Safe realizability of high-level message sequence charts [⋆]

Markus Lohrey

Institut für Informatik, Universität Stuttgart,
Breitwiesenstr. 20-22, 70565 Stuttgart, Germany
`lohrey@informatik.uni-stuttgart.de`

Abstract. We study the notion of safe realizability for high-level message sequence charts (HMSCs), which was introduced in [2]. We prove that safe realizability is EXPSPACE-complete for bounded HMSCs but undecidable for the class of all HMSCs. This solves two open problems from [2]. Moreover we prove that safe realizability is also EXPSPACE-complete for the larger class of transition-connected HMSCs.

1 Introduction

Message sequence charts (MSCs) are a popular visual formalism for specifying the communication of asynchronous processes, where most of the details (variables, timing constraints, etc) are abstracted away. They are part of the ITU standard [14]. *High-level message sequence charts* (HMSCs) extend MSCs by allowing iteration and non-deterministic choices. In this way infinite sets of MSCs can be described.

Due to the abstract nature of HMSCs, the question of realizability (or implementability) arises: Given an HMSC (the specification), is it possible to implement it as a communicating protocol (the implementation), which shows the same behaviour as the original HMSC? This is a highly nontrivial problem, properties like for instance non-local choices in HMSCs [4] may constitute nontrivial obstacles for obtaining realizations, see e.g. [10].

Concerning the formal definition of realizability, we follow Alur et al [1, 2], which define two notions of realizability: *weak realizability* and *safe realizability*. Both are based on the model of *communicating finite state machines* (CFMs) with FIFO-queues for describing the implementation. CFMs appeared as one of the earliest abstract models for concurrent systems [5, 18], and are used for instance in the specification language SDL [13]. An accepting run of a CFM generates in a canonical way an MSC. Thus, in [2] an HMSC H is called weakly realizable, if there exists a CFM $\mathcal{A}$ such that the set of all MSCs generated by the accepting runs of $\mathcal{A}$ is precisely the set of MSCs defined by H. In practice, such an implementation may be considered as being too weak. A very desirable further property of the implementation $\mathcal{A}$ is *deadlock-freeness*: every partial run

[⋆] This work was done while the author was on leave at IRISA, Campus de Beaulieu, 35042 Rennes, France and supported by the INRIA cooperative research action FISC.

of $\mathcal{A}$ can be completed to a run that terminates in a final state of $\mathcal{A}$. Thus, in [2] an HMSC H is called safely realizable, if there exists a *deadlock-free* CFM $\mathcal{A}$ such that the set of all MSCs generated by the accepting runs of $\mathcal{A}$ is precisely the set of MSCs defined by H.

In [2] it is shown that weak realizability is already undecidable for *bounded HMSCs*, a class of HMSCs, which was introduced in [3, 16] because of its nice model-checking properties. As shown in [15], FIFO communication is crucial for this result: for non-FIFO communication weak realizability is decidable for bounded HMSCs. Concerning safe realizability, Alur et al prove in [2] that for bounded HSMCs safe realizability is in EXPSPACE but PSPACE-hard, but the exact complexity remained open. In Section 3.1, we will prove that safe realizability is in fact EXPSPACE-complete for bounded HMSCs. Moreover, using the same technique, we will also prove that safe realizability is undecidable for the class of all HMSCs, which solves the second open problem from [2]. Finally, in Section 3.2, we will establish EXPSPACE-completeness of safe realizability also for the class of *transition-connected HMSCs* [8, 15]. This class strictly contains the class of bounded HSMC but shares many of the nice properties of the latter.

Let us also remark that the notion of realizability used in this paper is a quite strict one in the sense that it allows on the implementation-side neither the introduction of new messages nor the addition of further content to already existing messages. More liberal realizations that allow the latter were studied in [8]. Another approach to realization based on Petri nets is studied in [6].

Proofs that are omitted in this extended abstract will appear in the full version of this paper.

2 Preliminaries

For complexity results we will use standard classes like PSPACE (polynomial space) and EXPSPACE (exponential space), see [17] for definitions.

Let Σ be an alphabet of symbols and $\Gamma \subseteq \Sigma$. We denote with $\pi_\Gamma : \Sigma^* \to \Gamma^*$ the projection morphism onto the subalphabet Γ. The empty word is denoted by ϵ. The length of the word $w \in \Sigma^*$ is $|w|$. For $k \in \mathbb{N}$ let $w[1, k]$ be the prefix of w of length $\min\{k, |w|\}$. For $u, v \in \Sigma^*$ we write $u \sqsubseteq v$, if u is a prefix of v.

A *pomset* is a labeled partial order $\mathcal{P} = (A, \lambda, \prec)$, i.e., $(A, \prec)$ is a partial order and $\lambda : A \to \Sigma$ is a labeling function. For $B \subseteq A$ we define the restricted pomset $\mathcal{P}\restriction_B = (B, \lambda\restriction_B, \prec\restriction_B)$. A word $\lambda(a_1)\lambda(a_2)\cdots\lambda(a_n) \in \Sigma^*$ is a *linearization* of $\mathcal{P}$ if $A = \{a_1, a_2, \ldots, a_n\}$, $a_i \neq a_j$ for $i \neq j$, and $a_i \prec a_j$ implies $i < j$ for all i, j. With $\mathrm{lin}(\mathcal{P}) \subseteq \Sigma^*$ we denote the set of all linearizations of $\mathcal{P}$.

For this paper, we use some basic notions from trace theory, see [7] for more details. An *independence relation* on the alphabet Σ is a symmetric and irreflexive relation $I \subseteq \Sigma \times \Sigma$. The complementary relation $(\Sigma \times \Sigma) \backslash I$ is also called a *dependence relation.* On Σ^* we define the equivalence relation $\equiv_I$ as the transitive reflexive closure of the symmetric relation $\{(uabv, ubav) \mid u, v \in \Sigma^*, (a, b) \in I\}$. The *I-closure* of $L \subseteq \Sigma^*$ is $[L]_I = \{v \in \Sigma^* \mid \exists u \in L : u \equiv_I v\} \subseteq \Sigma^*$. Let $\mathcal{A}$ be a finite automaton over the alphabet Σ, and assume that $\to \subseteq Q \times \Sigma \times Q$ is the transition relation of $\mathcal{A}$. Then $\mathcal{A}$ is called *loop-connected with respect to I*, if for every loop $q_1 \xrightarrow{a_1} q_2 \xrightarrow{a_2} \cdots \xrightarrow{a_{n-1}} q_n \xrightarrow{a_n} q_1$ of $\mathcal{A}$, the set $\{a_1, \ldots, a_n\} \subseteq \Sigma$ induces a connected subgraph of $(\Sigma, (\Sigma \times \Sigma)\backslash I)$. For a loop connected automaton $\mathcal{A}$, one can construct an automaton $\mathcal{A}'$ of size bounded exponentially in the size of $\mathcal{A}$ such that $L(\mathcal{A}') = [L(\mathcal{A})]_I$ [16]. In general, this exponential blow-up cannot be avoided, see [16] for an example.

