A Unifying Framework for Deciding Synchronizability

Benedikt Bollig, Cinzia Di Giusto, Alain Finkel, Laetitia Laversa, Etienne Lozes, Amrita Suresh

A Unifying Framework for Deciding Synchronizability

Benedikt Bollig
Université Paris-Saclay, ENS Paris-Saclay, CNRS, LMF, France

Cinzia Di Giusto
Université Côte d'Azur, CNRS, I3S, France

Alain Finkel
Université Paris-Saclay, ENS Paris-Saclay, CNRS, LMF, France
Institut Universitaire de France

Laetitia Laversa
Université Côte d'Azur, CNRS, I3S, France

Etienne Lozes
Université Côte d'Azur, CNRS, I3S, France

Amrita Suresh
Université Paris-Saclay, ENS Paris-Saclay, CNRS, LMF, France

Abstract

Several notions of synchronizability of a message-passing system have been introduced in the literature. Roughly, a system is called synchronizable if every execution can be rescheduled so that it meets certain criteria, e.g., a channel bound. We provide a framework, based on MSO logic and (special) tree-width, that unifies existing definitions, explains their good properties, and allows one to easily derive other, more general definitions and decidability results for synchronizability.

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory

Keywords and phrases communicating finite-state machines, message sequence charts, synchronizability, MSO logic, special tree-width

1 Introduction

Communication systems. The model of concurrent processes communicating asynchronously through FIFO channels is used since the 1960s in applications such as communication protocols [28], hardware design, MPI programs, and more recently for designing and verifying session types [23], web contracts, choreographies, concurrent programs, Erlang, Rust, etc. Since communication systems use FIFO channels, it is well known that all non-trivial properties (e.g., are all channels bounded?) are undecidable [9], essentially because a FIFO channel may simulate the tape of Turing machines and the counters of Minsky machines. However, there are many subclasses of communication systems for which the control-state reachability problem becomes decidable: e.g., synchronizable systems and existentially bounded systems (executions can be reorganized or decomposed into a finite number of sequences in which all channels are bounded), flat FIFO machines [15, 17] (the graph of the machine does not contain nested loops), channel-recognizable systems [4], unreliable (lossy, insertion, duplication) FIFO systems [11], input-bounded FIFO machines [5], and half-duplex systems [10].

On the boundedness problem. We focus on the boundedness problem, which is known to be undecidable. We could limit our analysis to decide whether for a given integer $k \geq 0$, known in advance, the FIFO channels are k-bounded, and this property is generally decidable in

PSPACE. Unfortunately, the k-boundedness property is too binding since we could want to design an *unbounded* system that is able, for example, to make unbounded iterations of sending and receiving messages. Hence, to cope with this limitation, one can find variants of the boundedness property that essentially reduce to say that every unbounded execution of a system (i.e., channels are unbounded along the execution) is equivalent (for instance, causally equivalent) to another *bounded* execution.

About synchronizability. To mention some examples, Lohrey and Muscholl introduced *existentially k-bounded* systems [25] (see also [18, 19, 24]) where all accepting executions leading to a stable (with empty channels) final configuration can be re-ordered into a k-bounded execution. This property is undecidable, even for a given k [18]. A more general definition, still called existentially bounded, is given in 2014 where the considered executions are *not* supposed to be final or stable [22]. In [21, 25], the notion of *universally k-bounded* (all possible schedulings of an execution are k-bounded) is also discussed and the authors show that the property is undecidable in general. In 2011, Basu and Bultan introduced *synchronizable* systems [3], for which every execution is equivalent (for the projection on sending messages) to one of the same system but communicating by rendezvous; to avoid ambiguity, we call such systems *send-synchronizable.* In 2018, Bouajjani et al., called a system $\mathcal{S}$ *k-synchronizable* [8] (to avoid confusion we call such systems *weakly k-synchronizable*) if every MSC of $\mathcal{S}$ admits a linearization (which is not necessarily an execution) that can be divided into blocks of at most k messages. After each block, a message is either read, or will never be read. This constraint seems to imply that buffers are bounded to k messages. However, as the linearization need not be an execution, this implies that a weakly k-synchronizable execution, even with the more efficient reschedule, can need unbounded channels to be run by the system.

Communication architecture and variants. A key difference between these works is that they consider different communication architectures. Existentially bounded systems have been studied for p2p (with one queue per pair of processes), whereas k-synchronizability has been studied for mailbox communication, for which each process merges all its incoming messages in a unique queue. The decidability results for k-synchronizability have been extended to p2p communications [14], but it is unknown whether the decidability results for existentially bounded systems extend to mailbox communication. Moreover, variants of those definitions can be obtained depending on if we consider messages that are sent but never read, called unmatched messages. Indeed the challenges that arise in [8] are due to mailbox communication and unmatched messages blocking a channel so that all messages sent afterwards will never be read. To clarify and overcome this issue, we propose *strong k-synchronizability*, a new definition that is suitable for mailbox communication: an execution is called *strongly k-synchronizable* if it can be rescheduled into another k-bounded execution such that there are at most k messages in the channels before emptying them.

Contributions. Our contributions can be summarized as follows:

- In order to unify the notions of synchronizability, we introduce a general framework based on monadic second-order (MSO) logic and (special) tree-width that captures most existing definitions of systems that may work with bounded channels. Moreover, reachability and model checking are shown decidable in this framework.
- We show that existentially bounded systems can be expressed in our framework and, as a consequence, the existentially k-bounded property is decidable by using the generic proof.

- We generalize the existing notion of (weak) k-synchronizability in [8] and we introduce three new classes of synchronizable systems: weakly synchronizable (which are more general than weakly k-synchronizable), strongly synchronizable and strongly k-synchronizable (which are particular cases of weakly synchronizable). We then prove that these properties all fit in our framework and are all shown decidable using the generic proof.
- We then deduce that reachability and model checking are decidable for these classes (only control-state reachability was shown to be decidable for weakly k-synchronizable in [8] and it is clearly also decidable for existentially/universally bounded systems but reachability properties are generally not studied for these classes of systems).
- In order to obtain better complexity results for some classes (strongly and weakly synchronizable systems), we also use the fragment of propositional dynamic logic with loop and converse (LCPDL) instead of MSO logic in our framework.
- We provide a comparison between synchronizable classes both for p2p and mailbox semantics (see Fig. 8 for p2p systems and Fig. 9 for mailbox systems). In particular, we clarify the link between weakly synchronizable and existentially bounded systems for both p2p and mailbox systems, which was left open in [8] and solved only for p2p systems in [23, Theorem 7] where weakly synchronizable systems are shown to be included into existentially bounded ones when considering executions (and not MSCs as in our case).

Outline. Section 2 defines some preliminary notions such as p2p/mailbox message sequence charts (MSCs), and communicating systems. Section 3 presents the unifying MSO framework and two general theorems on k-synchronizability and model checking. In Section 4, we apply the MSO framework to different existing definitions of synchronizability, and we introduce a new decidable one. Section 5 studies the relations between the classes. In Section 6, we conclude with some final remarks. Missing proofs are given in the appendix.

2 Preliminaries

2.1 Message Sequence Charts

Assume a finite set of processes $\mathbb{P}$ and a finite set of messages $\mathbb{M}$. The set of (p2p) channels is $\mathbb{C} = \{(p,q) \in \mathbb{P} \times \mathbb{P} \mid p \neq q\}$. A send action is of the form $\mathit{send}(p,q,m)$ where $(p,q) \in \mathbb{C}$ and $m \in \mathbb{M}$. It is executed by p and sends message m to q. The corresponding receive action, executed by q, is $\mathit{rec}(p,q,m)$. For $(p,q) \in \mathbb{C}$, let $\mathit{Send}(p,q,\_) = \{\mathit{send}(p,q,m) \mid m \in \mathbb{M}\}$ and $\mathit{Rec}(p,q,\_) = \{\mathit{rec}(p,q,m) \mid m \in \mathbb{M}\}$. For $p \in \mathbb{P}$, we set $\mathit{Send}(p,\_,\_) = \{\mathit{send}(p,q,m) \mid q \in \mathbb{P} \setminus \{p\} \text{ and } m \in \mathbb{M}\}$, etc. Moreover, $\Sigma_p = \mathit{Send}(p,\_,\_) \cup \mathit{Rec}(\_,p,\_)$ will denote the set of all actions that are executed by p. Finally, $\Sigma = \bigcup_{p\in\mathbb{P}} \Sigma_p$ is the set of all the actions.

