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 FIFO 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, 14].

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

* 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".

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 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 FIFO channels and global acceptance condition 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, 12] 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 [16] 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 MPAs.

Organization of the Paper. The next two sections introduce some basic notions and recall the definition of message sequence charts and (existential) monadic second-order logic. Section 4 deals with message-passing automata and their expressive equivalence to existential monadic second-order logic, while Section 5 studies the gap between monadic second-order formulas and their existential fragment.

Acknowledgment. We would like to thank Dietrich Kuske for valuable remarks and pointing out some innaccuracies in a previous version of this paper. We also thank the anonymous referees for their helpful suggestions and comments.

2 Message Sequence Charts

Forthcoming definitions are all made wrt. a fixed finite set $\mathcal{P}$ of at least two *processes*. (Note that, in one proof, we assume the existence of at least three processes.) We denote by Ch the set $\{(p,q) \mid p,q \in \mathcal{P},\ p \neq q\}$ 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 $p \in \mathcal{P}$, set Act_p to be the set $\{\sigma \in \mathit{Act} \mid P(\sigma) = p\}$.

For a total order $\leq$ on a finite set E, $\lessdot$ denotes the *covering relation* of $\leq$: for $e, e' \in E$, $e \lessdot e'$ if both $e < e'$ and, for any $e'' \in E$, $e < e'' \leq e'$ implies $e'' = e'$.

Definition 1 (Message Sequence Chart). *A* message sequence chart *(MSC) is a structure* $(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda)$ *such that*

- E *is a nonempty finite set of* events*,*
- $\lambda : E \to \mathit{Act}$ *is a* labeling function*,*
- $\lessdot_p$ *is the covering relation of some total order* $\leq_p$ *on* $E_p := \{e \in E \mid \lambda(e) \in \mathit{Act}_p\}$*,*
- $<_{\mathrm{c}} \subseteq E \times E$ *such that, for any* $e, e' \in E$*,* $e <_{\mathrm{c}} e'$ *iff* $(\lambda(e), \lambda(e')) \in \mathit{Com}$ *and* $|{\downarrow}e \cap \lambda^{-1}(\lambda(e))| = |{\downarrow}e' \cap \lambda^{-1}(\lambda(e'))|$ *(where, for* $e \in E$*,* ${\downarrow}e$ *is the set of events* $e' \in E_{P(\lambda(e))}$ *with* $e' \leq_{P(\lambda(e))} e$*),*
- $(<_{\mathrm{c}} \cup \bigcup_{p\in\mathcal{P}} \lessdot_p)^*$ *is a partial order, and*
- $|\lambda^{-1}(p!q)| = |\lambda^{-1}(q?p)|$ *for each* $(p,q) \in \mathit{Ch}$*.*

Thus, events on one and the same process line are totally ordered, and events on distinct process lines that communicate with each other in a FIFO manner (wrt. $<_{\mathrm{c}}$) are labeled with actions related by Com.

Given an MSC $(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda)$ and $e \in E$, $P(e)$ will serve as a shorthand for $P(\lambda(e))$. The set of MSCs is denoted by $\mathbb{MSC}$ and a subset of $\mathbb{MSC}$ is called an *MSC language*.

Henceforth, we identify a structure of any kind with its isomorphism class.

3 (Existential) Monadic Second-Order Logic

Given supplies $\mathrm{Var} = \{x, y, \ldots, x_1, x_2, \ldots\}$ of *individual variables* and $\mathrm{VAR} = \{X, Y, \ldots, X_1, X_2, \ldots\}$ of *set variables*, formulas from MSO, the set of *monadic second-order formulas* (over MSCs) are built up from the atomic formulas

$$\lambda(x) = \sigma \text{ (for } \sigma \in Act) \qquad x \in X \qquad x \lessdot_p y \text{ (for } p \in \mathcal{P}) \qquad x <_{\mathrm{c}} y \qquad x = y$$

(where $x, y \in \mathrm{Var}$ and $X \in \mathrm{VAR}$) and furthermore allow the Boolean connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$ and the quantifiers $\exists$, $\forall$, which can be applied to either kind of variable.

Let $M = (E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_{\mathrm{c}}, \lambda)$ 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 a formula φ is given by $M \models_{\mathcal{I}} \lambda(x) = \sigma$ if $\lambda(\mathcal{I}(x)) = \sigma$, $M \models_{\mathcal{I}} x \lessdot_p y$ if $\mathcal{I}(x) \lessdot_p \mathcal{I}(y)$, and $M \models_{\mathcal{I}} x <_{\mathrm{c}} y$ if $\mathcal{I}(x) <_{\mathrm{c}} \mathcal{I}(y)$, while the remaining operators are defined as usual.

