RWTH Aachen

Department of Computer Science
Technical Report

Message-Passing Automata are expressively equivalent to EMSO logic

Benedikt Bollig and Martin Leucker

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2004-02

RWTH Aachen · Department of Computer Science · February 2004

Message-Passing Automata are expressively equivalent to EMSO logic

Benedikt Bollig[1*] and Martin Leucker[2**]

[1] Lehrstuhl für Informatik II, RWTH Aachen, Germany
bollig@informatik.rwth-aachen.de
[2] IT department, Uppsala University, Sweden
leucker@it.uu.se

Abstract. We study the expressiveness of finite message-passing automata with a priori unbounded channels and show them to capture exactly the class of MSC languages that are definable in existential monadic second-order logic interpreted over MSCs. Moreover, we prove the monadic quantifier-alternation hierarchy over MSCs to be infinite and conclude that the class of MSC languages accepted by message-passing automata is not closed under complement. Furthermore, we show that satisfiability for (existential) monadic seconder-order logic over MSCs is undecidable.

1 Introduction

A common design practice when developing communicating systems is to start with drawing scenarios showing the intended interaction of the system to be. The standardized notion of *message sequence charts* (MSCs, [7]) is widely used in industry to formalize such typical behaviors.

An MSC depicts a single partially-ordered execution sequence of a system. It defines a set of processes interacting with one another by communication actions. In the visual representation of an MSC, processes are drawn as vertical lines that are interpreted as time axes. A labeled arrow from one line to a second corresponds to the communication events of sending and receiving a message. Collections of MSCs are used to capture the scenarios that a designer might want the system to follow or to avoid. Several specification formalisms have been considered, such as *high-level MSCs* or *MSC graphs* [2, 13].

The next step in the design process usually is to derive an implementation of the system to develop [5], preferably automatically. In other words, we are interested in generating a distributed automaton *realizing* the behavior given in form of scenarios. This problem asks for the study of automata models that are suited for accepting the system behavior described by MSC specifications.

A common model that reflects the partially-ordered execution behavior of MSCs in a natural manner are *message-passing automata*, MPAs for short. They

* Part of this work was done while the author was on leave at the School of Computer Science, University of Birmingham, United Kingdom, and supported by the German Academic Exchange Service (DAAD).
** Supported by the European Research Training Network "Games".

consist of several components that communicate using channels. Several variants of MPAs have been studied in the literature: automata with a single or multiple initial states, with finitely or infinitely many states, bounded or unbounded channels, and systems with a global or local acceptance condition.

We focus on MPAs with a priori unbounded channels where each component employs a finite state space. Our model subsumes the one studied in [5] where a local acceptance condition is used. It coincides with the one used in [6, 9], although these papers characterize the fragment of channel-bounded automata. It extends the setting of [1, 11] in so far as we provide synchronization messages and a global acceptance condition to have the possibility to coordinate rather autonomous processes. Thus, our version covers most existing models of communicating automata for MSCs.

A fruitful way to study properties of automata is to establish logical characterizations. For example, finite word automata are known to be expressively equivalent to monadic second-order (MSO) logic over words. More precisely, the set of words satisfying some MSO formula can be defined by a finite automaton and vice versa. Since then the study of automata models for generalized structures such as graphs or, more specifically, labeled partial orders and their relation to MSO logic has been a research area of great interest aiming at a deeper understanding of their logical and algorithmic properties (see [15] for an overview).

In this paper, we show that MPAs accept exactly those MSC languages that are definable within the existential fragment of MSO (over MSCs), abbreviated by EMSO. We recall that emptiness for MPAs is undecidable and conclude that so is satisfiability for EMSO and universality for MSO logic.

Furthermore, we show that MSO is strictly more expressive than EMSO. More specifically, the monadic quantifier-alternation hierarchy turns out to be infinite. Thus, MPAs do *not* necessarily accept a set of MSCs defined by an MSO formula. Furthermore, we use this result to conclude that the class of MSC languages that corresponds to MPAs is not closed under complementation, answering the question posed in [9].

MPAs with a priori unbounded channels have been rather used as a model to implement a given (high-level) MSC specification [5]. Previous results lack an algebraic or logical characterization of the corresponding class of languages. They deal with MPAs and sets of MSCs that make use only of a bounded part of the actually unbounded channel [6, 9]. More specifically, when restricting to sets of so-called *bounded* MSCs, MSO captures exactly the class of those MSC languages that correspond to some bounded MPA. Together with our results, it follows that EMSO and MSO coincide on bounded MSC languages.

Organization of the Paper The next section introduces some basic notions and recalls the definition of message sequence charts, monadic second-order logic, and message-passing automata. Section 3 deals with the expressive equivalence of message-passing automata and existential monadic second-order logic, while

Section 4 studies the gap between monadic second-order formulas and their existential fragment.

2 Preliminaries

Let Σ be an alphabet. A (finite) *Σ-labeled poset* is a structure $(E, \leq, \lambda)$ such that $(E, \leq)$ is a (finite, respectively) partially-ordered set and λ is a mapping $E \to \Sigma$, called *labeling function*. Henceforth, we will identify a labeled poset with its isomorphism class. Given a Σ-labeled poset $(E, \leq, \lambda)$ and $e, e' \in E$, we say that e' *covers* e, written $e \lessdot e'$, if $e < e'$ and, for all $e'' \in E$, $e < e'' \leq e'$ implies $e'' = e'$. Furthermore, for $e \in E$, we define $\downarrow e := \{e' \in E \mid e' \leq e\}$ to be the *downwards-closure* of e. A *linearization* of M is a structure $(E, \leq', \lambda)$ such that $\leq'$ extends $\leq$ to a linear order. Depending on the context, $(E, \leq', \lambda)$ can be regarded as a word over Σ.

2.1 Message Sequence Charts

Forthcoming definitions are all made wrt. a fixed finite set *Proc* of at least two *processes*. (Note that, however, in one proof, we assume the existence of at least three processes.) We denote by *Ch* the set $\{(p, q) \mid p, q \in \mathit{Proc},\ p \neq q\}$ of reliable FIFO *channels*. Thus, a message exchange is allowed between distinct processes only. Let $\mathit{Act}^!$ denote the set $\{p!q \mid (p, q) \in \mathit{Ch}\}$ of *send actions* while $\mathit{Act}^?$ denotes the set $\{q?p \mid (p, q) \in \mathit{Ch}\}$ of *receive actions*. Hereby, $p!q$ and $q?p$ are to be read as *p sends a message to q* and *q receives a message from p*, respectively. They are related in the sense that they will label communicating events of an MSC, which are joint by a message arrow in its graphical representation. Accordingly, let $\mathit{Com} := \{(p!q, q?p) \mid (p, q) \in \mathit{Ch}\}$. Observe that an action $p\theta q$ ($\theta \in \{!, ?\}$) is performed by process p, which is indicated by $P(p\theta q) = p$. We let *Act* stand for the union of $\mathit{Act}^!$ and $\mathit{Act}^?$ and, for $p \in \mathit{Proc}$, set Act_p to be the set $\{\sigma \in \mathit{Act} \mid P(\sigma) = p\}$.

Let $(E, \leq, \lambda)$ be a finite *Act*-labeled poset. For $p \in \mathit{Proc}$, we define E_p to be $\lambda^{-1}(\mathit{Act}_p)$, i.e., the set of elements (also called *events* in the following) that are labeled with an action performed by process p. Furthermore, let $\leq_p$ ($<_p$) be (the largest irreflexive subset of) $\leq \cap (E_p \times E_p)$ and set $\lessdot_p$ to be $<_p \cap \lessdot$. For $e, e' \in E$, we write $e <_c e'$ if both $(\lambda(e), \lambda(e')) \in \mathit{Com}$ and $|\downarrow e \cap \lambda^{-1}(\lambda(e))| = |\downarrow e' \cap \lambda^{-1}(\lambda(e'))|$. For $e \in E$, $P(e)$ will serve as a shorthand for $P(\lambda(e))$.

Definition 1 (Message Sequence Chart). *A* message sequence chart *(MSC) is a finite Act-labeled poset $(E, \leq, \lambda)$ such that*

- *$\leq_p$ is a linear order for each $p \in \mathit{Proc}$,*
- *$\leq = (\bigcup_{p \in \mathit{Proc}} \leq_p \cup <_c)^*$, and*
- *$|\lambda^{-1}(p!q)| = |\lambda^{-1}(q?p)|$ for each $(p, q) \in \mathit{Ch}$.*

The set of MSCs is denoted by $\mathbb{MSC}$. In general, a set of MSCs is called an *MSC language*.

In other words, events on one and the same process line are linearly ordered, and events on distinct process lines that are immediately concerned with each other (wrt. $\lessdot$) are labeled with actions related by *Com*.

Given an MSC $M = (E, \leq, \lambda)$ and a process $p \in \mathit{Proc}$, the *projection* of M on p, denoted by $M \restriction p$, is the total order $(E_p, \leq_p, \lambda_{|E_p})$ where $\lambda_{|E_p}$ is the restriction of λ to E_p. Obviously, $M \restriction p$ can be regarded as a word over Act_p. For example, considering Figure 1, $M \restriction 1$ is represented by (1!2)(1!2)(1!2)(1!3). Projection can be extended towards MSC languages L defining, for $p \in \mathit{Proc}$, $L \restriction p := \{M \restriction p \mid M \in L\}$. Note that an MSC is uniquely determined by one of its linearizations and even by the collection of its projections.

2.2 (Existential) Monadic Second-Order Logic

Given a supply $\mathrm{Var} = \{x, y, \ldots, x_1, x_2, \ldots\}$ of *individual variables* and a supply $\mathrm{VAR} = \{X, Y, \ldots, X_1, X_2, \ldots\}$ of *set variables*, the syntax of *monadic second-order* (MSO) formulas is defined by

$$\varphi ::= L_\sigma(x) \mid x \in X \mid x \leq y \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x\varphi \mid \forall x\varphi \mid \exists X\varphi \mid \forall X\varphi$$

where $\sigma \in \mathit{Act}$, $x, y \in \mathrm{Var}$, and $X \in \mathrm{VAR}$. Moreover, we allow the formulas $\varphi_1 \rightarrow \varphi_2$, $x <_c y$, and, for $p \in \mathit{Proc}$, $P(x) = p$ as abbreviations for $\neg\varphi_1 \vee \varphi_2$, $\bigvee_{(\sigma,\tau)\in \mathit{Com}}(L_\sigma(x) \wedge L_\tau(y)) \wedge x \leq y \wedge \neg\exists z(x < z \wedge z < y)$, and $\bigvee_{\sigma \in \mathit{Act}_p} L_\sigma(x)$, respectively, while $x < y$ is a shorthand for $x \leq y \wedge \neg(y \leq x)$ and $x \lessdot_{\mathrm{p}} y$ is a shorthand for $\bigvee_{p \in \mathit{Proc}}(P(x) = p \wedge P(y) = p) \wedge x < y \wedge \neg\exists z(x < z \wedge z < y)$.

Let $M = (E, \leq, \lambda)$ be an MSC. Given an interpretation function $\mathcal{I}$, which assigns to an individual variable x an event $\mathcal{I}(x) \in E$ and to a set variable X a set of events $\mathcal{I}(X) \subseteq E$, the satisfaction relation $M \models_{\mathcal{I}} \varphi$ for an MSO formula φ is defined canonically. In particular, $M \models_{\mathcal{I}} L_\sigma(x)$ if $\lambda(\mathcal{I}(x)) = \sigma$ and $M \models_{\mathcal{I}} x \leq y$ if $\mathcal{I}(x) \leq \mathcal{I}(y)$, i.e., the predicate symbol $\leq$ is interpreted as the corresponding partial-order relation $\leq$.

In the following, we usually consider MSO *sentences*, i.e., MSO formulas without free variables, and accordingly replace $\models_{\mathcal{I}}$ with $\models$. For an MSO sentence φ, the *MSC language* of φ, denoted by $L(\varphi)$, is the set of MSCs M with $M \models \varphi$.

Given an MSO formula φ, the notation $\varphi(x_1, \ldots, x_m, X_1, \ldots, X_n)$ shall indicate that at most the variables $x_1, \ldots, x_m, X_1, \ldots, X_n$ occur free in φ. An MSO formula is called an *existential* MSO (EMSO) formula if it is of the form $\exists X_1 \ldots \exists X_n \varphi(X_1, \ldots, X_n, \overline{Y})$ where $\overline{Y}$ is a block of second-order variables and $\varphi(X_1, \ldots, X_n, \overline{Y})$ is a first-order formula, i.e., it contains no set quantifiers. In general, an MSO formula of the form $\exists\overline{X_1}\forall\overline{X_2}\ldots\exists/\forall\overline{X_k}\varphi(\overline{X_1}, \ldots, \overline{X_k}, \overline{Y})$ with first-order kernel $\varphi(\overline{X_1}, \ldots, \overline{X_k}, \overline{Y})$ is called a Σ_k-formula (again, $\overline{X_i}$ and $\overline{Y}$ are blocks of second-order variables). The language of a Σ_k-sentence is called Σ_k-*definable*. Note that the sets of Σ_1- and EMSO formulas coincide. We will show in a subsequent section that the monadic quantifier-alternation hierarchy is infinite when formulas are interpreted over MSCs, resuming a result by Matz and Thomas, who proved infinity of the hierarchy for grids [10]. In other words, the more alternation depth second-order quantification allows, the more expressive formulas become. However, it will turn out that, to cover the feasible area of realizable MSC languages (in terms of message-passing automata), we can restrict to EMSO-definable MSC languages. An MSC language L is called *EMSO-/MSO-definable* if $L = L(\varphi)$ for some EMSO/MSO sentence φ, respectively. The class of EMSO-definable MSC languages is denoted by $\mathcal{EMSO}$, the class of MSO-definable MSC languages by $\mathcal{MSO}$.