Peer-to-peer MSCs. A *p2p MSC* (or simply *MSC*) over $\mathbb{P}$ and $\mathbb{M}$ is a tuple $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ where $\mathcal{E}$ is a finite (possibly empty) set of *events* and $\lambda : \mathcal{E} \rightarrow \Sigma$ is a labeling function. For $p \in \mathbb{P}$, let $\mathcal{E}_p = \{e \in \mathcal{E} \mid \lambda(e) \in \Sigma_p\}$ be the set of events that are executed by p. We require that $\rightarrow$ (the *process relation*) is the disjoint union $\bigcup_{p\in\mathbb{P}} \rightarrow_p$ of relations $\rightarrow_p \subseteq \mathcal{E}_p \times \mathcal{E}_p$ such that $\rightarrow_p$ is the direct successor relation of a total order on $\mathcal{E}_p$. For an event $e \in \mathcal{E}$, a set of actions $A \subseteq \Sigma$, and a relation $R \subseteq \mathcal{E} \times \mathcal{E}$, let $\#_A(R,e) = |\{f \in \mathcal{E} \mid (f,e) \in R \text{ and } \lambda(f) \in A\}|$. We require that $\lhd \subseteq \mathcal{E} \times \mathcal{E}$ (the *message relation*) satisfies the following:

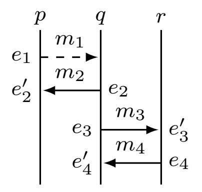
(1) for every pair $(e,f) \in \lhd$, there is a send action $\mathit{send}(p,q,m) \in \Sigma$ such that $\lambda(e) = \mathit{send}(p,q,m)$, $\lambda(f) = \mathit{rec}(p,q,m)$, and $\#_{\mathit{Send}(p,q,\_)}(\rightarrow^+, e) = \#_{\mathit{Rec}(p,q,\_)}(\rightarrow^+, f)$,

(2) for all $f \in \mathcal{E}$ such that $\lambda(f)$ is a receive action, there is $e \in \mathcal{E}$ such that $e \lhd f$.

Finally, letting $\leq_M = (\rightarrow \cup \lhd)^*$, we require that $\leq_M$ is a partial order.

Condition (1) above ensures that every (p2p) channel (p, q) behaves in a FIFO manner. By Condition (2), every receive event has a matching send event. Note that, however, there may be unmatched send events in an MSC. We let $\mathit{SendEv}(M) = \{e \in \mathcal{E} \mid \lambda(e)$ is a send action$\}$, $\mathit{RecEv}(M) = \{e \in \mathcal{E} \mid \lambda(e)$ is a receive action$\}$, $\mathit{Matched}(M) = \{e \in \mathcal{E} \mid$ there is $f \in \mathcal{E}$ such that $e \lhd f\}$, and $\mathit{Unm}(M) = \{e \in \mathcal{E} \mid \lambda(e)$ is a send action and there is no $f \in \mathcal{E}$ such that $e \lhd f\}$. We do not distinguish isomorphic MSCs and let MSC be the set of all MSCs over the given sets $\mathbb{P}$ and $\mathbb{M}$.

▶ **Example 1.** For a set of processes $\mathbb{P} = \{p, q, r\}$ and a set of messages $\mathbb{M} = \{m_1, m_2, m_3, m_4\}$, $M_1 = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ is an MSC where, for example, $e_2 \lhd e_2'$ and $e_3' \rightarrow e_4$. The dashed arrow means that the send event e_1 does not have a matching receive, so $e_1 \in \mathit{Unm}(M_1)$. Moreover, $e_2 \leq_{M_1} e_4$, but $e_1 \not\leq_{M_1} e_4$. We can find a total order $\rightsquigarrow \supseteq \leq_{M_1}$ such that $e_1 \rightsquigarrow e_2 \rightsquigarrow e_2' \rightsquigarrow e_3 \rightsquigarrow e_3' \rightsquigarrow e_4 \rightsquigarrow e_4'$. We call $\rightsquigarrow$ a linearization, which is formally defined below.



🟨 **Figure 1** MSC M_1

Mailbox MSCs. For an MSC $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$, we define an additional binary relation that represents a constraint under the mailbox semantics, where each process has only one incoming channel. Let $\sqsubset_M \subseteq \mathcal{E} \times \mathcal{E}$ be defined by: $e_1 \sqsubset_M e_2$ if there is $q \in \mathbb{P}$ such that $\lambda(e_1) \in \mathit{Send}(\_, q, \_)$, $\lambda(e_2) \in \mathit{Send}(\_, q, \_)$, and one of the following holds:

- $e_1 \in \mathit{Matched}(M)$ and $e_2 \in \mathit{Unm}(M)$, or
- $e_1 \lhd f_1$ and $e_2 \lhd f_2$ for some $f_1, f_2 \in \mathcal{E}_q$ such that $f_1 \rightarrow^+ f_2$.

We let $\preceq_M = (\rightarrow \cup \lhd \cup \sqsubset_M)^*$. Note that $\leq_M \subseteq \preceq_M$. We call $M \in \mathsf{MSC}$ a *mailbox MSC* if $\preceq_M$ is a partial order. Intuitively, this means that events can be scheduled in a way that corresponds to the mailbox semantics, i.e., with one incoming channel per process. Following the terminology in [8], we also say that a mailbox MSC satisfies *causal delivery*. The set of mailbox MSCs $M \in \mathsf{MSC}$ is denoted by $\mathsf{MSC}_{\mathsf{mb}}$.

▶ **Example 2.** MSC M_1 is a mailbox MSC. Indeed, even though the order $\rightsquigarrow$ defined in Example 1 does not respect all mailbox constraints, particularly the fact that $e_4 \sqsubset_{M_1} e_1$, there is a total order $\rightsquigarrow \supseteq \preceq_{M_1}$ such that $e_2 \rightsquigarrow e_3 \rightsquigarrow e_3' \rightsquigarrow e_4 \rightsquigarrow e_1 \rightsquigarrow e_2' \rightsquigarrow e_4'$. We call $\rightsquigarrow$ a mailbox linearization, which is formally defined below.

Linearizations, Prefixes, and Concatenation. Consider $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}$. A *p2p linearization* (or simply *linearization*) of M is a (reflexive) total order $\rightsquigarrow \subseteq \mathcal{E} \times \mathcal{E}$ such that $\leq_M \subseteq \rightsquigarrow$. Similarly, a *mailbox linearization* of M is a total order $\rightsquigarrow \subseteq \mathcal{E} \times \mathcal{E}$ such that $\preceq_M \subseteq \rightsquigarrow$. That is, every mailbox linearization is a p2p linearization, but the converse is not necessarily true (Example 2). Note that an MSC is a mailbox MSC iff it has at least one mailbox linearization.

Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}$ and consider $E \subseteq \mathcal{E}$ such that E is $\leq_M$*-downward-closed*, i.e, for all $(e, f) \in \leq_M$ such that $f \in E$, we also have $e \in E$. Then, the MSC $(E, \rightarrow \cap (E \times E), \lhd \cap (E \times E), \lambda')$, where λ' is the restriction of $\mathcal{E}$ to E, is called a *prefix* of M. In particular, the empty MSC is a prefix of M. We denote the set of prefixes of M by $\mathit{Pref}(M)$. This is extended to sets $L \subseteq \mathsf{MSC}$ as expected, letting $\mathit{Pref}(L) = \bigcup_{M \in L} \mathit{Pref}(M)$.

▶ **Lemma 3.** *Every prefix of a mailbox MSC is a mailbox MSC.*

Let $M_1 = (\mathcal{E}_1, \rightarrow_1, \lhd_1, \lambda_1)$ and $M_2 = (\mathcal{E}_2, \rightarrow_2, \lhd_2, \lambda_2)$ be two MSCs. Their *concatenation* $M_1 \cdot M_2 = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ is defined if, for all $(p, q) \in \mathbb{C}$, $e_1 \in \mathit{Unm}(M_1)$, and $e_2 \in \mathcal{E}_2$ such that $\lambda(e_1) \in \mathit{Send}(p, q, \_)$ and $\lambda(e_2) \in \mathit{Send}(p, q, \_)$, we have $e_2 \in \mathit{Unm}(M_2)$. As expected, $\mathcal{E}$ is the disjoint union of $\mathcal{E}_1$ and $\mathcal{E}_2$, $\lhd = \lhd_1 \cup \lhd_2$, λ is the "union" of λ_1 and λ_2, and $\rightarrow = \rightarrow_1 \cup \rightarrow_2 \cup R$. Here, R contains, for all $p \in \mathbb{P}$ such that $(\mathcal{E}_1)_p$ and $(\mathcal{E}_2)_p$ are non-empty, the pair (e_1, e_2) where e_1 is the maximal p-event in M_1 and e_2 is the minimal p-event in M_2. Note that $M_1 \cdot M_2$ is indeed an MSC and that concatenation is associative.

