Logic and Languages of Higher-Dimensional Automata

Amazigh Amrane[1], Hugo Bazille[1], Uli Fahrenberg[1] and Marie Fortin[2]

[1] EPITA Research Laboratory (LRE), Paris, France
[2] Université Paris Cité, CNRS, IRIF, France

Abstract. In this paper we study finite higher-dimensional automata (HDAs) from the logical point of view. Languages of HDAs are sets of finite bounded-width interval pomsets with interfaces ($\mathsf{iiPoms}_{\leq k}$) closed under order extension. We prove that languages of HDAs are MSO-definable. For the converse, we show that the order extensions of MSO-definable sets of $\mathsf{iiPoms}_{\leq k}$ are languages of HDAs. As a consequence, unlike the case of all pomsets, order extension of MSO-definable sets of $\mathsf{iiPoms}_{\leq k}$ is also MSO-definable.

1 Introduction

Connections between logic and automata play a key role in several areas of theoretical computer science – logic being used to specify the behaviours of automata models in formal verification, and automata being used to prove the decidability of various logics. The first and most well-known result of this kind is the equivalence in expressive power of finite automata and monadic second-order logic (MSO) over finite words, proved independently by Büchi [3], Elgot [7] and Trakhtenbrot [24] in the 60's. This was soon extended to infinite words [4] as well as finite and infinite trees [6, 20, 21].

Finite automata over words are a simple model of sequential systems with a finite memory, each word accepted by the automaton corresponding to an execution of the system. For concurrent systems, executions may be represented as *pomsets* (partially ordered sets). Several classes of pomsets and matching automata models have been defined in the literature, corresponding to different communication models or different views of concurrency. In that setting, logical characterisations of classes of automata in the spirit of the Büchi-Elgot-Trakhtenbrot theorem have been obtained for several cases, such as asynchronous automata and Mazurkiewicz traces [23, 27], branching automata and series-parallel pomsets [2, 17], step transition systems and local trace languages [12, 18], or communicating finite-state machines and message sequence charts [14].

Higher-dimensional automata (HDAs) [19, 25] are another automaton-based model of concurrent systems that matches more closely an interval-based view of events. Initially studied from a geometrical or categorical point of view, the language theory of HDAs has become another focus for research in the past few years [8]. The language of an HDA is defined as a set of *interval pomsets with interfaces (interval ipomsets)* [10]. The idea is that each event in the execution of an HDA corresponds to an interval of time where some process is active. In addition, if we shorten some intervals in one possible behaviour of the HDA, we obtain another valid behaviour for the HDA. In terms of pomsets, this means that the language of an HDA is closed under *subsumption* (expanding the partial order). In addition (for finite HDAs), it also has bounded width, meaning that each set of pairwise concurrent events has size at most k for some k.

Several theorems of classical automata theory have already been extended to HDAs, including a Kleene theorem [9] and a Myhill-Nerode theorem [11]. The closure properties of HDAs were also studied in [1]. In particular, regular languages are not closed under complement, but they are closed under *bounded width complement*: the subsumption closure of the complement of the language restricted to interval ipomsets of bounded width. In this paper, we explore the relationship between HDAs and MSO. We prove that a set of interval ipomsets is regular if and only if it is simultaneously MSO-definable, of bounded width, and downward-closed for subsumption. The latter two assumptions are necessary as it is possible to define in MSO sets with unbounded width or sets that are not downward-closed.

The HDA-to-MSO direction is proved similarly to the original Büchi-Elgot-Trakhtenbrot theorem. We use one second-order variable for each *upstep* (starting events) or *downstep* (terminating events) of the HDA. The main difference with words is that each upstep or downstep involves several events. We rely on the existence of a canonical *sparse step decomposition* for any interval ipomset. Intuitively, we prove that this decomposition can be "defined" in MSO.

On the other hand, the usual approach for the MSO-to-automata direction, which works by induction and relies on the closure properties of regular languages, does not work for HDAs, as they are not closed under complement. One could try to use the bounded-width complement instead, but the downward closures present some difficulties. Instead, we rely on a known connection [1] between regular languages of interval ipomsets and regular languages of *step decompositions*. A step decomposition of an ipomset P is a sequence of discrete ipomsets (that is, pomsets where all events are concurrent) such that their *gluing composition* is equal to P. We prove that for every MSO-definable language L of width at most k, the language of all step decompositions of ipomsets in L, viewed as words over a finite alphabet of discrete ipomsets, is regular. To do so, we give a translation from MSO formulas over ipomsets to MSO formulas over words with this new alphabet. It was shown in [1] that the downward closure of L is then regular.

The paper is organised as follows. Interval pomsets with interfaces and step decompositions are defined in Section 2, and higher-dimensional automata in Section 3. In Section 4, we introduce monadic second-order logic and state our main result. Section 5 gives the proof for the MSO-to-HDA direction, and Section 6 for the HDA-to-MSO one. Missing proofs can be found in the appendix.

2 Pomsets with Interfaces

We fix a finite alphabet Σ throughout this paper. A *pomset with interfaces*, or *ipomset*, is a structure $(P, <, \dashrightarrow, S, T, \lambda)$ comprising a finite set P, a (strict) partial order[3] $< \subseteq P \times P$ called the *precedence order*, a pseudo-order $\dashrightarrow \subseteq P \times P$ called the *event order*, subsets $S, T \subseteq P$ called *source* and *target* sets, and a *labelling* $\lambda : P \to \Sigma$. We require the following properties:

- for all $e \neq e' \in P$, exactly one of $e < e'$, $e' < e$, $e \dashrightarrow e'$, or $e' \dashrightarrow e$ holds;
- for all $e_1 \in S$, $e_2 \in P$, and $e_3 \in T$, $e_2 \not< e_1$ and $e_3 \not< e_2$.

That is, all points in P are related by precisely one of the orders, sources are $<$-minimal, and targets are $<$-maximal. We may add subscripts "$_P$" to the elements above if necessary.

Ipomsets are a generalisation of standard pomsets (see for example [15]) obtained by adding interfaces and event order. Both are needed in order to properly connect them with HDAs, see [8]. In particular, event order is necessary in order to define gluing composition, see below. In [8] and other works, a transitively closed event order is used instead of the pseudo-order we use here; we find it more convenient to use the non-transitive version which otherwise is equivalent.

[3] A strict pseudo-order is a relation which is irreflexive and asymmetric. It is a strict partial order if it is also transitive. We will omit the qualifier "strict".

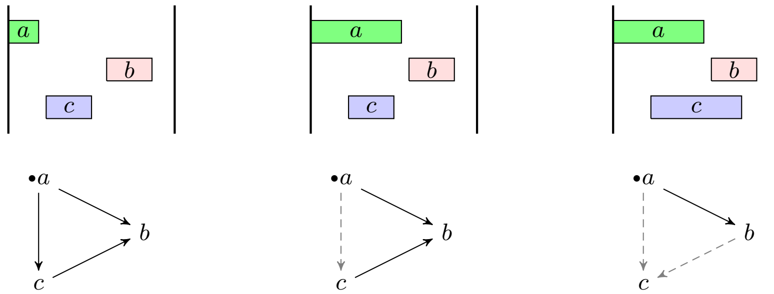


Fig. 1: Activity intervals of events (top) and corresponding ipomsets (bottom), *cf.* Ex. 1. Full arrows indicate precedence order; dashed arrows indicate event order; bullets indicate interfaces.

An ipomset P is a *word* (with interfaces) if $<$ is total and *discrete* if $< = \emptyset$ (then $\dashrightarrow$ is total). P is a *pomset* if $S = T = \emptyset$, a *conclist* (short for "concurrency list") if it is a discrete pomset, a *starter* if it is discrete and $T = P$, a *terminator* if it is discrete and $S = P$, and an *identity* if it is both a starter and a terminator. The source and target *interfaces* of P are the conclists $S_P = (S, \dashrightarrow_{\restriction S\times S}, \lambda_{\restriction S})$ and $T_P = (T, \dashrightarrow_{\restriction T\times T}, \lambda_{\restriction T})$, where "$_\restriction$" denotes restriction.

Figure 1 shows some simple examples. Source and target events are marked by "•" at the left or right side, and if the event order is not shown, we assume that it goes downwards. Precedence $<$ and event order $\dashrightarrow$ are intended to order sequential and concurrent events, respectively.

An ipomset P is *interval* if $<_P$ is an interval order [13]; that is, if it admits an interval representation given by functions $f, g : (P, <_P) \to (\mathbb{R}, <_\mathbb{R})$ such that $f(e) \le_\mathbb{R} g(e)$ for all $e \in P$ and $e_1 <_P e_2$ iff $g(e_1) <_\mathbb{R} f(e_2)$ for all $e_1, e_2 \in P$. Given that our ipomsets represent activity intervals of events, any of the ipomsets we will encounter will be interval, and we omit the qualification "interval". We emphasise that this is *not* a restriction, but rather induced by the semantics, [26]. The *width* $\mathsf{wid}(P)$ of an ipomset P is the cardinality of a maximal $<$-antichain.