For 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 *existential* 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. Let EMSO denote the class of existential MSO formulas. In general, Σ_k shall contain MSO formulas 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 $\varphi(\overline{X_1}, \ldots, \overline{X_k}, \overline{Y})$ (again, $\overline{X_i}$ and $\overline{Y}$ are blocks of second-order variables)[1].

In the following sections, we usually consider MSO *sentences*, i.e., 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 $M \models \varphi$. For a set of MSO formulas $\mathfrak{L}$, an MSC language L is called $\mathfrak{L}$-*definable* if $L = L(\varphi)$ for some sentence $\varphi \in \mathfrak{L}$. We will show in a subsequent section that the classes of Σ_k-definable languages form an infinite hierarchy when formulas are interpreted over MSCs, resuming a result by Matz and Thomas, who proved infinity of the hierarchy for grids [11]. 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. The class of MSO-definable MSC languages is denoted by $\mathcal{MSO}$, the one of EMSO-definable languages by $\mathcal{EMSO}$.

4 Message-Passing Automata and Their Expressiveness

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

[1] Note that Σ_1 and EMSO coincide.

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* $\mathcal{A} = ((\mathcal{A}_p)_{p\in\mathcal{P}}, \mathcal{D}, \overline{s}^{in}, F)$ *such that*

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

For a *global state* $\overline{s} = (s_p)_{p\in\mathcal{P}} \in \prod_{p\in\mathcal{P}} S_p$ of $\mathcal{A}$, $\overline{s}[p]$ will henceforth refer to s_p.

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 $\mathcal{A} = ((\mathcal{A}_p)_{p\in\mathcal{P}}, \mathcal{D}, \overline{s}^{in}, F)$, $\mathcal{A}_p = (S_p, \Delta_p)$, be an MPA and $M = (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda)$ be an MSC. For a function $r : E \to \bigcup_{p\in\mathcal{P}} S_p$, we define $r^- : E \to \bigcup_{p\in\mathcal{P}} S_p$ 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 $r : E \to \bigcup_{p\in\mathcal{P}} S_p$ 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
- $(r^-(e'), \lambda(e'), m((e,e')), r(e')) \in \Delta_{P(e')}$.

For $p \in \mathcal{P}$, 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 (r, m) *accepting* if $(f_p)_{p\in\mathcal{P}} \in F$.

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 $\mathcal{MPA} := \{L \subseteq \mathbb{MSC} \mid L = L(\mathcal{A})$ for some MPA $\mathcal{A}\}$ 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]. The proof can be easily adapted towards MPAs. □

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

Lemma 1. $\mathcal{MPA} \subseteq \mathcal{EMSO}$

Proof. Several instances of this problem have been considered in the literature and can be easily adapted to our setting. See [17], for example. $\square$

Corollary 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 Corollary 1 (a). Corollary 1 (b) follows from an easy reduction from the satisfiability problem. $\square$

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

Theorem 1. $\mathcal{MPA} = \mathcal{EMSO}$

The proof will be based on the concept of *graph acceptors* [16], a generalization of finite automata to labeled graphs, which are known to be expressively equivalent to existential monadic second-order logic wrt. graphs of bounded degree. We consider graph acceptors running on MSCs, thus, on structures of bounded degree[2], which makes them applicable to our setting. A graph acceptor works on a graph as follows: It first assigns to each node one of its control states and then checks if the local neighborhood of each node (incorporating the state assignment) corresponds to a pattern from a finite supply of so-called *spheres*. In our setting, such a pattern is a labeled graph. For an alphabet Σ, we assume in the following a *Σ-labeled graph* to be a nonempty and finite structure $(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda)$ of degree at most 3. In particular, λ is a mapping $E \to \Sigma$, while the edges can be considered to be $(\mathcal{P} \uplus \{c\})$-labeled. Note that an MSC is an *Act*-labeled graph, while the converse does not necessarily hold.

Let us become more concrete and let Σ and R be an alphabet and a natural, respectively. Given a Σ-labeled graph $G = (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda)$ (let in the following $\prec$ denote $<_c \cup \bigcup_{p\in\mathcal{P}} \lessdot_p$) and elements $e, e' \in E$, the *distance* $d_G(e', e)$ from e' to e is ∞ if it holds $(e, e') \notin (\prec \cup \prec^{-1})^*$ 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 $e_i \prec e_{i+1}$ or $e_{i+1} \prec e_i$ for each $i \in \{0, \ldots, k-1\}$. Sometimes, if it is clear from the context, we omit the subscript G. An *R-sphere* over Σ is a Σ-labeled graph $H = (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda, \gamma)$ together with a designated *sphere center* $\gamma \in E$ such that, for any $e \in E$, $d_H(e, \gamma) \leq R$. Two 2-spheres are shown in Figure 1 where the sphere centers are depicted as rectangles. For a Σ-labeled graph $G = (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda)$ and $e \in E$, let the *R-sphere of G around e* be given by $(E', \{\lessdot'_p\}_{p\in\mathcal{P}}, <'_c, \lambda', e)$ where $E' = \{e' \in E \mid d_G(e', e) \leq R\}$, $\lessdot'_p = \lessdot_p \cap (E' \times E')$ for each $p \in \mathcal{P}$, $<'_c = <_c \cap (E' \times E')$, and λ' is the restriction of λ to E'.