2.2 Communicating Systems

We now recall the definition of communicating systems (aka communicating finite-state machines or message-passing automata), which consist of finite-state machines A_p (one for every process $p \in \mathbb{P}$) that can communicate through the FIFO channels from $\mathbb{C}$.

▶ **Definition 4.** *A* communicating system *over $\mathbb{P}$ and $\mathbb{M}$ is a tuple $\mathcal{S} = (A_p)_{p \in \mathbb{P}}$. For each $p \in \mathbb{P}$, $A_p = (\mathit{Loc}_p, \delta_p, \ell_p^0)$ is a finite transition system where Loc_p is a finite set of local (control) states, $\delta_p \subseteq \mathit{Loc}_p \times \Sigma_p \times \mathit{Loc}_p$ is the transition relation, and $\ell_p^0 \in \mathit{Loc}_p$ is the initial state.*

Given $p \in \mathbb{P}$ and a transition $t = (\ell, a, \ell') \in \delta_p$, we let $\mathit{source}(t) = \ell$, $\mathit{target}(t) = \ell'$, $\mathit{action}(t) = a$, and $\mathit{msg}(t) = m$ if $a \in \mathit{Send}(\_, \_, m) \cup \mathit{Rec}(\_, \_, m)$.

There are in general two ways to define the semantics of a communicating system. Most often it is defined as a global infinite transition system that keeps track of the various local control states and all (unbounded) channel contents. As, in this paper, our arguments are based on a graph view of MSCs, we will define the language of $\mathcal{S}$ directly as a set of MSCs. These two semantic views are essentially equivalent, but they have different advantages depending on the context. We refer to [1] for a thorough discussion.

Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ be an MSC. A *run* of $\mathcal{S}$ on M is a mapping $\rho : \mathcal{E} \rightarrow \bigcup_{p \in \mathbb{P}} \delta_p$ that assigns to every event e the transition $\rho(e)$ that is executed at e. Thus, we require that (i) for all $e \in \mathcal{E}$, we have $\mathit{action}(\rho(e)) = \lambda(e)$, (ii) for all $(e, f) \in \rightarrow$, $\mathit{target}(\rho(e)) = \mathit{source}(\rho(f))$, (iii) for all $(e, f) \in \lhd$, $\mathit{msg}(\rho(e)) = \mathit{msg}(\rho(f))$, and (iv) for all $p \in \mathbb{P}$ and $e \in \mathcal{E}_p$ such that there is no $f \in \mathcal{E}$ with $f \rightarrow e$, we have $\mathit{source}(\rho(e)) = \ell_p^0$.

Letting run $\mathcal{S}$ directly on MSCs is actually very convenient. This allows us to associate with $\mathcal{S}$ its p2p language and mailbox language in one go. The *p2p language* of $\mathcal{S}$ is $L_{\mathsf{p2p}}(\mathcal{S}) = \{M \in \mathsf{MSC} \mid \text{there is a run of } \mathcal{S} \text{ on } M\}$. The *mailbox language* of $\mathcal{S}$ is $L_{\mathsf{mb}}(\mathcal{S}) = \{M \in \mathsf{MSC}_{\mathsf{mb}} \mid \text{there is a run of } \mathcal{S} \text{ on } M\}$.

Note that, following [8, 14], we do not consider final states or final configurations, as our purpose is to reason about all possible traces that can be *generated* by $\mathcal{S}$. The next lemma is obvious for the p2p semantics and follows from Lemma 3 for the mailbox semantics.

▶ **Lemma 5.** *For all* com $\in \{\mathsf{p2p}, \mathsf{mb}\}$*, $L_{\text{com}}(\mathcal{S})$ is prefix-closed: $\mathit{Pref}(L_{\text{com}}(\mathcal{S})) \subseteq L_{\text{com}}(\mathcal{S})$.*

▶ **Example 6.** Fig. 2 depicts $\mathcal{S}_1 = (A_p, A_q, A_r)$ such that MSC M_1 in Fig. 1 belongs to $L_{\mathsf{p2p}}(\mathcal{S}_1)$ and to $L_{\mathsf{mb}}(\mathcal{S}_1)$. There is a unique run ρ of $\mathcal{S}_1$ on M_1. We can see that $(e_3', e_4) \in \rightarrow$ and $\mathit{target}(\rho(e_3')) = \mathit{source}(\rho(e_4)) = \ell_r^1$, $(e_2, e_2') \in \lhd_{M_1}$, and $\mathit{msg}(\rho(e_2)) = \mathit{msg}(\rho(e_2')) = m_2$.

2.3 Conflict Graph

We now recall the notion of a conflict graph associated to an MSC defined in [8]. This graph is used to depict the causal dependencies between message exchanges. Intuitively, we have

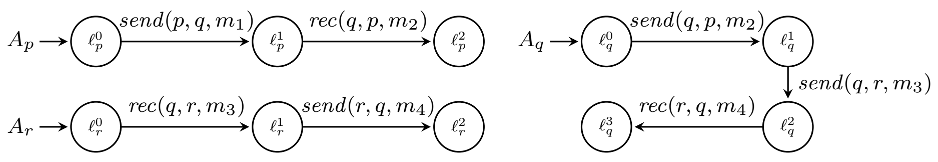


Figure 2 System $\mathcal{S}_1$

a dependency whenever two messages have a process in common. For instance, an $\xrightarrow{SS}$ dependency between message exchanges v and v' expresses the fact that v' has been sent after v, by the same process. This notion is of interest because it was seen in [8] that the notion of synchronizability in MSCs (which is studied in this paper) can be graphically characterized by the nature of the associated conflict graph. It is defined in terms of linearizations in [14], but we equivalently express it directly in terms of MSCs.

For an MSC $M = (\mathcal{E}, \to, \lhd, \lambda)$ and $e \in \mathcal{E}$, we define the type $\tau(e) \in \{S, R\}$ of e by $\tau(e) = S$ if $e \in \mathit{SendEv}(M)$ and $\tau(e) = R$ if $e \in \mathit{RecEv}(M)$. Moreover, for $e \in \mathit{Unm}(M)$, we let $\mu(e) = e$, and for $(e, e') \in \lhd$, we let $\mu(e) = \mu(e') = (e, e')$.

▶ **Definition 7** (Conflict graph). *The* conflict graph $\mathsf{CG}(M)$ *of an MSC* $M = (\mathcal{E}, \to, \lhd, \lambda)$ *is the labeled graph* $(\mathit{Nodes}, \mathit{Edges})$*, with* $\mathit{Edges} \subseteq \mathit{Nodes} \times \{S, R\}^2 \times \mathit{Nodes}$*, defined by* $\mathit{Nodes} = \lhd \cup \mathit{Unm}(M)$ *and* $\mathit{Edges} = \{(\mu(e), \tau(e)\tau(f), \mu(f)) \mid (e, f) \in \to^+\}$*. In particular, a node of* $\mathsf{CG}(M)$ *is either a single unmatched send event or a message pair* $(e, e') \in \lhd$*.*

3 Model Checking and Synchronizability

In this section, we survey two classical decision problems for communicating systems. The first problem is the model-checking problem, in which one checks whether a given system satisfies a given specification. A canonical specification language for MSCs is monadic second-order (MSO) logic. However, model checking in full generality is undecidable. A common approach is, therefore, to restrict the behavior of the given system to MSCs of bounded (special) tree-width. Next, we introduce MSO logic and special tree-width.