We let iiPoms denote the set of (interval) ipomsets and $\mathsf{iiPoms}_{\le k} = \{P \in \mathsf{iiPoms} \mid \mathsf{wid}(P) \le k\}$. We write $\mathsf{St}, \mathsf{Te}, \mathsf{Id} \subseteq \mathsf{iiPoms}$ for the sets of starters, terminators, and identities and let $\Omega = \mathsf{St} \cup \mathsf{Te}$. Further, for $S \in \{\mathsf{St}, \mathsf{Te}, \mathsf{Id}, \Omega\}$, $S_{\le k} = S \cap \mathsf{iiPoms}_{\le k}$. Note that $\mathsf{Id} = \mathsf{St} \cap \mathsf{Te}$ and $\mathsf{Id}_{\le k} = \mathsf{St}_{\le k} \cap \mathsf{Te}_{\le k}$. We introduce special notation for starters and terminators and write ${}_A{\uparrow}U = {}_{U\setminus A}U_U$ and $U{\downarrow}_B = {}_U U_{U\setminus B}$. The intuition is that ${}_A{\uparrow}U$ does nothing but start the events in $A = U \setminus S_U$ and $U{\downarrow}_B$ terminates the events in $B = U \setminus T_B$.

Ipomsets may be *refined* by shortening activity intervals, potentially removing concurrency and expanding precedence. The inverse to refinement is called *subsumption* and defined as follows. For ipomsets P and Q we say that Q subsumes P and write $P \sqsubseteq Q$ if there is a bijection $f : P \to Q$ for which

(1) $f(S_P) = S_Q$, $f(T_P) = T_Q$, and $\lambda_Q \circ f = \lambda_P$,
(2) $f(e_1) <_Q f(e_2) \implies e_1 <_P e_2$, and $e_1 \dashrightarrow_P e_2 \implies f(e_1) \dashrightarrow_Q f(e_2)$.

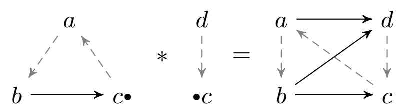


Fig. 2: Gluing composition of ipomsets.

This definition adapts the one of [15] to event orders and interfaces. Intuitively, P has more order and less concurrency than Q.

Example 1. In Fig. 1 there is a sequence of subsumptions from left to right: $\bullet acb \sqsubseteq \left[\begin{smallmatrix}\bullet a\\ c\end{smallmatrix}\right] b \sqsubseteq \left[\begin{smallmatrix}\bullet a \to b\\ c\end{smallmatrix}\right]$. An event e_1 is smaller than e_2 in the precedence order if e_1 is terminated before e_2 is started; e_1 is smaller than e_2 in the event order if they are concurrent and e_1 is above e_2 in the respective conclist.

Isomorphisms of ipomsets are invertible subsumptions, *i.e.*, bijections f for which the second item above is strengthened to

$$(2') \quad f(e_1) <_Q f(e_2) \iff e_1 <_P e_2 \text{ and } e_1 \dashrightarrow_P e_2 \iff f(e_1) \dashrightarrow_Q f(e_2).$$

We write $P \simeq Q$ if P and Q are isomorphic. Because of the requirement that all elements are related by $<$ or $\dashrightarrow$, there is at most one isomorphism between any two ipomsets. That means that we may without danger switch between ipomsets and their isomorphism classes, and we will do so often in the sequel.

The *gluing* $P * Q$ of ipomsets P and Q is defined if $T_P = S_Q$ *as conclists* (hence $\dashrightarrow_{P\restriction T_P \times T_P} = \dashrightarrow_{Q\restriction S_Q \times S_Q}$ and $\lambda_{P\restriction T_P} = \lambda_{Q\restriction S_Q}$), and then $P * Q = (P \cup Q, <, \dashrightarrow, S_P, T_Q, \lambda)$, where $< = (<_P \cup <_Q \cup (P \setminus T_P) \times (Q \setminus S_Q))^+$, $\dashrightarrow = \dashrightarrow_P \cup \dashrightarrow_Q$, and $\lambda = \lambda_P \cup \lambda_Q$. (Here $^+$ denotes transitive closure.) Ipomsets in Id are identities for $*$. Figure 2 shows an example.

Any ipomset P can be decomposed as a gluing of starters and terminators $P = P_1 * \cdots * P_n$ [10, 16]. Such a presentation we call a *step decomposition*. If starters and terminators are alternating, the step decomposition is called *sparse*.

Lemma 2 ([11]). *Every ipomset P has a unique sparse step decomposition.*

We will also use the following notion, introduced in [1]. A word $P_1 \dots P_n \in \Omega^*$ is *coherent* if the gluing $P_1 * \cdots * P_n$ is defined. We denote by $\mathsf{Coh} \subseteq \Omega^*$ the set of coherent words and $\mathsf{Coh}_{\leq k} = \mathsf{Coh} \cap \mathsf{iiPoms}_{\leq k}$.

3 Higher-dimensional automata

Let $\square$ denote the set of conclists. A *precubical set*

$$\mathcal{H} = (\mathcal{H}, \mathsf{ev}, \{\delta^0_{A,U}, \delta^1_{A,U} \mid U \in \square, A \subseteq U\})$$

consists of a set of *cells* $\mathcal{H}$ together with a function $\mathsf{ev} : \mathcal{H} \to \square$ which to every cell assigns a conclist of concurrent events which are active in it. We write

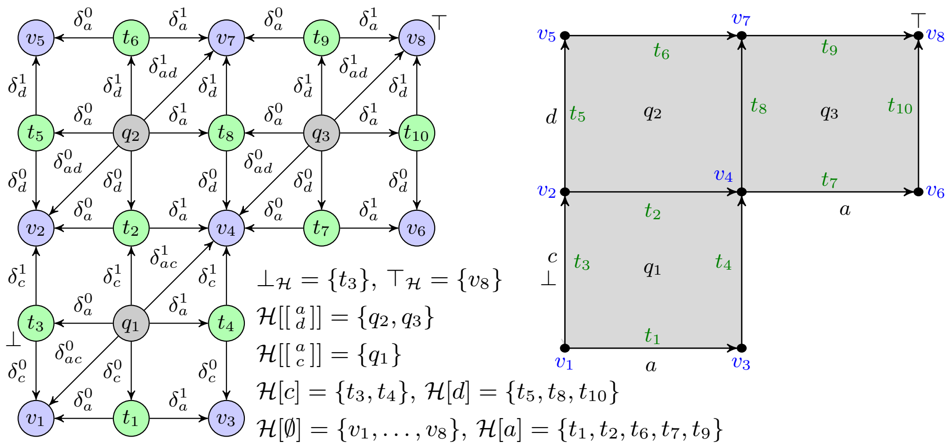


Fig. 3: A two-dimensional HDA $\mathcal{H}$ on $\Sigma = \{a, c, d\}$, see Ex. 3.

$\mathcal{H}[U] = \{q \in \mathcal{H} \mid \mathsf{ev}(q) = U\}$ for the cells of type U. For every $U \in \square$ and $A \subseteq U$ there are *face maps* $\delta^0_A, \delta^1_A : \mathcal{H}[U] \to \mathcal{H}[U \setminus A]$ which satisfy $\delta^\nu_A \delta^\mu_B = \delta^\mu_B \delta^\nu_A$ for $A \cap B = \emptyset$ and $\nu, \mu \in \{0, 1\}$. The *upper* face maps δ^1_A terminate events in A and the *lower* face maps δ^0_A transform a cell q into one in which the events in A have not yet started. A *higher-dimensional automaton* (*HDA*) $\mathcal{H} = (\mathcal{H}, \bot_{\mathcal{H}}, \top_{\mathcal{H}})$ is a finite precubical set together with subsets $\bot_{\mathcal{H}}, \top_{\mathcal{H}} \subseteq \mathcal{H}$ of *start* and *accept* cells. The *dimension* of an HDA $\mathcal{H}$ is $\dim(\mathcal{H}) = \sup\{|\mathsf{ev}(q)| \mid q \in \mathcal{H}\} \in \mathbb{N}$.

A standard automaton is the same as a one-dimensional HDA $\mathcal{H}$ with the property that for all $q \in \bot_{\mathcal{H}} \cup \top_{\mathcal{H}}$, $\mathsf{ev}(q) = \emptyset$: cells in $\mathcal{H}[\emptyset]$ are states, cells in $\mathcal{H}[\{a\}]$ for $a \in \Sigma$ are a-labelled transitions, and face maps $\delta^0_{\{a\}}$ and $\delta^1_{\{a\}}$ attach source and target states to transitions. In contrast to ordinary automata we allow start and accept *transitions* instead of merely states, so languages of one-dimensional HDAs may contain words with interfaces.