[2] Any node of the *graph* of an MSC has at most three direct neighbors.

A *graph acceptor* (over *Act*) is a structure $\mathcal{GA} = (Q, R, \mathfrak{S}, Occ)$ such that Q is a nonempty finite set of *(control) states*, $R \in \mathbb{N}$, $\mathfrak{S}$ is a finite set of R-spheres over $Act \times Q$ (as we identify isomorphic structures, we actually deal with a finite set of isomorphism classes), and Occ is a Boolean combination of *conditions* of the form "sphere $H \in \mathfrak{S}$ occurs at least $\geq n$ times" where $n \in \mathbb{N}$. A *run* of $\mathcal{GA}$ on an Act-labeled graph $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda)$ is a mapping $\rho : E \to Q$ such that, for each $e \in E$, the R-sphere of $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, (\lambda, \rho))$ around e is isomorphic to some $H \in \mathfrak{S}$. We call ρ *accepting* if it satisfies the constraints imposed by Occ. The language of $\mathcal{GA}$ wrt. a class $\mathcal{K}$ of Act-labeled graphs, denoted by $L_{\mathcal{K}}(\mathcal{GA})$, is the set of Act-labeled graphs $G \in \mathcal{K}$ on which there is an accepting run of $\mathcal{GA}$.

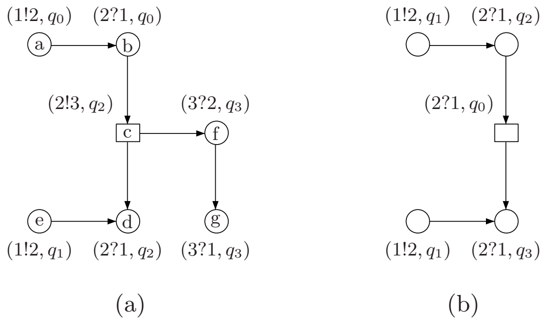

Fig. 1. The sphere(s) of a graph acceptor

The rest of this section is dedicated to the proof of Theorem 1.

Proof. It remains to show inclusion from right to left. So let φ be an EMSO sentence. We can assume the existence of a graph acceptor $\mathcal{GA}$ over Act that, running on MSCs, recognizes the MSC language defined by φ. In turn, $\mathcal{GA}$ will be translated into an MPA $\mathcal{A}$ that captures the application of $\mathcal{GA}$ to MSCs, i.e., $L(\mathcal{A}) = L_{\mathbb{MSC}}(\mathcal{GA})$. So let $\mathcal{GA} = (Q, R, \mathfrak{S}, Occ)$ be a graph acceptor over Act.

For our purpose, it suffices to consider only those R-spheres $H \in \mathfrak{S}$ for which there is an *extended* MSC $M = (E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda)$, which has an extended labeling function $\lambda : E \to Act \times Q$, and an event $e \in E$ such that H is the R-sphere of M around e. Other spheres cannot contribute to an MSC. Because, to become part of a run on some MSC M, an R-sphere has to admit an embedding into M. In this sense, the 2-sphere illustrated in Figure 1 (a) may contribute to a run on an MSC (it can be complemented by a 1!3-labeled event arranged in order between the two other events of process 1), while the 2-sphere illustrated aside is irrelevant and will be ignored in the following. This assumption is essential, as it ensures that, for each $H = (E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda, \gamma) \in \mathfrak{S}$ and $e \in E$, $d_H(e, \gamma) < R$ implies that E also contains a communication partner of e wrt. $<_c$.