3.1 Logic and Special Tree-Width

Monadic Second-Order Logic. The set of MSO formulas over MSCs (over $\mathbb{P}$ and $\mathbb{M}$) is given by the grammar $\varphi ::= x \to y \mid x \lhd y \mid \lambda(x) = a \mid x = y \mid x \in X \mid \exists x.\varphi \mid \exists X.\varphi \mid \varphi \vee \varphi \mid \neg\varphi$, where $a \in \Sigma$, x and y are first-order variables, interpreted as events of an MSC, and X is a second-order variable, interpreted as a set of events. We assume that we have an infinite supply of variables, and we use common abbreviations such as $\wedge$, $\forall$, etc. The satisfaction relation is defined in the standard way and self-explanatory. For example, the formula $\neg\exists x.(\bigvee_{a \in \mathit{Send}(\_,\_,\_)} \lambda(x) = a \ \wedge\ \neg \mathit{matched}(x))$ with $\mathit{matched}(x) = \exists y. x \lhd y$ says that there are no unmatched send events. It is not satisfied by MSC M_1 of Fig. 1, as message m_1 is not received, but by M_4 from Fig. 6.

Given a sentence φ, i.e., a formula without free variables, we let $L(\varphi)$ denote the set of (p2p) MSCs that satisfy φ. It is worth mentioning that the (reflexive) transitive closure of a binary relation defined by an MSO formula with free variables x and y, such as $x \to y$, is MSO-definable so that the logic can freely use formulas of the form $x \to^+ y$ or $x \le y$ (where $\le$ is interpreted as $\le_M$ for the given MSC M). Therefore, the definition of a mailbox MSC can be readily translated into the formula $\varphi_{\mathsf{mb}} = \neg\exists x.\exists y.(\neg(x = y) \wedge x \preceq y \wedge y \preceq x)$ so that

$$
\begin{array}{llll}
M \models \mathsf{E}\sigma \text{ if } [\![\sigma]\!]_M \neq \emptyset & & [\![\rightarrow]\!]_M := \rightarrow \quad \text{and} \quad [\![\lhd]\!]_M := \lhd & \\
[\![a]\!]_M & := \{e \in \mathcal{E} \mid \lambda(e) = a\} & [\![\mathsf{test}(\sigma)]\!]_M & := \{(e,e) \mid e \in [\![\sigma]\!]_M\} \\
[\![\langle\pi\rangle\sigma]\!]_M & := \{e \in \mathcal{E} \mid \exists f \in [\![\sigma]\!]_M : (e,f) \in [\![\pi]\!]_M\} & [\![\mathsf{jump}]\!]_M & := \mathcal{E} \times \mathcal{E} \\
[\![\mathsf{Loop}\langle\pi\rangle]\!]_M & := \{e \in \mathcal{E} \mid (e,e) \in [\![\pi]\!]_M\} & [\![\pi_1 + \pi_2]\!]_M & := [\![\pi_1]\!]_M \cup [\![\pi_2]\!]_M \\
[\![\pi^{-1}]\!]_M & := \{(e,f) \in \mathcal{E} \times \mathcal{E} \mid (f,e) \in [\![\pi]\!]_M\} & [\![\pi^*]\!]_M & := \bigcup_{n\in\mathbb{N}} [\![\pi]\!]_M^n \\
[\![\pi_1 \cdot \pi_2]\!]_M & := \{(e,f) \in \mathcal{E} \times \mathcal{E} \mid \exists g \in \mathcal{E} : (e,g) \in [\![\pi_1]\!]_M \text{ and } (g,f) \in [\![\pi_2]\!]_M\} & &
\end{array}
$$

Figure 3 Semantics of LCPDL

we have $L(\varphi_{\mathsf{mb}}) = \mathsf{MSC}_{\mathsf{mb}}$. Here, $x \preceq y$ is obtained as the MSO-definable reflexive transitive closure of the union of the MSO-definable relations $\rightarrow$, $\lhd$, and $\sqsubset$. In particular, we may define $x \sqsubset y$ by

$$x \sqsubset y = \bigvee_{\substack{q\in\mathbb{P} \\ a,b\in Send(\_,q,\_)}} \lambda(x) = a \ \wedge\ \lambda(y) = b \wedge \left(\begin{array}{ll} & matched(x) \wedge \neg matched(y) \\ \vee & \exists x'.\exists y'.(x \lhd x' \ \wedge\ y \lhd y' \ \wedge\ x' \rightarrow^+ y') \end{array} \right).$$

Propositional Dynamic Logic (PDL). For better complexity, we also consider PDL with Loop and Converse, henceforth called LCPDL (cf. [6,7,27] for more details). Its syntax is:

$$
\begin{array}{lr}
\Phi ::= \mathsf{E}\sigma \mid \Phi \vee \Phi \mid \neg\Phi & \text{(sentence)} \\
\sigma ::= a \mid \sigma \vee \sigma \mid \neg\sigma \mid \langle\pi\rangle\sigma \mid \mathsf{Loop}\langle\pi\rangle & \text{(event formula)} \\
\pi ::= \rightarrow \mid \lhd \mid \mathsf{test}(\sigma) \mid \mathsf{jump} \mid \pi + \pi \mid \pi \cdot \pi \mid \pi^* \mid \pi^{-1} & \text{(path formula)}
\end{array}
$$

where $a \in \Sigma$. We use the symbol $\top$ to denote a tautology event formula (such as $a \vee \neg a$). We describe the semantics for the logic in Fig. 3 (apart from the obvious cases). A sentence Φ is evaluated wrt. an MSC $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$. An event formula σ is evaluated wrt. M and an event $e \in \mathcal{E}$ so that it defines a unary relation $[\![\sigma]\!]_M \subseteq \mathcal{E}$. Finally, a path formula π is evaluated over two events, and so it defines a binary relation $[\![\pi]\!]_M \subseteq \mathcal{E} \times \mathcal{E}$. Finally, we let $L(\Phi) = \{M \in \mathsf{MSC} \mid M \models \Phi\}$. Note that every LCPDL-definable property is MSO-definable.

It can be seen below that the mailbox semantics can be readily translated into the LCPDL formula $\Phi_{\mathsf{mb}} = \neg\mathsf{E}\,(\mathsf{Loop}\langle(\lhd + \rightarrow + \sqsubset)^+\rangle)$ such that $L(\Phi_{\mathsf{mb}}) = \mathsf{MSC}_{\mathsf{mb}}$. Hereby, we let

$$\sqsubset\, = \lhd \cdot \rightarrow^+ \cdot \lhd^{-1} \ + \sum_{\substack{q\in\mathbb{P} \\ a,b\in Send(\_,q,\_)}} \mathsf{test}(a) \cdot \lhd \cdot \mathsf{jump} \cdot \mathsf{test}(b \wedge \neg\langle\lhd\rangle\top)\,.$$

Special Tree-Width. *Special tree-width* [12], is a graph measure that indicates how close a graph is to a tree (we may also use classical *tree-width* instead). This or similar measures are commonly employed in verification. For instance, tree-width and split-width have been used in [26] and, respectively, [2,13] to reason about graph behaviors generated by pushdown and queue systems. There are several ways to define the special tree-width of an MSC. We adopt the following game-based definition from [7].

Adam and Eve play a two-player turn based “decomposition game” whose positions are MSCs with some pebbles placed on some events. More precisely, Eve’s positions are *marked MSC fragments* (M, U), where $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ is an *MSC fragment* (an MSC with possibly some edges from $\lhd$ or $\rightarrow$ removed) and $U \subseteq \mathcal{E}$ is the subset of marked events. Adam’s positions are pairs of marked MSC fragments. A move by Eve consists in the following steps:

1. marking some events of the MSC resulting in (M, U') with $U \subseteq U' \subseteq \mathcal{E}$,
2. removing (process and/or message) edges whose endpoints are marked,
3. dividing (M, U) in (M_1, U_1) and (M_2, U_2) such that M is the disjoint (unconnected) union of M_1 and M_2 and marked nodes are inherited.

When it is Adam's turn, he simply chooses one of the two marked MSC fragments. The initial position is $(M, \emptyset)$ where M is the (complete) MSC at hand. A terminal position is any position belonging to Eve such that all events are marked. For $k \in \mathbb{N}$, we say that the game is k-winning for Eve if she has a (positional) strategy that allows her, starting in the initial position and independently of Adam's moves, to reach a terminal position such that, in every single position visited along the play, there are at most $k + 1$ marked events.

▶ **Fact 1** ([7]). *The special tree-width of an MSC is the least k such that the associated game is k-winning for Eve.*

The set of MSCs whose special tree-width is at most k is denoted by $\mathsf{MSC}^{k\text{-}\mathsf{stw}}$.