Example 3. Figure 3 shows a two-dimensional HDA as a combinatorial object (left) and in a geometric realisation (right). It consists of 21 cells: states $\mathcal{H}_0 = \{v_1, \dots, v_8\}$ in which no event is active ($\mathsf{ev}(v_i) = \emptyset$), transitions $\mathcal{H}_1 = \{t_1, \dots, t_{10}\}$ in which one event is active (*e.g.*, $\mathsf{ev}(t_3) = \mathsf{ev}(t_4) = c$), squares $\mathcal{H}_2 = \{q_1, q_2, q_3\}$ where $\mathsf{ev}(q_1) = [\begin{smallmatrix} a \\ c \end{smallmatrix}]$ and $\mathsf{ev}(q_2) = \mathsf{ev}(q_3) = [\begin{smallmatrix} a \\ d \end{smallmatrix}]$. The arrows between cells in the left representation correspond to the face maps connecting them. For example, the upper face map δ^1_{ac} maps q_1 to v_4 because the latter is the cell in which the active events a and c of q_1 have been terminated. On the right, face maps are used to glue cells, so that for example $\delta^1_{ac}(q_1)$ is glued to the top right of q_1. In this and other geometric realisations, when we have two concurrent events a and c with $a \dashrightarrow c$, we will draw a horizontally and c vertically.

Computations of HDAs are *paths*, *i.e.*, sequences $\alpha = (q_0, \varphi_1, q_1, \dots, q_{n-1}, \varphi_n, q_n)$ consisting of cells $q_i \in \mathcal{H}$ and symbols φ_i which indicate face map types: for every $i \in \{1, \dots, n\}$, $(q_{i-1}, \varphi_i, q_i)$ is either

– $(\delta^0_A(q_i), \nearrow^A, q_i)$ for $A \subseteq \mathsf{ev}(q_i)$ (an *upstep*)
– or $(q_{i-1}, \searrow_A, \delta^1_A(q_{i-1}))$ for $A \subseteq \mathsf{ev}(q_{i-1})$ (a *downstep*).

Downsteps terminate events, following upper face maps, whereas upsteps start events by following inverses of lower face maps. We denote by $\mathsf{ups}(\mathcal{H})$ and $\mathsf{downs}(\mathcal{H})$ the finite set of upsteps and downsteps of $\mathcal{H}$.

The *source* and *target* of α as above are $\mathsf{src}(\alpha) = q_0$ and $\mathsf{tgt}(\alpha) = q_n$. A path α is *accepting* if $\mathsf{src}(\alpha) \in \bot_{\mathcal{H}}$ and $\mathsf{tgt}(\alpha) \in \top_{\mathcal{H}}$. Paths α and β may be concatenated if $\mathsf{tgt}(\alpha) = \mathsf{src}(\beta)$; their concatenation is written $\alpha * \beta$.

Path equivalence is the congruence $\simeq$ generated by $(q \nearrow^A r \nearrow^B p) \simeq (q \nearrow^{A\cup B} p)$, $(p \searrow_A r \searrow_B q) \simeq (p \searrow_{A\cup B} q)$, and $\gamma\alpha\delta \simeq \gamma\beta\delta$ whenever $\alpha \simeq \beta$. This relation allows to assemble subsequent upsteps or downsteps into one bigger step.

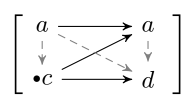
The *event ipomset* $\mathsf{ev}(\alpha)$ of a path α is defined recursively as follows:

– if $\alpha = (q)$, then $\mathsf{ev}(\alpha) = \mathsf{id}_{\mathsf{ev}(q)}$;
– if $\alpha = (q \nearrow^A p)$, then $\mathsf{ev}(\alpha) = {}_A{\uparrow}\mathsf{ev}(p)$;
– if $\alpha = (p \searrow_B q)$, then $\mathsf{ev}(\alpha) = \mathsf{ev}(p){\downarrow}_B$;
– if $\alpha = \alpha_1 * \cdots * \alpha_n$ is a concatenation, then $\mathsf{ev}(\alpha) = \mathsf{ev}(\alpha_1) * \cdots * \mathsf{ev}(\alpha_n)$.

Note that upsteps in α correspond to starters in $\mathsf{ev}(\alpha)$ and downsteps correspond to terminators. Path equivalence $\alpha \simeq \beta$ implies $\mathsf{ev}(\alpha) = \mathsf{ev}(\beta)$ [9].

Example 4. The HDA X of Ex. 3 (Fig. 3) admits several accepting paths, for example $t_3 \nearrow^a q_1 \searrow_c t_2 \nearrow^d q_2 \searrow_a t_8 \nearrow^a q_3 \searrow_{ad} v_8$. Its event ipomset is

$${}_a{\uparrow}\left[\begin{smallmatrix} a \\ c \end{smallmatrix}\right] * \left[\begin{smallmatrix} a \\ c \end{smallmatrix}\right]{\downarrow}_c * {}_d{\uparrow}\left[\begin{smallmatrix} a \\ d \end{smallmatrix}\right] * \left[\begin{smallmatrix} a \\ d \end{smallmatrix}\right]{\downarrow}_a * {}_a{\uparrow}\left[\begin{smallmatrix} a \\ d \end{smallmatrix}\right] * \left[\begin{smallmatrix} a \\ d \end{smallmatrix}\right]{\downarrow}_{ad} =$$



which is a sparse step decomposition. This path is equivalent to $t_3 \nearrow^a q_1 \searrow_c t_2 \nearrow^d q_2 \searrow_a t_8 \nearrow^a q_3 \searrow_a t_{10} \searrow_d v_8$ which induces the coherent word w_1 of Fig.4.

The *language* of an HDA $\mathcal{H}$ is $L(\mathcal{H}) = \{\mathsf{ev}(\alpha) \mid \alpha \text{ accepting path in } \mathcal{H}\}$.

For $A \subseteq \mathsf{iiPoms}$ we let

$$A{\downarrow} = \{P \in \mathsf{iiPoms} \mid \exists Q \in A : P \sqsubseteq Q\}.$$

A *language* is a subset $L \subseteq \mathsf{iiPoms}$ for which $L{\downarrow} = L$. The *width* of L is $\mathsf{wid}(L) = \sup\{\mathsf{wid}(P) \mid P \in L\}$. For $k \geq 0$ and $L \in \mathsf{iiPoms}$, denote $L_{\leq k} = \{P \in L \mid \mathsf{wid}(P) \leq k\}$. The *singleton ipomsets* are $[a]$ $[\bullet a]$, $[a\bullet]$ and $[\bullet a \bullet]$, for all $a \in \Sigma$.

A language is *regular* if it is the language of a finite HDA. It is *rational* if it is constructed from $\emptyset$, $\{\mathsf{id}_\emptyset\}$ and discrete ipomsets using $\cup$, $*$ and $^+$ (Kleene plus) [9]. Languages of HDAs are closed under subsumption, that is, if L is regular, then $L{\downarrow} = L$ [8, 9]. The rational operations above have to take this closure into account.

Theorem 5 ([9]). *A language is regular if and only if it is rational.*

Lemma 6 ([9]). *Any regular language has finite width.*

It immediately follows that the universal language iiPoms is *not* rational.

4 MSO

Monadic second-order (MSO) logic is an extension of first-order logic allowing to quantify existentially and universally over elements as well as subsets of the domain of the structure. It uses second-order variables $X, Y, \dots$ interpreted as subsets of the domain in addition to the first-order variables $x, y, \dots$ interpreted as elements of the domain of the structure, and a new binary predicate $x \in X$ interpreted commonly. We refer the reader to [22] for more details about MSO.

We interpret MSO over iiPoms. Thus we consider the signature $\mathcal{S} = \{<, \dashrightarrow, (a)_{a \in \Sigma}, s, t\}$ where $<$ and $\dashrightarrow$ are binary relation symbols and the a's, s and t are unary predicates (over first-order variables). We associate to every ipomset $(P, <, \dashrightarrow, S, T, \lambda)$ the relational structure $S = (P; <; \dashrightarrow; (a)_{a \in \Sigma}; \mathsf{s}; \mathsf{t})$ where $<$ and $\dashrightarrow$ are interpreted as the orderings $<$ and $\dashrightarrow$ over P, and $a(x)$, $\mathsf{s}(x)$ and $\mathsf{t}(x)$ hold respectively if and only if $\lambda(x) = a$, $x \in S$ and $x \in T$. We say that a relation $R \subseteq P^n \times (2^P)^m$ is *MSO-definable* in S if and only if there exists an MSO-formula $\psi(x_1, \dots, x_n, X_1, \dots, X_m)$, where the x_i's (resp. X_j's) are free first (resp. second) order variables, such that their interpretation in S is a tuple of R. The well-formed MSO formulas are built using the following grammar:

$$\begin{aligned} \psi ::= {} & a(x) \mid \mathsf{s}(x) \mid \mathsf{t}(x) \mid x < y \mid x \dashrightarrow y \mid x \in X \\ & \exists x.\, \psi \mid \forall x.\, \psi \mid \exists X.\, \psi \mid \forall X.\, \psi \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \neg \psi \end{aligned}$$

In order to shorten formulas we use several notations and shortcuts such as $\psi_1 \implies \psi_2$. We define $x \to y := x < y \wedge \neg(\exists z. x < z < y)$.