In the following, we use notions that we have introduced for MSCs also for spheres $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda, \gamma)$ over $Act \times Q$, such as $P(e)$, E_p, and $\leq_p$ (to indicate the process of $e \in E$ and as abbreviations for $\lambda^{-1}(Act_p \times Q)$ and the reflexive transitive closure of $\lessdot_p$, respectively)[3]. For example, considering the 2-sphere from Figure 1 (a), $P(\mathrm{a}) = 1$, $E_1 = \{\mathrm{a}, \mathrm{e}\}$, and $\mathrm{b} \leq_2 \mathrm{d}$, but *not* $\mathrm{a} \leq_1 \mathrm{e}$. Let $\mathit{maxE} := \max\{|E| \mid (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma) \in \mathfrak{S}\}$ and let $\mathfrak{S}^+$ be the set of *extended R-spheres*, i.e., the set of structures $((E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma, e), i)$ where $(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma) \in \mathfrak{S}$, $e \in E$ is the *active node*, and $i \in \{1, \ldots, 4 \cdot \mathit{maxE}^2 + 1\}$ is the current *instance*. For $p \in \mathcal{P}$, we define $\mathfrak{S}_p := \{(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma) \in \mathfrak{S} \mid P(\gamma) = p\}$ and, furthermore, $\mathfrak{S}_p^+ := \{((E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma, e), i) \in \mathfrak{S}^+ \mid P(e) = p\}$. Finally, let $\max(\mathit{Occ})$ denote the least threshold n such that Occ does not distinguish occurrence numbers $\geq n$.

For readability, we let in the following $\prec$ denote the collection of relations $(\{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}})$ and just write $(E, \prec, \lambda, \gamma)$ instead of $(E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_{\mathrm{c}}, \lambda, \gamma)$.

The idea of the transformation 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 $((E, \prec, \lambda, \gamma, e), i) \in \mathfrak{S}^+$ with $\gamma = e$, which means that a run of $\mathcal{GA}$ assigns $(E, \prec, \lambda, \gamma)$ to the event that corresponds to s. To establish isomorphism between $(E, \prec, \lambda, \gamma)$ and the R-sphere induced by s, s transfers/obtains its obligations in form of an extended R-sphere $((E, \prec, \lambda, \gamma, e'), i)$ to/from its immediate neighbors, respectively. For example, provided e is labeled with a send action and there is $e' \in E$ with $e <_{\mathrm{c}} e'$, the message to be sent in state s will contain $((E, \prec, \lambda, \gamma, e'), i)$, which, in turn, the receiving process understands as a requirement to be satisfied. As there may be an overlapping of isomorphic R-spheres, a state can hold several instances of one and the same 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.

Let us turn to the construction of $\mathcal{A} = ((\mathcal{A}_p)_{p\in\mathcal{P}}, \mathcal{D}, \overline{s}^{in}, F)$, $\mathcal{A}_p = (S_p, \Delta_p)$, which is given as follows: For $p \in \mathcal{P}$, 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 $H \in \mathfrak{S}_p$ 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 $((E, \prec, \lambda, \gamma, e), i) \in \mathcal{S}$ with $\gamma = e$ (whose component $(E, \prec, \lambda, \gamma)$ we identify by $\varsigma(\mathcal{S})$ from now on) and
  - for any two $((E, \prec, \lambda, \gamma, e), i), ((E', \prec', \lambda', \gamma', e'), i') \in \mathcal{S}$,
    (a) $\lambda(e) = \lambda'(e') \in \mathit{Act}_p \times Q$ (so that we can assign a well-defined unique label $\lambda(\mathcal{S}) \in \mathit{Act}_p \times Q$ to $\mathcal{S}$, namely the labeling $\lambda(e)$ for some $((E, \prec, \lambda, \gamma, e), i) \in \mathcal{S}$) and
    (b) if $(E, \prec, \lambda, \gamma) \cong (E', \prec', \lambda', \gamma')$ 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 re-

[3] Note that, wrt. spheres, $\leq_p$ is not necessarily a total order.

ceiving 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. $\lambda(\mathcal{S}') = (\sigma, s)$ for some $s \in Q$.
2. For any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}$ and $e' \in E_p$, if $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{S}'$, then $e \lessdot_p e'$.
3. For any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}'$, if $\mathcal{S} \neq \emptyset$ and e is minimal in $(E_p,\leq_p)$, then $d(e,\gamma) = R$.
4. For any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}$, if e is maximal in $(E_p,\leq_p)$, then $d(e,\gamma) = R$.
5. For any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}'$, if e is not minimal in $(E_p,\leq_p)$, then we have $((E,\prec,\lambda,\gamma,e^-),i) \in \mathcal{S}$ where $e^- \in E_p$ is the unique event with $e^- \lessdot_p e$.
6. For any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}$, if e is not maximal in $(E_p,\leq_p)$, then we have $((E,\prec,\lambda,\gamma,e^+),i) \in \mathcal{S}'$ 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 $q \in \mathcal{P}$:
   (a) for any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}'$ and any $e' \in E$, if $e <_{\mathrm{c}} e'$, then we have $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{P}$,
   (b) for any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{S}'$ and any $e' \in E$, if $e \not<_{\mathrm{c}} e'$, then we have $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{N}$, and
   (c) for any $((E,\prec,\lambda,\gamma,e),i) \in \mathcal{P}$, there is $e' \in E$ such that $e' <_{\mathrm{c}} e$ and $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{S}'$.