Example 1. Let L be the set of MSCs whose linearizations are of the form $(1!2)^m(1!3)(3?1)(3!2)(2?3)(2?1)^m$, $m \in \mathbb{N}$. Thus, Figure 1 depicts an MSC contained in L. Indeed, L is EMSO-definable (it is even definable by a first-order sentence). Note that L can also be recognized by some simple *message-passing automaton* as illustrated in the following section. So let φ be the conjunction of the formulas φ_1, φ_2, and φ_3, which describe the behavior of processes 1, 2, and 3, respectively, and are given as follows:

$$
\begin{aligned}
\varphi_1 = \exists x(\quad & L_{1!3}(x) \\
& \wedge \forall y(P(y) = 1 \wedge y < x \rightarrow L_{1!2}(y)) \\
& \wedge \forall y(P(y) = 1 \wedge x \leq y \rightarrow y = x))
\end{aligned}
$$

$$
\begin{aligned}
\varphi_2 = \exists x(\quad & L_{2?3}(x) \\
& \wedge \forall y(P(y) = 2 \wedge x < y \rightarrow L_{2?1}(y)) \\
& \wedge \forall y(P(y) = 2 \wedge y \leq x \rightarrow y = x))
\end{aligned}
$$

$$
\begin{aligned}
\varphi_3 = \exists x \exists y(\quad & L_{3?1}(x) \\
& \wedge L_{3!2}(y) \\
& \wedge x \lessdot_{\mathrm{p}} y \\
& \wedge \forall z(P(z) = 3 \wedge z \leq x \rightarrow z = x) \\
& \wedge \forall z(P(z) = 3 \wedge y \leq z \rightarrow z = y))
\end{aligned}
$$

In fact, it holds $L(\varphi) = L$. Note that, as L is a set of total orders that can be considered as a nonregular word language over Act, L is not MSO-definable when MSO formulas are interpreted over arbitrary words. However, as we interpret φ over MSCs, only those words have to be considered that are linearizations of MSCs. Accordingly, φ is rather meant to define total orders *generated* by the regular expression $(1!2)^*(1!3)(3?1)(3!2)(2?3)(2?1)^*$ while only restricting to MSCs rules out posets that are not valid MSCs.

2.3 Message-Passing Automata

In this section, we study distributed automata, called *message-passing automata*, which, as we will see, generate MSC languages in a natural manner.

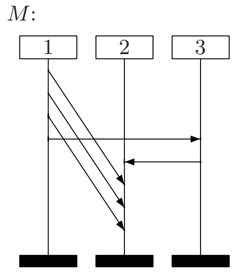

Fig. 1. An EMSO-definable MSC language

A message-passing automaton is a collection of finite-state machines that share one global initial state and several global final states. The machines are connected pairwise with a priori unbounded reliable FIFO buffers. The transitions of each component are labeled with send or receive actions. A send action $p!q$ puts a message at the end of the channel from p to q. A receive action can be taken provided the requested message is found in the channel. To extend the expressive power, message-passing automata can send certain synchronization messages. Let us be more precise:

Definition 2 (Message-Passing Automaton). *A* message-passing automaton *(MPA) is a structure* $\mathcal{A} = ((\mathcal{A}_p)_{p \in Proc}, \mathcal{D}, \overline{s}^{in}, F)$ *such that*

- $\mathcal{D}$ *is a nonempty finite set of* synchronization messages *(or* data*),*
- *for each* $p \in Proc$*,* $\mathcal{A}_p$ *is a pair* (S_p, Δ_p) *where*
  - S_p *is a nonempty finite set of* (p-)local states *and*
  - $\Delta_p \subseteq S_p \times Act_p \times \mathcal{D} \times S_p$ *is the set of* (p-)local transitions*,*
- $\overline{s}^{in} \in \prod_{p \in Proc} S_p$ *is the* global initial state*, and*
- $F \subseteq \prod_{p \in Proc} S_p$ *is the set of* global final states.

Given a *global state* $\overline{s} = (s_p)_{p \in Proc} \in \prod_{p \in Proc} S_p$ of $\mathcal{A}$, $\overline{s}[p]$ will in the following refer to s_p. An MPA with set of synchronization messages $\{\circ, \bullet\}$ is illustrated in Figure 2. Note that its MSC language cannot be recognized by some MPA with only one synchronization message (even if we allowed infinite local state spaces).

We now define the behavior of message-passing automata and, in doing so, adhere to the style of [9]. In particular, an automaton will run on MSCs rather than on linearizations of MSCs, allowing for its distributed behavior. Let $\mathcal{A} = ((\mathcal{A}_p)_{p \in Proc}, \mathcal{D}, \overline{s}^{in}, F)$, $\mathcal{A}_p = (S_p, \Delta_p)$, be an MPA and $M = (E, \leq, \lambda)$ be an MSC. For a function $r : E \to \bigcup_{p \in Proc} S_p$, we define, provided $E \neq \emptyset$, $r^- : E \to \bigcup_{p \in Proc} S_p$ to map an event $e \in E$ onto $\overline{s}^{in}[P(e)]$ if e is minimal wrt. $\leq_{P(e)}$ and, otherwise, onto $r(e')$ where $e' \in E_{P(e)}$ is the unique event with $e' \lessdot_{P(e)} e$. A *run* of $\mathcal{A}$ on M is a pair (r, m) of mappings $r : E \to \bigcup_{p \in Proc} S_p$ with $r(e) \in S_{P(e)}$ for each $e \in E$ and $m : <_c \to \mathcal{D}$ such that, for any $e, e' \in E$, $e <_c e'$ implies

- $(r^-(e), \lambda(e), m((e, e')), r(e)) \in \Delta_{P(e)}$ and
- $(r^-(e'), \lambda(e'), m((e, e')), r(e')) \in \Delta_{P(e')}$.

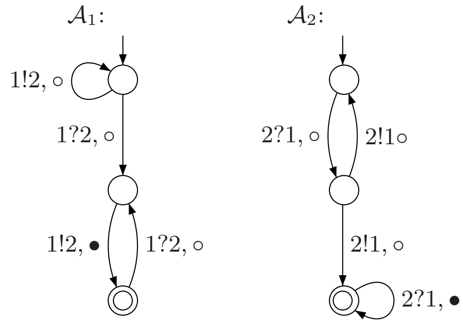

Fig. 2. A message-passing automaton

For $p \in \mathit{Proc}$, let f_p denote $\overline{s}^{in}[p]$ if E_p is empty. Otherwise, let f_p denote $r(e)$ where $e \in E_p$ is the maximal event wrt. $\leq_p$. We call (m, r) *accepting* if $(f_p)_{p \in \mathit{Proc}} \in F$.

For an MPA $\mathcal{A}$, we denote by $L(\mathcal{A}) := \{M \in \mathbb{MSC} \mid$ there is an accepting run of $\mathcal{A}$ on $M\}$ the *language* of $\mathcal{A}$. Let furthermore $\mathcal{L}(\text{MPA}) := \{L \subseteq \mathbb{MSC} \mid$ there is an MPA $\mathcal{A}$ such that $L(\mathcal{A}) = L\}$ denote the class of languages that are *realizable* as MPAs.

Remark 1. The emptiness problem for MPAs is undecidable.

Proof. Several decidability questions were studied for *communicating finite-state machines*, a slightly different variant of MPAs. Among them, (a problem related to) the emptiness problem for communicating finite-state machines turned out to be undecidable [3]. Their proof can be easily adapted towards MPAs.

3 The Expressiveness of Message-Passing Automata

We now turn towards one of our main results and first show that an MPA can be effectively transformed into an equivalent EMSO sentence.

Lemma 1. $\mathcal{L}(\text{MPA}) \subseteq \mathcal{EMSO}$

Proof. Let $N \geq 1$ and let $\mathcal{A} = ((\mathcal{A}_p)_{p \in \mathit{Proc}}, \{1, \ldots, N\}, \overline{s}^{in}, F)$, $\mathcal{A}_p = (S_p, \Delta_p)$, be an MPA. We assume $F \neq \emptyset$. Note that $\mathcal{A}_p$, once equipped with an initial state $s \in S_p$ and a set of final states $F_p \subseteq S_p$, can be considered as a finite word automaton generating a regular word language over $\mathit{Act}_p \times \{1, \ldots, N\}$ (which can be seen as a regular set of $(\mathit{Act}_p \times \{1, \ldots, N\})$-labeled total orders) in the obvious manner. In case that $N = 1$, we even understand $\mathcal{A}_p$ to recognize a word language over Act_p ignoring the respective message component in the transition relation Δ_p.

Our aim is to exhibit an EMSO sentence Ψ such that $L(\Psi) = L(\mathcal{A})$. It is well-known that the class of word languages that are EMSO-definable (wrt. EMSO formulas over total orders or words) coincides with the class of regular word languages. Clearly, $L(\mathcal{A}) = \bigcup_{\overline{s}\in F} L(((\mathcal{A}_p)_{p\in Proc}, \{1,\ldots,N\}, \overline{s}^{in}, \{\overline{s}\}))$. So let, for each global final state $\overline{s} \in F$ and each process $p \in Proc$,

$$\varphi(\overline{s}, p) = \exists X_1^{\overline{s},p} \ldots \exists X_{n_{\overline{s},p}}^{\overline{s},p} \psi^{\overline{s},p}(X_1^{\overline{s},p}, \ldots, X_{n_{\overline{s},p}}^{\overline{s},p})$$

be an EMSO sentence (over words) with first-order kernel $\psi^{\overline{s},p}$ such that $\varphi(\overline{s}, p)$ defines the language of the finite (word) automaton $\mathcal{A}_p$ with initial state $\overline{s}^{in}[p]$ and set of final states $\{\overline{s}[p]\}$. The EMSO formula Ψ now requires the existence of an assignment of synchronization messages to the events that, on the one hand, respects that communication events have to be equally labeled and, on the other hand, meets the restrictions imposed by the formulas $\varphi(\overline{s}, p)$ for at least one final state $\overline{s} \in F$. It is given by

$$\begin{aligned}\Psi = {} & \exists X_1 \ldots \exists X_N \; \exists \overline{X} \\ & \textstyle\bigvee_{\overline{s}\in F} \Big(\; \text{partition}(X_1, \ldots, X_N) \wedge \\ & \qquad\quad \text{consistent}(X_1, \ldots, X_N) \wedge \\ & \qquad\quad \textstyle\bigwedge_{p\in Proc} \text{process}_{\overline{s},p}(X_1, \ldots, X_N, X_1^{\overline{s},p}, \ldots, X_{n_{\overline{s},p}}^{\overline{s},p}) \Big)\end{aligned}$$

where $\overline{X}$ is the block of all the second-order variables $X_i^{\overline{s},p}$.

The formula partition$(X_1, \ldots, X_N)$ ensures that the variables $X_1, \ldots, X_N$ define a mapping from the set of events of an MSC to the set of synchronization messages $\{1, \ldots, N\}$:

$$\text{partition}(X_1, \ldots, X_N) = \left(\forall x \bigvee_{i\in\{1,\ldots,N\}} x \in X_i \right) \wedge \left(\forall x \bigwedge_{1\leq i<j\leq N} \neg(x \in X_i \wedge x \in X_j) \right)$$

Then consistent$(X_1, \ldots, X_N)$ guarantees the mapping is consistent, i.e., a send event and its corresponding receive event are equally labeled wrt. the alphabet of synchronization messages:

$$\text{consistent}(X_1, \ldots, X_N) = \forall x \forall y \left(x <_c y \rightarrow \bigvee_{i\in\{1,\ldots,N\}} (x \in X_i \wedge y \in X_i) \right)$$

For $\overline{s} \in F$ and $p \in Proc$, the formula $\text{process}_{\overline{s},p}(X_1, \ldots, X_N, X_1^{\overline{s},p}, \ldots, X_{n_{\overline{s},p}}^{\overline{s},p}) = \|\psi^{\overline{s},p}\|_p$ ensures that the total order that process p induces wrt. an MSC corresponds to what the word automaton $\mathcal{A}_p$ with initial state $\overline{s}^{in}[p]$ and set of final states $\{\overline{s}[p]\}$ recognizes. It is a kind of projection of the formula $\psi^{\overline{s},p}$ on p and inductively derived from $\psi^{\overline{s},p}$ as follows (recall that $\psi^{\overline{s},p}$ is the kernel of a first-order formula interpreted over words (or total orders) from $(Act_p \times \{1, \ldots, N\})^*$):

- $\|L_{(\sigma,n)}(x)\|_p = L_\sigma(x) \wedge x \in X_n$
- $\|x \in X\|_p = x \in X$

- $\|x \leq y\|_p = x \leq y$
- $\|\neg\varphi\|_p = \neg\|\varphi\|_p$
- $\|\varphi \vee \psi\|_p = \|\varphi\|_p \vee \|\psi\|_p$
- $\|\varphi \wedge \psi\|_p = \|\varphi\|_p \wedge \|\psi\|_p$
- $\|\exists x\varphi\|_p = \exists x(P(x) = p \wedge \|\varphi\|_p)$
- $\|\forall x\varphi\|_p = \forall x(P(x) = p \rightarrow \|\varphi\|_p)$

Note that, though Ψ allows the set variables $X_i^{\overline{s},p}$ to range over arbitrary subsets of events, in $\|\psi^{\overline{s},p}\|_p$, their interpretation is restricted to elements of process p. □

Example 1 provides an EMSO sentence (even a first-order sentence) that reflects the principle of the above construction and defines an MSC language recognized by some simple MPA with one synchronization message and local acceptance condition where process 1 performs a sequence $(1!2)^m(1!3)$ before it stays in a final state, process 2, similarly, reads sequences of the form $(2?3)(2?1)^m$, and the language of process 3 contains $(3?1)(3!2)$ only.