Let $\psi(x_1, \dots, x_n, X_1, \dots, X_m)$ be an MSO formula whose free variables are $x_1, \dots, x_n, X_1, \dots, X_m$ and let $P \in \mathsf{iiPoms}$. The pair of functions $\nu = (\nu_1, \nu_2)$ where $\nu_1 : \{x_1, \dots, x_n\} \to P$ and $\nu_2 : \{X_1, \dots, X_m\} \to 2^P$ is called a *valuation* or an *interpretation*. We write $P \models_\nu \psi$, or, by a slight abuse of notation, $P \models \psi\big(\nu(x_1), \dots, \nu(x_n), \nu(X_1), \dots, \nu(X_m)\big)$, if ψ holds when x_i and X_j are interpreted as $\nu(x_i)$ and $\nu(X_j)$. A *sentence* is a formula without free variables. In this case no valuation is needed. Given an MSO sentence ψ, we define $L(\psi) = \{P \in \mathsf{iiPoms} \mid P \models \psi\}$. Note that this may not be closed under subsumption, hence not a language in our sense. A set $L \in \mathsf{iiPoms}$ is MSO-definable if and only if there exists an MSO sentence ψ over $\mathcal{S}$ such that $L = L(\psi)$.

Example 7. Let $\varphi = \exists x \, \exists y.\, a(x) \wedge b(y) \wedge \neg(x < y) \wedge \neg(y < x)$. That is, there are at least two concurrent events, one labelled a and the other b. $L(\varphi)$ is not width-bounded, as φ is satisfied, among others, by any conclist which contains at least one a and one b, nor closed under subsumption, given that $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] \models \varphi$ but $ab, ba \not\models \varphi$. Note, however, that $L(\varphi)_{\le k}{\downarrow}$ is a regular language for any k.

We will use also MSO over words of $\Omega_{\le k}^*$. The definitions above can be easily adapted to this case by considering the words as structures of the form $(w, <, \lambda : w \to \Omega_{\le k})$: totally ordered pomsets over the alphabet $\Omega_{\le k}$, and the signature $\{<, (D)_{D \in \Omega_{\le k}}\}$: the atomic predicates are $D(x)$ for $D \in \Omega_{\le k}$, $x < y$ and $x \in X$, with first-order variables ranging over positions in the word and second-order variables over sets of positions. We denote by MSO_Ω^k the set of MSO formulas over $\Omega^*_{\leq k}$. For example the following MSO^2_Ω formula where $P_i \in \Omega_{\leq 2}$ stands for the ith discrete ipomset of w_1 in Fig. 4 is satisfied only by w_1.

$$\varphi' := \exists y_1, \dots, y_7. \bigwedge_{1\leq i\leq 7} P_i(y_i) \wedge y_1 \to \dots \to y_7 \wedge \forall y. \bigvee_{1\leq i\leq 7} y = y_i$$

The main result of this paper is the following:

Theorem 8. *For all $L \subseteq \mathsf{iiPoms}$,*

1. *if L is MSO-definable, then $L_{\leq k}\downarrow$ is regular for all $k \in \mathbb{N}$.*
2. *if L is regular, then it is MSO-definable.*

Corollary 9. *For all $k \in \mathbb{N}$, a language $L \subseteq \mathsf{iiPoms}_{\leq k}$ is regular if and only if it is MSO-definable.*

The next two sections are devoted to the proof of Thm. 8. For the first assertion we effectively build an HDA $\mathcal{H}$ from a sentence φ such that $L(\mathcal{H}) = L(\varphi)_{\leq k}\downarrow$ for all $k \in \mathbb{N}$. Since emptiness of HDAs is decidable [1], we have that for MSO sentences φ such that $L(\varphi) = L(\varphi)_{\leq k}\downarrow$, the satisfiability problem (asking given such a formula φ, if there exists P such that $P \models \varphi$), and the model-checking problem for HDAs (given φ and an HDA $\mathcal{H}$, do we have $L(\mathcal{H}) \subseteq L(\varphi)$) are both decidable. Actually, looking more closely at our construction which goes through finite automata accepting step sequences, we get the same result for MSO formulas even without the assumption that $L(\varphi)$ is downward-closed (but still over $\mathsf{iiPoms}_{\leq k}$, and not iiPoms). This could also be shown alternatively by observing that $\mathsf{iiPoms}_{\leq k}$ has bounded treewidth (in fact, even bounded pathwidth), and applying Courcelle's theorem [5]. In fact our implied proof of decidability is relatively similar, using step sequences instead of path decompositions.

For the second assertion of the theorem, we show that regular languages of HDAs are MSO-definable, again using an effective construction. Thus, using both directions of Thm. 8 and the closure properties of HDAs, we also get the for all $k \in \mathbb{N}$ and MSO-definable $L \subseteq \mathsf{iiPoms}_{\leq k}$, $L\downarrow$ is MSO-definable. Note that this property does *not* hold for the class of *all* pomsets [12].

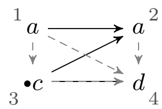
5 From MSO to HDAs

Given an MSO sentence φ over iiPoms we build an HDA $\mathcal{H}$ such that $L(\mathcal{H}) = L(\varphi)_{\leq k}\downarrow$. The first step is to define an MSO-interpretation of interval ipomsets of width at most k into words of $\Omega^+_{\leq k}$, so that:

Lemma 10. *For every MSO sentence φ over iiPoms and every k there exists $\widehat{\varphi} \in \mathrm{MSO}^k_\Omega$ such that for all $P_1 \dots P_n \in (\Omega_{\leq k} \setminus \{\mathsf{id}_\emptyset\})^+$, we have $P_1 \dots P_n \models \widehat{\varphi}$ if and only if $P = P_1 * \dots * P_n$ is well-defined and $P \models \varphi$.*

We will treat the case of the empty ipomset $\mathsf{id}_\emptyset$ separately. We want $\widehat{\varphi}$ to accept only coherent words. This is MSO^k_Ω-definable by:

$$\mathtt{Coh}_k := \forall x \forall y.\, x \to y \implies \bigvee_{P_1 P_2 \in \mathsf{Coh}_{\leq k} \cap \Omega^2_{\leq k}} P_1(x) \wedge P_2(y).$$



$$w_1 = \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \begin{bmatrix} a\bullet \\ \bullet c\bullet \end{bmatrix} & \begin{bmatrix} \bullet a\bullet \\ \bullet c \end{bmatrix} & \begin{bmatrix} \bullet a\bullet \\ d\bullet \end{bmatrix} & \begin{bmatrix} \bullet a \\ \bullet d\bullet \end{bmatrix} & \begin{bmatrix} a\bullet \\ \bullet d\bullet \end{bmatrix} & \begin{bmatrix} \bullet a \\ \bullet d\bullet \end{bmatrix} & [\bullet d] \end{array}$$

Fig. 4: Ipomset and corresponding coherent words. (Numbers indicate positions.)

That is, discrete ipomsets of $\Omega_{\leq k}$ at consecutive positions x and y may be glued.

We let $\widehat{\varphi} := \mathtt{Coh}_k \wedge \varphi'$, where φ' is built by induction on φ. Therefore, we have to consider formulas φ that contain free variables. The free variables of φ' will be all the free first-order variables of φ and second-order variables $X_1, \ldots, X_k$ for every free second-order variable X of φ. .

To be precise, let $w = P_1 \ldots P_n \in \mathsf{Coh}_{\leq k}$ and $P = P_1 * \cdots * P_n$. Let $E = \{1, \ldots, n\} \times \{1, \ldots, k\}$. Our construction is built on a partial function $evt : E \to P$ defined as follows: if P_ℓ consists of events $e_1 \dashrightarrow \cdots \dashrightarrow e_r$, then for every $i \leq r$, $evt(\ell, i) = e_i$. We sometimes abuse notation and write $evt(P_\ell, i)$. Since $e \in P$ may occur in consecutive P_ℓ within w, one must determine when $evt(\ell, i) = evt(\ell', j)$. This can be done when $\ell' = \ell + 1$ as follows. For all $i, j \leq k$, let $M_{i,j} = \{P_1 P_2 \in \Omega^2_{\leq k} \mid evt(1, i) = evt(2, j)\}$. Then

$$glue_{i,j}(x, y) := x \to y \wedge \bigvee_{P_1 P_2 \in M_{i,j}} P_1(x) \wedge P_2(y)\,.$$

More generally, let us define the equivalence relation $\sim$ on E generated by $(\ell, i) \sim (\ell', i')$ if and only if $glue_{i,i'}(\ell, \ell')$ holds. Then for all $(\ell_1, i), (\ell_2, j) \in E$, $(\ell_1, i) \sim (\ell_2, j)$ if and only if $evt(\ell_1, i) = evt(\ell_2, j)$. We have $(\ell, i) \sim (\ell', i')$ is MSO-definable (see Annex.A).