2.1 Message sequence charts

For the rest of this paper let P be a finite set of *processes* ($|P| \geq 2$) and $\mathfrak{C}$ be a finite set of *message contents*. With $\mathrm{Ch} = \{(p,q) \in P \times P \mid p \neq q\}$ we denote the set of all *channels*. With $\Sigma_p = \{p!q(c), p?q(c) \mid q \in P\backslash\{p\}, c \in \mathfrak{C}\}$ we denote the set of all *types of process* $p \in P$. The set of all *types* is $\Sigma = \bigcup_{p \in P} \Sigma_p$. With $p!q(c)$ we denote the type of an event that sends from process p a message with content c to process q, whereas $p?q(c)$ denotes the type of an event that receives on process p a message with content c from process q. A *partial message sequence chart (pMSC)* over P and $\mathfrak{C}$ is a tuple $M = (E, t, m, \prec)$, where:

- E is a finite set of *events*.
- $t : E \to \Sigma$ labels each event with its type. The set of events *located on process* $p \in P$ is $E_p = t^{-1}(\Sigma_p)$. Let $E_! = \{e \in E \mid \exists p, q \in P, c \in \mathfrak{C} : t(e) = p!q(c)\}$ be the set of *send events* and $E_? = E\backslash E_!$ be the set of *receive events*.
- $m : D \to E_?$ is a bijection between a subset $D \subseteq E_!$ of the send events and the receive events such that $m(s) = r$ and $t(s) = p!q(c)$ implies $t(r) = q?p(c)$. In this case we also say that (s, r) is a *message* from process p to q with content c. If $s \in E_!\backslash D$, then s is called an *unmatched send event* in M from p to q.
- $\prec$ is a partial order on E, called the *visual order of* M, such that for every $p \in P$, the restriction of $\prec$ to E_p is a total order, and $\prec$ is equal to the transitive closure of

$$\{(e_1, e_2) \mid e_1 \prec e_2,\ \exists p \in P : e_1, e_2 \in E_p\} \ \cup\ \{(s, m(s)) \mid s \in D\}.$$

Often pMSCs are further restricted to satisfy the *FIFO-condition*, which means that for all $s_1, s_2 \in E_!$, if $s_1 \prec s_2$, $t(s_1) = p!q(c)$, $t(s_2) = p!q(d)$, and $s_2 \in D$, then also $s_1 \in D$ and $m(s_1) \prec m(s_2)$, i.e., message overtaking on any channel is disallowed. *For the main part of this paper we always assume the FIFO-restriction without mention it explicitly*, only in Section 4 we briefly discuss the non-FIFO case. The pMSC definition may also include local actions, however this is not important in the present setting. We use the usual graphical representation of pMSCs, where time flows top-down and processes are drawn as vertical lines.

Let $M = (E, t, m, \prec)$ be a pMSC, where $m : D \to E_?$ for $D \subseteq E_!$. We also write $E(M) = E$. We identify M with the pomset $(E, t, \prec)$, and we identify pMSCs if they are isomorphic as pomsets. If $D = E_!$, i.e., if there are no unmatched send events, then M is called a *message sequence chart* (MSC) over P and $\mathfrak{C}$. With $\mathrm{p}\mathbb{MSC}_{P,\mathfrak{C}}$ (resp. $\mathbb{MSC}_{P,\mathfrak{C}}$) we denote the set of all pMSCs (resp. MSCs) over P and $\mathfrak{C}$. In the sequel, we will omit the subscripts P and

$\mathfrak{C}$, if they are clear from the context. Let $|M| = |E|$ denote the *size of* M. Let $P(M) = \{p \in P \mid E_p \neq \emptyset\}$, more generally, let $P(M\restriction_F) = \{p \in P \mid E_p \cap F \neq \emptyset\}$ for $F \subseteq E$. The *communication graph* $G(M)$ of M is defined as the directed graph $G(M) = (P(M), \mapsto)$, where $p \mapsto q$ if and only if there exists in M a message from p to q (with arbitrary content). For $p \in P$ let $\pi_p(M) = \pi_{\Sigma_p}(w)$, where $w \in \text{lin}(M)$ is chosen arbitrarily (the actual choice of $w \in \text{lin}(M)$ is irrelevant).

Let $M_i = (E_i, t_i, m_i, \prec_i)$, $i = 1, 2$, be two pMSCs over P and $\mathfrak{C}$ such that $E_1 \cap E_2 = \emptyset$ and for all $(p, q) \in \text{Ch}$, if there is an unmatched send event from p to q in M_1, then there is no message from p to q in M_2 (there may be unmatched sends from p to q in M_2). Then the *concatenation* of M_1 and M_2 is the pMSC $M_1 \cdot M_2 = (E_1 \cup E_2,\ t_1 \cup t_2,\ m_1 \cup m_2,\ \prec)$, where $\prec$ is the transitive closure of

$$\prec_1 \cup \prec_2 \cup \{(e_1, e_2) \in E_1 \times E_2 \mid \exists p \in P : e_1 \text{ and } e_2 \text{ are located on process } p\}.$$

For the case that $M_1, M_2 \in \mathbb{MSC}$ this corresponds to the usual definition of MSC-concatenation. Note that concatenation is only partially defined on p$\mathbb{MSC}$.

Let $F \subseteq E(M)$ be an arbitrary set of events of the pMSC M. Note that the pomset $N = M\restriction_F$ is in general not a pMSC. On the other hand, if F is *downward-closed*, i.e., $e \prec f \in F$ implies $e \in F$, then $N = M\restriction_F$ it is again a pMSC over P and $\mathfrak{C}$. We write $N \leq M$ in this case, this defines a partial order (p$\mathbb{MSC}, \leq$) on the set of pMSCs. The pomset $M\restriction_{E\setminus F}$ will be denoted by $M\setminus N$. In general, $M\setminus N$ is not a pMSC. On the other hand, if a send event $s \in F$ is unmatched in M whenever it is unmatched in N (i.e., no message arrows are crossing from F to its complement $E\setminus F$, this happens in particular if N is an MSC), then $M\setminus N \in$ p$\mathbb{MSC}$ and moreover $M = N \cdot (M\setminus N)$.

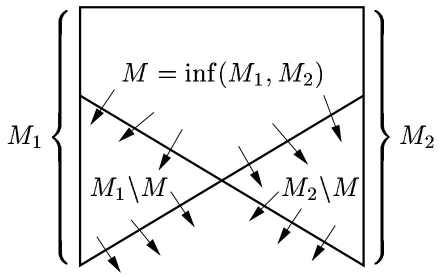
We say that an MSC $M \in \mathbb{MSC}$ is *atomic* if M cannot be written as $M = M_1 \cdot M_2$ for MSCs $M_1, M_2 \in \mathbb{MSC}\setminus\{\emptyset\}$, where $\emptyset$ stands for the MSC with an empty set of events. With $\mathbb{A}_{P,\mathfrak{C}}$ (briefly $\mathbb{A}$) we denote the set of atomic MSCs over P and $\mathfrak{C}$. Already for $|P| = 2$, the set $\mathbb{A}$ is easily seen to be infinite, see e.g. [9, Sec. 3] for an example. On $\mathbb{A}$ we define an independence relation $\mathcal{I}$ by $(A, B) \in \mathcal{I}$ if $P(A) \cap P(B) = \emptyset$. Obviously, every $M \in \mathbb{MSC}$ can be written as $M = A_1 \cdot A_2 \cdots A_m$, where $A_i \in \mathbb{A}$. Furthermore, this factorization is unique up to $\mathcal{I}$-commutations, a fact which will be crucial in Section 3.2, see [11, 15]:

Lemma 1. *If $A_1, \ldots, A_m, B_1, \ldots, B_n \in \mathbb{A}$ are such that the MSCs $A_1 \cdot A_2 \cdots A_m$ and $B_1 \cdot B_2 \cdots B_n$ are equal then the words $A_1 A_2 \cdots A_n$, $B_1 B_2 \cdots B_m \in \mathbb{A}^*$ satisfy $A_1 A_2 \cdots A_n \equiv_{\mathcal{I}} B_1 B_2 \cdots B_m$.*

The *supremum* (resp. *infimum*) of two pMSCs $M_1, M_2 \in$ p$\mathbb{MSC}$ in the partial order (p$\mathbb{MSC}, \leq$) is denoted by $\sup(M_1, M_2)$ (resp. $\inf(M_1, M_2)$). In general, $\sup(M_1, M_2)$ does not exist:

Lemma 2. *Let $M_1, M_2 \in$ p$\mathbb{MSC}$. Then $\sup(M_1, M_2)$ exists if and only if for all $p \in P$, either $\pi_p(M_1) \sqsubseteq \pi_p(M_2)$ or $\pi_p(M_2) \sqsubseteq \pi_p(M_1)$. Moreover, if $\sup(M_1, M_2)$ exists, then $\inf(M_1, M_2) = \emptyset$ if and only if $P(M_1) \cap P(M_2) = \emptyset$.*

The following picture visualizes the general situation. Arrows that are leafing some region correspond to unmatched sends, and the whole region corresponds to the supremum.