3.2 Model Checking

In general, even simple verification problems, such as control-state reachability, are undecidable for communicating systems [9]. However, they are decidable when we restrict to behaviors of bounded special tree-width, which motivates the following definition of a generic **bounded model-checking problem** for com $\in \{\mathsf{p2p}, \mathsf{mb}\}$:
Input: Two finite sets $\mathbb{P}$ and $\mathbb{M}$, a communicating system $\mathcal{S}$, an MSO sentence φ, and $k \in \mathbb{N}$ (given in unary).
Question: Do we have $L_{\mathrm{com}}(\mathcal{S}) \cap \mathsf{MSC}^{k\text{-}\mathsf{stw}} \subseteq L(\varphi)$?

▶ **Fact 2** ([7]). *The bounded model-checking problem for* com $= \mathsf{p2p}$ *is decidable. When the formulas φ are from LCPDL, then the problem is solvable in exponential time.*

Note that [7] does not employ the LCPDL modality jump, but it can be integrated easily. Using φ_{mb} or Φ_{mb}, we obtain the corresponding result for mailbox systems as a corollary:

▶ **Theorem 8.** *The bounded model-checking problem for* com $= \mathsf{mb}$ *is decidable. When the formulas φ are from LCPDL, then the problem is solvable in exponential time.*

3.3 Synchronizability

The above model-checking approach is incomplete in the sense that a positive answer does not imply correctness of the whole system. The system may still produce behaviors of special tree-width greater than k that violate the given property. However, if we know that a system only generates behaviors from a class whose special tree-width is bounded by k, we can still conclude that the system is correct.

This motivates the *synchronizability problem.* Several notions of synchronizability have been introduced in the literature. However, they all amount to asking whether all behaviors generated by a given communicating system have a particular shape, i.e., whether they are all included in a fixed (or given) set of MSCs $\mathcal{C}$. Thus, the synchronizability problem is essentially an inclusion problem, namely $L_{\mathsf{p2p}}(\mathcal{S}) \subseteq \mathcal{C}$ or $L_{\mathsf{mb}}(\mathcal{S}) \subseteq \mathcal{C}$. We show that, for decidability, it is enough to have that $\mathcal{C}$ is MSO-definable and special-tree-width-bounded (STW-bounded): We call $\mathcal{C} \subseteq \mathsf{MSC}$ (i) *MSO-definable* if there is an MSO-formula φ such that $L(\varphi) = \mathcal{C}$, (ii) *LCPDL-definable* if there is an an LCPDL-formula Φ such that $L(\Phi) = \mathcal{C}$, (iii) *STW-bounded* if there is $k \in \mathbb{N}$ such that $\mathcal{C} \subseteq \mathsf{MSC}^{k\text{-}\mathsf{stw}}$.

Table 1 Summary of the decidability of the synchronizability problem in various classes

	Peer-to-Peer	Mailbox
Weakly synchronous	Undecidable [Thm. 20]	EXPTIME [Thm. 19]
Weakly k-synchronous	Decidable [8, 14] and [Thm. 27]	
Strongly k-synchronous	—	Decidable [Thm. 33]
Existentially k-p2p-bounded	Decidable [18, Prop. 5.5]	
Existentially k-mailbox-bounded	—	Decidable [Prop. 38]

An important component of the decidability proof is the following lemma, which shows that we can reduce synchronizability wrt. an STW-bounded class to bounded model-checking.

▶ **Lemma 9.** *Let $\mathcal{S}$ be a communicating system,* com $\in \{\mathsf{p2p}, \mathsf{mb}\}$*, $k \in \mathbb{N}$, and $\mathcal{C} \subseteq \mathsf{MSC}^{k\text{-stw}}$. Then, $L_{\mathrm{com}}(\mathcal{S}) \subseteq \mathcal{C}$ iff $L_{\mathrm{com}}(\mathcal{S}) \cap \mathsf{MSC}^{(k+2)\text{-stw}} \subseteq \mathcal{C}$.*

The result follows from the following lemma. Note that a similar property was shown in [18, Proposition 5.4] for the specific class of existentially k-bounded MSCs.

▶ **Lemma 10.** *Let $k \in \mathbb{N}$ and $\mathcal{C} \subseteq \mathsf{MSC}^{k\text{-stw}}$. For all $M \in \mathsf{MSC} \setminus \mathcal{C}$, we have $(\mathit{Pref}(M) \cap \mathsf{MSC}^{(k+2)\text{-stw}}) \setminus \mathcal{C} \neq \emptyset$.*

We now have all ingredients to state a generic decidability result for synchronizability:

▶ **Theorem 11.** *Fix finite sets $\mathbb{P}$ and $\mathbb{M}$. Suppose* com $\in \{\mathsf{p2p}, \mathsf{mb}\}$ *and let $\mathcal{C} \subseteq \mathsf{MSC}$ be an MSO-definable and STW-bounded class (over $\mathbb{P}$ and $\mathbb{M}$). The following problem is decidable: Given a communicating system $\mathcal{S}$, do we have $L_{\mathrm{com}}(\mathcal{S}) \subseteq \mathcal{C}$?*

Proof. Consider the MSO-formula φ such that $L(\varphi) = \mathcal{C}$, and let $k \in \mathbb{N}$ such that $\mathcal{C} \subseteq \mathsf{MSC}^{k\text{-stw}}$. We have $L_{\mathrm{com}}(\mathcal{S}) \subseteq \mathcal{C} \overset{\text{Lemma 9}}{\iff} L_{\mathrm{com}}(\mathcal{S}) \cap \mathsf{MSC}^{(k+2)\text{-stw}} \subseteq \mathcal{C} \iff L_{\mathrm{com}}(\mathcal{S}) \cap \mathsf{MSC}^{(k+2)\text{-stw}} \subseteq L(\varphi)$. The latter can be solved thanks to Fact 2 and Theorem 8. ◀

▶ Remark 12. Note that, in some cases (cf. Section 4), $\mathbb{P}$ and $\mathbb{M}$ are part of the input and the concrete class $\mathcal{C}$ may be parameterized by a natural number so that it is part of the input, too. Then, we need to be able to compute the MSO formula characterizing the class as well as the bound on the special tree-width.

4 Application to Concrete Classes of Synchronizability

In this section, we instantiate our general framework by specific classes. Table 1 gives a summary of the results.

4.1 A New General Class: Weakly Synchronous MSCs

We first introduce the class of weakly synchronous MSCs. This is a generalization of synchronous MSCs studied earlier, in [8, 14], which we shall discuss later. We say an MSC is weakly synchronous if it is breakable into *exchanges* where an exchange is an MSC that allows one to schedule all sends before all receives. Let us define this formally:

▶ **Definition 13** (exchange)**.** *Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ be an MSC. We say that M is an* exchange *if $\mathit{SendEv}(M)$ is a $\leq_M$-downward-closed set.*

▶ **Definition 14** (weakly synchronous)**.** *We say that $M \in \mathsf{MSC}$ is* weakly synchronous *if it is of the form $M = M_1 \cdot \ldots \cdot M_n$ such that every M_i is an exchange.*

We use the term *weakly* to distinguish from variants introduced later.

▶ **Example 15.** Consider the MSC M_2 in Fig. 4. It is is weakly synchronous. Indeed, m_1, m_2, and m_5 are independent and can be put alone in an exchange. Repetitions of m_3 and m_4 are interlaced, but they constitute an exchange, as we can do all sends and then all receptions.

An easy adaptation of a characterization from [14] yields the following result for weakly synchronous MSCs:

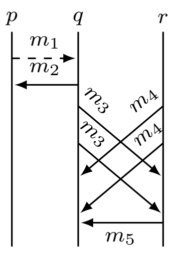


Figure 4 MSC M_2

▶ **Proposition 16.** *Let M be an MSC. Then, M is weakly synchronous iff no RS edge occurs on any cyclic path in the conflict graph* $\mathsf{CG}(M)$.

It is easily seen that the characterization from Proposition 16 is LCPDL-definable:

▶ **Corollary 17.** *The sets of weakly synchronous MSCs and weakly synchronous* mailbox *MSCs are LCPDL-definable. Both formulas have polynomial size.*

Moreover, under the mailbox semantics, we can show:

▶ **Proposition 18.** *The set of weakly synchronous mailbox MSCs is STW-bounded (in fact, it is included in* $\mathsf{MSC}^{4|\mathbb{P}|\text{-stw}}$*).*

Proof. Let M be fixed, and let us sketch Eve's winning strategy. Let $n = |\mathbb{P}|$.