Actually, we construct a formula φ'_τ relative to a function τ which associates with every free first-order variable x of φ some $\tau(x) \in \{1, \ldots, k\}$. We sometimes leave τ implicit. Our aim is to have the following *invariant property* at each step of the induction: $P \models_\nu \varphi$ if and only if $w \models_{\nu'} \varphi'$ for any valuations ν, ν' satisfying the following: (1) $evt(\nu'(x), \tau(x)) = \nu(x)$ and (2) $\bigcup_{1 \leq i \leq k} \{evt(e, i) \mid e \in \nu'(X_i)\} = \nu(X)$.

Example 11. Figure 4 displays an ipomset P and the coherent word $w_1 = P_1 \ldots P_7$ such that $P_1 * \cdots * P_7 = P$. Let $e_1, \ldots, e_4$ be the events of P labelled respectively by the left a, the right a, c, and d and let $p_1, \ldots, p_7$ the positions on w_1 from left to right. Assume that $P \models_\nu \varphi(x, X)$ for some MSO-formula φ and the valuation $\nu(x) = e_1$ and $\nu(X) = \{e_2, e_3\}$. Then, $w_1 \models_{\nu'} \varphi'_{[x \mapsto 1]}(x, X_1, X_2)$ when, for example, $\nu'(x) = p_2$, $\nu'(X_1) = \{p_6\}$ and $\nu'(X_2) = \{p_3\}$ since this valuation satisfies the invariant property. For $\sim$ we have $(p_1, 1) \sim \cdots \sim (p_4, 1)$, $(p_1, 2) \sim (p_2, 2)$, $(p_3, 2) \sim \cdots \sim (p_6, 2) \sim (p_7, 1)$ and $(p_5, 1) \sim (p_6, 1)$. In particular $(p_1, 1) \not\sim (p_5, 1)$ since neither $glue_{1,1}(p_4, p_5)$ nor $glue_{2,1}(p_4, p_5)$ hold.

We are now ready to build φ' by induction on φ. When φ is $\psi_1 \vee \psi_2$ or $\neg\psi$, then we let φ' be $\psi'_1 \vee \psi'_2$ or $\neg\psi'$, respectively. For $\varphi = \exists X\, \psi$ we let $\varphi' :=$

$\exists X_1, \ldots, X_k.\psi'$. The function τ emerges in the case $\varphi = \exists x\, \psi$, where we let $\varphi'_\tau := \bigvee_{1\le i\le k} \exists x\, \psi'_{[x\mapsto i]}$. When $\varphi = x \in X$, we let

$$\varphi'_{[x\mapsto i]} := \bigvee_{1\le j\le k} \exists y\, (x,i) \sim (y,j) \wedge y \in X_j$$

For $\varphi = \mathsf{s}(x)$, we let $\varphi'_{[x\mapsto i]} := \bigwedge_{1\le j\le k} \forall y\, (x,i) \sim (y,j) \implies \mathsf{s}(y,j)$, where $\mathsf{s}(y,j)$ is defined as the disjunction of all $D(y)$ where $evt(D,j) \in S_D$. We define $\varphi'_{[x\mapsto i]}$ similarly when $\varphi = \mathsf{t}(x)$. For $\varphi = x < y$ we let

$$\varphi'_{[x\mapsto i, y\mapsto j]} := \bigwedge_{1\le i',j'\le k} \forall x',y'.\big((x',i') \sim (x,i) \wedge (y',j') \sim (y,j)\big) \implies x' < y'.$$

For $\varphi = x \dashrightarrow y$ we let

$$\varphi'_{[x\mapsto i, y\mapsto j]} := \bigvee_{1\le i'<j'\le k} \exists z\, (z,i') \sim (x,i) \wedge (z,j') \sim (y,j).$$

Finally, when $\varphi = a(x)$, then we let $\varphi'_{[x\mapsto i]}$ be the disjunction of all $D(x)$ where $evt(D,i)$ is labelled by a. As a consequence, we obtain:

Proposition 12. *Let φ be an MSO sentence over* iiPoms*, $k \in \mathbb{N}$, and $L = \{P \in \mathsf{iiPoms}_{\le k} \mid P \models \varphi\}{\downarrow}$. Then L is regular.*

Proof. Let $K = \{P \in \mathsf{iiPoms}_{\le k} \mid P \models \varphi\}$. By Lem. 10, $L' = \{P_1 \ldots P_n \in (\Omega_{\le k} \setminus \{\mathsf{id}_\emptyset\})^+ \mid P_1 * \cdots * P_n \in K\}$ is MSO^k_Ω-definable, and thus so is $L'' = \{P_1 \ldots P_n \in \Omega^+_{\le k} \mid P_1 * \cdots * P_n \in K\}$. By the standard Büchi and Kleene theorems, L'' is obtained from $\emptyset$ and $\Omega_{\le k}$ using $\cup$, $\cdot$ and $^+$. Replacing concatenation of words by gluing composition, we see that L is rational and thus regular by Thm. 5. $\square$

6 From HDAs to MSO

In this section we prove the second assertion of Thm. 8. The proof adapt the classical construction, encoding accepting paths of an automaton, to the case of HDAs. Our construction relies on the uniqueness of the sparse step decomposition (Lem. 2) and the MSO-definability of the relation: "an event is started/terminated before another event is started/terminated" in a sparse step decomposition (Lem. 15 below).

More formally, let $P \in \mathsf{iiPoms}$, then P admits a unique sparse step decomposition $P = P_1 * \cdots * P_n$. Given $e \in P \setminus S_P$, we denote by $\mathsf{St}(e)$ the step where e is started in the decomposition, *i.e.*, the minimal i such that $e \in P_i$. For $e \in P \setminus T_P$, we similarly denote by $\mathsf{Te}(e)$ the step where e is terminated. For $x \in S_P$ we let $\mathsf{St}(x) = -\infty$ and for $x \in T_P$, $\mathsf{Te}(x) = +\infty$. Then P_i contains precisely all $e \in P$ such that $\mathsf{St}(e) \le i \le \mathsf{Te}(e)$, that is all events which are started before or at P_i (or never) and are terminated after or at P_i (or never). In particular, if P_i is a starter, then it starts all e such that $\mathsf{St}(e) = i$, and if it is a terminator, it terminates all e such that $\mathsf{Te}(e) = i$. Note that $\mathsf{St}(e) < \mathsf{Te}(e)$ for all $e \in P$.

Example 13. Proceeding with Ex. 11, let $w_2 = P_1 \dots P_6 = \left[\begin{smallmatrix} a\bullet \\ \bullet c\bullet \end{smallmatrix}\right]\left[\begin{smallmatrix} \bullet a\bullet \\ \bullet c \end{smallmatrix}\right]\left[\begin{smallmatrix} \bullet a\bullet \\ d\bullet \end{smallmatrix}\right]\left[\begin{smallmatrix} \bullet a \\ \bullet d\bullet \end{smallmatrix}\right]$ $\left[\begin{smallmatrix} a\bullet \\ \bullet d\bullet \end{smallmatrix}\right]\left[\begin{smallmatrix} \bullet a \\ \bullet d \end{smallmatrix}\right]$ be the sparse step decomposition of P (see also Ex. 4). We have $\mathsf{St}(e_3) = -\infty, \mathsf{St}(e_1) = 1, \mathsf{St}(e_4) = 3$ and $\mathsf{St}(e_2) = 5$. Also, $\mathsf{Te}(e_3) = 2, \mathsf{Te}(e_1) = 4$ and $\mathsf{Te}(e_2) = \mathsf{Te}(e_4) = 6$. Further, P_1 contains e_1 since $\mathsf{St}(e_1) = 1$ and e_3 because $\mathsf{St}(e_3) \leq 1 \leq \mathsf{Te}(e_3)$; P_4 contains e_1 since $\mathsf{Te}(e_1) = 4$ and e_4 because $\mathsf{St}(e_4) \leq 4 \leq \mathsf{Te}(e_4)$.

The next lemma describes the existence of an accepting path inducing a sparse step decomposition as the existence of labellings $\rho_\nearrow$ and $\rho_\searrow$ mapping each started or terminated event of P to the upstep or downstep of the HDA performing it.