The ITU standard Z.120 defines *high-level message sequence charts (HMSCs)* as finite transition systems with nodes labeled by MSCs. Here we prefer to label edges by MSCs, which does not change the expressive power of HMSCs. Thus, an HMSC H over P and $\mathfrak{C}$ is a tuple $H = (V, \to, v_0, F)$, where V is a finite set of nodes, $\to \subseteq V \times \mathbb{MSC}_{P,\mathfrak{C}} \times V$ is a finite set of labeled edges, $v_0 \in V$ is the initial state, and $F \subseteq V$ is the set of final nodes. Instead of $(u, M, v) \in \to$, we write $u \xrightarrow{M}_H v$. The MSC-language $\mathrm{msc}(H)$ defined by H is the set of all MSCs $M_1 \cdot M_2 \cdots M_n$, where $v_0 \xrightarrow{M_1}_H v_1 \xrightarrow{M_2}_H \cdots \xrightarrow{M_n}_H v_n \in F$ for some $v_1, \ldots, v_n \in V$. We impose the restriction that for every node $v \in V$, v is accessible from the initial node v_0, and some node from F is accessible from v. Furthermore, we assume that $\to \subseteq V \times \mathbb{A}_{P,\mathfrak{C}} \times V$. Both of these assumptions do not change the expressiveness of HMSCs and can be easily established by polynomial time constructions. Let $\mathbb{A}_H = \{A \in \mathbb{A} \mid \exists u, v \in V : u \xrightarrow{A}_H v\}$. We may view H also as a finite automaton over the alphabet $\mathbb{A}_H$ of atoms, which accepts a set $L(H) \subseteq \mathbb{A}_H^*$ of words over $\mathbb{A}_H$. We will denote this automaton by H as well. An HMSC H is called *bounded* [3, 16] if for every cycle

$$v_1 \xrightarrow{A_1}_H v_2 \xrightarrow{A_2}_H \cdots \xrightarrow{A_{n-1}}_H v_n \xrightarrow{A_n}_H v_1,$$

the communication graph $G(A_1 \cdot A_2 \cdots A_n)$ is strongly connected (recall that the set of nodes of $G(M)$ is $P(M)$). In [3] it is shown that for a bounded HMSC H the language $\mathrm{lin}(\mathrm{msc}(H)) \subseteq \Sigma^*$ of all linearizations of MSCs generated by H is regular. We say that H is *transition-connected* [8] if H, viewed as a finite automaton over the alphabet $\mathbb{A}_H$, is loop-connected with respect to the independence relation $\mathcal{I} \subseteq \mathbb{A} \times \mathbb{A}$. It is easy to see that every bounded HMSC is transition-connected. Finally, H is called *$\mathcal{I}$-closed* if H, viewed as a finite automaton over $\mathbb{A}_H$, satisfies $L(H) = [L(H)]_{\mathcal{I}}$. Thus, by [16], for a transition-connected HMSC H there exists an $\mathcal{I}$-closed HMSC H' of size bounded exponentially in the size of H such that $L(H') = [L(H)]_{\mathcal{I}}$ and thus, $\mathrm{msc}(H) = \mathrm{msc}(H')$.

2.2 Communicating finite state machines

In this section we briefly introduce *communicating finite state machines* (CFMs) The tight relationship between CFMs and the theory of MSCs is well-known, see e.g. [12].

The set of *buffer configurations* is the set $(\mathfrak{C}^*)^{\mathrm{Ch}}$ of all functions from the set of channels Ch to $\mathfrak{C}^*$. The buffer configuration $\mathcal{B} \in (\mathfrak{C}^*)^{\mathrm{Ch}}$ such that $\mathcal{B}(p,q) = \epsilon$ for all $(p,q) \in \mathrm{Ch}$ is denoted by $\mathcal{B}_\emptyset$. A CFM over P and $\mathfrak{C}$ is a tuple $\mathcal{A} = (\mathcal{A}_p)_{p \in P}$, where $\mathcal{A}_p = (S_p, \Sigma_p, \delta_p, s_{0,p}, F_p)$ is a not necessarily finite automaton over the alphabet Σ_p of types of process p. The set of states of $\mathcal{A}_p$ is S_p, the transition relation of $\mathcal{A}_p$ is $\delta_p \subseteq S_p \times \Sigma_p \times S_p$, the initial state is $s_{0,p} \in S_p$, and the set of final states is $F_p \subseteq S_p$. We will always assume w.l.o.g. that every local automaton $\mathcal{A}_p$ is *reduced*, i.e., for every $s \in S_p$, s is accessible from the initial state $s_{0,p}$ and some state from F_p is accessible from s. The infinite set $\mathbf{S}$ of *global states of* $\mathcal{A}$ and the set $\mathbf{F}$ of *final states of* $\mathcal{A}$ are defined by

$$\mathbf{S} = \prod_{p \in P} S_p \times (\mathfrak{C}^*)^{\mathrm{Ch}} \quad \text{and} \quad \mathbf{F} = \prod_{p \in P} F_p \times \{\mathcal{B}_\emptyset\}.$$

The *initial state of* $\mathcal{A}$ is $(\mathbf{s}_0, \mathcal{B}_\emptyset)$, where $\mathbf{s}_0 = (s_{0,p})_{p \in P}$. The *global transition relation* δ *of* $\mathcal{A}$ is defined as follows: Let $(\mathbf{s}, \mathcal{B}) \in \mathbf{S}$, where $\mathbf{s} = (s_p)_{p \in P}$. If $(s_i,\ i!j(c),\ t) \in \delta_i$ then $((\mathbf{s},\mathcal{B}),\ i!j(c),\ (\mathbf{t},\mathcal{C})) \in \delta$, where $\mathbf{t} = (t_p)_{p \in P}$, $t_p = s_p$ for $p \neq i$, $t_i = t$, $\mathcal{C}(p,q) = \mathcal{B}(p,q)$ for $(p,q) \neq (i,j)$, and $\mathcal{C}(i,j) = c\mathcal{B}(i,j)$. On the other hand, if $(s_i,\ i?j(c),\ t) \in \delta_i$ and $\mathcal{B}(j,i) = wc$ for some $w \in \mathfrak{C}^*$, then $((\mathbf{s},\mathcal{B}),\ i?j(c),\ (\mathbf{t},\mathcal{C})) \in \delta$, where $\mathbf{t} = (t_p)_{p \in P}$, $t_p = s_p$ for $p \neq i$, $t_i = t$, $\mathcal{C}(q,p) = \mathcal{B}(q,p)$ for $(q,p) \neq (j,i)$, and $\mathcal{C}(j,i) = w$. We extend the relation $\delta \subseteq \mathbf{S} \times \Sigma \times \mathbf{S}$ in the usual way to a relation $\delta \subseteq \mathbf{S} \times \Sigma^* \times \mathbf{S}$. Instead of $((\mathbf{s},\mathcal{B}), w, (\mathbf{t},\mathcal{C})) \in \delta$, $w \in \Sigma^*$, we write $(\mathbf{s},\mathcal{B}) \xrightarrow{w}_{\mathcal{A}} (\mathbf{t},\mathcal{C})$. We write $(\mathbf{s},\mathcal{B}) \xrightarrow{*}_{\mathcal{A}} (\mathbf{t},\mathcal{C})$ if $(\mathbf{s},\mathcal{B}) \xrightarrow{w}_{\mathcal{A}} (\mathbf{t},\mathcal{C})$ for some $w \in \Sigma^*$. We write $(\mathbf{s},\mathcal{B}) \xrightarrow{w}_{\mathcal{A}}$ if $(\mathbf{s},\mathcal{B}) \xrightarrow{w}_{\mathcal{A}} (\mathbf{t},\mathcal{C})$ for some $(\mathbf{t},\mathcal{C})$. Let $L(\mathcal{A}) = \{w \in \Sigma^* \mid \exists (\mathbf{t},\mathcal{B}_\emptyset) \in \mathbf{F} : (\mathbf{s}_0,\mathcal{B}_\emptyset) \xrightarrow{w}_{\mathcal{A}} (\mathbf{t},\mathcal{B}_\emptyset)\}$.