The first step for Eve is to split M in exchanges. She first disconnects the first exchange from the rest of the graph ($2n$ pebbles are needed), then she disconnects the second exchange from the rest of the graph ($2n$ pebbles needed, plus n pebbles remaining from the first round), and so on for each exchange.

So we are left with designing a winning strategy for Eve with $4n+1$ pebbles on the graph of an exchange M_0, where initially there are (at most) n pebbles placed on the first event of each process and also (at most) n pebbles placed on the last event of each process. Eve also places (at most) n pebbles on the last send event of each process and also (at most) n pebbles on the first receive event of each process. Eve erases the (at most) n $\rightarrow$-edges between the last send event and the first receive event.

We are now in a configuration that will be our invariant.

Let us fix a mailbox linearization of M_0 and let e be the first send event in this linearization.

- if e is an unmatched send of process p, Eve places her last pebble on the next send event of p (if it exists), let us call it e'. Then Eve erases the $\rightarrow$-edge (e, e'), and now e is completely disconnected, so it can be removed and the pebble can be taken back.
- if $e \lhd e'$, with e' a receive event of process q, then due to the mailbox semantics e' is the first receive event of q, so it has a pebble placed on it. Eve removes the $\lhd$-edge between e and e', then using the extra pebble she disconnects e and places a pebble on the $\rightarrow$-successor of e, then she also disconnects e' and places a pebble on the $\rightarrow$-successor of e'.

After that, we are back to our invariant, so we can repeat the same strategy with the second send event of the linearization, and so on until all edges have been erased. ◀

We obtain the following result as a corollary. Note that it assumes the mailbox semantics.

▶ **Theorem 19.** *The following problem is decidable in exponential time: Given $\mathbb{P}$, $\mathbb{M}$, and a communicating system $\mathcal{S}$ (over $\mathbb{P}$ and $\mathbb{M}$), is every MSC in $L_{\mathsf{mb}}(\mathcal{S})$ weakly synchronous?*

Proof. According to Corollary 17, we determine the LCPDL formula Φ_{wsmb} such that $L(\Phi_{\mathsf{wsmb}})$ is the set of weakly synchronous mailbox MSCs. Moreover, recall from Proposition 18 that the special tree-width of all weakly synchronous mailbox MSCs is bounded by $4|\mathbb{P}|$. By Lemma 9, $L_{\mathsf{mb}}(\mathcal{S}) \subseteq L(\Phi_{\mathsf{wsmb}})$ iff $L_{\mathsf{mb}}(\mathcal{S}) \cap \mathsf{MSC}^{(4|\mathbb{P}|+2)\text{-}\mathsf{stw}} \subseteq L(\Phi_{\mathsf{wsmb}})$. The latter is an instance of the bounded model-checking problem. As the length of Φ_{wsmb} is polynomial in $|\mathbb{P}|$, we obtain that the original problem is decidable in exponential time by Theorem 8. ◀

For the same reasons, the model-checking problem for "weakly synchronous" systems is decidable. Interestingly, a reduction from Post's correspondence problem shows that decidability fails when adopting the p2p semantics:

▶ **Theorem 20.** *The following problem is undecidable: Given finite sets $\mathbb{P}$ and $\mathbb{M}$ as well as a communicating system $\mathcal{S}$, is every MSC in $L_{\mathsf{p2p}}(\mathcal{S})$ weakly synchronous?*

4.2 Weakly k-Synchronous MSCs

This negative result for the p2p semantics motivates the study of other classes. In fact, our framework captures several classes introduced in the literature.

▶ **Definition 21** (k-exchange). *Let $M = (\mathcal{E}, \to, \lhd, \lambda)$ be an MSC and $k \in \mathbb{N}$. We call M a* k-exchange *if M is an exchange and $|SendEv(M)| \leq k$.*

Let us now recall the definition from [8, 14], but (equivalently) expressed directly in terms of MSCs rather than via *executions*. It differs from the weakly synchronous MSCs in that here, we insist on constraining the number of messages sent per exchange to be at most k.

▶ **Definition 22** (weakly k-synchronous). *Let $k \in \mathbb{N}$. We say that $M \in \mathsf{MSC}$ is weakly k-synchronous if it is of the form $M = M_1 \cdot \ldots \cdot M_n$ such that every M_i is a k-exchange.*

▶ **Example 23.** MSC M_3 in Fig. 5 is weakly 1-synchronous, as it can be decomposed into three 1-exchanges (the decomposition is depicted by the horizontal dashed lines). We remark that $M_3 \in \mathsf{MSC}_{\mathsf{mb}}$. Note that there is a p2p linearization that respects the decomposition. On the other hand, a mailbox linearization needs to reorganize actions from different MSCs: the sending of m_3 needs to be done before the sending of m_1. Note that M_1 in Fig. 1 is also weakly 1-synchronous.

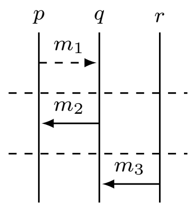


Figure 5 MSC M_3

▶ **Proposition 24.** *Let $k \in \mathbb{N}$. The set of weakly k-synchronous p2p (mailbox, respectively) MSCs is effectively MSO-definable.*

In fact, MSO-definability essentially follows from the following known theorem:

▶ **Theorem 25** ([14]). *Let M be an MSC. Then, M is weakly k-synchronous iff every SCC in its conflict graph $\mathsf{CG}(M)$ is of size at most k and no RS edge occurs on any cyclic path.*

This property is similar to the graphical characterization of weakly synchronous MSCs, except for the condition that every SCC in the conflict graph is of size at most k. Furthermore, it is easy to establish a bound on the special tree-width:

▶ **Proposition 26.** *Let $k \in \mathbb{N}$. The set of MSCs that are weakly k-synchronous have special tree-width bounded by $2k + |\mathbb{P}|$.*

Hence, we can conclude that the class of weakly k-synchronous MSCs is MSO-definable and STW-bounded. As a corollary, we get the following (known) decidability result, but via an alternative proof:

▶ **Theorem 27** ([8, 14])**.** *For* $\mathrm{com} \in \{\mathsf{p2p}, \mathsf{mb}\}$*, the following problem is decidable: Given finite sets* $\mathbb{P}$ *and* $\mathbb{M}$*, a communicating system* $\mathcal{S}$*, and* $k \in \mathbb{N}$*, is every MSC in* $L_{\mathrm{com}}(\mathcal{S})$ *weakly* k*-synchronous?*

Proof. We proceed similarly to the proof of Theorem 19. For the given $\mathbb{P}$, $\mathbb{M}$, and k, we first determine, using Proposition 24, the MSO formula φ_k such that $L(\varphi_k)$ is the set of weakly k-synchronous p2p/mailbox MSCs. From Proposition 26, we know that the special tree-width of all weakly k-synchronous MSCs is bounded by $2k + |\mathbb{P}|$. By Lemma 9, we have $L_{\mathrm{com}}(\mathcal{S}) \subseteq L(\varphi_k)$ iff $L_{\mathrm{com}}(\mathcal{S}) \cap \mathsf{MSC}^{(2k+|\mathbb{P}|+2)\text{-}\mathsf{stw}} \subseteq L(\varphi_k)$. The latter is an instance of the bounded model-checking problem. By Fact 2 and Theorem 8, we obtain decidability. ◀

▶ Remark 28. The set of weakly k-synchronous MSCs is not directly expressible in LCPDL (the reason is that LCPDL does not have a built-in counting mechanism). However, its *complement* is expressible in the extension of LCPDL with existentially quantified propositions (we need $k + 1$ of them). The model-checking problem for this kind of property is still in EXPTIME and, therefore, so is the problem from Theorem 27 when k is given in unary. It is very likely that our approach can also be used to infer the PSPACE upper bound from [8] by showing bounded *path width* and using finite word automata instead of tree automata. Finally, note that the problem to decide whether there exists an integer $k \in \mathbb{N}$ such that all MSCs in $L_{\mathrm{com}}(\mathcal{S})$ are weakly k-synchronous has recently been studied in [20] and requires different techniques.

Observe also that we can remove the constraint of all the sends preceding all the receives in a k-exchange, and still have decidability. We then have the following definition.