Lemma 14. *Let $\mathcal{H}$ be an HDA and $P \in \mathsf{iiPoms} \setminus \mathsf{Id}$ whose sparse step decomposition is $P_1 * \dots * P_n$. We have $P \in L(\mathcal{H})$ if and only if there exist $\rho_\nearrow : P \setminus S_P \to \mathsf{ups}(X)$ and $\rho_\searrow : P \setminus T_P \to \mathsf{downs}(X)$ such that, for all $e_1, e_2 \in P$:*

1. *if $\mathsf{St}(e_1) = \mathsf{St}(e_2)$ then $\rho_\nearrow(e_1) = \rho_\nearrow(e_2)$;*
2. *if $\mathsf{Te}(e_1) = \mathsf{Te}(e_2)$ then $\rho_\searrow(e_1) = \rho_\searrow(e_2)$;*
3. *if $\mathsf{St}(e_2) = \mathsf{Te}(e_1) + 1$ then $\mathsf{src}(\rho_\nearrow(e_2)) = \mathsf{tgt}(\rho_\searrow(e_1))$;*
4. *if $\mathsf{Te}(e_2) = \mathsf{St}(e_1) + 1$ then $\mathsf{src}(\rho_\searrow(e_2)) = \mathsf{tgt}(\rho_\nearrow(e_1))$;*
5. *if $\rho_\nearrow(e_1) = (p, \nearrow^A, q)$ then*
$$\begin{aligned} A &= (U = \{e \mid \mathsf{St}(e) = \mathsf{St}(e_1)\}, \dashrightarrow_{P_{\restriction U}}, \lambda_{P_{\restriction U}}), \\ \mathsf{ev}(q) &= (V = \{e \mid \mathsf{St}(e) \leq \mathsf{St}(e_1) < \mathsf{Te}(e)\}, \dashrightarrow_{P_{\restriction V}}, \lambda_{P_{\restriction V}}); \end{aligned}$$
6. *if $\rho_\searrow(e_1) = (p, \searrow_A, q)$ then*
$$\begin{aligned} A &= (U = \{e \mid \mathsf{Te}(e) = \mathsf{Te}(e_1)\}, \dashrightarrow_{P_{\restriction U}}, \lambda_{P_{\restriction U}}), \\ \mathsf{ev}(p) &= (V = \{e \mid \mathsf{St}(e) < \mathsf{Te}(e_1) \leq \mathsf{Te}(e)\}, \dashrightarrow_{P_{\restriction V}}, \lambda_{P_{\restriction V}}); \end{aligned}$$
7. *if $\mathsf{St}(e_1) = 1$ then $\mathsf{src}(\rho_\nearrow(e_1)) \in \bot_\mathcal{H}$;*
8. *if $\mathsf{Te}(e_1) = 1$ then $\mathsf{src}(\rho_\searrow(e_1)) \in \bot_\mathcal{H}$;*
9. *if $\mathsf{St}(e_1) = n$ then $\mathsf{tgt}(\rho_\nearrow(e_1)) \in \top_\mathcal{H}$;*
10. *if $\mathsf{Te}(e_1) = n$ then $\mathsf{tgt}(\rho_\searrow(e_1)) \in \top_\mathcal{H}$.*

As $P \notin \mathsf{Id}$, $\rho_\nearrow$ or $\rho_\searrow$ must be defined for at least one element of P above.

Our goal is to show that the conditions given by Lem. 14 can be expressed in MSO. We want to define a formula $\exists X_1 \dots \exists X_m . \exists Y_1 \dots \exists Y_n . \varphi$ with one X_i (resp. Y_j) for each upstep (resp. downstep) of the HDA. Intuitively, each X_i (Y_j) will contain all the events started (terminated) by performing the corresponding upstep (downstep). The sentence φ expresses that each event belongs to exactly one X_i (unless it is a source, in which case it belongs to none) and one Y_i (unless it is a target), and that the resulting labellings $\rho_\nearrow$ and $\rho_\searrow$ satisfy the conditions of the lemma. Hence, identity events do not belong to any X_i or Y_j. Nevertheless, conditions 5 and 6 ensure that they are consistent with the encoded path. Let us first prove that the relations used in Lem. 14 are MSO-definable.

Lemma 15. *For $f, g \in \{\mathsf{St}, \mathsf{Te}\}$ and $\bowtie \in \{=, <, >\}$, the relations $f(x) \bowtie g(y)$, $\min(f)$ and $\max(f)$ are MSO-definable.*

Proof. We first define $\mathsf{Te}(x) < \mathsf{St}(y)$ as the formula $x < y$, together with $\mathsf{St}(x) < \mathsf{Te}(y) := \neg(\mathsf{Te}(y) < \mathsf{St}(x))$. Because starters and terminators alternate in the sparse step decomposition, we can then let

$$\begin{aligned}
\mathsf{St}(x) < \mathsf{St}(y) &:= \exists z.\, \mathsf{St}(x) < \mathsf{Te}(z) \land \mathsf{Te}(z) < \mathsf{St}(y),\\
\mathsf{St}(x) = \mathsf{St}(y) &:= \neg(\mathsf{St}(x) < \mathsf{St}(y)) \land \neg(\mathsf{St}(y) < \mathsf{St}(x)) \land \neg\mathsf{s}(x) \land \neg\mathsf{s}(y)\\
\min(\mathsf{St}(x)) &:= \neg\mathsf{s}(x) \land \neg\exists y.\, \mathsf{Te}(y) < \mathsf{St}(x)\\
\max(\mathsf{Te}(x)) &:= \neg\mathsf{t}(x) \land \neg\exists y.\, \mathsf{St}(y) > \mathsf{Te}(x)\,.
\end{aligned}$$

The other formulas are defined similarly. □

We can also define $\mathsf{St}(y) = \mathsf{Te}(x) + 1$ and $\mathsf{Te}(y) = \mathsf{St}(x) + 1$ using standard techniques. Observe that $\mathsf{Te}(x) < \mathsf{St}(y)$ implies $\neg t(x) \land \neg\mathsf{s}(y)$, given that the end of the x-event precedes the beginning of the y-event. As a consequence $\mathsf{St}(x) < \mathsf{St}(y)$ implies $\neg\mathsf{s}(y)$. On the other hand $\mathsf{St}(x) < \mathsf{Te}(y)$ holds in particular when x or y are interpreted as identities.

Proposition 16. *Given an HDA $\mathcal{H}$, one can construct an MSO sentence φ such that $L(\mathcal{H}) = \{P \in \mathsf{iiPoms} \mid P \models \varphi\}$.*

Proof. We define

$$\begin{aligned}
\varphi := (\exists x.\, \neg\mathsf{s}(x) \lor \neg\mathsf{t}(x)) \implies \exists X_1, \dots, X_m.\, \exists Y_1, \dots, Y_n.\, \textstyle\bigwedge_{i=0,\dots,10} \varphi_i\\
\land\, (\forall y.\, \mathsf{s}(y) \land \mathsf{t}(y)) \implies \textstyle\bigvee_{\substack{p \in \bot_{\mathcal{H}} \cap \top_{\mathcal{H}}\\ \mathsf{ev}(p) \neq \emptyset}} \exists y_1, \dots, y_{|\mathsf{ev}(p)|}.\mathsf{ev}(p)(y_1, \dots, y_{|\mathsf{ev}(p)|}).
\end{aligned}$$

where φ_0 checks that the X_i's and Y_i's define labellings $\rho_{\nearrow}$ and $\rho_{\searrow}$ as in Lem. 14, that is, each event belongs to at most one X_i (is associated with at most one upstep) and one Y_i, and to no X_i iff it is a source and to no Y_i iff it is a target. The other formulas φ_i check condition i of Lem. 14. The second line of φ is satisfied by all non-empty identities accepted by $\mathcal{H}$. Thus $L(\varphi) = L(\mathcal{H}) \setminus \{\mathsf{id}_\emptyset\}$. If $\mathsf{id}_\emptyset \in L(\mathcal{H})$ then $L(\mathcal{H}) = L(\varphi \lor \neg\exists x.\, \mathtt{true})$.

7 Conclusion

This paper enriches the language theory of higher-dimensional automata with a Büchi-Elgot-Trakhtenbrot-like theorem. We have shown that the subsumption closures of MSO-definable subsets of $\mathsf{iiPoms}_{\leq k}$ are regular and that regular languages of HDAs are MSO-definable, both with effective constructions. Also, the MSO theory of $\mathsf{iiPoms}_{\leq k}$ and the MSO model-checking for HDAs are decidable.

Theorem 8 induces also a construction, for an MSO sentence φ over $\mathsf{iiPoms}_{\leq k}$, of $\varphi{\downarrow}$ such that $L(\varphi{\downarrow}) = L(\varphi){\downarrow}$. This property fails when we consider non-interval pomsets. However, the construction of $\varphi{\downarrow}$ is not efficient, as the current workflow is to transform φ to an HDA and then get $\varphi{\downarrow}$. We are wondering whether a more direct construction is possible.