It is easy to see that for every word $w \in \Sigma^*$ such that $(\mathbf{s},\mathcal{B}_\emptyset) \xrightarrow{w}_{\mathcal{A}}$ for some $\mathbf{s}$, there exists a unique pMSC $\mathrm{pmsc}(w)$ with $w \in \mathrm{lin}(\mathrm{pmsc}(w))$. Furthermore, if $(\mathbf{s},\mathcal{B}_\emptyset) \xrightarrow{w}_{\mathcal{A}} (\mathbf{t},\mathcal{B}_\emptyset)$ for some $\mathbf{s},\mathbf{t}$, then $\mathrm{pmsc}(w) \in \mathbb{MSC}$, and we write $\mathrm{msc}(w)$ instead of $\mathrm{pmsc}(w)$. Thus, we can define $\mathrm{msc}(\mathcal{A}) = \{\mathrm{msc}(w) \mid w \in L(\mathcal{A})\}$. It is also easy to see that if $w_1, w_2 \in \mathrm{lin}(M)$ for $M \in \mathrm{p}\mathbb{MSC}$, then $(\mathbf{s},\mathcal{B}_\emptyset) \xrightarrow{w_1}_{\mathcal{A}} (\mathbf{t},\mathcal{B})$ if and only if $(\mathbf{s},\mathcal{B}_\emptyset) \xrightarrow{w_2}_{\mathcal{A}} (\mathbf{t},\mathcal{B})$. Thus, we may write $(\mathbf{s},\mathcal{B}_\emptyset) \xrightarrow{M}_{\mathcal{A}} (\mathbf{t},\mathcal{B})$ in this case. Finally, we say that $\mathcal{A}$ is *deadlock-free* if for all states $(\mathbf{s},\mathcal{B}) \in \mathbf{S}$ such that $(\mathbf{s}_0,\mathcal{B}_\emptyset) \xrightarrow{*}_{\mathcal{A}} (\mathbf{s},\mathcal{B})$ we have $(\mathbf{s},\mathcal{B}) \xrightarrow{*}_{\mathcal{A}} (\mathbf{t},\mathcal{B}_\emptyset)$ for some $(\mathbf{t},\mathcal{B}_\emptyset) \in \mathbf{F}$.

3 Weak and safe realizability

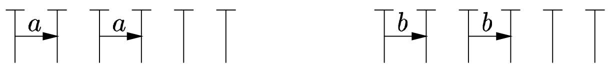
Let $L \subseteq \mathbb{MSC}_{P,\mathfrak{C}}$. Following [1], we say that L is *weakly realizable* if there exists a CFM $\mathcal{A}$ over P and $\mathfrak{C}$ such that $\mathrm{msc}(\mathcal{A}) = L$. We say that L is *safely realizable* if there exists a deadlock-free CFM $\mathcal{A}$ over P and $\mathfrak{C}$ such that $\mathrm{msc}(\mathcal{A}) = L$. These definitions allow local automata with infinite state sets, but this case will never occur in this paper, since we only consider sets of MSCs that are generated by HMSCs. An HMSC H is called *weakly realizable* (*safely realizable*) if $\mathrm{msc}(H)$ is weakly realizable (safely realizable). Given H, we can construct in polynomial time *finite* automata $\mathcal{A}_p$, $p \in P$, with $L(\mathcal{A}_p) = \pi_p(\mathrm{msc}(H))$. We call the CFM $\mathcal{A} = (\mathcal{A}_p)_{p \in P}$ the *canonical implementation* of H. Then H is weakly realizable if and only if $\mathrm{msc}(\mathcal{A}) = \mathrm{msc}(H)$ [1]. Note that the inclusion $\mathrm{msc}(H) \subseteq \mathrm{msc}(\mathcal{A})$ always holds. Furthermore, H is safely realizable if and only if $\mathcal{A}$ is deadlock-free and $\mathrm{msc}(\mathcal{A}) = \mathrm{msc}(H)$ [1]. In [1] it is also shown that $L \subseteq \mathbb{MSC}$ is weakly realizable if and only if the following closure condition CC_w (called CC2 in [1]) holds:

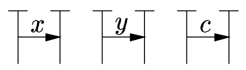
Closure condition CC_w. If $M \in \mathbb{MSC}$ is such that for all $p \in P$ there exists $N \in L$ with $\pi_p(M) = \pi_p(N)$ then $M \in L$.

Furthermore, it is claimed that $L \subseteq \mathbb{MSC}$ is safely realizable if and only if the following closure condition CC_s (called CC3 in [1]) holds:

Closure condition CC_s. If $M \in \mathrm{p}\mathbb{MSC}$ is such that for all $p \in P$ there exists $N \in L$ with $\pi_p(M) \sqsubseteq \pi_p(N)$ then $M \leq N$ for some $N \in L$.

But this is in fact false, the set L consisting of the following 6 MSCs satisfies CC_s but it does not satisfy CC_w, and hence, it is not even weakly realizable.





for all $x, y \in \{a, b\}$

On the other hand, using arguments from [1], one can easily prove

Lemma 3. *$L \subseteq \mathbb{MSC}$ is safely realizable if and only if L satisfies CC_w and CC_s.*

As already mentioned, the notions of weak and safe realizability were introduced in [1], where it was shown that for finite sets of MSCs, safe realizability can be tested in polynomial time, whereas weak realizability is coNP-complete. In [2], realizability was studied for HMSCs. It was shown that weak realizability is already undecidable for bounded HMSCs, [1] whereas safe realizability for bounded HMSCs is in EXPSPACE but PSPACE-hard. In Section 3.1, we will close this latter gap by proving that safe realizability for bounded HMSCs is EXPSPACE-complete. The proof technique used for this result will be also applied in order to show that safe realizability is undecidable for the class of all HMSCs.

3.1 Bounded HMSCs

Theorem 1. *The following problem is EXPSPACE-complete:*

INPUT: Set P of processes, set $\mathfrak{C}$ of message contents, and a bounded HMSC H over P and $\mathfrak{C}$

QUESTION: Is H safely realizable?

Furthermore, this problem is also EXPSPACE-complete if P and $\mathfrak{C}$ are fixed, i.e., do not belong to the input.

[1] For this result, FIFO-communication is important: Under non-FIFO communication, weak realizability is decidable for bounded HMSCs [15].

Proof. Membership in EXPSPACE is shown in [2] (for variable P and $\mathfrak{C}$), or follows from Theorem 5. For the lower bound we combine ideas from [2] and [16, 19]. Let $\mathcal{M}$ be a fixed Turing-machine with an EXPSPACE-complete acceptance problem (such a machine exists, take any machine, which accepts an EXPSPACE-complete language). W.l.o.g. $\mathcal{M}$ works on an input of length n in space $2^n - 1$. Let Q be the set of states of $\mathcal{M}$ and let Δ be the tape alphabet. Furthermore, let q_0 be the initial state of $\mathcal{M}$ and q_f be the final state of $\mathcal{M}$. Let $\Box \in \Delta$ be the blank symbol. The machine $\mathcal{M}$ accepts if it reaches the final state q_f. Let us fix an input $w \in \Delta^*$ for $\mathcal{M}$ with $|w| = n$ for the further discussion. Configurations of $\mathcal{M}$ are represented as a word from $\Delta^* Q \Delta^*$ of length 2^n. A sequence $(u_1, \ldots, u_m)$ of words $u_i \in \Delta^* Q \Delta^*$ is called an *accepting computation* of $\mathcal{M}$ if $u_1 = q_0 w \Box^{2^n - n - 1}$, $|u_i| = 2^n$ $(1 \leq i \leq m)$, u_{i+1} is a successor configuration of u_i with respect to $\mathcal{M}$ $(1 \leq i < m)$, and $u_m \in \Delta^* q_f \Delta^*$.