   (ii) In case that $\sigma = p?q$ for some $q \in \mathcal{P}$:
   (a) $\mathcal{P} \subseteq \mathcal{S}'$,
   (b) $\mathcal{N} \cap \mathcal{S}' = \emptyset$, and
   (c) for any $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{S}'$, if there is $e \in E$ with $e <_{\mathrm{c}} e'$, then $((E,\prec,\lambda,\gamma,e'),i) \in \mathcal{P}$.
8. $\nu' = \nu[\varsigma(\mathcal{S}')/\min\{\nu(\varsigma(\mathcal{S}'))+1, \max(\mathit{Occ})\}]$ (i.e., ν' maps $\varsigma(\mathcal{S}')$ to the minimum of $\nu(\varsigma(\mathcal{S}'))+1$ and $\max(\mathit{Occ})$ and, otherwise, coincides with ν).

Thus, Condition 1. guarantees that any state within a run has the same labeling as the event it is assigned to. Condition 2. makes sure that, whenever there is a $\lessdot_p$-edge in the input MSC, then there is a corresponding edge in the extended sphere that is passed from the source to the target state of the corresponding transition. Conversely, if there is no $\lessdot_p$-edge between two nodes in the extended sphere, then it must not be passed directly to impose the same behavior on the MSC, i.e., the corresponding events in the MSC must not touch each other. Conditions 3. and, dually, 4. make sure that a sphere that does not make use of the whole radius R is employed in the initial or final phase of a run only. By Conditions 5. and 6., extended spheres must be passed along a process line as far as possible, hereby starting in a minimal and ending in a maximal active node. Condition 7. ensures the corresponding beyond process lines, i.e., for messages. Finally, Condition 8. guarantees that the second component of each state correctly keeps track the number of spheres used so far.

Furthermore, $\overline{s}^{in} = ((\emptyset,\nu_p^0))_{p\in\mathcal{P}}$ and, for $(\mathcal{S}_p,\nu_p) \in S_p$, $((\mathcal{S}_p,\nu_p))_{p\in\mathcal{P}} \in F$ if the union of mappings ν_p satisfies the requirements imposed by Occ and, for all

$p \in \mathcal{P}$ and $((E, \prec, \lambda, \gamma, e), i) \in \mathcal{S}_p$, e is maximal in $(E_p, \leq_p)$. In fact, it holds $L(\mathcal{A}) = L_{\mathbb{MSC}}(\mathcal{GA})$.

Let $\rho : \widetilde{E} \to Q$ be an accepting run of $\mathcal{GA}$ on $M = (\widetilde{E}, \{\widetilde{\lessdot}_p\}_{p\in\mathcal{P}}, \widetilde{<}_c, \widetilde{\lambda}) \in \mathbb{MSC}$ and let $\widehat{\rho}$ denote the mapping $\widetilde{E} \to \mathfrak{S}$ that maps an event $e \in \widetilde{E}$ onto the R-sphere of $(\widetilde{E}, \{\widetilde{\lessdot}_p\}_{p\in\mathcal{P}}, \widetilde{<}_c, (\widetilde{\lambda}, \rho))$ around e. In an accepting run (r, m) of $\mathcal{A}$ on M, r basically assigns to an event $e \in \widetilde{E}$—apart from the obvious mapping ν—the set of those extended spheres $((E, \prec, \lambda, \gamma, e_0), i) \in \mathfrak{S}^+$ such that there is an event $e' \in \widetilde{E}$ with both $d_M(e', e) \leq R$ and $(E, \prec, \lambda, \gamma, e_0)$ is isomorphic to $(\widehat{\rho}(e'), e)$. Hereby, $maxE$ is sufficiently large to guarantee an instance labeling that is consistent with the transition relation of $\mathcal{A}$. If we suppose $m : \widetilde{<}_c \to \mathcal{D}$ to map a pair $(e_s, e_r) \in \widetilde{<}_c$ onto $(\mathcal{P}, \mathcal{N})$ where (set $(\mathcal{S}, \nu)$ to be $r(e_s)$) $\mathcal{P} = \{((E, \prec, \lambda, \gamma, e_0'), i) \in \mathfrak{S}^+ \mid$ there is $e_0 \in E$ with $((E, \prec, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 <_c e_0'\}$ and $\mathcal{N} = \{((E, \prec, \lambda, \gamma, e_0'), i) \in \mathfrak{S}^+ \mid$ there is $e_0 \in E$ such that $((E, \prec, \lambda, \gamma, e_0), i) \in \mathcal{S}$ and $e_0 \not<_c e_0'\}$, (r, m) is an accepting run of $\mathcal{A}$ on M.