▶ **Definition 29** (modified k-exchange)**.** *Let* $M = (\mathcal{E}, \rightarrow, \lhd, \lambda)$ *be an MSC and* $k \in \mathbb{N}$*. We call* M *a* modified k-exchange *if* $|\mathit{SendEv}(M)| \leq k$*.*

We extend this notion to consider modified weakly k-synchronous executions as before, and the graphical characterization of this property is that there are at most k nodes in every SCC of the conflict graph. Hence, this class is also MSO-definable, and since each modified k-exchange has at most $2k$ events, it also has bounded special tree-width.

4.3 Strongly k-Synchronous MSCs and Other Classes

Our framework can be applied to a variety of other classes. Here we show how the decidability results can be shown for a variant of the class of weakly k-synchronous MSCs.

▶ **Definition 30.** *Let* $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}_{\mathsf{mb}}$*. We call* M strongly k-synchronous *if it can be written as* $M = M_1 \cdot \ldots \cdot M_n$ *such that every MSC* $M_i = (\mathcal{E}_i, \rightarrow_i, \lhd_i, \lambda_i)$ *is a* k*-exchange and, for all* $(e, f) \in \sqsubset_M$*, there are* $1 \leq i \leq j \leq n$ *such that* $e \in \mathcal{E}_i$ *and* $f \in \mathcal{E}_j$*.*

▶ **Example 31.** MSC $M_4 \in \mathsf{MSC}_{\mathsf{mb}}$ in Fig. 6 is strongly 1-synchronous. Indeed, we can decompose it into 1-exchanges and this decomposition allows for a total order compatible with $\sqsubset_{M_4}$. Moreover, MSC M_3 in Fig. 5, which is weakly 1-synchronous, is strongly 3-synchronous. Indeed, we need to put the three messages in the same k-exchange to regain our total order. Finally, for all k, MSC M_1 in Fig. 1 is not strongly k-synchronous, as we cannot put all messages in the same k-exchange, where all sends are followed by all receptions. Here, this is not possible as the reception of m_3 has to take place before the sending of m_4.

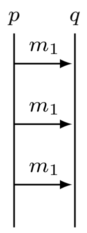


Figure 6 MSC M_4

▶ **Proposition 32.** *For all $k \in \mathbb{N}$, the set of strongly k-synchronous mailbox MSCs is MSO-definable and STW-bounded.*

The proof proceeds similarly to what has been shown in the previous cases, but MSO-definability now relies on an *extended* conflict graph. As a corollary, we thus obtain:

▶ **Theorem 33.** *The following problem is decidable: Given finite sets $\mathbb{P}$ and $\mathbb{M}$, a communicating system $\mathcal{S}$, and $k \in \mathbb{N}$, is every MSC in $L_{\mathsf{mb}}(\mathcal{S})$ strongly k-synchronous?*

▶ Remark 34. Only mailbox MSCs are considered for the definition of strongly k-synchronous MSCs for the following reason: A natural p2p analogue of Definition 30 would require from the decomposition that, for all $(e, f) \in \leq_M$, there are indices $1 \leq i \leq j \leq n$ such that $e \in \mathcal{E}_i$ and $f \in \mathcal{E}_j$. But this is always satisfied. So the natural definition of "strongly k-synchronous MSCs" would coincide with weakly k-synchronous MSCs.

Like the variant for the case of weakly synchronous MSCs, we can also generalize strongly k-synchronous MSCs by removing the restriction on the number of messages per exchange:

▶ **Definition 35.** *Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}_{\mathsf{mb}}$. We call M* strongly synchronous *if it can be written as $M = M_1 \cdot \ldots \cdot M_n$ such that every MSC $M_i = (\mathcal{E}_i, \rightarrow_i, \lhd_i, \lambda_i)$ is an exchange and, for all $(e, f) \in \sqsubset_M$, there are indices $1 \leq i \leq j \leq n$ such that $e \in \mathcal{E}_i$ and $f \in \mathcal{E}_j$.*

Similarly to the constructions for strongly k-synchronous MSCs, we can obtain a graphical characterization where we only look for the absence of RS-edges in a cycle. Hence, this class is also MSO-definable (in fact, even LCPDL-definable) and STW-bounded.

4.4 Existentially k-Bounded MSCs

Now, we turn to existentially k-bounded MSCs [18,19,24]. Synchronizability has been studied for the p2p case in [18], so we only consider the mailbox case here. A linearization $\rightsquigarrow$ of an MSC $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}$ is called *k-mailbox-bounded* if, for all $e \in \mathit{Matched}(M)$, say with $\lambda(e) = \mathit{send}(p, q, m)$, we have $\#_{\mathit{Send}(\_,q,\_)}(\rightsquigarrow, e) - \#_{\mathit{Rec}(\_,q,\_)}(\rightsquigarrow, e) \leq k$.

▶ **Definition 36.** *Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}$ and $k \in \mathbb{N}$. We call M* existentially k-mailbox-bounded *if it has some mailbox linearization that is k-mailbox-bounded.*

Note that every existentially k-mailbox-bounded MSC is a mailbox MSC.

▶ **Example 37.** MSC M_5 in Fig. 7 is existentially 1-mailbox-bounded, as witnessed by the (informally given) linearization $\mathsf{s}(q, p, m_2) \rightsquigarrow \mathsf{s}(p, q, m_1) \rightsquigarrow \mathsf{s}(q, r, m_3) \rightsquigarrow \mathsf{r}(q, r, m_3) \rightsquigarrow \mathsf{r}(p, q, m_1) \rightsquigarrow \mathsf{s}(p, q, m_1) \rightsquigarrow \mathsf{r}(q, p, m_2) \rightsquigarrow \mathsf{s}(q, r, m_3) \ldots$ Note that M_5 is neither weakly nor strongly synchronous as we cannot divide it into exchanges.

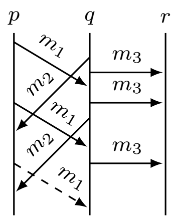


Figure 7 MSC M_5

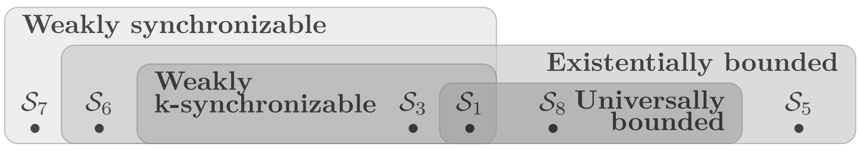


Figure 8 Hierarchy of classes for p2p systems

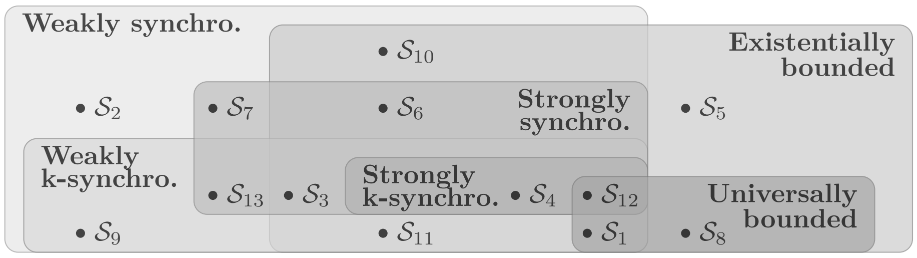


Figure 9 Hierarchy of classes for mailbox systems

▶ **Proposition 38.** *For all $k \in \mathbb{N}$, the set of existentially k-mailbox-bounded MSCs is MSO-definable and STW-bounded.*

This extension is also valid for the p2p definition of existentially k-bounded MSCs, which were addressed in [18]. Finally, our framework can also be adapted to treat universally bounded systems [21, 24].

5 Relations Between Classes

In this section we study how the classes introduced and recalled so far are related to each other. Notably, depending on the semantics (p2p or mailbox), we obtain two different classifications. The results are summed up in Figures 8 and 9. Here, we define existentially k-p2p-bounded MSCs and universally bounded counterparts as expected (formal definitions are available in Appendices C.8 and C.9).

To refer to those systems we use the following terminology: a system $\mathcal{S}$ is called weakly synchronizable (resp. strongly synchronizable) if all MSCs M in the respective language are weakly synchronous (resp. strongly synchronous). A system is called weakly k-synchronizable (resp. strongly k-synchronizable, existentially bounded or universally bounded) if all MSCs are weakly k-synchronous (resp. strongly k-synchronous, existentially k-bounded or universally k-bounded). A similar comparison relating existentially bounded systems, weakly k-synchronizable systems, as well as other systems that have not been described here, can also be found in [23] for p2p systems.