For a number $0 \leq i < 2^n$ let $\langle i \rangle \in \{0,1\}^n$ denote the binary representation of i of length n, where moreover the least significant bit is the left-most bit. For $w = a_0 \cdots a_{2^n-1}$, $a_i \in Q \cup \Delta$, let $\beta(w) = \langle 0 \rangle a_0 \cdots \langle 2^n - 1 \rangle a_{2^n-1}$. Let $\Gamma = Q \cup \Delta \cup \{0,1\}$ and define the set $\mathfrak{C}$ of message contents as $\mathfrak{C} = \Gamma \cup \{\$, \ell, r\}$. [2] We will deal with the fixed set of processes $P = \{0, \ldots, 5\}$. For a symbol $a \in \Gamma$ we define the MSC $a^{(2,1)}$ (resp. $a^{(4,5)}$) over P and $\mathfrak{C}$ as the unique MSC with the only linearization $2!1(a)\, 1?2(a)\, 1!2\, 2?1$ (resp. $4!5(a)\, 5?4(a)\, 5!4\, 4?5$), thus, the symbol a is send from 2 to 1 (resp. 4 to 5) and immediately confirmed. For $C = b_1 \cdots b_m \in \Gamma^*$ define $C^{(2,1)} = b_1^{(2,1)} \cdots b_m^{(2,1)}$ and $C^{(4,5)} = b_1^{(4,5)} \cdots b_m^{(4,5)}$. For words $C_1, D_1, \ldots, C_m, D_m \in \Gamma^*$ $(m \geq 1)$ we define the MSC $M(C_1, D_1, \ldots, C_m, D_m)$ over P and $\mathfrak{C}$ as shown in Figure 1, where the case $m = 3$ is shown (process 0 is not involved into this MSC, and hence not drawn). Finally define the following two sets of MSCs:

$$\begin{aligned}
L_\ell &= \{M(C_1, D_1, \ldots, C_m, D_m) \mid m \geq 1,\ C_1, D_1, \ldots, C_m, D_m \in \Gamma^*\} \\
L_r &= L_\ell \backslash \{M(\beta(u_1), \beta(u_1), \ldots, \beta(u_m), \beta(u_m)) \mid (u_1, \ldots, u_m) \text{ is an} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{accepting computation of } \mathcal{M}\}
\end{aligned}$$

Claim 1. There exist bounded HMSCs H_ℓ and H_r that can be constructed in time polynomial in n such that $\mathrm{msc}(H_\ell) = L_\ell$ and $\mathrm{msc}(H_r) = L_r$.

For L_ℓ this is clear, since all messages are immediately confirmed by messages back to the sending process. For L_r the construction follows [16, Prop. 7].

Claim 2. L_ℓ is safely realizable.

We will only check condition CC_w, condition CC_s can be verified analogously. Thus, assume that M is an MSC such that for each $p \in \{0, \ldots, 5\}$ there exists $N \in L_\ell$ with $\pi_p(M) = \pi_p(N)$. Thus, $\pi_3(M) = (3!2\ 3?2\ 3!4\ 3?4)^k$ for some $k \geq 1$.

[2] In the following, we will also use messages without any content, the corresponding types are written as $p!q$ and $p?q$, respectively. Formally, one can introduce an additional message content nil for these messages.

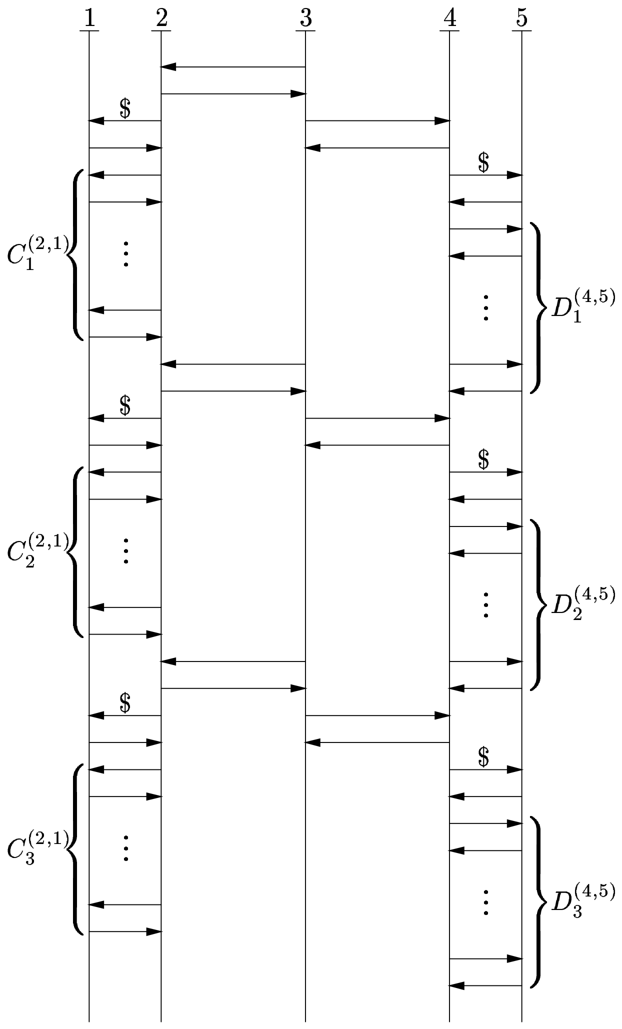


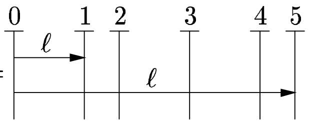
Fig. 1. $M(C_1, D_1, C_2, D_2, C_3, D_3)$

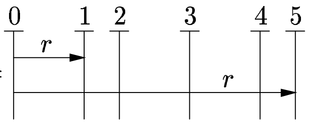
Since M is an MSC, it follows that

$$\begin{aligned}
\pi_2(M) =&(2?3\ 2!3\ 2!1(\$)\ 2?1\ 2!1(a_{1,1})\ 2?1 \cdots 2!1(a_{1,i_1})\ 2?1) \cdots \\
&(2?3\ 2!3\ 2!1(\$)\ 2?1\ 2!1(a_{k,1})\ 2?1 \cdots 2!1(a_{k,i_k})\ 2?1) \\
\pi_4(M) =&(4?3\ 4!3\ 4!5(\$)\ 4?5\ 4!5(b_{1,1})\ 4?5 \cdots 4!5(b_{1,j_1})\ 4?5) \cdots \\
&(4?3\ 4!3\ 4!5(\$)\ 4?5\ 4!5(b_{k,1})\ 4?5 \cdots 4!5(b_{k,j_k})\ 4?5) \\
\pi_1(M) =&(1?2(\$)\ 1!2\ 1?2(a_{1,1})\ 1!2 \cdots 1?2(a_{1,i_1})\ 1!2) \cdots \\
&(1?2(\$)\ 1!2\ 1?2(a_{k,1})\ 1!2 \cdots 1?2(a_{k,i_k})\ 1!2) \\
\pi_5(M) =&(5?4(\$)\ 5!4\ 5?4(b_{1,1})\ 5!4 \cdots 5?4(b_{1,j_1})\ 5!4) \cdots \\
&(5?4(\$)\ 5!4\ 5?4(b_{k,1})\ 5!4 \cdots 5?4(b_{k,j_k})\ 5!4)
\end{aligned}$$

for some $i_1, j_1, \ldots, i_k, j_k \geq 0$ and $a_{1,1}, \ldots, a_{k,i_k}, b_{1,1}, \ldots, b_{k,j_k} \in \Gamma$. Thus, $M \in L_\ell$. This proves Claim 2.