Our work could be continued by considering logics weaker than MSO. For example, the study of the expressive power of first order logic over $\mathsf{iiPoms}_{\leq k}$ would be useful for model-checking purposes. In this regard, another operational model that would naturally arise is a class of ω-HDAs: HDAs over infinite ipomsets.

References

1. Amazigh Amrane, Hugo Bazille, Uli Fahrenberg, and Krzysztof Ziemiański. Closure and decision properties for higher-dimensional automata. In Erika Ábrahám, Clemens Dubslaff, and Silvia Lizeth Tapia Tarifa, editors, *ICTAC*, volume 14446 of *Lecture Notes in Computer Science*, pages 295–312. Springer, 2023.
2. Nicolas Bedon. Logic and branching automata. *Log. Methods Comput. Sci.*, 11(4), 2015.
3. J. Richard Büchi. Weak second order arithmetic and finite automata. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.
4. J. Richard Büchi. On a decision method in restricted second order arithmetic. In Ernest Nagel, Patrick Suppes, and Alfred Tarski, editors, *LMPS'60*, pages 1–11. Stanford University Press, 1962.
5. Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.
6. John Doner. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4(5):406–451, 1970.
7. Calvin C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–52, 1961.
8. Uli Fahrenberg, Christian Johansen, Georg Struth, and Krzysztof Ziemiański. Languages of higher-dimensional automata. *Mathematical Structures in Computer Science*, 31(5):575–613, 2021.
9. Uli Fahrenberg, Christian Johansen, Georg Struth, and Krzysztof Ziemiański. A Kleene theorem for higher-dimensional automata. In Bartek Klin, Sławomir Lasota, and Anca Muscholl, editors, *CONCUR*, volume 243 of *Leibniz International Proceedings in Informatics*, pages 29:1–29:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.
10. Uli Fahrenberg, Christian Johansen, Georg Struth, and Krzysztof Ziemiański. Posets with interfaces as a model for concurrency. *Information and Computation*, 285(B):104914, 2022.
11. Uli Fahrenberg and Krzysztof Ziemiański. A Myhill-Nerode theorem for higher-dimensional automata. In Luís Gomes and Robert Lorenz, editors, *PETRI NETS*, volume 13929 of *Lecture Notes in Computer Science*, pages 167–188. Springer, 2023.
12. Jean Fanchon and Rémi Morin. Pomset languages of finite step transition systems. In Giuliana Franceschinis and Karsten Wolf, editors, *PETRI NETS*, volume 5606 of *Lecture Notes in Computer Science*, pages 83–102. Springer, 2009.
13. Peter C. Fishburn. *Interval Orders and Interval Graphs: A Study of Partially Ordered Sets*. Wiley, 1985.
14. Blaise Genest, Dietrich Kuske, and Anca Muscholl. A Kleene theorem and model checking algorithms for existentially bounded communicating automata. *Information and Computation*, 204(6):920–956, 2006.
15. Jan Grabowski. On partial languages. *Fundamentae Informatica*, 4(2):427, 1981.
16. Ryszard Janicki and Maciej Koutny. Operational semantics, interval orders and sequences of antichains. *Fundamentae Informatica*, 169(1-2):31–55, 2019.
17. Dietrich Kuske. Infinite series-parallel posets: Logic and languages. In *ICALP*, volume 1853 of *Lecture Notes in Computer Science*, pages 648–662. Springer, 2000.
18. Dietrich Kuske and Rémi Morin. Pomsets for local trace languages. *J. Autom. Lang. Comb.*, 7(2):187–224, 2002.

19. Vaughan R. Pratt. Modeling concurrency with geometry. In *POPL*, pages 311–322, New York City, 1991. ACM Press.
20. Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.
21. James W. Thatcher and Jesse B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81, 1968.
22. W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume III, pages 389–455. Springer, 1997.
23. Wolfgang Thomas. On logical definability of trace languages. In *Algebraic and Syntactic Methods in Computer Science (ASMICS)*, Report TUM-I9002, Technical University of Munich, pages 172–182, 1990.
24. Boris A. Trakhtenbrot. Finite automata and monadic second order logic. *Siberian Mathematical Journal*, 3:103–131, 1962. In Russian; English translation in Amer. Math. Soc. Transl. 59, 1966, 23–55.
25. Rob J. van Glabbeek. Bisimulations for higher dimensional automata. Email message, June 1991. `http://theory.stanford.edu/~rvg/hda`.
26. Norbert Wiener. A contribution to the theory of relative position. *Proceedings of the Cambridge Philosophical Society*, 17:441–449, 1914.
27. Wiesław Zielonka. Notes on finite asynchronous automata. *RAIRO – Informatique Théorique et Applications*, 21(2):99–135, 1987.

Appendix

This appendix is provided for the convenience of the referees. It should not be considered as part of the paper for publication. It contains formulas and proofs omitted from the paper due to space constraints.

A From MSO to HDAs

Let $\texttt{Gclosed}(X_1, \dots, X_k)$ be the following:

$$\bigwedge_{i,j \le k} \forall x, y.\, x \in X_i \wedge (glue_{i,j}(x, y) \vee glue_{j,i}(y, x)) \implies y \in X_j\,.$$

This formula is satisfied by a transitively closed interpretation of $X_1, \dots, X_k$ under $\bigvee_{i,j \le k} glue_{i,j}$ where if $x \in X_i$ and $glue_{i,j}(x, y)$ or $glue_{j,i}(y, x)$ hold then $y \in X_j$. Hence:

$$(x, i) \sim (y, j) := \forall X_1, \dots, X_k.\, (x \in X_i \wedge \texttt{Gclosed}(X_1, \dots, X_k)) \implies y \in X_j\,.$$

The formula above is thus satisfied by all $P_1 \dots P_n \in \Omega^+_{\le k}$ for which there exist $l_1, l_2 \le n$ with $l_1 < l_2$ and $e \in \bigcap_{l_1 \le \ell \le l_2} P_\ell$ such that $evt(l_1, i) = evt(l_2, j) = e$. In particular e must be an identity in all of $P_{l_1+1}, \dots, P_{l_2-1}$ when $l_1 < l_2 + 1$. Note that such words are not always coherent. The non-coherent words are, however, excluded by $\texttt{Coh}_k$.

Example 1. Figure 4 displays an ipomset P and the coherent word $w_1 = P_1 \cdots P_7$ such that $P_1 * \cdots * P_7 = P$. Let $e_1, \dots, e_4$ be the events of P labeled respectively by the left a, the right a, c, and d and let $p_1, \dots, p_7$ the positions on w_1 from left to right. Let $\nu(X_1) = \{p_5\}$ and $\nu(X_2) = \{p_2\}$. Observe that $w_1 \not\models_\nu \texttt{Gclosed}(X_1, X_2)$ since for example $glue_{2,2}(p_1, p_2)$ but $p_1 \notin \nu(X_2)$. The smallest good valuation including the previous one is $\nu(X_1) = \{p_5, p_6\}$ and $\nu(X_2) = \{p_1, p_2\}$.

B From HDAs to MSO

Proof (of Lem. 14).
For the direction from the left to the right, since P is accepted by X it admits a sparse accepting path $\alpha_1 * \cdots * \alpha_n$ such that $\mathsf{ev}(\alpha_i) = P_i$. Let us define $\rho_\nearrow(e) = \alpha_{\mathsf{St}(e)}$ for all $xe \in P \setminus S_P$ and $\rho_\searrow(e) = \alpha_{\mathsf{Te}(e)}$ for all $e \in P \setminus T_P$. Conditions 1-4 are satisfied by definition, and conditions 7-10 follow from the fact that α is an accepting path. Finally, $\mathsf{ev}(\alpha_i) = P_i$ implies conditions 5 and 6.