Theorem 1. *The following two problems are undecidable:*

(a) Satisfiability for EMSO sentences over $\mathbb{MSC}$
(b) Universality for MSO sentences over $\mathbb{MSC}$

Proof. Using Remark 1 and Lemma 1, we obtain Theorem 1 (a). Theorem 1 (b) follows from an easy reduction from the satisfiability problem. □

In fact, any EMSO-definable MSC language is realizable as an MPA and, vice versa, any MSC language recognized by some MPA has an appropriate EMSO counterpart.

Theorem 2. $\mathcal{L}(\mathrm{MPA}) = \mathcal{EMSO}$

From Theorem 2, it follows that, restricting to *bounded* [6] MSC languages, EMSO- and MSO-definability coincide, refining a result by Henriksen et al. [6]. We leave as an open problem whether EMSO- and MSO-definability still coincide if we restrict to *finitely-generated* languages [6]. Due to results by Morin [11], this is the case if any *connected* high-level MSC is realizable as an MPA. Genest et al. provide an algorithm that realizes a *locally-cooperative* high-level MSC in terms of an MPA with local acceptance condition [5]. However, EMSO- and MSO-definability do not coincide for the whole class of MSC languages, which will be shown in the following section.

The rest of this section is dedicated to the proof of Theorem 2.

Proof. It remains to show inclusion from right to left.

Graph acceptors [15], an automata-theoretic generalization of finite automata to labeled graphs and, in particular, labeled posets, are known to be equivalent to EMSO logic wrt. expressiveness. So, given an EMSO formula interpreted over

Act-labeled posets, we can assume the existence of an equivalent graph acceptor $\mathcal{GA}$, which, in turn, will be translated into an MPA $\mathcal{A}$ that captures the application of $\mathcal{GA}$ to MSCs, i.e., $L(\mathcal{A}) = L(\mathcal{GA}) \cap \mathbb{MSC}$.

Let R be a natural and Σ be an alphabet. For a Σ-labeled poset $(E, \leq, \lambda)$ and elements $e, e' \in E$, the *distance* $d(e', e)$ from e to e' is ∞ if $(e, e') \notin (\leq \cup \leq^{-1})^*$ and, otherwise, the minimal natural number k such that there is a sequence of elements $e_0, \ldots, e_k \in E$ with $e_0 = e$, $e_k = e'$, and $e_i \lessdot e_{i+1}$ or $e_{i+1} \lessdot e_i$ for each $i \in \{0, \ldots, k-1\}$. An *R-sphere* over Σ is a nonempty Σ-labeled poset $(E, \leq, \lambda, \gamma)$ together with a designated *sphere center* $\gamma \in E$ such that, for any $e \in E$, $d(e, \gamma) \leq R$. For a Σ-labeled poset $M = (E, \leq, \lambda)$ and $e \in E$, let $is_{M,R}(e)$ denote the R-sphere $(E', \leq', \lambda', e)$ of M *induced* by e, i.e., $E' = \{e' \in E \mid d(e', e) \leq R\}$ and $\leq' = (\lessdot \cap (E' \times E'))^*$.

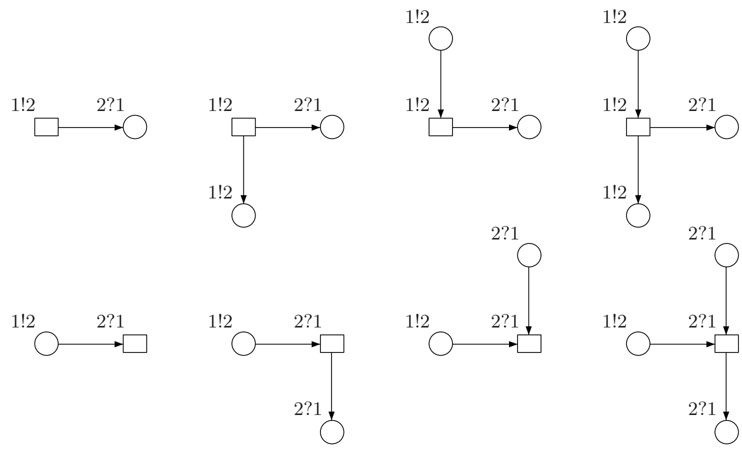

Fig. 3. A graph acceptor

A *graph acceptor* over Act is a structure $\mathcal{GA} = (Q, R, \mathfrak{S}, Occ)$ where Q is a nonempty finite set of *states*, $R \in \mathbb{N}$, $\mathfrak{S}$ is a finite set of R-spheres over $Act \times Q$ (where we identify isomorphic structures so that, more precisely, we deal with a set of isomorphism classes), and Occ is a boolean combination of *conditions* of the form "sphere $g \in \mathfrak{S}$ occurs at least $\geq n$ times" where $n \in \mathbb{N}$. Let $\max(Occ)$ denote the least threshold n such that Occ does not distinguish occurrence numbers $\geq n$. A run of $\mathcal{GA}$ on an Act-labeled poset $M = (E, \leq, \lambda)$ is a mapping $\rho : E \to \mathfrak{S}$ for which there is $\mu : E \to Q$—inducing an extended labeling function $(\lambda, \mu) : E \to Act \times Q$ of M—such that, for each $e \in E$, $is_{(E,\leq,(\lambda,\mu)),R}(e)$ is isomorphic to $\rho(e)$. We call ρ *accepting* if it satisfies the constraints imposed by Occ. The language of $\mathcal{GA}$, denoted by $L(\mathcal{GA})$, is the set of Act-labeled posets on which there is an accepting run of $\mathcal{GA}$. (Recall that we consider graph acceptors running on posets only.) For example, consider the graph acceptor $\mathcal{GA} = (Q, R, \mathfrak{S}, Occ)$ where Q is a singleton, $R = 1$, $\mathfrak{S}$ is given by the 1-spheres shown in Figure 3 (where the center of a sphere is depicted as a rectangle and the states are removed from the labelings), and Occ is true. We easily verify that $L(\mathcal{GA}) \cap \mathbb{MSC}$ is the set of MSCs in which each event is labeled either with 1!2 or with 2?1. Figure 4 provides an example run of $\mathcal{GA}$. Note that, however, $L(\mathcal{GA})$ can also be recognized by a graph acceptor equipped with 0-spheres.

We proceed by translating any graph acceptor $\mathcal{GA}$ over Act into an MPA $\mathcal{A}$ such that $L(\mathcal{A}) = L(\mathcal{GA}) \cap \mathbb{MSC}$, which proves the theorem. So let $\mathcal{GA} = (Q, R, \mathfrak{S}, Occ)$ be a graph acceptor over Act. Without loss of generality, we assume that, for each R-sphere $(E, \leq, \lambda, \gamma) \in \mathfrak{S}$, there is an MSC $M = (E', \leq', \lambda')$ (with extended labeling function $\lambda' : E' \to Act \times Q$) and $e' \in E'$ such that $(E, \leq, \lambda, \gamma) \sim is_{M,R}(e')$. Because, to become part of a run on an MSC M, an R-sphere has to be embedded into M. In this sense, the poset illustrated in Figure 5 (a) (again, states are omitted) is a sphere (it can be complemented by a 1!3-labeled event arranged in order between the two other events of process 1), while the partial order illustrated aside is not a sphere, as it will never be part of a run on some MSC. This is essential in the proof. Let $maxE := \max\{|E| \mid (E, \leq, \lambda, \gamma) \in \mathfrak{S}\}$ and let $\mathfrak{S}^+$ be the set of *extended R-spheres*, i.e., the set of structures $((E, \leq, \lambda, \gamma, e), i)$ where $(E, \leq, \lambda, \gamma) \in \mathfrak{S}$, $e \in E$, and $i \in \{1, \ldots, 4 \cdot maxE + 1\}$.

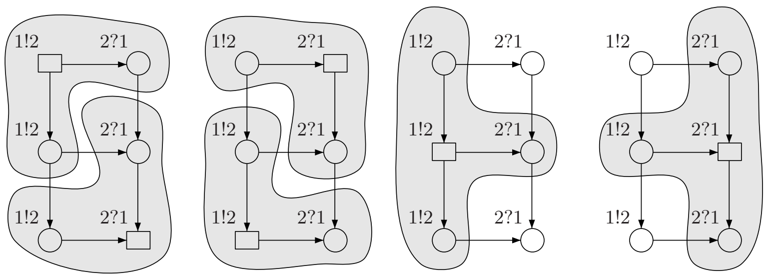

Fig. 4. The run of a graph acceptor

The idea in the following is that, roughly speaking, $\mathcal{A}$ guesses a tiling of the MSC to be read and then verifies that the tiling corresponds to an accepting run of $\mathcal{GA}$. Accordingly, a local state of $\mathcal{A}$ holds a set of *active* R-spheres, i.e., a set of spheres that play a role in its immediate environment of distance at most R. Each local state s (apart from the initial states, as we will see) carries exactly one extended R-sphere $((E, \leq, \lambda, \gamma, e), i)$ with $\gamma = e$, which means that a run of $\mathcal{GA}$ assigns $(E, \leq, \lambda, \gamma)$ to the event that corresponds to s. To establish isomorphism between $(E, \leq, \lambda, \gamma)$ and the R-sphere induced by s, s refers/obtains its obligations in form of an extended R-sphere $((E, \leq, \lambda, \gamma, e'), i)$ to/from its immediate neighbors, respectively. For example, provided e is labeled with a send action and there is $e' \in E$ with $e <_c e'$, the message to be sent in state s will contain $((E, \leq, \lambda, \gamma, e'), i)$, which, in turn, the receiving process understands as a requirement to be satisfied. A state may hold several instances of one and the same R-sphere, which then refer to distinct states/events as corresponding sphere center. Those instances will be distinguished by means of the natural i. The benefit of i will become clear before long.

For $p \in \mathit{Proc}$, we define $\mathfrak{S}_p := \{(E, \leq, \lambda, \gamma) \in \mathfrak{S} \mid P(\gamma) = p\}$ and $\mathfrak{S}_p^+ := \{((E, \leq, \lambda, \gamma, e), i) \in \mathfrak{S}^+ \mid P(e) = p\}$. Let us now turn towards the construction of $\mathcal{A} = ((\mathcal{A}_p)_{p \in \mathit{Proc}}, \mathcal{D}, \overline{s}^{in}, F)$, $\mathcal{A}_p = (S_p, \Delta_p)$, which is given as follows: For $p \in \mathit{Proc}$, a local state of $\mathcal{A}_p$ is a pair $(\mathcal{S}, \nu)$ where

- ν is a mapping $\mathfrak{S}_p \to \{0, \ldots, \max(\mathit{Occ})\}$ (let in the following ν_p^0 denote the function that maps each R-sphere $g \in \mathfrak{S}_p$ to 0) and
- $\mathcal{S}$ is either the empty set or it is a subset of $\mathfrak{S}_p^+$ such that
  - there is exactly one extended R-sphere $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}$ with $\gamma = e$ (whose component $(E, \leq, \lambda, \gamma)$ we identify by $\varsigma(\mathcal{S})$ from now on) and
  - for any two $((E, \leq, \lambda, \gamma, e), i), ((E', \leq', \lambda', \gamma', e'), i') \in \mathcal{S}$,
    (a) $\lambda(e) = \lambda'(e')$ (so that we can assign a unique label $\lambda(\mathcal{S}) \in \mathit{Act} \times Q$ to $\mathcal{S}$) and
    (b) if $(E, \leq, \lambda, \gamma) \sim (E', \leq', \lambda', \gamma')$ and $i = i'$, then $e = e'$.