Now define the MSCs M_ℓ and M_r as follows:

$M_\ell =$ 

$M_r =$ 

From the bounded HMSCs H_ℓ and H_r in Claim 1 it is straight-forward to construct a bounded HMSC H such that $\mathrm{msc}(H) = (M_\ell \cdot L_\ell) \cup (M_r \cdot L_r)$, where concatenation is lifted to sets of MSCs in the obvious way.

Claim 3. H is safely realizable if and only if $\mathcal{M}$ does not accept the input w.

If $\mathcal{M}$ does not accept w, then $L_\ell = L_r$ and $\mathrm{msc}(H) = \{M_\ell, M_r\} \cdot L_\ell$. Since L_ℓ is safely realizable by Claim 2, also $\mathrm{msc}(H)$ is safely realizable. Now assume that $\mathcal{M}$ accepts w. Thus, there exists an accepting computation $(u_1, \ldots, u_m)$ of $\mathcal{M}$. Let $M = M(\beta(u_1), \beta(u_1), \beta(u_2), \beta(u_2), \ldots, \beta(u_m), \beta(u_m))$. Since $M \notin L_r$, we have $M_r \cdot M \notin \mathrm{msc}(H)$. On the other hand for all $p \in \{0, \ldots, 5\}$ there exists $N \in \mathrm{msc}(H)$ such that $\pi_p(M_r \cdot M) = \pi_p(N)$, for instance for $p \in \{0,1,2,3\}$ take $N = M_r \cdot M(\beta(u_1), C, \beta(u_2), \beta(u_2), \ldots, \beta(u_m), \beta(u_m))$ for some $C \neq \beta(u_1)$. Thus, $\mathrm{msc}(H)$ is not weakly realizable and hence not safely realizable. This proves Claim 3 and hence the theorem. □

By applying the reduction from the previous proof (without the use of binary counters) to a Turing-machine with an undecidable acceptance problem, we obtain the following result.

Theorem 2. *There exist fixed sets P and $\mathfrak{C}$ of processes and message contents, respectively, such that the following problem is undecidable:*
INPUT: An HMSC H over P and $\mathfrak{C}$
QUESTION: Is H safely realizable?

3.2 Transition-connected HMSCs

In [15] it is shown that weak realizability can be decided for transition-connected HMSCs if non-FIFO communication is supposed. Moreover, it is argued that the methods used in the proof of this result can be also applied in order to show that safe realizability is decidable for transition-connected HMSCs, both for FIFO and non-FIFO communication. In this section, we will prove that safe realizability is in fact EXPSPACE-complete for transition-connected HMSCs.

For the further discussion we have to introduce a few new notations. Let us fix an arbitrary HMSC $H = (V, \to, v_0, F)$ over P and $\mathfrak{C}$, which is not necessarily transition-connected. Recall that $\mathbb{A}_H = \{A \in \mathbb{A} \mid \exists u, v \in V : u \xrightarrow{A}_H v\}$. With $\langle \mathbb{A}_H \rangle$ we denote the set of all MSCs of the form $A_1 \cdot A_2 \cdots A_n$ with $A_i \in \mathbb{A}_H$ (possibly $n = 0$, i.e., $\emptyset \in \langle \mathbb{A}_H \rangle$). Let $\mathcal{A} = (\mathcal{A}_p)_{p \in P}$ be the canonical implementation of H, thus $L(\mathcal{A}_p) = \pi_p(\mathrm{msc}(H))$. Let $\mathcal{A}_p = (S_p, \Sigma_p, \delta_p, s_{0,p}, F_p)$. Recall that $\mathcal{A}$ can be constructed in polynomial time from H, in particular the size of every S_p is bounded polynomially in the size of H. Finally, following [15], let us define a finite automaton $\mathcal{A}_\emptyset = (\mathbf{S}_\emptyset, \mathbb{A}_H, \delta_\emptyset, \mathbf{s}_0, \mathbf{F}_\emptyset)$ over the alphabet of atoms $\mathbb{A}_H$, where $\mathbf{s}_0 = (s_{0,p})_{p \in P}$ is the initial state, $\mathbf{S}_\emptyset \subseteq \prod_{p \in P} S_p$ is the set of all tuples $\mathbf{s}$ such that there exists $K \in \langle \mathbb{A}_H \rangle$ with $(\mathbf{s}_0, \mathcal{B}_\emptyset) \xrightarrow{K}_{\mathcal{A}} (\mathbf{s}, \mathcal{B}_\emptyset)$, $\mathbf{F}_\emptyset = \mathbf{S}_\emptyset \cap \prod_{p \in P} F_p$, and the transition relation $\delta_\emptyset$ is defined as follows: If $\mathbf{s}, \mathbf{t} \in \mathbf{S}_\emptyset$ and $A \in \mathbb{A}_H$ then $(\mathbf{s}, A, \mathbf{t}) \in \delta_\emptyset$ if and only if $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{A}_{\mathcal{A}} (\mathbf{t}, \mathcal{B}_\emptyset)$. Notations like $\mathbf{s} \xrightarrow{A}_{\mathcal{A}_\emptyset} \mathbf{t}$ are defined as for CFMs in Section 2.2. Note that $u \equiv_{\mathcal{I}} v$ for words $u, v \in \mathbb{A}_H^*$ implies that for all $\mathbf{s}, \mathbf{t} \in \mathbf{S}_\emptyset$, $\mathbf{s} \xrightarrow{u}_{\mathcal{A}_\emptyset} \mathbf{t}$ if and only if $\mathbf{s} \xrightarrow{v}_{\mathcal{A}_\emptyset} \mathbf{t}$, in fact, $\mathcal{A}_\emptyset$ is an asynchronous automaton in the sense of [20]. Thus, by Lemma 1, for $K \in \langle \mathbb{A}_H \rangle$ and $\mathbf{s}, \mathbf{t} \in \mathbf{S}_\emptyset$ we can write $\mathbf{s} \xrightarrow{K}_{\mathcal{A}_\emptyset} \mathbf{t}$, with the obvious meaning.

The main technical result of this section is stated in the following theorem. Note that it does not restrict to transition-connected HMSCs.

Theorem 3. *The following problem is in PSPACE:*

INPUT: Set P of processes, set $\mathfrak{C}$ of message contents, and an arbitrary HMSC H over P and $\mathfrak{C}$

QUESTION: Does the canonical implementation $\mathcal{A}$ of H satisfy the following two properties: (i) $\mathcal{A}$ is deadlock-free and (ii) $\mathrm{msc}(\mathcal{A}) \subseteq \langle \mathbb{A}_H \rangle$?

Using Theorem 3, we can prove the next two results.

Theorem 4. *The following problem is PSPACE-complete:*

INPUT: Set P of processes, set $\mathfrak{C}$ of message contents, and an $\mathcal{I}$-closed HMSC H over P and $\mathfrak{C}$

QUESTION: Is H safely realizable?

Furthermore, this problem is also PSPACE-complete if P and $\mathfrak{C}$ are fixed.