Conversely, assume that there exist $\rho_\nearrow$ and $\rho_\searrow$ satisfying conditions 1-10. From conditions 1-4, we can define a path $\alpha = \alpha_1 * \cdots * \alpha_n$ such that $\rho_\nearrow(e) = \alpha_{\mathsf{St}(e)}$ and $\rho_\searrow(e) = \alpha_{\mathsf{Te}(e)}$. By conditions 7-10, this path is accepting. It remains now to prove that $\mathsf{ev}(\alpha_i) = P_i$. Assume that $i = \mathsf{St}(e)$ for some $e \in P \setminus S_P$. Then $\alpha_i = p_i \nearrow^A q_i$ is chosen by condition 5 such that A is a conclist isomorphic to the conclist of all events started at position i and $\mathsf{ev}(q_i)$ is isomorphic to the conclist of all e' such that $\mathsf{St}(e') \le \mathsf{St}(e) = i < \mathsf{Te}(e')$ that is the conclist of all events that are started at position i, started before i, or never started (sources), and which are not terminated yet. Thus $\mathsf{ev}(\alpha_i) = {}_A{\uparrow}\mathsf{ev}(q_i)$ which is exactly P_i. The arguments are similar when $i = \mathsf{Te}(e)$ for some $e \in P \setminus T_P$. □

Proof (of Lem. 15). We first define $\mathsf{Te}(x) < \mathsf{St}(y)$ as the formula $x < y$, together with $\mathsf{St}(x) < \mathsf{Te}(y) \equiv \neg(\mathsf{Te}(y) < \mathsf{St}(x))$. Because starters and terminators alternate in the sparse step decomposition, we can then let

$$
\begin{aligned}
\mathsf{St}(x) < \mathsf{St}(y) &:= \exists z.\, \mathsf{St}(x) < \mathsf{Te}(z) \wedge \mathsf{Te}(z) < \mathsf{St}(y)\\
\mathsf{St}(x) = \mathsf{St}(y) &:= \neg(\mathsf{St}(x) < \mathsf{St}(y)) \wedge \neg(\mathsf{St}(y) < \mathsf{St}(x)) \wedge \neg\mathsf{s}(x) \wedge \neg\mathsf{s}(y)\\
\mathsf{Te}(x) < \mathsf{Te}(y) &:= \exists z.\, \mathsf{Te}(x) < \mathsf{St}(z) \wedge \mathsf{St}(z) < \mathsf{Te}(y)\\
\mathsf{Te}(x) = \mathsf{Te}(y) &:= \neg(\mathsf{Te}(x) < \mathsf{Te}(y)) \wedge \neg(\mathsf{Te}(y) < \mathsf{Te}(x)) \wedge \neg\mathsf{t}(x) \wedge \neg\mathsf{t}(y)\\
\min(\mathsf{St}(x)) &:= \neg\mathsf{s}(x) \wedge \neg\exists y.\, \mathsf{Te}(y) < \mathsf{St}(x)\\
\min(\mathsf{Te}(x)) &:= \neg\mathsf{t}(x) \wedge \neg\exists y.\, \mathsf{St}(y) < \mathsf{Te}(x)\\
\max(\mathsf{St}(x)) &:= \neg\mathsf{s}(x) \wedge \neg\exists y.\, \mathsf{Te}(y) > \mathsf{St}(x)\\
\max(\mathsf{Te}(x)) &:= \neg\mathsf{t}(x) \wedge \neg\exists y.\, \mathsf{St}(y) > \mathsf{Te}(x)\,.
\end{aligned}
$$

□

We can also define

$$
\begin{aligned}
\mathsf{St}(y) = \mathsf{Te}(x) + 1 := {}& \mathsf{Te}(x) < \mathsf{St}(y) \wedge\\
& \neg\exists z. \mathsf{Te}(x) < \mathsf{St}(z) \wedge \mathsf{St}(z) < \mathsf{St}(y)\\
\mathsf{Te}(y) = \mathsf{St}(x) + 1 := {}& \neg\mathsf{s}(x) \wedge \neg\mathsf{t}(y) \wedge \mathsf{St}(x) < \mathsf{Te}(y) \wedge\\
& \neg\exists z. \mathsf{St}(x) < \mathsf{Te}(z) \wedge \mathsf{Te}(z) < \mathsf{Te}(y)\,.
\end{aligned}
$$

Example 2. Continuing Ex. 13, observe that $P \models \mathsf{St}(e) = \mathsf{St}(e)$ for $e \in \{e_1, e_3, e_4\}$. This is not the case when $e = e_2$ since e_2 is a source neither when x and y are interpreted differently since there is no starter in w_2 starting two different events. We have also $P \models \mathsf{Te}(e) = \mathsf{Te}(e)$ for all $e \in P$ and $P \models \mathsf{Te}(e_2) = \mathsf{Te}(e_4)$. Let $\nu(x) = e_1$ and $\nu(y) = e_2$. Then $P \models_\nu \mathsf{St}(x) < \mathsf{Te}(y)$ but $P \not\models_\nu \mathsf{Te}(x) = \mathsf{St}(y) + 1$ since e_1, e_3 are terminating before e_2. Nevertheless $P \models \mathsf{Te}(e) = \mathsf{St}(e_2) + 1$ for $e \in \{e_2, e_4\}$ and $P \models \mathsf{St}(e_4) = \mathsf{Te}(e_3) + 1$. We have also $P \models \mathsf{St}(e_1) < \mathsf{Te}(e)$ for $e \in \{e_2, e_3, e_4\}$. Finally we have $P \models \min(\mathsf{St}(e_1))$ and $P \models \max(\mathsf{Te}(e))$ for $e \in \{e_2, e_4\}$.

Proof (of Prop. 16). Let $\mathsf{ups}(X) = \{(u_1, \nearrow^{A_1}, v_1), \dots, (u_m, \nearrow^{A_m}, v_m)\}$ and $\mathsf{downs}(X) = \{(p_1, \searrow^{B_1}, q_1), \dots, (p_n, \searrow^{B_n}, q_n)\}$. Then

$$
\begin{aligned}
\varphi := (\exists x. \neg\mathsf{s}(x) \vee \neg\mathsf{t}(x)) &\implies \exists X_1 \dots \exists X_m.\, \exists Y_1 \dots \exists Y_n.\, \varphi_0 \wedge \varphi_1 \wedge \cdots \wedge \varphi_{10}\\
\wedge\, (\forall y.\ \mathsf{s}(y) \wedge \mathsf{t}(y)) &\implies \bigvee_{\substack{p \in \bot_X \cap \top_X\\ \mathsf{ev}(p) \neq \emptyset}} \exists y_1, \dots, y_{|\mathsf{ev}(p)|}. \mathsf{ev}(p)(y_1, \dots, y_{|\mathsf{ev}(p)|}).
\end{aligned}
$$

where the second line of φ is satisfied by all the non-empty identities accepted by X and

- φ_0 checks that the X_i's and Y_i's define labellings $\rho_{\nearrow}$ and $\rho_{\searrow}$, that is, each event belongs to at most one X_i (is associated at most one upstep) and one Y_i (is associated at most one downstep), and to no X_i iff it is a source / no Y_i iff it is a target:

$$\begin{aligned}\varphi_0 = \forall x. & \bigwedge_{1\le i<j\le m} \neg(x \in X_i \land x \in X_j)\\ \land & \bigwedge_{1\le i<j\le n} \neg(x \in Y_i \land x \in Y_j)\\ \land & \neg \mathsf{s}(x) \iff \bigvee_{1\le i\le m} x \in X_i\\ \land & \neg \mathsf{t}(x) \iff \bigvee_{1\le i\le n} x \in Y_i\,.\end{aligned}$$

- φ_1 checks condition 1 from Lemma 14:

$$\varphi_1 = \forall x. \forall y. (\mathsf{St}(x) = \mathsf{St}(y)) \implies \bigwedge_{1\le i\le m} x \in X_i \iff y \in X_i\,.$$

- φ_2 similarly checks condition 2 from Lemma 14.
- φ_3 checks condition 3 from Lemma 14:

$$\varphi_3 = \forall x, y. \mathsf{St}(y) = \mathsf{Te}(x) + 1 \implies \bigvee_{u_i = q_j} y \in X_i \land x \in Y_j\,.$$

- φ_4 similarly checks condition 4 from Lemma 14.
- φ_5 checks condition 5 from Lemma 14:

$$\begin{aligned}\varphi_5 = \bigwedge_{1\le i\le m} & \forall x. x \in X_i \implies \exists x_1, \dots, x_{|A_i|}, y_1, \dots, y_{|\mathsf{ev}(v_i)|}.\\ & \Big(\forall y. \mathsf{St}(y) = \mathsf{St}(x) \iff \bigvee_{1\le i\le |A_i|} y = x_i\Big) \land A_i(x_1, \dots, x_{|A_i|})\\ \land & \Big(\forall y. \mathsf{St}(y) \le \mathsf{St}(x) < \mathsf{Te}(y) \iff \bigvee_{1\le i\le |\mathsf{ev}(v_i)|} y = y_i\Big)\\ \land & \,\mathsf{ev}(v_i)(y_1, \dots, y_{|\mathsf{ev}(v_i)|})\end{aligned}$$

where for a conclist $A = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix}$,

$$A(x_1, \dots, x_k) = \bigwedge_{1\le i<k} x_i \dashrightarrow x_{i+1} \land \bigwedge_{1\le i\le k} a_i(x_i)\,.$$

- φ_6 similarly checks condition 6 from Lemma 14.

– φ_7 checks condition 7 from Lemma 14:

$$\varphi_7 = \forall x.\ \min(\mathsf{St}(x)) \implies \bigvee_{u_i \in \perp_X} x \in X_i\,.$$

– φ_8, φ_9 and φ_{10} similarly check conditions 8, 9 and 10 of Lemma 14. We have $L(\varphi) = L(\mathcal{H}) \setminus \{\mathsf{id}_\emptyset\}$. If $\mathsf{id}_\emptyset \in L(\mathcal{H})$ then $L(\mathcal{H}) = L(\varphi \vee \neg\exists x.\ \mathtt{true})$. $\square$