The set $\mathcal{D}$ of synchronization messages is the cartesian product $2^{\mathfrak{S}^+} \times 2^{\mathfrak{S}^+}$. Roughly speaking, the first component of a message contains obligations the receiving state/event has to satisfy, while the second component imposes requirements that must *not* be satisfied by the receiving process to ensure isomorphism. We now turn towards the definition of Δ_p and define $((\mathcal{S}, \nu), \sigma, (\mathcal{P}, \mathcal{N}), (\mathcal{S}', \nu')) \in \Delta_p$ if the following hold:

1. $\lambda(\mathcal{S}') = (\sigma, q)$ for some $q \in Q$.
2. For any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}$ and $e' \in E_p$, if $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{S}'$, then $e \lessdot_p e'$.
3. For any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}'$, if $\mathcal{S} \neq \emptyset$ and e is minimal in $(E_p, <_p)$, then $d(e, \gamma) = R$.
4. For any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}$, if e is maximal in $(E_p, <_p)$, then $d(e, \gamma) = R$.
5. For any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}'$, if e is not minimal in $(E_p, <_p)$, then we have $((E, \leq, \lambda, \gamma, e^-), i) \in \mathcal{S}$ where $e^- \in E_p$ is the unique event with $e^- \lessdot_p e$.
6. For any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}$, if e is not maximal in $(E_p, <_p)$, then we have $((E, \leq, \lambda, \gamma, e^+), i) \in \mathcal{S}'$ where $e^+ \in E_p$ is the unique event such that $e \lessdot_p e^+$.
7. (i) In case that $\sigma = p!q$ for some $q \in \mathit{Proc}$:
   (a) for any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}'$ and any $e' \in E$, if $e <_c e'$, then we have $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{P}$,
   (b) for any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}'$ and any $e' \in E$, if $e \not<_c e'$, then we have $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{N}$, and
   (c) for any $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{P}$, there is $e' \in E$ such that $e' <_c e$ and $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{S}'$.

   (ii) In case that $\sigma = p?q$ for some $q \in \mathit{Proc}$:
   (a) $\mathcal{P} \subseteq \mathcal{S}'$,
   (b) $\mathcal{N} \cap \mathcal{S}' = \emptyset$, and
   (c) for any $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{S}'$, if there is $e \in E$ with $e <_c e'$, then $((E, \leq, \lambda, \gamma, e'), i) \in \mathcal{P}$.

8. $\nu' = \nu[\varsigma(\mathcal{S}')/\min\{\nu(\varsigma(\mathcal{S}'))+1, \max(Occ)\}]$ (i.e., ν' maps $\varsigma(\mathcal{S}')$ to the minimum of $\nu(\varsigma(\mathcal{S}'))+1$ and $\max(Occ)$ and, otherwise, coincides with ν).

Furthermore, $\overline{s}^{in} = ((\emptyset, \nu_p^0))_{p\in Proc}$ and, for $(\mathcal{S}_p, \nu_p) \in S_p$, $((\mathcal{S}_p, \nu_p))_{p\in Proc} \in F$ if the union of mappings ν_p satisfies the requirements imposed by Occ and, for all $p \in Proc$ and $((E, \leq, \lambda, \gamma, e), i) \in \mathcal{S}_p$, e is maximal in $(E_p, <_p)$.

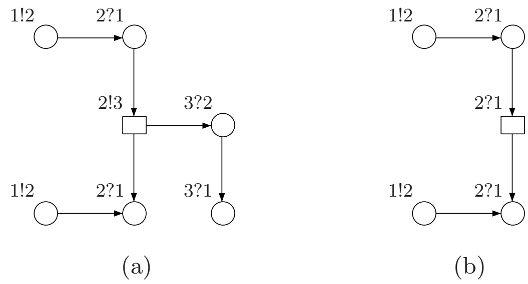

Fig. 5. The sphere(s) of a graph acceptor

Claim 1. $L(\mathcal{A}) = L(\mathcal{GA}) \cap \mathbb{MSC}$

Proof of Claim 1. "$\supseteq$": Let $\rho : \widetilde{E} \to \mathfrak{S}$ be an accepting run of $\mathcal{GA}$ on an MSC $M = (\widetilde{E}, \widetilde{\leq}, \widetilde{\lambda})$.

Claim 2. There is a mapping $i_{M,\rho} : \widetilde{E} \to \{1, \ldots, 4 \cdot maxE + 1\}$ such that, for any $e, e', e_0, e_0' \in \widetilde{E}$ with $\rho(e) = \rho(e')$, $e \neq e'$, $d(e_0, e) \leq R$, and $d(e_0', e') \leq R$, if $e_0 \mathrel{\widetilde{\lessdot}} e_0'$ or $e_0' \mathrel{\widetilde{\lessdot}} e_0$ or $e_0 = e_0'$, then $i_{M,\rho}(e) \neq i_{M,\rho}(e')$.

Proof of Claim 2. We can reduce the existence of $i_{M,\rho}$ to the existence of a graph coloring. Recall some basic definitions: A *graph* G is a structure $(V, Arcs)$ where V is a finite set of *vertices* and $Arcs \subseteq V \times V$ is a set of *arcs*. For a natural $n \geq 1$, a graph $G = (V, Arcs)$ is called *n-colorable* if there is a mapping $\chi : V \to \{1, \ldots, n\}$ such that $(u, v) \in Arcs$ implies $\chi(u) \neq \chi(v)$ for any two nodes $u, v \in V$ (we then say that G is *n-colored* by χ). Furthermore, for $d \in \mathbb{N}$, G is said to be of *degree* d if $d = \max\{|Arcs(u)| \mid u \in V\}$ where, for $u \in V$, $Arcs(u) = \{v \in V \mid (u, v) \in Arcs$ or $(v, u) \in Arcs\}$. It is easy to show that, for any $d \in \mathbb{N}$ and any graph G of degree d, G is $(d+1)$-colorable. The mapping $i_{M,\rho}$ can now be constructed as follows: Let G be the graph $(\widetilde{E}, Arcs)$ where, for any $e, e' \in \widetilde{E}$, $(e, e') \in Arcs$ iff $e \neq e'$, $\rho(e) = \rho(e')$, and there is $e_0, e_0' \in \widetilde{E}$ with $d(e_0, e) \leq R$, $d(e_0', e') \leq R$, and ($e_0 \mathrel{\widetilde{\lessdot}} e_0'$ or $e_0' \mathrel{\widetilde{\lessdot}} e_0$ or $e_0 = e_0'$). As G is of degree not greater than $4 \cdot maxE$ (for each $e \in \widetilde{E}$, there are at most four distinct events $e' \in \widetilde{E}$ such that $e \mathrel{\widetilde{\lessdot}} e'$, $e' \mathrel{\widetilde{\lessdot}} e$, or $e = e'$), it can be $(4 \cdot maxE + 1)$-colored by some mapping $\chi : \widetilde{E} \to \{1, \ldots, 4 \cdot maxE + 1\}$. Now set $i_{M,\rho}$ to be χ. This concludes the proof of Claim 2. Now let $i_{M,\rho}$ be the mapping from the above construction. For $g \in \mathfrak{S}$ and $e \in \widetilde{E}$, let furthermore $le_M(g,e) = |\{e' \in \widetilde{E}_{P(e)} \mid e' \mathrel{\widetilde{\leq}}_{P(e)} e,\ g \sim \rho(e')\}|$. For ease of notation, we assume that, for each $e \in \widetilde{E}$, $\rho(e)$ is exactly $is_{M,R}(e)$ where the labeling function is replaced with the corresponding extended mapping induced by ρ. When we define mappings with domain $\widetilde{E}$, we will suppose that $\widetilde{E}$ is nonempty, as those mappings are otherwise not significant. Let the mapping $r : \widetilde{E} \to \bigcup_{p \in Proc} S_p$ be given as follows: for $e \in \widetilde{E}$, we define $r(e) = (\mathcal{S}, \nu)$ where

1. $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ iff there is an event $e' \in \widetilde{E}$ such that $d(e', e) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e)$, and $i = i_{M,\rho}(e')$, and
2. for $g \in \mathfrak{S}$, $\nu(g) = \min\{le_M(g, e), \max(Occ)\}$.

For $e \in \widetilde{E}$, we first verify that, in fact, $r(e) = (\mathcal{S}, \nu)$ is a valid state of $\mathcal{A}$. So suppose there are extended R-spheres $((E, \leq, \lambda, \gamma, e_0), i), ((E', \leq', \lambda', \gamma', e_0'), i') \in \mathcal{S}$. Of course, it holds $\lambda(e_0) = \lambda(e_0')$. Assume now that both $\gamma = e_0$ and $\gamma' = e_0'$. But then the requirements $(E, \leq, \lambda, \gamma, \gamma) \sim (\rho(e), e)$ and $(E', \leq', \lambda', \gamma', \gamma') \sim (\rho(e), e)$ imply $(E, \leq, \lambda, \gamma, \gamma) \sim (E', \leq', \lambda', \gamma', \gamma')$. In particular, it holds $\varsigma(\mathcal{S}) = (E, \leq, \lambda, \gamma) \sim \rho(e)$. Furthermore, $i = i' = i_{M,\rho}(e)$. Now assume that $(E, \leq, \lambda, \gamma) \sim (E', \leq', \lambda', \gamma')$ and $i = i'$. There are events $e_1, e_2 \in \widetilde{E}$ such that $d(e_1, e) \leq R$, $d(e_2, e) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e_1), e)$, $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e_2), e)$, and $i = i_{M,\rho}(e_1) = i_{M,\rho}(e_2)$. Clearly, we have $\rho(e_1) \sim \rho(e_2)$. Furthermore, $e_1 = e_2$ and, consequently $e_0 = e_0'$. Because $e_1 \neq e_2$, according to Claim 2, implies $i_{M,\rho}(e_1) \neq i_{M,\rho}(e_2)$, which contradicts the premise.

Let $m : \mathrel{\widetilde{\lessdot}}_c \to \mathcal{D}$ map a pair $(e_s, e_r) \in \mathrel{\widetilde{\lessdot}}_c$ onto $(\mathcal{P}, \mathcal{N})$ where (set $(\mathcal{S}, \nu)$ to be $r(e_s)$) $\mathcal{P} = \{((E, \leq, \lambda, \gamma, e_0'), i) \in \mathfrak{S}^+ \mid$ there is $e_0 \in E$ with $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 <_c e_0'\}$ and $\mathcal{N} = \{((E, \leq, \lambda, \gamma, e_0'), i) \in \mathfrak{S}^+ \mid$ there is $e_0 \in E$ such that $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 \not<_c e_0'\}$. In the following, we verify that (r, m) is a run of $\mathcal{A}$ on M. For any distinct processes $p, q \in Proc$, $e \in \widetilde{E}_p$, and $e_r \in \widetilde{E}_q$ with $e \mathrel{\widetilde{\lessdot}}_c e_r$, we check that $(r^-(e), \lambda(e), m((e, e_r)), r(e)) \in \Delta_p$. So set $(\mathcal{S}, \nu)$ to be $r^-(e)$ and $(\mathcal{S}', \nu')$ to be $r(e)$.

1. Of course, $\lambda(\mathcal{S}') = (\widetilde{\lambda}(e), q)$ for some $q \in Q$.
2. Let $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{S}'$ for some $e_0' \in E_p$ and let $e^- \in \widetilde{E}_p$ such that $e^- \mathrel{\widetilde{\lessdot}}_p e$ (as $\mathcal{S} \neq \emptyset$, such an e^- must exist). There is $e^{-\prime}, e' \in \widetilde{E}$ such that $d(e^{-\prime}, e^-) \leq R$, $d(e', e) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e^{-\prime}), e^-)$, $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e'), e)$, and $i = i_{M,\rho}(e^{-\prime}) = i_{M,\rho}(e')$. We just have to show $e^{-\prime} = e'$, as then $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e^-)$, $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e'), e)$, and $e^- \mathrel{\widetilde{\lessdot}}_p e$ imply $e_0 \lessdot_p e_0'$. Because $e^{-\prime} \neq e'$, according to Claim 2, implies $i_{M,\rho}(e') \neq i_{M,\rho}(e)$, which leads to a contradiction.
3. Suppose $\mathcal{S} \neq \emptyset$ and suppose there is $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}'$ with e_0 minimal in $(E_p, <_p)$. There is $e' \in \widetilde{E}$ such that $d(e', e) \leq R$ and $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e)$. As $\mathcal{S} \neq \emptyset$, e is not minimal in $(\widetilde{E}_p, \mathrel{\widetilde{\lessdot}}_p)$ and, consequently, $d(\gamma, e_0) = d(e', e) = R$ (if $d(e', e) < R$, e would have to be minimal in $(\widetilde{E}_p, \mathrel{\widetilde{\lessdot}}_p)$).
4. Let $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ with e_0 maximal in $(E_p, <_p)$ and let $e^- \in \widetilde{E}_p$ such that $e^- \mathrel{\widetilde{\lessdot}}_p e$. Furthermore, as $r^-(e) = r(e^-)$, there is $e^{-\prime} \in \widetilde{E}$ such that both

$d(e^{-\prime}, e^-) \leq R$ and $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e^{-\prime}), e^-)$. As e^- is not maximal in $(\widetilde{E}_p, \widetilde{\lessdot}_p)$, $d(e_0, \gamma) = d(e^{-\prime}, e^-) = R$ (analogously to 3., if $d(e^{-\prime}, e^-) < R$, e^- would have to be maximal in $(\widetilde{E}_p, \widetilde{\lessdot}_p)$).