Conversely, let (r, m) be an accepting run of $\mathcal{A}$ on $M = (E, \{\lessdot_p\}_{p\in\mathcal{P}}, <_c, \lambda) \in \mathbb{MSC}$. If we define $\rho : E \to Q$ to map an event $e \in E$ to the control state that is associated with the sphere center of $\varsigma(\mathcal{S})$ where $r(e) = (\mathcal{S}, \nu)$ for some ν, then ρ turns out to be an accepting run of $\mathcal{GA}$ on M. $\square$

Example 1. In the following, let H denote the 2-sphere from Figure 1 (a). Figure 2, showing some MSC M with four processes, illustrates the transition behavior of the MPA $\mathcal{A}$ from the above proof. It demonstrates how a run of $\mathcal{A}$ on M transfers extensions of H from one event of M to a neighboring one to make sure that the 2-sphere around event e_{c} (which is indicated by solid edges) is isomorphic to H. For example, the state that is taken on event e_{a} may contain the extended sphere (H, a). (For clarity, control states and the natural i to distinguish different instances of spheres are omitted.) As $\mathrm{a} <_{\mathrm{c}} \mathrm{b}$ (wrt. the edge relation of H), $\mathcal{A}$ passes (H, b) in form of a message to process 2. Receiving (H, b), process 2 becomes aware it should bind e_{b} to some state that contains (H, b) (conditions 7. (i) (a) and 7. (ii) (a) from the definition of the transition relation). As, in H, b is followed by c, so e_{c} has to be associated with a state containing (H, c) (condition 6.). In contrast, e_{h} is not allowed to carry the extended sphere (H, e), unless it belongs to a different instance of H (condition 2.). Now consider e_{d}, which holds the extended sphere (H, d). Due to condition 5., the preceding state, which is associated to e_{c}, must contain (H, c), which means that a run cannot simply enter H beginning with d. Moreover, as e_{d} is a receive event, $\mathcal{A}$ has to receive a message containing (H, d) (condition 7. (ii) (c)). In turn, the corresponding send event e_{e} has to be associated with a state that holds (H, e) (condition 7. (i) (c)). Note that, as $d(a, c) = d(e, c) = 2$, the (illustrated parts of the) states assigned to e_{a} and e_{e} satisfy conditions 3. and 4.

5 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

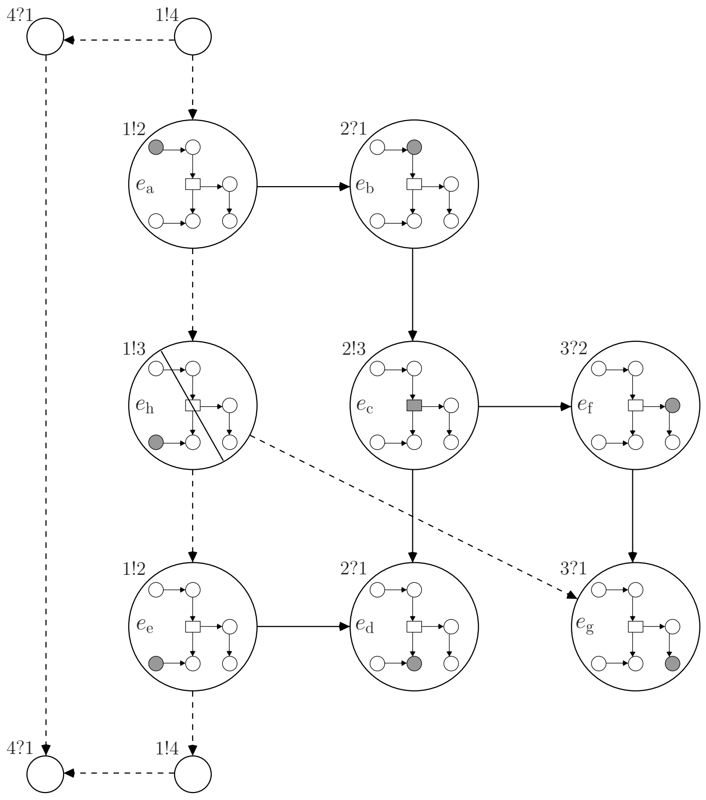

Fig. 2. Simulating a graph acceptor

to show that MPAs cannot be complemented in general. More specifically, we show that quantifier alternation forms a hierarchy:

Theorem 2. *The monadic quantifier-alternation hierarchy over* $\mathbb{MSC}$ *is infinite.*

Proof. Matz and Thomas proved infinity of the monadic quantifier-alternation hierarchy over *grids* [11, 16]. Using an idea from [15], we show how grids can be encoded into MSCs and then rewrite their result in terms of MSCs adapting their proof to our setting.

