi\rangle}))]$.*

To conclude the proof, let $\varphi_1 = \langle\pi\rangle$, $\varphi_2 = \mathsf{Loop}(\mathsf{max}\ \pi)$, $\varphi_3 = \mathsf{Loop}(\mathsf{min}\ (\xrightarrow{+} \cdot \pi^{-1}))$, and $\varphi_4 = \mathsf{Loop}(\mathsf{max}\ ((\mathsf{max}\ \pi) \cdot \xrightarrow{\neg\langle\pi\rangle}))$. By Lemmas 20 and 21, we already have transducers $\mathcal{A}_{\varphi_i}$ for $i \in \{1, 2, 3, 4\}$. We let $\mathcal{A}_\psi = \mathcal{A} \circ (\mathcal{A}_{\varphi_1} \times \mathcal{A}_{\varphi_2} \times \mathcal{A}_{\varphi_3} \times \mathcal{A}_{\varphi_4})$, where, at an event f labeled (b_1, b_2, b_3, b_4), the transducer $\mathcal{A}$ outputs 1 if $b_3 = 1$ or if $(b_1, b_2, b_3, b_4) = (1, 0, 0, 0)$ and the output was 1 at the last event e on the same process satisfying φ_1 (to do so, each process keeps in its state the output at the last event where b_1 was 1), and 0 otherwise.

Consider the formula φ' over $\Sigma \times \{0, 1\}$ obtained from φ by replacing ψ by $\bigvee_{a\in\Sigma}(a, 1)$, and all event formulas a, with $a \in \Sigma$, by $(a, 0) \vee (a, 1)$. It contains fewer Loop operators than φ, so by induction hypothesis, we have a transducer $\mathcal{A}_{\varphi'}$ for φ'. We then let $\mathcal{A}_\varphi = \mathcal{A}_{\varphi'} \circ (\mathcal{A}_{Id} \times \mathcal{A}_\psi)$, where $\mathcal{A}_{Id}$ is the transducer for the identity relation. ◀

Proof of Proposition 4. By Theorem 7, every FO$[\rightarrow, \lhd, \leq]$ formula $\Phi(x)$ with a single free variable is equivalent to some PDL$_{\mathsf{sf}}[\mathsf{Loop}]$ state formula, for which we obtain a transducer $\mathcal{A}_\Phi$ using Theorem 25. It is easy to build from $\mathcal{A}_\Phi$ CFMs for the sentences $\forall x.\Phi(x)$ and $\exists x.\Phi(x)$. Closure of $\mathcal{L}(\text{CFM})$ under union and intersection takes care of disjunction and conjunction. ◀

5 Discussion

Though the translation of EMSO/FO formulas into CFMs is interesting on its own, it allows us to obtain some difficult results for bounded CFMs as corollaries. We will briefly sketch some of them. For details, we refer to [2].

First, note that, for a given channel bound, the set of existentially bounded MSCs is FO-definable (essentially due to [26]). By Theorem 3, we obtain [14, Proposition 5.14] stating that this set is recognized by some CFM. Second, we obtain [14, Proposition 5.3], a Kleene theorem for existentially bounded MSCs, as a corollary of Theorem 3 in combination with a linearization normal form from [30].

Since (bounded) MSCs can be seen as a special case of Mazurkiewicz traces [9], we also get Zielonka's theorem [33] (though a weaker, nondeterministic version, and without guarantee on the size of the constructed automaton).

We leave open whether there is a one-dimensional temporal logic over MSCs, with a finite set of FO-definable modalities, that is expressively complete for FO$[\rightarrow, \lhd, \leq]$.

References

1 B. Bollig, M. Fortin, and P. Gastin. Communicating finite-state machines and two-variable logic. In *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *Leibniz International Proceedings in Informatics*, pages 17:1–17:14. Leibniz-Zentrum für Informatik, 2018.

2 B. Bollig, M. Fortin, and P. Gastin. It is easy to be wise after the event: Communicating finite-state machines capture first-order logic with "happened before". *CoRR*, abs/1804.10076, 2018. `arXiv:1804.10076`.