5. Suppose there is an extended R-sphere $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}'$ with e_0 not minimal in $(E_p, <_p)$. Let $e_0^- \in E$ such that $e_0^- \lessdot_p e_0$. As $r(e) = (\mathcal{S}', \nu')$, there is $e' \in \widetilde{E}$ with $d(e', e) \leq R$ such that $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e)$ and $i = i_{M,\rho}(e')$. As a consequence, e is not minimal in $(\widetilde{E}_p, \widetilde{\lessdot}_p)$ so that there is $e^- \in \widetilde{E}$ with $e^- \mathrel{\widetilde{\lessdot}_p} e$. As furthermore $d(e', e^-) = d(\gamma, e_0^-) \leq R$ and $(E, \leq, \lambda, \gamma, e_0^-) \sim (\rho(e'), e^-)$, it holds $((E, \leq, \lambda, \gamma, e_0^-), i) \in \mathcal{S}$.
6. Suppose there is an extended R-sphere $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ (then e is not minimal in $(\widetilde{E}_p, \widetilde{\lessdot}_p)$, so let $e^- \in \widetilde{E}_p$ such that $e^- \mathrel{\widetilde{\lessdot}_p} e$) with e_0 not maximal in $(E_p, <_p)$. Let $e_0^+ \in E$ such that $e_0 \lessdot_p e_0^+$. As we have $r^-(e) = r(e^-) = (\mathcal{S}, \nu)$, there exists $e^{-\prime} \in \widetilde{E}$ with $d(e^{-\prime}, e^-) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e^{-\prime}), e^-)$, and $i = i_{M,\rho}(e^{-\prime})$. Since then $d(e^{-\prime}, e) = d(\gamma, e_0^+) \leq R$ and also $(E, \leq, \lambda, \gamma, e_0^+) \sim (\rho(e^{-\prime}), e)$, we have $((E, \leq, \lambda, \gamma, e_0^+), i) \in \mathcal{S}'$.
7. Let $\mathcal{P}, \mathcal{N} \subseteq \mathfrak{S}^+$ such that $m((e, e_r)) = (\mathcal{P}, \mathcal{N})$.
   (a) Let $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}'$ and $e_0' \in E$. According to the definition of m, $e_0 <_c e_0'$ implies $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{P}$.
   (b) Let $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}'$ and $e_0' \in E$. According to the definition of m, $e_0 \not<_c e_0'$ implies $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{N}$.
   (c) Let $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{P}$. Then, due to the definition of $\mathcal{P}$, there is $e_0' \in E$ with $e_0' <_c e_0$ and $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{S}$.
8. As $\varsigma(\mathcal{S}') \sim \rho(e)$ and $|\{e' \mathrel{\widetilde{\leq}_p} e \mid \varsigma(\mathcal{S}') \sim \rho(e')\}| = |\{e' \mathrel{\widetilde{<}_p} e \mid \varsigma(\mathcal{S}') \sim \rho(e')\}|+1$, we have $\nu'(\varsigma(\mathcal{S}')) = \min\{|\{e' \mathrel{\widetilde{<}_p} e \mid \varsigma(\mathcal{S}') \sim \rho(e')\}|+1, \max(Occ)\}$. Furthermore, $\nu'(g) = \nu(g)$ if $g \neq \varsigma(\mathcal{S}')$.

Verifying $(r^-(e), \lambda(e), m((e_s, e)), r(e)) \in \Delta_p$ for any $e \in E_p$ and $e_s \in E$ with $e_s \mathrel{\widetilde{<}_c} e$ differs from the above scheme only in point 7. (set $(\mathcal{S}, \nu)$ to be $r(e_s)$ and $(\mathcal{S}', \nu')$ to be $r(e)$ and let $\mathcal{P}, \mathcal{N} \subseteq \mathfrak{S}^+$ such that $m((e_s, e)) = (\mathcal{P}, \mathcal{N})$):

7. (a) Suppose there is $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{P}$. Then there exists $e_0 \in E$ with $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 <_c e_0'$. Due to $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$, there is $e_s' \in \widetilde{E}$ with $d(e_s', e_s) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e_s'), e_s)$, and $i = i_{M,\rho}(e_s')$. As then $d(e_s', e) = d(\gamma, e_0') \leq R$ and $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e_s'), e)$, $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{S}'$.
   (b) Suppose there is $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{N} \cap \mathcal{S}'$. Then there is $e_0 \in E$ with $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 \not<_c e_0'$. Due to $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$, there is $e_s' \in \widetilde{E}$ satisfying $d(e_s', e_s) \leq R$, $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e_s'), e_s)$, and $i = i_{M,\rho}(e_s')$. Due to $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{S}'$, there is also $e' \in \widetilde{E}$ with $d(e', e) \leq R$, $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e'), e)$, and $i = i_{M,\rho}(e')$. Suppose $e_s' \neq e'$. But then, as $\rho(e_s') \sim \rho(e')$, $i_{M,\rho}(e_s') \neq i_{M,\rho}(e')$, which leads to a contradiction. Now suppose $e_s' = e'$. But then $e_s \mathrel{\widetilde{<}_c} e$ implies $e_0 <_c e_0'$, also contradicting the premise.
   (c) Suppose there is $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{S}'$ and $e_0 \in E$ with $e_0 <_c e_0'$. Then there is $e' \in \widetilde{E}$ with $d(e', e) \leq R$, $(E, \leq, \lambda, \gamma, e_0') \sim (\rho(e'), e)$, and

$i = i_{M,\rho}(e')$. As $d(e', e_s) = d(\gamma, e_0) \leq R$ and $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e_s)$, it holds $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and, thus, $((E, \leq, \lambda, \gamma, e_0'), i) \in \mathcal{P}$.

In the following, we verify that (r, m) is accepting. So set, given $p \in \mathit{Proc}$, $(\mathcal{S}_p, \nu_p)$ to be $(\emptyset, \nu_p^0)$ if $\widetilde{E}_p$ is empty and, otherwise, $(\mathcal{S}_p, \nu_p)$ to be $r(e_p)$ where $e_p \in \widetilde{E}_p$ is the maximal event wrt. $\widetilde{\leq}_p$. Clearly, the union of mappings ν_p carries, for each $g \in \mathfrak{G}$, the number of occurrences of g in ρ. Furthermore, for all $p \in \mathit{Proc}$ and $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}_p$, e_0 is maximal in $(E_p, <_p)$. Because suppose there is $e_0' \in E$ with $e_0 \lessdot_p e_0'$. But then, as there exists no $e^+ \in \widetilde{E}$ satisfying $e_p \mathrel{\widetilde{\lessdot}_p} e^+$, there is no $e' \in \widetilde{E}$ either with $d(e', e_p) \leq R$ such that $(E, \leq, \lambda, \gamma, e_0) \sim (\rho(e'), e_p)$, which contradicts the definition of r.

"$\subseteq$": Let (r, m) be an accepting run of $\mathcal{A}$ on an MSC $M = (\widetilde{E}, \widetilde{\leq}, \widetilde{\lambda})$. Then $\rho : \widetilde{E} \to \mathfrak{G}$ with $\rho(e) = g$ if there is $\mathcal{S}$ and ν such that $r(e) = (\mathcal{S}, \nu)$ and $g = \varsigma(\mathcal{S})$ turns out to be an accepting run of $\mathcal{GA}$ on M. For $((E, \leq, \lambda, \gamma, e_0), i) \in \mathfrak{G}^+$ and $e \in \widetilde{E}$, we sometimes write $((E, \leq, \lambda, \gamma, e_0), i) \in r(e)$ if there is $\mathcal{S}$ and ν such that $r(e) = (\mathcal{S}, \nu)$ and $((E, \leq, \lambda, \gamma, e_0), i) \in \mathcal{S}$.

Claim 3. For each $e \in \widetilde{E}$, $((E, \leq, \lambda, \gamma, \bar{e}), i) \in r(e)$, and $d \in \mathbb{N}$, if there is a sequence of events $e_0, \ldots, e_d \in E$ such that $e_0 = \bar{e}$ and, for each $k \in \{0, \ldots, d-1\}$, $e_k \lessdot e_{k+1}$ or $e_{k+1} \lessdot e_k$, then there is a unique sequence of events $\widehat{e}_0, \ldots, \widehat{e}_d \in \widetilde{E}$ with

- $\widehat{e}_0 = e$,
- for each $k \in \{0, \ldots, d\}$, $((E, \leq, \lambda, \gamma, e_k), i) \in r(\widehat{e}_k)$, and
- for each $k \in \{0, \ldots, d-1\}$, $\widehat{e}_k \mathrel{\widetilde{\lessdot}} \widehat{e}_{k+1}$ iff $e_k \lessdot e_{k+1}$ and $\widehat{e}_{k+1} \mathrel{\widetilde{\lessdot}} \widehat{e}_k$ iff $e_{k+1} \lessdot e_k$.

Proof of Claim 3. We proceed by induction. Obviously, the statement holds for $d = 0$. Now assume there is a sequence of events $e_0, \ldots, e_d, e_{d+1} \in E$ such that $e_0 = \bar{e}$ and, for each $k \in \{0, \ldots, d\}$, $e_k \lessdot e_{k+1}$ or $e_{k+1} \lessdot e_k$. By induction hypothesis, there is a unique sequence of events $\widehat{e}_0, \ldots, \widehat{e}_d \in \widetilde{E}$ with

- $\widehat{e}_0 = e$,
- for each $k \in \{0, \ldots, d\}$, $((E, \leq, \lambda, \gamma, e_k), i) \in r(\widehat{e}_k)$ (in particular, $\lambda(e_k) = (\widetilde{\lambda}(\widehat{e}_k), q)$ for some $q \in Q$), and
- for each $k \in \{0, \ldots, d-1\}$, $\widehat{e}_k \mathrel{\widetilde{\lessdot}} \widehat{e}_{k+1}$ iff $e_k \lessdot e_{k+1}$ (which implies, for one thing, $\widehat{e}_k \mathrel{\widetilde{<}_c} \widehat{e}_{k+1}$ iff $e_k <_c e_{k+1}$) and $\widehat{e}_{k+1} \mathrel{\widetilde{\lessdot}} \widehat{e}_k$ iff $e_{k+1} \lessdot e_k$.

Suppose that

- $e_d \lessdot_p e_{d+1}$ for some $p \in \mathit{Proc}$. As e_d is not maximal in $(E_p, <_p)$, $r(\widehat{e}_d)$ cannot be part of a final state so that there is a (unique) event $\widehat{e}_{d+1} \in \widetilde{E}$ with $\widehat{e}_d \mathrel{\widetilde{\lessdot}_p} \widehat{e}_{d+1}$. Furthermore, due to item 6. from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e_{d+1}), i) \in r(\widehat{e}_{d+1})$.
- $e_{d+1} \lessdot_p e_d$ for some $p \in \mathit{Proc}$. As e_d is not minimal in $(E_p, <_p)$, there is, according to item 5. from the definition of Δ_p, a (unique) event $\widehat{e}_{d+1} \in \widetilde{E}$ with $\widehat{e}_{d+1} \mathrel{\widetilde{\lessdot}_p} \widehat{e}_d$ and $((E, \leq, \lambda, \gamma, e_{d+1}), i) \in r(\widehat{e}_{d+1})$.

- $e_d <_c e_{d+1}$. There is a (unique) event $\widehat{e}_{d+1} \in \widetilde{E}$ with $\widehat{e}_d \mathrel{\widetilde{<}_c} \widehat{e}_{d+1}$. Set $(\mathcal{P},\mathcal{N})$ to be $m((\widehat{e}_d, \widehat{e}_{d+1}))$. According to item 7. (i) (a) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e_{d+1}), i) \in \mathcal{P}$. With 7. (ii) (a), it follows $((E, \leq, \lambda, \gamma, e_{d+1}), i) \in r(\widehat{e}_{d+1})$.
- $e_{d+1} <_c e_d$. There is a (unique) event $\widehat{e}_{d+1} \in \widetilde{E}$ with $\widehat{e}_{d+1} \mathrel{\widetilde{<}_c} \widehat{e}_d$. Set $(\mathcal{P},\mathcal{N})$ to be $m((\widehat{e}_{d+1}, \widehat{e}_d))$. According to item 7. (ii) (c) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e_d), i) \in \mathcal{P}$. With 7. (i) (c), it follows $((E, \leq, \lambda, \gamma, e_{d+1}), i) \in r(\widehat{e}_{d+1})$.

This concludes the proof of Claim 3.

Let $\mu : \widetilde{E} \to Q$ be given by $\mu(e) = q$ if there is $\mathcal{S}$, ν, and σ such that $r(e) = (\mathcal{S}, \nu)$ and $\lambda(\mathcal{S}) = (\sigma, q)$. We have to show that, for each $e \in \widetilde{E}$, $is_{(\widetilde{E},\widetilde{\leq},(\widetilde{\lambda},\mu)),R}(e)$ is isomorphic to $\rho(e)$. So let $e \in \widetilde{E}$ and set $(E, \leq, \lambda, \gamma)$ to be $\rho(e)$ and $i \in \{1, \ldots, 4 \cdot maxE + 1\}$ to be the unique element with $((E, \leq, \lambda, \gamma, \gamma), i) \in r(e)$.

Claim 4. For each $d \in \{0, \ldots, R\}$, there is an isomorphism

$$h : is_{(\widetilde{E},\widetilde{\leq},(\widetilde{\lambda},\mu)),d}(e) \to is_{(E,\leq,\lambda),d}(\gamma)$$

such that, for each $\widehat{e} \in \widetilde{E}$ with $d(\widehat{e}, e) \leq d$, $((E, \leq, \lambda, \gamma, h(\widehat{e})), i) \in r(\widehat{e})$.

Proof of Claim 4. We proceed by induction. The statement holds for $d = 0$. Now assume $d < R$ and there is an isomorphism $h : is_{(\widetilde{E},\widetilde{\leq},(\widetilde{\lambda},\mu)),d}(e) \to is_{(E,\leq,\lambda),d}(\gamma)$ such that, for each $\widehat{e} \in \widetilde{E}$ with $d(\widehat{e}, e) \leq d$, $((E, \leq, \lambda, \gamma, h(\widehat{e})), i) \in r(\widehat{e})$.