For a positive natural $n \in \mathbb{N}_{\geq 1}$, we use $[n]$ as a shorthand for $\{1, \ldots, n\}$. Given $n, m \in \mathbb{N}_{\geq 1}$, the (n, m)-*grid* (with n rows and m columns) is the structure $g(n, m) := ([n] \times [m], S_1, S_2)$ where $S_1, S_2 \subseteq ([n] \times [m])^2$ contain the pairs $((i, j), (i + 1, j)) \in ([n] \times [m])^2$ and $((i, j), (i, j + 1)) \in ([n] \times [m])^2$, respectively. A relation $R \subseteq \mathbb{N}_{\geq 1} \times \mathbb{N}_{\geq 1}$ may be represented by the grid language $\{g(n, m) \mid (n, m) \in R\}$. 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* $\mathcal{G}(f)$ of f to be the set $\{g(n, f(n)) \mid n \in \mathbb{N}_{\geq 1}\}$. A grid $g(n, m)$ can be folded to an MSC $M(n, m)$ as exemplarily shown for $g(3, 5)$ in Figure 3.

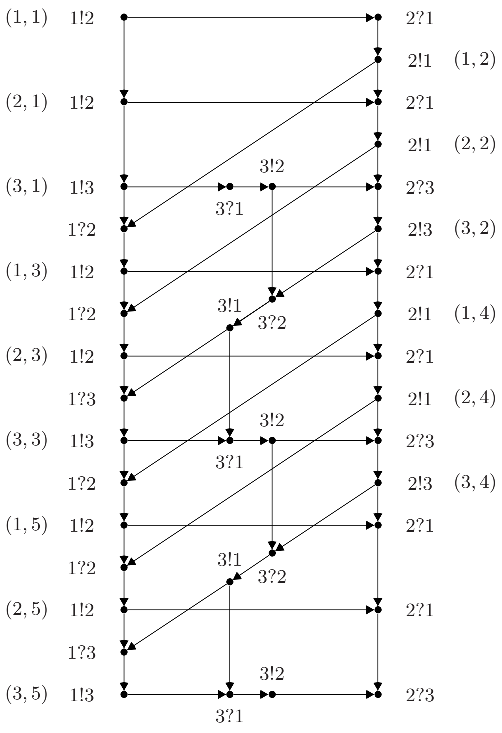

Fig. 3. Folding the $(3, 5)$-grid

A similar encoding is used by Kuske to prove infinity of the monadic quantifier-alternation hierarchy for certain *pomsets* over at least two processes [8]. However, we introduce a third process to obtain distinguished labelings of events that mark the end of a column in the grid to be encoded, which is signalized by sending a message to process 3. By the type of an event, we furthermore recognize which events really correspond to a node of the grid, namely those that are labeled with a send action performed by process 1 or 2.

A grid language $\mathcal{G}$ defines the MSC language $L(\mathcal{G}) := \{M(n, m) \mid g(n, m) \in \mathcal{G}\}$. 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 $L(\mathcal{G}(f))$. We now closely follow [16], which resumes the result of [11]. 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 1. For each $k \in \mathbb{N}$, the MSC language $L(f_k)$ is Σ_{2k+3}-definable.

Proof of Claim 1. It is easy to prove that the set of possible grid foldings is EMSO-definable (or, equivalently, the language of some MPA). As, furthermore, a grid is interpretable in a grid folding by first-order formulas, we can show that, for any $k \geq 1$, if a grid language $\mathcal{G}$ is Σ_k-definable (over grids), then $L(\mathcal{G})$ is Σ_k-

definable (over MSCs). The claim follows from the fact that any grid language $\mathcal{G}(f_k)$ is Σ_{2k+3}-definable [16].

Claim 2. Let $f : \mathbb{N}_{\geq 1} \to \mathbb{N}_{\geq 1}$ 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 2. Let $k \geq 1$ and let in the following the events of an MSC $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda)$ be labeled with elements from $Act \times \{0,1\}^i$ for some $i \in \mathbb{N}_{\geq 1}$, i.e., $\lambda : E \to Act \times \{0,1\}^i$. 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 $M(3,5)$ as shown in Figure 3, $\mathcal{A}_3$ first reads the *letter* $[(1!2)^2(1!3)(3?1)(3!2)]$ (recall that each action is still provided with an extra labeling, 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 $\exists \overline{X_k} \forall \overline{X_{k-1}} \ldots \exists/\forall \overline{X_1} \psi(Y_1, \ldots, Y_i, \overline{X_k}, \ldots, \overline{X_1})$ or, equivalently, $\exists \overline{X_k} \neg \exists \overline{X_{k-1}} \ldots \neg \exists \overline{X_1} \psi'(Y_1, \ldots, Y_i, \overline{X_k}, \ldots, \overline{X_1})$. We proceed by induction on k. For $k = 1$, $\varphi(Y_1, \ldots, Y_i)$ is an EMSO formula. According to [16], 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 1 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 $\mathcal{A} = ((\mathcal{A}_p)_{p \in \mathcal{P}}, \mathcal{D}, \overline{s}^{in}, F)$. The automaton $\mathcal{A}_n$ can now be obtained 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 2.