Proof. For PSPACE-hardness we can use the construction from the proof of [2, Thm. 3]. In fact, the HMSC H constructed there satisfies the property that $u \xrightarrow{A}_H v \xrightarrow{B}_H w$ implies $P(A) \cap P(B) \neq \emptyset$, thus, H is $\mathcal{I}$-closed. Moreover, P and $\mathfrak{C}$ are fixed in the construction. Thus, it remains to show membership in

PSPACE. Using Theorem 3, we first verify in PSPACE whether the canonical implementation $\mathcal{A}$ of H is both deadlock-free and satisfies $\mathrm{msc}(\mathcal{A}) \subseteq \langle \mathbb{A}_H \rangle$. If this is not the case then we can reject. Thus, let us assume that $\mathcal{A}$ is deadlock-free and $\mathrm{msc}(\mathcal{A}) \subseteq \langle \mathbb{A}_H \rangle$. It remains to show that $\mathrm{msc}(\mathcal{A}) = \mathrm{msc}(H)$, where the inclusion $\mathrm{msc}(H) \subseteq \mathrm{msc}(\mathcal{A})$ is trivial. Thus, we have to check whether $\mathrm{msc}(\mathcal{A}) \subseteq \mathrm{msc}(H)$. Since $\mathrm{msc}(\mathcal{A}) \subseteq \langle \mathbb{A}_H \rangle$, this is equivalent to $\mathrm{msc}(\mathcal{A}) \cap \langle \mathbb{A}_H \rangle \subseteq \mathrm{msc}(H)$. The following argument follows [15]. First note that for all $A_1, \ldots, A_m \in \mathbb{A}_H$, we have $A_1 \cdot A_2 \cdots A_m \in \mathrm{msc}(\mathcal{A})$ if and only if the *word* $A_1 A_2 \cdots A_m \in \mathbb{A}_H^*$ belongs to $L(\mathcal{A}_\emptyset)$. Thus, we have $\mathrm{msc}(\mathcal{A}) \cap \langle \mathbb{A}_H \rangle \subseteq \mathrm{msc}(H)$ if and only if $L(\mathcal{A}_\emptyset) \subseteq [L(H)]_\mathcal{I}$ (where H is viewed as a finite automaton over the alphabet $\mathbb{A}_H$) if and only if $L(\mathcal{A}_\emptyset) \subseteq L(H)$ (H is $\mathcal{I}$-closed) if and only if $L(\mathcal{A}_\emptyset) \cap (\mathbb{A}_H^* \backslash L(H)) = \emptyset$. This can be checked in polynomial space by guessing a word in the intersection and storing only the current state of $\mathcal{A}_\emptyset$ and the current state of the automaton for $\mathbb{A}_H^* \backslash L(H)$ resulting from the subset construction for H, which is a subset of the set of states of H. □

Theorem 5. *The following problem is EXPSPACE-complete:*

INPUT: Set P of processes, set $\mathfrak{C}$ of message contents, and a transition-connected HMSC H over P and $\mathfrak{C}$

QUESTION: Is H safely realizable?

Furthermore, this problem is also EXPSPACE-complete if P and $\mathfrak{C}$ are fixed.

Proof. The lower bound follows from Theorem 1. For the upper bound we can argue as follows: By the proof of Theorem 4, we have to check whether $L(\mathcal{A}_\emptyset) \subseteq [L(H)]_\mathcal{I}$. But since H is assumed to be transition-connected, we can construct an HMSC H' of at most exponential size such that $L(H') = [L(H)]_\mathcal{I}$, and then verify $L(\mathcal{A}_\emptyset) \subseteq L(H')$ in space bounded polynomially in the size of H' (and thus, in space bounded exponentially in the size of H). □

The proof of Theorem 3 is based on the following lemma.

Lemma 4. *The following two statements are equivalent:*

(A) $\mathcal{A}$ *is deadlock-free and* $\mathrm{msc}(\mathcal{A}) \subseteq \langle \mathbb{A}_H \rangle$.

(B) $\mathcal{A}_\emptyset$ *is deadlock-free, and for all* $\mathbf{s} \in \mathbf{S}_\emptyset$ *and all* $M \in \mathrm{p}\mathbb{MSC} \backslash \{\emptyset\}$ *such that* $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{M}_\mathcal{A}$ *it holds*

$$\exists K \in \langle \mathbb{A}_H \rangle \; \exists A \in \mathbb{A}_H \left\{ \begin{array}{l} \mathbf{s} \xrightarrow{K \cdot A}_{\mathcal{A}_\emptyset}, \; P(K) \cap P(M) = \emptyset, \\ \sup(A, M) \textit{ exists and, } \inf(A, M) \neq \emptyset \end{array} \right\}. \qquad (1)$$

For the further consideration, assume that $\mathbf{s} \in \mathbf{S}_\emptyset$ and $M \in \mathrm{p}\mathbb{MSC} \backslash \{\emptyset\}$ are such that $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{M}_\mathcal{A}$ but (1) from Lemma 4 *is not satisfied* for $\mathbf{s}$ and M. Furthermore, let us assume that M is chosen such that $|M|$ is minimal. We will show that we can bound the size of M. For this we need the following lemma, which can be shown by induction on $|N|$.

Lemma 5. *Let $\mathbf{t} \in \mathbf{S}_\emptyset$ and $N \in \mathrm{p}\mathbb{MSC}$ such that $(\mathbf{t}, \mathcal{B}_\emptyset) \xrightarrow{N}_{\mathcal{A}}$ and $|N| < |M|$. Then there exist atoms $A_1, \dots, A_m \in \mathbb{A}_H$ and non-empty prefixes $B_i \le A_i$, $1 \le i \le m$, such that the following holds:*

- *For all send types $p!q(c) \in \Sigma$, if there is an unmatched send event of type $p!q(c)$ in B_i then $q \notin P(B_{i+1} \cdots B_m)$.*
- *$N = B_1 \cdot B_2 \cdots B_m$ (by the first point, concatenation of the B_i is defined)*

Now choose an arbitrary maximal event e of $M \neq \emptyset$, and let $N = M{\restriction}_{E(M)\setminus\{e\}} \in \mathrm{p}\mathbb{MSC}$, i.e., remove e from M. Since $|N| < |M|$ and $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{N}_{\mathcal{A}}$, Lemma 5 applies to N. Thus, we get the following two properties (C1) and (C2) for M:

(C1) There is a maximal event e of M such that $N = M{\restriction}_{E(M)\setminus\{e\}}$ satisfies $N = B_1 \cdot B_2 \cdots B_m$ for non-empty prefixes $B_i \le A_i$ of atoms $A_i \in \mathbb{A}_H$.

(C2) For all send types $p!q(c) \in \Sigma$, if there is an unmatched send event of type $p!q(c)$ in B_i then $q \notin P(B_{i+1} \cdots B_m)$.

Now let $(\mathbf{s}, \mathcal{B}_\emptyset) = (\mathbf{s}_1, \mathcal{B}_1) \xrightarrow{B_1}_{\mathcal{A}} (\mathbf{s}_2, \mathcal{B}_2) \xrightarrow{B_2}_{\mathcal{A}} \cdots \xrightarrow{B_m}_{\mathcal{A}} (\mathbf{s}_{m+1}, \mathcal{B}_{m+1})$ be a run of $\mathcal{A}$ and assume that $\mathbf{s}_k = \mathbf{s}_\ell$ (but possibly $\mathcal{B}_k \neq \mathcal{B}_\ell$) for some $k < \ell$. Due to (C2), the CFM $\mathcal{A}$ can process, starting from $(\mathbf{s}_k, \mathcal{B}_k)$, also the suffix $B_\ell \cdots B_m$, i.e., $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{B_1 \cdots B_{k-1} \cdot B_\ell \cdots B_m}_{\mathcal{A}} (\mathbf{s}_{m+1}, \mathcal{C})$ for some buffer configuration $\mathcal{C}$ (in general $\mathcal{C} \neq \mathcal{B}_{m+1}$). We can use this observation for a kind of pumping argument, in order to prove that additionally to (C1) and (C2), the following property (C3) holds, where $\alpha = \max\{|A| \mid A \in \mathbb{A}_H\}$.

(C3) The number m in (C1) satisfies $m < (|P| + \alpha \cdot |P| + 2) \cdot (1 + \prod_{p \in P} |S_p|)$.