Extended sphere simulates MSC Suppose there is $\widehat{e}_1, \widehat{e}'_1, \widehat{e}_2, \widehat{e}'_2 \in \widetilde{E}$ such that $d(\widehat{e}_1, e) = d(\widehat{e}_2, e) = d$, $d(\widehat{e}'_1, e) = d(\widehat{e}'_2, e) = d + 1$, ($\widehat{e}_1 \mathrel{\widetilde{\lessdot}} \widehat{e}'_1$ or $\widehat{e}'_1 \mathrel{\widetilde{\lessdot}} \widehat{e}_1$) and ($\widehat{e}_2 \mathrel{\widetilde{\lessdot}} \widehat{e}'_2$ or $\widehat{e}'_2 \mathrel{\widetilde{\lessdot}} \widehat{e}_2$). Furthermore, suppose (let e_1 and e_2 denote $h(\widehat{e}_1)$ and $h(\widehat{e}_2)$, respectively)

- $\widehat{e}_1 \mathrel{\widetilde{\lessdot}_p} \widehat{e}'_1$ for some $p \in Proc$. As $d(\widehat{e}_1, e) < R$, we have $d(e_1, \gamma) < R$. Due to item 4. from the definition of Δ_p, e_1 is not maximal in $(E_p, <_p)$ so that there is $e'_1 \in E$ with $e_1 \lessdot_p e'_1$ and, due to item 6. and $((E, \leq, \lambda, \gamma, e_1), i) \in r(\widehat{e}_1)$, $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $\widehat{e}'_1 \mathrel{\widetilde{\lessdot}_p} \widehat{e}_1$ for some $p \in Proc$. As $d(\widehat{e}_1, e)$ is less than R, so is $d(e_1, \gamma)$. Due to item 3. from the definition of Δ_p, e_1 is not minimal in $(E_p, <_p)$ so that there is $e'_1 \in E$ with $e'_1 \lessdot_p e_1$ and, due to item 5. and $((E, \leq, \lambda, \gamma, e_1), i) \in r(\widehat{e}_1)$, $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $\widehat{e}_1 \mathrel{\widetilde{<}_c} \widehat{e}'_1$. Set $(\mathcal{P},\mathcal{N})$ to be $m((\widehat{e}_1, \widehat{e}'_1))$. As $d(\widehat{e}_1, e) < R$ and, thus, $d(e_1, \gamma) < R$, there is $e'_1 \in E$ such that $e_1 <_c e'_1$. (This is because $(E, \leq, \lambda, \gamma)$ can be embedded into some MSC.) According to item 7. (i) (a) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e'_1), i) \in \mathcal{P}$. Due to item 7. (ii) (a), it then follows from $((E, \leq, \lambda, \gamma, e_1), i) \in r(\widehat{e}_1)$ that $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $\widehat{e}'_1 \mathrel{\widetilde{<}_c} \widehat{e}_1$. Set $(\mathcal{P},\mathcal{N})$ to be $m((\widehat{e}'_1, \widehat{e}_1))$. As $d(\widehat{e}_1, e) < R$ and, consequently, $d(e_1, \gamma) < R$, there is also $e'_1 \in E$ such that $e'_1 <_c e_1$. (Recall that $(E, \leq, \lambda, \gamma)$

can be embedded into some MSC.) According to item 7. (ii) (c) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e_1), i) \in \mathcal{P}$. Due to item 7. (i) (c), it then follows that $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.

Thus, depending on $\widehat{e}'_1$, we obtain from e_1 a unique event $e'_1 \in E$, which we denote by $h'(\widehat{e}'_1)$. According to the above scheme, we obtain from e_2 a unique event $e'_2 \in E$, denoted by $h'(\widehat{e}'_2)$. It holds $d(e'_1, \gamma) = d(e'_2, \gamma) = d+1$. Now suppose

- $\widehat{e}'_1 \mathrel{\widetilde{\lessdot}_p} \widehat{e}'_2$ for some $p \in \mathit{Proc}$. As already $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$ and $((E, \leq, \lambda, \gamma, e'_2), i) \in r(\widehat{e}'_2)$, it follows from item 2. of the definition of Δ_p that $e'_1 \lessdot_p e'_2$.
- $\widehat{e}'_1 \mathrel{\widetilde{<}_c} \widehat{e}'_2$. Set $(\mathcal{P}, \mathcal{N})$ to be $m((\widehat{e}'_1, \widehat{e}'_2))$ and suppose $e'_1 <_c e'_2$ does not hold. But then, according to items 7. (i) (b) and 7. (ii) (b) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e'_2), i) \in \mathcal{N}$ and $((E, \leq, \lambda, \gamma, e'_2), i) \notin r(\widehat{e}'_2)$, resulting in a contradiction.
- $\widehat{e}'_1 = \widehat{e}'_2$. Then $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$ and $((E, \leq, \lambda, \gamma, e'_2), i) \in r(\widehat{e}'_1)$ implies $e'_1 = e'_2$ (otherwise, $r(\widehat{e}'_1)$ would not be a valid state of $\mathcal{A}$).

The cases $\widehat{e}'_2 \mathrel{\widetilde{\lessdot}_p} \widehat{e}'_1$ and $\widehat{e}'_2 \mathrel{\widetilde{<}_c} \widehat{e}'_1$ are handled analogously.

MSC simulates extended sphere Suppose there is $e_1, e'_1, e_2, e'_2 \in E$ such that $d(e_1, \gamma) = d(e_2, \gamma) = d$, $d(e'_1, \gamma) = d(e'_2, \gamma) = d + 1$, ($e_1 \lessdot e'_1$ or $e'_1 \lessdot e_1$) and ($e_2 \lessdot e'_2$ or $e'_2 \lessdot e_2$). We now proceed as in the proof of Claim 3. So suppose (let $\widehat{e}_1$ and $\widehat{e}_2$ denote $h^{-1}(e_1)$ and $h^{-1}(e_2)$, respectively)

- $e_1 \lessdot_p e'_1$ for some $p \in \mathit{Proc}$. As e_1 is not maximal in $(E_p, <_p)$, $r(\widehat{e}_1)$ cannot be part of a final state so that there is $\widehat{e}'_1 \in \widetilde{E}$ with $\widehat{e}_1 \mathrel{\widetilde{\lessdot}_p} \widehat{e}'_1$. Furthermore, due to item 6. from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $e'_1 \lessdot_p e_1$ for some $p \in \mathit{Proc}$. As e_1 is not minimal in $(E_p, <_p)$ there is, according to item 5. from the definition of Δ_p, $\widehat{e}'_1 \in \widetilde{E}$ with $\widehat{e}'_1 \mathrel{\widetilde{\lessdot}_p} \widehat{e}_1$ and $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $e_1 <_c e'_1$. There is $\widehat{e}'_1 \in \widetilde{E}$ with $\widehat{e}_1 \mathrel{\widetilde{<}_c} \widehat{e}'_1$. Set $(\mathcal{P}, \mathcal{N})$ to be $m((\widehat{e}_1, \widehat{e}'_1))$. According to item 7. (i) (a) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e'_1), i) \in \mathcal{P}$. With 7. (ii) (a), it follows $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.
- $e'_1 <_c e_1$. There is $\widehat{e}'_1 \in \widetilde{E}$ with $\widehat{e}'_1 \mathrel{\widetilde{<}_c} \widehat{e}_1$. Set $(\mathcal{P}, \mathcal{N})$ to be $m((\widehat{e}'_1, \widehat{e}_1))$. According to item 7. (ii) (c) from the definition of Δ_p, $((E, \leq, \lambda, \gamma, e_1), i) \in \mathcal{P}$. With 7. (i) (c), it follows $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$.

According to the above scheme, we obtain from $\widehat{e}_2$ a unique event $\widehat{e}'_2$. Now suppose

- $e'_1 \lessdot_p e'_2$ for some $p \in \mathit{Proc}$. Assume $\widehat{e}'_1 \mathrel{\widetilde{\not\lessdot}_p} \widehat{e}'_2$. According to the definition of the set of states of $\mathcal{A}$, $e'_1 \neq e'_2$, $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$, and $((E, \leq, \lambda, \gamma, e'_2), i) \in r(\widehat{e}'_2)$ implies $\widehat{e}'_1 \neq \widehat{e}'_2$. But then, following the scheme depicted in Figure 6, we can construct an infinite sequence $x_1, x_2, \ldots \in \widetilde{E}$ inducing an infinite set of (pairwise distinct) events: Suppose $\widehat{e}'_1 \mathrel{\widetilde{<}_p} \widehat{e}'_2$. (The other case is handled analogously.) Set $x_1 \in \widetilde{E}$ to be the unique event satisfying $\widehat{e}'_1 \mathrel{\widetilde{\lessdot}_p} x_1$. We have $((E, \leq, \lambda, \gamma, e'_2), i) \in r(x_1)$ and $x_1 \mathrel{\widetilde{<}_p} \widehat{e}'_2$. According to Claim 3, there is $x_2 \in \widetilde{E}$ such that $((E, \leq, \lambda, \gamma, \gamma), i) \in r(x_2)$ and $x_2 \mathrel{\widetilde{<}} e$. (There is a path in $(E, \leq, \lambda)$ from e'_2 to γ that, according to Claim 3, takes M from $\widehat{e}'_2$ to e. Apply this path to x_1 yielding a path to a unique event $x_2 \in \widetilde{E}$ with $((E, \leq, \lambda, \gamma, \gamma), i) \in r(x_2)$. From $x_1 \mathrel{\widetilde{<}_p} \widehat{e}'_2$, it easily follows that $x_2 \mathrel{\widetilde{<}} e$.) Similarly, there is $x_3 \in \widetilde{E}$ with $((E, \leq, \lambda, \gamma, e'_1), i) \in r(x_3)$ and $x_3 \mathrel{\widetilde{<}_p} \widehat{e}'_1$. Now let $x_4 \in \widetilde{E}$ be the unique event such that $x_3 \mathrel{\widetilde{\lessdot}_p} x_4$ and $((E, \leq, \lambda, \gamma, e'_2), i) \in r(x_4)$ (as already $((E, \leq, \lambda, \gamma, e'_1), i) \in r(\widehat{e}'_1)$, it holds $x_4 \mathrel{\widetilde{<}_p} \widehat{e}'_1$) and let, again following Claim 3, $x_5 \in \widetilde{E}$ be an event with $((E, \leq, \lambda, \gamma, \gamma), i) \in r(x_5)$ and $x_5 \mathrel{\widetilde{<}} x_2$ and $x_6 \in \widetilde{E}$ be an event with $((E, \leq, \lambda, \gamma, e'_1), i) \in r(x_6)$ and $x_6 \mathrel{\widetilde{<}_p} x_3$. Continuing this scheme yields an infinite set of events, contradicting the premise that we deal with finite MSCs.

- $e'_1 <_c e'_2$. Assuming $\widehat{e}'_1 \mathrel{\not\widetilde{<}_c} \widehat{e}'_2$, we proceed according to the very same scheme as in case $e'_1 \lessdot_p e'_2$ to generate an infinite sequence $x_1, x_2, \ldots \in \widetilde{E}$ inducing an infinite set of events, i.e., set $x_1 \in \widetilde{E}$ to be the unique event such that $\widehat{e}'_1 \mathrel{\widetilde{<}_c} x_1$ and $((E, \leq, \lambda, \gamma, e'_2), i) \in r(x_1)$. Assuming $x_1 \mathrel{\widetilde{<}} \widehat{e}'_2$, we can find $x_2 \in \widetilde{E}$ with $((E, \leq, \lambda, \gamma, \gamma), i) \in r(x_2)$ and $x_2 \mathrel{\widetilde{<}} e$ and so on.
- $e'_1 = e'_2$. Again, assuming $\widehat{e}'_1 \neq \widehat{e}'_2$, we can generate an infinite sequence $x_1, x_2, \ldots \in \widetilde{E}$ inducing an infinite set of events as follows: Suppose $\widehat{e}'_1 \mathrel{\widetilde{<}} \widehat{e}'_2$. According to Claim 3, we can find $x_1 \in \widetilde{E}$ such that $((E, \leq, \lambda, \gamma, \gamma), i) \in r(x_1)$ and $x_1 \mathrel{\widetilde{<}} e$. Furthermore, there is $x_2 \in \widetilde{E}$ satisfying $((E, \leq, \lambda, \gamma, e'_1), i) \in r(x_2)$ and $x_2 \mathrel{\widetilde{<}} \widehat{e}'_1$ and so on.