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

Corollary 2. $\mathcal{MPA} \subsetneq \mathcal{MSO}$

As $\mathcal{MPA} = \mathcal{EMSO}$, it follows that the *complement* $\overline{L} := \{M \in \mathbb{MSC} \mid M \notin L\}$ of an MSC language $L \in \mathcal{MPA}$, is not necessarily contained in $\mathcal{MPA}$, too [15]. Thus, we get the answer to an open question, which has been raised by Kuske [9].

Theorem 3. *$\mathcal{MPA}$ is not closed under complement.*

6 Discussion

Recall that we consider an MSC to be a graph, which corresponds to the view taken in [10] but is different from the one in [6, 9], who model an MSC as a labeled partial order $(E, \leq, \lambda)$. However, while the way to define an MSC immediately affects the syntax and expressivity of (fragments of) the corresponding monadic second-order logic, Theorem 3 holds independently of that modeling, for the following reason: there is a one-to-one-correspondence between an MSC structure $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda)$ and its counterpart $(E, \leq, \lambda)$ with $\leq\, = (<_c \cup \bigcup_{p \in \mathcal{P}} \lessdot_p)^*$. This correspondence carries over to MSO logic in the signature proposed in this paper. In other words, an MSO formula is satisfied by $(E, \{\lessdot_p\}_{p \in \mathcal{P}}, <_c, \lambda)$ iff it is satisfied by $(E, \leq, \lambda)$ (where a formula will be interpreted over labeled partial orders $(E, \leq, \lambda)$ of MSCs in the obvious manner). As the definition of a message-passing automaton is robust against the concrete modeling, too, Theorem 3 can be applied to any common definition of what an MSC is. However, our logic can only be considered to be the canonical (existential) monadic second-order logic if MSCs are given by their graphs.

If, for some $B \geq 1$, we restrict to B-bounded MSCs (see [6] for details), EMSO[$\lessdot_p, <_c$], MSO[$\lessdot_p, <_c$], EMSO[$\leq$], and MSO[$\leq$] coincide wrt. expressiveness. Thus, our work subsumes the work by Henriksen et al. [6].

Note that, for clarity, an MSC does not carry any information about the concrete messages to be sent. However, preceding results can be easily extended towards MSCs that are equipped with message information, as they are provided in [1, 2, 5], for example.

Let us recall the results of the previous sections: We have studied the class of MSC languages that corresponds to EMSO logic and MPAs. By means of graph acceptors, we have shown that MPAs are expressively equivalent to EMSO logic. In particular, for every EMSO sentence, there exists an equivalent MPA. Our proof is based on results by Thomas, which, in turn, refer to Hanf's Theorem. For practical applications, it would be desirable to have a simple effective transformation from (fragments of) EMSO to MPAs of reasonable complexity.

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].

It remains to discuss the relation between the nondeterministic automata model with a deterministic one in the unbounded setting. In [13, 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.

It would also be interesting to have logics that capture formalisms such as locally- and globally-synchronized HMSCs and related automata models [5].

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. P. Madhusudan. Reasoning about sequential and branching behaviours of message sequence graphs. In *ICALP 2000*, volume 2076 of *LNCS*. Springer, 2001.
11. O. Matz and W. Thomas. The monadic quantifier alternation hierarchy over graphs is infinite. In *LICS 1997*. IEEE Computer Society Press, 1997.
12. R. Morin. Recognizable sets of message sequence charts. In *STACS 2002*, volume 2285 of *LNCS*. Springer, 2002.
13. M. Mukund, K. Narayan Kumar, and M. Sohoni. Synthesizing distributed finite-state systems from MSCs. In *CONCUR 2000*, volume 1877 of *LNCS*. Springer, 2000.
14. A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *MFCS 1999*, volume 1672 of *LNCS*. Springer, 1999.
15. W. Thomas. Elements of an automata theory over partial orders. In *POMIV 1996*, volume 29 of *DIMACS*. AMS, 1996.
16. W. Thomas. Automata theory on trees and partial orders. In *TAPSOFT 1997*, volume 1214 of *LNCS*. Springer, 1997.
17. W. Thomas. Languages, automata and logic. In A. Salomaa and G. Rozenberg, editors, *Handbook of Formal Languages*, volume 3, Beyond Words. Springer, Berlin, 1997.