3 B. Bollig, D. Kuske, and I. Meinecke. Propositional dynamic logic for message-passing systems. *Logical Methods in Computer Science*, 6(3:16), 2010.

4 B. Bollig and M. Leucker. Message-passing automata are expressively equivalent to EMSO logic. *Theoretical Computer Science*, 358(2-3):150–172, 2006.

5 D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2), 1983.

6 J. Büchi. Weak second order logic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.

7 G. De Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1.*, pages 205–212. AAAI Press / The MIT Press, 1994.

8 V. Diekert and P. Gastin. First-order definable languages. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives*, volume 2 of *Texts in Logic and Games*, pages 261–306. Amsterdam University Press, 2008.

9 V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, Singapore, 1995.

10 C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–52, 1961.

11 M. J. Fischer and R. E. Ladner. Propositional Dynamic Logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194–211, 1979.

12 D. M. Gabbay. Expressive functional completeness in tense logic. In Uwe Mönnich, editor, *Aspects of Philosophical Logic: Some Logical Forays into Central Notions of Linguistics and Philosophy*, pages 91–117. Springer Netherlands, Dordrecht, 1981.

13 D. M. Gabbay, I. Hodkinson, and M. A. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects, vol. 1.* Oxford University Press, 1994.

14 B. Genest, D. Kuske, and A. Muscholl. A Kleene theorem and model checking algorithms for existentially bounded communicating automata. *Information and Computation*, 204(6):920–956, 2006.

15 B. Genest, D. Kuske, and A. Muscholl. On communicating automata with bounded channels. *Fundamenta Informaticae*, 80(1-3):147–167, 2007.

16 S. Göller, M. Lohrey, and C. Lutz. PDL with intersection and converse: satisfiability and infinite-state model checking. *Journal of Symbolic Logic*, 74(1):279–314, 2009.

17 E. Grädel and M. Otto. On logics with two variables. *Theoretical Computer Science*, 224(1-2):73–113, 1999.

18 J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artif. Intell.*, 54(2):319–379, 1992.

19 W. Hanf. Model-theoretic methods in the study of elementary logic. In J. W. Addison, L. Henkin, and A. Tarski, editors, *The Theory of Models.* North-Holland, Amsterdam, 1965.

20 J. G. Henriksen, M. Mukund, K. Narayan Kumar, M. Sohoni, and P. S. Thiagarajan. A theory of regular MSC languages. *Information and Computation*, 202(1):1–38, 2005.

21 H. Kamp. *Tense Logic and the Theory of Linear Order.* PhD thesis, University of California, Los Angeles, 1968.

22 D. Kuske. Regular sets of infinite message sequence charts. *Information and Computation*, 187:80–109, 2003.

23 L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, 1978.

24 M. Lange. Model checking propositional dynamic logic with all extras. *Journal of Applied Logic*, 4(1):39–49, 2006.

25 M. Lange and C. Lutz. 2-ExpTime lower bounds for Propositional Dynamic Logics with intersection. *Journal of Symbolic Logic*, 70(5):1072–1086, 2005.

26 M. Lohrey and A. Muscholl. Bounded MSC Communication. *Information and Computation*, 189(2):160–181, 2004.

27 R. Mennicke. Propositional dynamic logic with converse and repeat for message-passing systems. *Logical Methods in Computer Science*, 9(2:12):1–35, 2013.

28 L. J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic*. PhD thesis, MIT, 1974.

29 R. S. Streett. Propositional dynamic logic of looping and converse. In *Proceedings of STOC'81*, pages 375–383. ACM, 1981.

30 P. S. Thiagarajan and I. Walukiewicz. An expressively complete linear time temporal logic for Mazurkiewicz traces. *Inf. Comput.*, 179(2):230–249, 2002.

31 W. Thomas. Languages, automata and logic. In A. Salomaa and G. Rozenberg, editors, *Handbook of Formal Languages*, volume 3, pages 389–455. Springer, 1997.

32 B. A. Trakhtenbrot. Finite automata and monadic second order logic. *Siberian Math. J*, 3:103–131, 1962. In Russian; English translation in *Amer. Math. Soc. Transl.* 59, 1966, 23–55.

33 W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21:99–135, 1987.