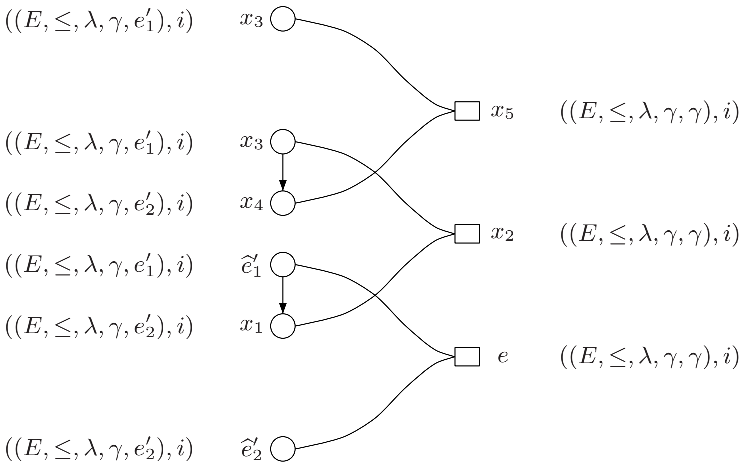

Fig. 6. An infinite sequence of events

The cases $e'_2 \lessdot_p e'_1$ and $e'_2 <_c e'_1$ are handled analogously. From the above results, we conclude that the mapping $\widehat{h} : is_{(\widetilde{E}, \widetilde{\leq}, (\widetilde{\lambda}, \mu)), d+1}(e) \to is_{(E, \leq, \lambda), d+1}(\gamma)$ given by

$$\widehat{h}(\widehat{e}) = \begin{cases} h(\widehat{e}) & \text{if } d(\widehat{e}, e) \leq d \\ h'(\widehat{e}) & \text{if } d(\widehat{e}, e) = d + 1 \end{cases}$$

(for $\widehat{e} \in \widetilde{E}$ with $d(\widehat{e}, e) \leq d+1$) is an isomorphism satisfying, for any $\widehat{e} \in \widetilde{E}$ with $d(\widehat{e}, e) \leq d+1$, $((E, \leq, \lambda, \gamma, \widehat{h}(\widehat{e})), i) \in r(\widehat{e})$. This concludes the proof of Claim 4.

As $((\mathcal{S}_p, \nu_p))_{p \in \mathit{Proc}} \in F$ only if the union of mappings ν_p is a model of Occ, an accepting run of $\mathcal{A}$ makes sure that the number of occurrences of an R-sphere meets the obligations imposed by $\mathcal{GA}$. This concludes the proof of Claim 1. $\square$

4 Beyond Realizability

In this section, we show that MSO logic over MSCs is strictly more expressive than EMSO. Together with the results of the previous section, this will be used to show that MPAs cannot be complemented in general. More specifically, we show that quantifier-alternation forms a hierarchy:

Theorem 3. *The monadic quantifier-alternation hierarchy over* $\mathbb{MSC}$ *is infinite.*

Proof. Matz and Thomas proved infinity of the monadic quantifier-alternation hierarchy over *grids* [10]. We show how grids can be encoded into MSCs and then rewrite their result in terms of MSCs.

For a positive natural $n \in \mathbb{N}_{\geq 1}$, we use in the following $[n]$ as a shorthand for $\{1, \ldots, n\}$. Given positive naturals $m, n \in \mathbb{N}_{\geq 1}$, the (m,n)-*grid* $g(m,n)$ (with m columns and n rows) is the structure $g(m,n) := ([m] \times [n], S_1, S_2)$ where $S_1, S_2 \subseteq ([m] \times [n])^2$ contain the pairs $((i,j),(i+1,j))$ with $i+1 \leq m$ and $j \leq n$ and, respectively, $((i,j),(i,j+1))$ with $i \leq m$ and $j+1 \leq n$. A relation $R \subseteq \mathbb{N}_{\geq 1} \times \mathbb{N}_{\geq 1}$ may be represented by the grid language $\{g(m,n) \mid (m,n) \in R\}$. As a unary function $f : \mathbb{N}_{\geq 1} \to \mathbb{N}_{\geq 1}$ can be considered as a binary relation, we define the *grid language* $G(f)$ of f to be the set $\{g(f(n), n) \mid n \in \mathbb{N}_{\geq 1}\}$. A grid $g(m,n)$ can be folded to an MSC $M(m,n)$ as exemplarily shown in Figure 8. Formally, $M(m,n)$ is given by its projections as follows:

$$M(m,n) \upharpoonright 1 = \begin{cases} (1!2)^{n-1}(1!3)\left[((1?2)(1!2))^{n-1}(1?3)(1!3)\right]^{(m-1)/2} & \text{if } m \text{ is odd} \\ (1!2)^{n-1}(1!3)\left[((1?2)(1!2))^{n-1}(1?3)(1!3)\right]^{(m/2)-1}(1?2)^{n-1}(1?3) & \\ & \text{if } m \text{ is even} \end{cases}$$

$$M(m,n) \upharpoonright 2 = \begin{cases} \left[((2?1)(2!1))^{n-1}(2?3)(2!3)\right]^{(m-1)/2}(2?1)^{n-1}(2?3) & \text{if } m \text{ is odd} \\ \left[((2?1)(2!1))^{n-1}(2?3)(2!3)\right]^{m/2} & \text{if } m \text{ is even} \end{cases}$$

$$M(m,n) \upharpoonright 3 = \begin{cases} ((3?1)(3!2)(3?2)(3!1))^{(m-1)/2}(3?1)(3!2) & \text{if } m \text{ is odd} \\ ((3?1)(3!2)(3?2)(3!1))^{m/2} & \text{if } m \text{ is even} \end{cases}$$

A similar encoding is used by Kuske [8], who proves infinity of the monadic quantifier-alternation hierarchy for certain *pomsets* over at least two processes.

However, considering MSCs, things become more complicated. In particular, we need to introduce a third process to obtain distinguished labelings of events that mark the end of a column in the grid to be encoded. By the type of an event, we furthermore recognize which event really corresponds to the grid's node, namely those that are labelled with a send action performed by process 1 or 2. EMSO-definability of the set of all grid encodings, which is needed in the proof, is witnessed by a corresponding MPA.

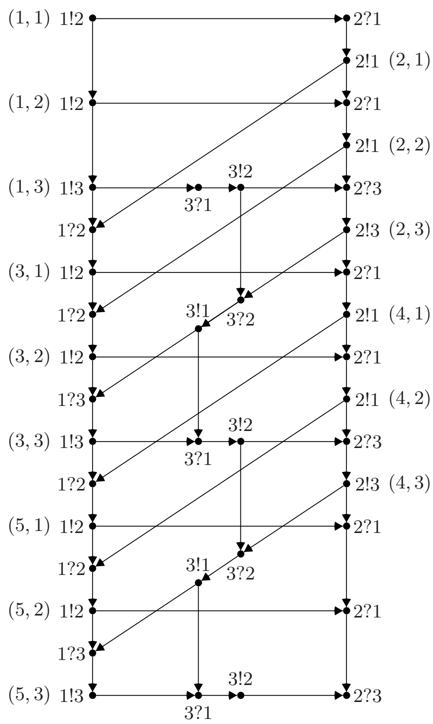

Fig. 7. Folding the $(5, 3)$-grid

A grid language G defines the MSC language $L(G) := \{M(m, n) \mid g(m, n) \in G\}$. For a function $f : \mathbb{N}_{\geq 1} \to \mathbb{N}_{\geq 1}$, we furthermore write $L(f)$ as a shorthand for the MSC language $L(G(f))$. We now closely follow [15], which resumes the result of [10]. So let, for $k \in \mathbb{N}$, the functions $s_k, f_k : \mathbb{N}_{\geq 1} \to \mathbb{N}_{\geq 1}$ be inductively defined via $s_0(n) = n$, $s_{k+1}(n) = 2^{s_k(n)}$, $f_0(n) = n$, and $f_{k+1}(n) = f_k(n) \cdot 2^{f_k(n)}$.

Claim 5. For each $k \in \mathbb{N}$, the MSC language $L(f_k)$ is Σ_{2k+3}-definable.

Proof of Claim 5. We show that, for any $k \geq 1$, if a grid language G is Σ_k-definable (over grids), then $L(G)$ is Σ_k-definable (over MSCs). The claim then follows from the fact that any grid language $G(f_k)$ is Σ_{2k+3}-definable [15].

So let $k \in \mathbb{N}_{\geq 1}$. Figure 8 shows the MPA $\mathcal{A}_{\mathcal{GF}}$, which recognizes the set of all possible grid foldings. As the part of process 3 is the easy one, it is omitted. Note that synchronization messages are only needed when processes 1 and 2 synchronize. For clarity, ε-transitions are employed, which can easily be eliminated without affecting the recognized language. Note that a global final state is depicted by a dashed line. Moreover, its labelling indicates, which grid foldings it accepts, while m and n range over $\mathbb{N}$ and $\mathbb{N}_{\geq 1}$, respectively. According to Lemma 1, let $\varphi_{\mathcal{GF}} = \exists \overline{X} \psi_{\mathcal{GF}}(\overline{X})$ be an EMSO sentence (over MSCs) with first-order kernel $\psi_{\mathcal{GF}}(\overline{X})$ that defines the language of $\mathcal{A}_{\mathcal{GF}}$. Let furthermore $\varphi = \exists \overline{Y_1} \forall \overline{Y_2} \ldots \exists/\forall \overline{Y_k} \varphi'(\overline{Y_1}, \ldots, \overline{Y_k})$ be a Σ_k-sentence (over grids) where $\varphi'(\overline{Y_1}, \ldots, \overline{Y_k})$ contains no set quantifiers. Without loss of generality, $\varphi_{\mathcal{GF}}$ and φ employ distinct sets of variables. We now determine the Σ_k-sentence Ψ_φ over MSCs with $L(\Psi_\varphi) = L(G(\varphi))$ (where $G(\varphi)$ is the set of models of φ interpreted over grids), i.e., the foldings of $G(\varphi)$ form exactly the MSC language defined by Ψ_φ. So let Ψ_φ be given by

$$\exists \overline{X} \exists \overline{Y_1} \forall \overline{Y_2} \ldots \exists/\forall \overline{Y_k} (\psi_{\mathcal{GF}}(\overline{X}) \wedge \|\varphi'(\overline{Y_1}, \ldots, \overline{Y_k})\|)$$

where $\|\varphi'(\overline{Y_1}, \ldots, \overline{Y_k})\|$ is inductively derived from $\varphi'(\overline{Y_1}, \ldots, \overline{Y_k})$ as follows:

- $\|S_1(x,y)\| = \quad L_{1!2}(x) \wedge L_{2!1}(y) \wedge \exists z(x <_c z \wedge z \lessdot_{\mathrm{p}} y)$
  $\vee\, L_{2!1}(x) \wedge L_{1!2}(y) \wedge \exists z(x <_c z \wedge z \lessdot_{\mathrm{p}} y)$
  $\vee\, L_{1!3}(x) \wedge L_{2!3}(y) \wedge$
  $\exists z \exists z' \exists z''(x <_c z \wedge z \lessdot_{\mathrm{p}} z' \wedge z' <_c z'' \wedge z'' \lessdot_{\mathrm{p}} y)$
  $\vee\, L_{2!3}(x) \wedge L_{1!3}(y) \wedge$
  $\exists z \exists z' \exists z''(x <_c z \wedge z \lessdot_{\mathrm{p}} z' \wedge z' <_c z'' \wedge z'' \lessdot_{\mathrm{p}} y)$
- $\|S_2(x,y)\| = \quad L_{1!2}(x) \wedge (\bigvee_{\sigma \in Act^!} L_\sigma(y)) \wedge (x \lessdot_{\mathrm{p}} y \vee \exists z(x \lessdot_{\mathrm{p}} z \wedge z \lessdot_{\mathrm{p}} y))$
  $\vee\, L_{2!1}(x) \wedge \exists z(x \lessdot_{\mathrm{p}} z \wedge z \lessdot_{\mathrm{p}} y)$
- $\|x \in X\| = x \in X$
- $\|\neg\varphi\| = \neg\|\varphi\|$
- $\|\varphi \vee \psi\| = \|\varphi\| \vee \|\psi\|$
- $\|\varphi \wedge \psi\| = \|\varphi\| \wedge \|\psi\|$
- $\|\exists x\varphi\| = \exists x((\bigvee_{\sigma \in \{1!2, 2!1, 1!3, 2!3\}} L_\sigma(x)) \wedge \|\varphi\|)$
- $\|\forall x\varphi\| = \forall x((\bigvee_{\sigma \in \{1!2, 2!1, 1!3, 2!3\}} L_\sigma(x)) \rightarrow \|\varphi\|)$

Similarly to the proof of Lemma 1, the above inductive derivation makes sure that only elements that correspond to grid nodes are assigned to the variables $\overline{Y_1}, \ldots, \overline{Y_k}$. This concludes the proof of Claim 5.

Claim 6. Let $f : \mathbb{N}_{\geq 1} \rightarrow \mathbb{N}_{\geq 1}$ be a function. If $L(f)$ is Σ_k-definable (over MSCs) for some $k \geq 1$, then $f(n)$ is in $s_k(\mathcal{O}(n))$.