Proof of Theorem 3 (sketch). It suffices to check property (B) from Lemma 4 in PSPACE. Whether $\mathcal{A}_\emptyset$ is deadlock-free can be easily checked in PSPACE without explicitly constructing $\mathcal{A}_\emptyset$ (states of $\mathcal{A}_\emptyset$ can be encoded in polynomial space). It remains to check, whether a situation of the form $(\mathbf{s}, \mathcal{B}_\emptyset) \xrightarrow{M}_{\mathcal{A}}$ exists such that (1) from Lemma 4 becomes false. A first approach would be to guess such a situation, but note that the size bound for M that results from (C3) is exponential in the size of H. On the other hand, all one has to remember from M in order to check, whether $\mathbf{s}$ and M do not satisfy (1) from Lemma 4, is the set of processes $P(M)$ and the tuple of prefixes $(\pi_p(M)[1, \alpha_p])_{p \in P}$ of the projections onto the processes, where $\alpha_p = \max\{|\pi_p(A)| \mid A \in \mathbb{A}_H\}$ for $p \in P$ (whether $\sup(A, M)$ exists for some $A \in \mathbb{A}_H$ depends by Lemma 2 only on the prefixes $\pi_p(M)[1, \alpha_p]$), which can be stored in polynomial space. Hence, one can guess M "slice by slice", according to (C1), (C2), and (C3), and thereby accumulate only the data $P(M)$ and $(\pi_p(M)[1, \alpha_p])_{p \in P}$ (and forget everything else). $\square$

4 Non-FIFO communication

For all results in Section 3 we have restricted to FIFO communication. In this section we briefly discuss the non-FIFO case. Note that the obvious fact that under FIFO communication, every MSC M can be recovered from its projections $\pi_p(M)$, $p \in P$, is false for non-FIFO communication (take two messages with identical contents, which are received in M_1 in the order in which they were sent, whereas in M_2 they are received in reverse order). On the other hand if we forbid at least overtaking of messages with identical message contents, this fact still holds, see also [15]. Let us assume this for the further discussion. Note also that for the non-FIFO case, our CFM-model has to be slightly altered. The set $\mathfrak{C}^{\mathrm{Ch}}$ of buffer configurations has to be replaced by $\mathbb{N}^{\mathrm{Ch} \times \mathfrak{C}}$. For a given buffer configuration $\mathcal{B} \in \mathbb{N}^{\mathrm{Ch} \times \mathfrak{C}}$, the value $\mathcal{B}((p,q),c)$, where $(p,q) \in \mathrm{Ch}$ and $c \in \mathfrak{C}$, represents the number of messages with content c in the channel from p to q, see also [15]. Transitions in this CFM model are defined analogously to the FIFO-case in Section 2.2.

With the modifications described above, all results from Section 3 can be also shown for non-FIFO communication. First, thanks to our assumption that overtaking of identical messages is disallowed, Lemma 3 remains true. Concerning the EXPSPACE-hardness proof for Theorem 1, note that in the construction there, every message is immediately confirmed, which implies that the absence of the FIFO-restriction has no effect. Of course, the same holds for the undecidability proof of Theorem 2. Note also that the HMSC H in the proof of Theorem 2 (resp. Theorem 1) is either safely realizable (if $\mathcal{M}$ does not accept w) or not even weakly realizable (if $\mathcal{M}$ accepts w). It follows that also under non-FIFO communication, weak realizability is undecidable for the class of all HMSCs and EXPSPACE-hard for bounded HMSCs. For the latter problem, no primitive recursive upper bound is presently known, since the decidability proof in [15] uses a reduction to the reachability problem for Petri nets, for which no primitive recursive upper bound is known. Finally, also the proof of Theorem 3 (and hence of Theorem 4 and Theorem 5) works after some slight adaptations for non-FIFO communication.

Acknowledgments. I am grateful to Anca Muscholl for many fruitful discussions on the topic of this paper.

References

1. R. Alur, K. Etessami, and M. Yannakakis. Inference of message sequence charts. In *Proceedings of the 22nd International Conference on on Software Engineering (ICSE 2000), Limerick (Ireland)*, pages 304–313. ACM Press, 2000.
2. R. Alur, K. Etessami, and M. Yannakakis. Realizability and verification of MSC graphs. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP 2001), Crete (Greece)*, number 2076 in Lecture Notes in Computer Science, pages 797–808. Springer, 2001.
3. R. Alur and M. Yannakakis. Model checking of message sequence charts. In *Proceedings of the 9th International Conference on Concurrency Theory (CONCUR 99), Eindhoven (The Netherlands)*, number 1664 in Lecture Notes in Computer Science, pages 114–129. Springer, 1999.
4. H. Ben-Abdallah and S. Leue. Syntactic detection of process divergence and non-local choice in message sequence charts. In *Proceedings of the Third Interna-*

tional Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS '97), Enschede (The Netherlands), number 1217 in Lecture Notes in Computer Science, pages 259–274, 1997.

5. D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the Association for Computing Machinery*, 30(2):323–342, 1983.
6. B. Caillaud, P. Darondeau, L. Hélouët, and G. Lesventes. HMSCs as partial specifications . . . with Petri nets as completion. In *Modelling and Verification of Parallel Processes (MOVEP), Nantes (France)*, number 2067 in Lecture Notes in Computer Science, pages 125–152, 2000.
7. V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, Singapore, 1995.
8. B. Genest, A. Muscholl, H. Seidl, and M. Zeitoun. Infinite-state high-level MSCs: Model-checking and realizability. to appear in Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP 2002), Malaga (Spain), 2002.
9. E. Gunter, A. Muscholl, and D. Peled. Compositional message sequence charts. In T. Margaria and W. Yi, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 7th International Conference (TACAS), Genova (Italy)*, volume 2031 of *Lecture Notes in Computer Science*, pages 496–511. Springer, 2001.
10. L. Hélouët and C. Jard. Conditions for synthesis of communicating automata from HMSCs. In *5th International Workshop on Formal Methods for Industrial Critical Systems (FMICS), Berlin (Germany)*, 2000.
11. L. Hélouët and P. Le Maigat. Decomposition of message sequence charts. In *2nd Workshop on SDL and MSC (SAM 2000), Grenoble (France)*, pages 46–60, 2000.
12. J. G. Henriksen, M. Mukund, K. N. Kumar, and P. Thiagarajan. Regular collections of message sequence charts. In U. Montanari, J. D. P. Rolim, and E. Welzl, editors, *Proceedings of the 25th International Symposium onMathematical Foundations of Computer Science (MFCS'2000), Bratislava, (Slovakia)*, number 1893 in Lecture Notes in Computer Science, pages 675–686. Springer, 2000.
13. ITU. Recommendation Z.100. Specification and Description Language (SDL). 1994.
14. ITU. Recommendation Z.120. Message Sequence Charts. 1996.
15. R. Morin. Recognizable sets of message sequence charts. In H. Alt and A. Ferreira, editors, *Proceedings of the19th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2002), Juan les Pins (France)*, number 2285 in Lecture Notes in Computer Science, pages 523–534. Springer, 2002.
16. A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz trace. In M. Kutylowski, L. Pacholski, and T. Wierzbicki, editors, *Proceedings of the 24th Mathematical Foundations of Computer Science (MFCS'99), Szklarska Poreba (Poland)*, number 1672 in Lecture Notes in Computer Science, pages 81–91. Springer, 1999.
17. C. H. Papadimitriou. *Computational Complexity.* Addison Wesley, 1994.
18. G. von Bochmann. Finite state description of communication protocols. *Computer Networks*, 2:361–372, 1978.
19. I. Walukiewicz. Difficult configurations – on the complexity of LTrL. In *Proceedings of the25th International Colloquium on Automata, Languages and Programming (ICALP 98), Aalborg (Denmark)*, number 1443 in Lecture Notes in Computer Science, pages 140–151. Springer, 1998.
20. W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 27:99–135, 1985.