Proof of Claim 6. Again, the proof is an adaption of the proof of [15]. So let $k \geq 1$ and let in the following the events of an MSC $(E, \leq, \lambda)$ be labelled with elements from $Act \times \{0,1\}^i$ for some $i \in \mathbb{N}_{\geq 1}$, i.e., $\lambda : E \rightarrow Act \times \{0,1\}^i$. But note that the type of an event still depends on the type of its communication action only. Let furthermore $\varphi(Y_1, \ldots, Y_i)$ be a Σ_k-formula defining a set of MSCs over the new label alphabet that are foldings of grids. For a fixed column length $n \geq 1$, we will build a nondeterministic finite word automaton $\mathcal{A}_n$ over $(Act \times \{0,1\}^i)^n$ with $s_{k-1}(c^n)$ states (for some constant c) that reads grid-folding MSCs column by column and is equivalent to $\varphi(Y_1, \ldots, Y_i)$ wrt. grid foldings with column length n. Column here means a sequence of communication actions, each provided with an additional label, that represents a column in the corresponding grid. For example, running on the MSC $M(5,3)$

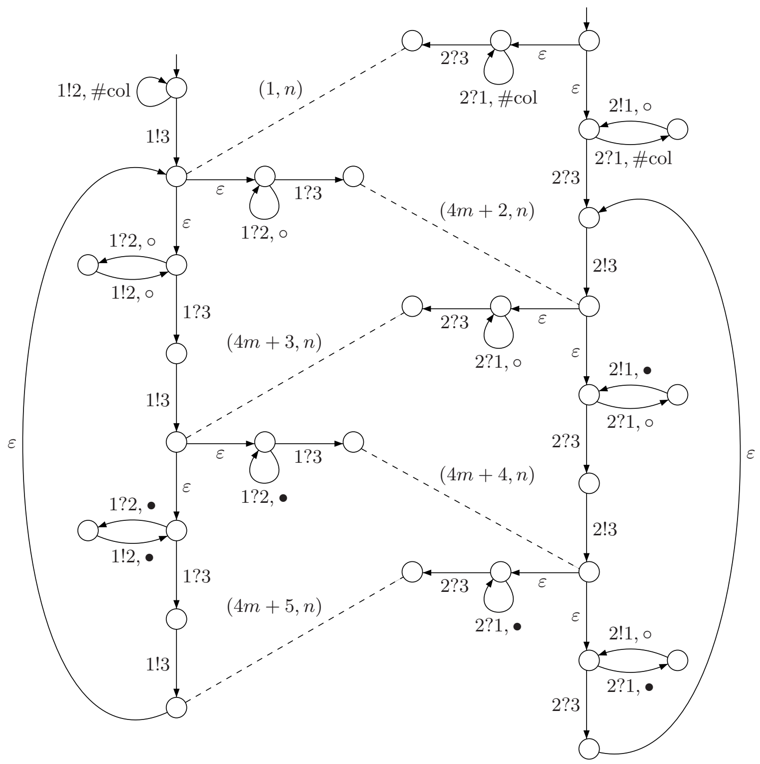

Fig. 8. A message-passing automaton regognizing $\mathcal{GF}$

shown in Figure 7, $\mathcal{A}_3$ first reads the *letter* $[(1!2)^2(1!3)(3?1)(3!2)]$ (recall that each action is still provided with an extra labelling, which we omit here for the sake of clarity), then continues reading $[((2?1)(2!1))^2(2?3)(2!3)(3?2)(3!1)]$ and so on. Then the shortest word accepted by $\mathcal{A}_n$ has length $\leq s_{k-1}(c^n)$ so that, if $\varphi(Y_1, \ldots, Y_i)$ defines an MSC language $L(f)$ for some f, we have $f(n) \in s_k(\mathcal{O}(n))$. Let us now turn to the construction of $\mathcal{A}_n$. The formula $\varphi(Y_1, \ldots, Y_i)$ is of the form $\varphi(Y_1, \ldots, Y_i) = \exists\overline{X_k}\forall\overline{X_{k-1}} \ldots \exists/\forall\overline{X_1}\psi(Y_1, \ldots, Y_i, \overline{X_k}, \ldots, \overline{X_1})$ or, equivalently, $\varphi(Y_1, \ldots, Y_i) = \exists\overline{X_k}\neg\exists\overline{X_{k-1}} \ldots \neg\exists\overline{X_1}\psi'(Y_1, \ldots, Y_i, \overline{X_k}, \ldots, \overline{X_1})$. We proceed by induction on k. For $k = 1$, $\varphi(Y_1, \ldots, Y_i)$ is an EMSO formula. According to [15], its MSC language (consisting of MSCs with extended labelings) coincides with the MSC language of some graph acceptor. The transformation from graph acceptors to MPAs from the proof of Theorem 2 can be easily adapted to handle the extended labeling. Thus, $\varphi(Y_1, \ldots, Y_i)$ defines a language that is realizable by an MPA $\mathcal{A} = ((\mathcal{A}_p)_{p \in Proc}, \mathcal{D}, \overline{s}^{in}, F)$. The automaton $\mathcal{A}_n$ can now be gained from $\mathcal{A}$ using a part of its *global transition relation* $\Longrightarrow_{\mathcal{A}} \subseteq (S_{\mathcal{A}} \times \mathcal{C}_{\mathcal{A}}) \times ((Act \times \{0,1\}^i) \times \mathcal{D}) \times (S_{\mathcal{A}} \times \mathcal{C}_{\mathcal{A}})$ (as it is defined, for example, in [6]) where $S_{\mathcal{A}}$ is the cartesian product of the local state spaces of $\mathcal{A}$ and $\mathcal{C}_{\mathcal{A}} := \{\chi \mid \chi : Ch \to (\mathcal{D} \uplus \{\bot\})^n\}$ is the set of possible *channel contents*. Note that only a bounded number of channel contents has to be considered, as the set of grid foldings with column length n forms a $\max\{1, n-1\}$-*bounded* MSC language (cf. [6] for the definition of boundedness). Due to $|S_{\mathcal{A}} \times \mathcal{C}_{\mathcal{A}}| \leq (|S_{\mathcal{A}}| \cdot (|\mathcal{D}|+1))^{|Ch| \cdot n} \leq c^n$ for some constant c, $c^n = s_0(c^n)$ is an upper bound for the number of states of $\mathcal{A}_n$, which only depends on the automaton $\mathcal{A}$ and, thus, on $\varphi(Y_1, \ldots, Y_i)$. The induction steps respectively involve both a complementation step (for negation) and a projection step (concerning existential quantification). While the former increases the number of states exponentially, the latter leaves it constant so that, altogether, the required number of states is obtained. This concludes the proof of Claim 6.

As $f_{k+1}(n)$ is not in $s_k(\mathcal{O}(n))$, it follows from Claims 5 and 6 that the hierarchy of classes of Σ_k-definable MSC languages ($k = 1, 2, \ldots$) is infinite. $\square$

As a consequence, we get the answer to two open questions, which have been raised by Kuske [9].

Corollary 1. $\mathcal{L}(\text{MPA}) \subsetneq \mathcal{MSO}$

As $\mathcal{L}(\text{MPA}) = \mathcal{EMSO}$, it follows directly that the *complement* $\overline{L} := \{M \in \mathbb{MSC} \mid M \notin L\}$ of an MSC language $L \in \mathcal{L}(\text{MPA})$, is not necessarily contained in $\mathcal{L}(\text{MPA})$, too [14].

Theorem 4. $\mathcal{L}(\text{MPA})$ *is not closed under complement.*

5 Conclusion

We have studied the classes of MSC languages that correspond to MSO logic, EMSO logic, and MPAs. We have shown that MPAs are expressively equivalent to EMSO logic. Furthermore, we proved that the class of MSC languages definable in MSO logic is strictly larger. Consequently, MPAs cannot be complemented in general. This question was raised in [9].

Since emptiness for MPAs is undecidable, we further conclude that satisfiability for EMSO and MSO and universality for MSO formulas are undecidable.

Figure 9 summarizes some of the results of this paper. Thus, for bounded languages, MSO and EMSO coincide and capture exactly the class of languages realizable by bounded [6] MPAs.

It remains to compare the nondeterministic automata model with a deterministic one in the unbounded setting. In [12, 9], it was shown that deterministic MPAs suffice to realize regular bounded MSC languages. This question was also addressed in [4] regarding the related model of asynchronous cellular automata for pomsets without autoconcurrency.

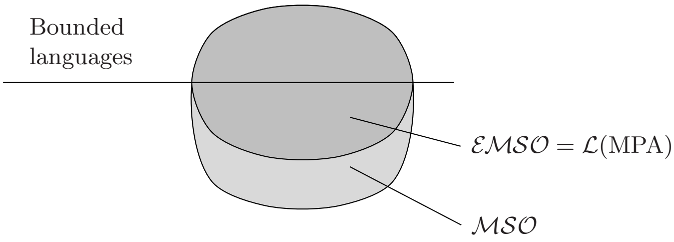

Fig. 9. An overview

References

1. R. Alur, K. Etessami, and M. Yannakakis. Inference of message sequence charts. In *22nd International Conference on Software Engineering*. ACM, 2000.
2. R. Alur and M. Yannakakis. Model checking of message sequence charts. In *CONCUR 1999*, volume 1664 of *LNCS*. Springer, 1999.
3. D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2), 1983.
4. M. Droste, P. Gastin, and D. Kuske. Asynchronous cellular automata for pomsets. *Theoretical Computer Science*, 247(1–2), 2000.
5. B. Genest, A. Muscholl, H. Seidl, and M. Zeitoun. Infinite-state high-level MSCs: Model-checking and realizability. In *ICALP 2002*, volume 2380 of *LNCS*. Springer, 2002.
6. J. G. Henriksen, M. Mukund, K. Narayan Kumar, and P. S. Thiagarajan. Regular collections of message sequence charts. In *MFCS 2000*, volume 1893 of *LNCS*. Springer, 2000.
7. ITU-TS. ITU-TS Recommendation Z.120: Message Sequence Chart 1999 (MSC99). Technical report, ITU-TS, Geneva, 1999.
8. D. Kuske. Asynchronous cellular automata and asynchronous automata for pomsets. In *CONCUR 1998*, volume 1466 of *LNCS*, 1998.
9. D. Kuske. Regular sets of infinite message sequence charts. *Information and Computation*, 187:80–109, 2003.
10. O. Matz and W. Thomas. The monadic quantifier alternation hierarchy over graphs is infinite. In *LICS 1997*. IEEE Computer Society Press, 1997.
11. R. Morin. Recognizable sets of message sequence charts. In *STACS 2002*, volume 2285 of *LNCS*. Springer, 2002.
12. M. Mukund, K. Narayan Kumar, and M. Sohoni. Synthesizing distributed finite-state systems from MSCs. In *CONCUR 2000*, volume 1877 of *LNCS*. Springer, 2000.
13. Anca Muscholl and Doron Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *MFCS 1999*, volume 1672 of *LNCS*. Springer, 1999.
14. W. Thomas. Elements of an automata theory over partial orders. In *POMIV 1996*, volume 29 of *DIMACS*. AMS, 1996.
15. W. Thomas. Automata theory on trees and partial orders. In *TAPSOFT 1997*, volume 1214 of *LNCS*. Springer, 1997.

Aachener Informatik-Berichte

This is a list of recent technical reports. To obtain copies of technical reports please consult http://aib.informatik.rwth-aachen.de/ or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

2000-01 *	Jahresbericht 1999
2000-02	Jens Vöge / Marcin Jurdzinski: A Discrete Strategy Improvement Algorithm for Solving Parity Games
2000-04	Andreas Becks / Stefan Sklorz / Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach
2000-05	Mareike Schoop: Cooperative Document Management
2000-06	Mareike Schoop / Christoph Quix (eds.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling
2000-07 *	Markus Mohnen / Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages
2000-08	Thomas Arts / Thomas Noll: Verifying Generic Erlang Client-Server Implementations
2001-01 *	Jahresbericht 2000
2001-02	Benedikt Bollig / Martin Leucker: Deciding LTL over Mazurkiewicz Traces
2001-03	Thierry Cachat: The power of one-letter rational languages
2001-04	Benedikt Bollig / Martin Leucker / Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus
2001-05	Benedikt Bollig / Martin Leucker / Thomas Noll: Regular MSC Languages
2001-06	Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic
2001-07	Martin Grohe / Stefan Wöhrle: An Existential Locality Theorem
2001-08	Mareike Schoop / James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling
2001-09	Thomas Arts / Jürgen Giesl: A collection of examples for termination of term rewriting using dependency pairs
2001-10	Achim Blumensath: Axiomatising Tree-interpretable Structures
2001-11	Klaus Indermark / Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung
2002-01 *	Jahresbericht 2001
2002-02	Jürgen Giesl / Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems
2002-03	Benedikt Bollig / Martin Leucker / Thomas Noll: Generalised Regular MSC Languages

2002-04 Jürgen Giesl / Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting
2002-05 Horst Lichter / Thomas von der Maßen / Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines
2002-06 Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata
2002-07 Jörg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities
2002-08 Markus Mohnen: An Open Framework for Data-Flow Analysis in Java
2002-09 Markus Mohnen: Interfaces with Default Implementations in Java
2002-10 Martin Leucker: Logics for Mazurkiewicz traces
2002-11 Jürgen Giesl / Hans Zantema: Liveness in Rewriting
2003-01 * Jahresbericht 2002
2003-02 Jürgen Giesl / René Thiemann: Size-Change Termination for Term Rewriting
2003-03 Jürgen Giesl / Deepak Kapur: Deciding Inductive Validity of Equations
2003-04 Jürgen Giesl / René Thiemann / Peter Schneider-Kamp / Stephan Falke: Improving Dependency Pairs
2003-05 Christof Löding / Philipp Rohde: Solving the Sabotage Game is PSPACE-hard
2003-06 Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates
2003-07 Horst Lichter / Thomas von der Maßen / Alexander Nyßen / Thomas Weiler: Vergleich von Anstzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung
2003-08 Jürgen Giesl / René Thiemann / Peter Schneider-Kamp / Stephan Falke: Mechanizing Dependency Pairs

* These reports are only available as a printed version.
Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.