We give some results showing the inclusion of certain classes. Recall that strong k-synchronizability is tailored to mailbox systems (cf. also Remark 34) so that, for p2p systems, we only consider the case of weak (k-)synchronizability.

▶ **Proposition 39.** *Every weakly k-synchronous MSC is existentially k-p2p-bounded. Moreover, every strongly k-synchronous mailbox MSC is existentially k-mailbox-bounded.*

Finally, if a system is weakly synchronizable and universally k-bounded then, there is a k' such that it is also weakly k'-synchronizable. The equivalent property is also valid for strong classes.

▶ **Proposition 40.** *Every weakly (resp. strongly) synchronizable and universally k-bounded system is weakly (resp. strongly) k'-synchronizable for a k'.*

6 Conclusion and Perspectives

We have presented a unifying framework based on MSO logic and (special) tree-width, that brings together existing definitions, explains their good properties, and allows one to easily derive other, more general definitions and decidability results for synchronizability. Let us notice that the send-synchronizability does not fit in our framework because the question $L_{\mathsf{p2p}}(\mathcal{S}) \subseteq \mathcal{C}_0$ would be decidable (by Theorem 11), where $\mathcal{C}_0$ is the set of send-synchronizable MSCs, but this property is equivalent to checking whether the system $\mathcal{S}$ is send-synchronizable and this last property is undecidable [16].

Many other related questions could be studied in the future. For example, we could think about the hypotheses to add to our general framework to make the problem *"does there exist an $k \geq 0$ such that $L_{\mathsf{p2p}}(\mathcal{S}) \subseteq \mathcal{C}_k$?"* decidable. From very recent work [20], one knows that the problem *"does there exist an $k \geq 0$ such that the system is (weakly/strongly) k-synchronizable?"* is decidable; but it remains to be seen if it would be possible to obtain these results by showing that these properties can be expressed in a decidable extension of our framework. Let us remark that the decidability of the question whether there exists an $k \geq 0$ such that $L_{\mathsf{p2p}}(\mathcal{S}) \subseteq \mathcal{C}_k$ allows us to build a bounded model checking strategy by first deciding whether there exists such an $k \geq 0$ and then by testing if $L_{\mathsf{p2p}}(\mathcal{S}) \subseteq \mathcal{C}_k$ for $k = 0, 1, 2 \ldots$. One may use this strategy for weakly/strongly synchronizable systems, but not for existentially bounded systems (except for deadlock-free systems) or for deterministic deadlock-free universally bounded systems. In [23], Lange and Yoshida introduced an *asynchronous compatibility* property and it would also be interesting to verify whether this property could be expressed into our framework.

References

1 C. Aiswarya and Paul Gastin. Reasoning about distributed systems: WYSIWYG (invited talk). In Venkatesh Raman and S. P. Suresh, editors, *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014, December 15-17, 2014, New Delhi, India*, volume 29 of *LIPIcs*, pages 11–30. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. `doi:10.4230/LIPIcs.FSTTCS.2014.11`.

2 C. Aiswarya, Paul Gastin, and K. Narayan Kumar. Verifying communicating multi-pushdown systems via split-width. In *Automated Technology for Verification and Analysis - 12th International Symposium, ATVA 2014*, volume 8837 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2014.

3 Samik Basu and Tevfik Bultan. Choreography conformance via synchronizability. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 795–804. ACM, 2011.

4 Bernard Boigelot and Patrice Godefroid. Symbolic verification of communication protocols with infinite state spaces using qdds (extended abstract). In Rajeev Alur and Thomas A. Henzinger, editors, *Computer Aided Verification, 8th International Conference, CAV '96, New Brunswick, NJ, USA, July 31 - August 3, 1996, Proceedings*, volume 1102 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 1996. `doi:10.1007/3-540-61474-5\_53`.

5 Benedikt Bollig, Alain Finkel, and Amrita Suresh. Bounded reachability problems are decidable in FIFO machines. In Igor Konnov and Laura Kovacs, editors, *Proceedings of the 31st International Conference on Concurrency Theory (CONCUR'20)*, volume 171 of *Leibniz*

International Proceedings in Informatics, pages 49:1–49:17, Vienna, Austria, September 2020. Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2020/12861`.

6 Benedikt Bollig, Marie Fortin, and Paul Gastin. Communicating finite-state machines, first-order logic, and star-free propositional dynamic logic. *J. Comput. Syst. Sci.*, 115:22–53, 2021.

7 Benedikt Bollig and Paul Gastin. Non-sequential theory of distributed systems. *CoRR*, abs/1904.06942, 2019. URL: `http://arxiv.org/abs/1904.06942`, `arXiv:1904.06942`.

8 Ahmed Bouajjani, Constantin Enea, Kailiang Ji, and Shaz Qadeer. On the completeness of verifying message passing programs under bounded asynchrony. In Hana Chockler and Georg Weissenbacher, editors, *Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part II*, volume 10982 of *Lecture Notes in Computer Science*, pages 372–391. Springer, 2018. `doi:10.1007/978-3-319-96142-2\_23`.

9 Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *J. ACM*, 30(2):323–342, 1983. URL: `http://doi.acm.org/10.1145/322374.322380`, `doi:10.1145/322374.322380`.

10 Gérard Cécé and Alain Finkel. Verification of programs with half-duplex communication. *Information and Computation*, 202(2):166–190, November 2005. URL: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/CF-icomp05.pdf`, `doi:10.1016/j.ic.2005.05.006`.

11 Gérard Cécé, Alain Finkel, and S. Purushothaman Iyer. Unreliable channels are easier to verify than perfect channels. *Information and Computation*, 124(1):20–31, January 1996. URL: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/PS/CFP-IC96.ps`.

12 Bruno Courcelle. Special tree-width and the verification of monadic second-order graph properties. In *FSTTCS*, volume 8 of *LIPIcs*, pages 13–29, 2010.

13 Aiswarya Cyriac, Paul Gastin, and K. Narayan Kumar. MSO decidability of multi-pushdown systems via split-width. In Maciej Koutny and Irek Ulidowski, editors, *CONCUR 2012 - Concurrency Theory - 23rd International Conference, CONCUR 2012, Newcastle upon Tyne, UK, September 4-7, 2012. Proceedings*, volume 7454 of *Lecture Notes in Computer Science*, pages 547–561. Springer, 2012. `doi:10.1007/978-3-642-32940-1\_38`.

14 Cinzia Di Giusto, Laetitia Laversa, and Étienne Lozes. On the k-synchronizability of systems. In Jean Goubault-Larrecq and Barbara König, editors, *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020, Proceedings*, volume 12077 of *Lecture Notes in Computer Science*, pages 157–176. Springer, 2020. `doi:10.1007/978-3-030-45231-5\_9`.

15 Javier Esparza, Pierre Ganty, and Rupak Majumdar. A perfect model for bounded verification. In *Proceedings of the 2012 27th Annual IEEE/ACM Symposium on Logic in Computer Science*, LICS '12, pages 285–294, Washington, DC, USA, 2012. IEEE Computer Society. `doi:10.1109/LICS.2012.39`.

16 Alain Finkel and Étienne Lozes. Synchronizability of communicating finite state machines is not decidable. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 122:1–122:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

17 Alain Finkel and M. Praveen. Verification of Flat FIFO Systems. *Logical Methods in Computer Science*, 20(4), October 2020. URL: `https://lmcs.episciences.org/6839`, `doi:10.23638/LMCS-16(4:4)2020`.

18 Blaise Genest, Dietrich Kuske, and Anca Muscholl. On communicating automata with bounded channels. *Fundamenta Informaticae*, 80(1-3):147–167, 2007.

19 Blaise Genest, Anca Muscholl, and Dietrich Kuske. A kleene theorem for a class of communicating automata with effective algorithms. In Cristian Calude, Elena Calude, and Michael J. Dinneen, editors, *Developments in Language Theory, 8th International Conference, DLT 2004,*

Auckland, New Zealand, December 13-17, 2004, Proceedings, volume 3340 of *Lecture Notes in Computer Science*, pages 30–48. Springer, 2004. doi:10.1007/978-3-540-30550-7_4.

20 Cinzia Di Giusto, Laetitia Laversa, and Étienne Lozes. Guessing the buffer bound for k-synchronizability. In *Implementation and Application of Automata - 25th International Conference, CIAA 2021, Proceedings*, Lecture Notes in Computer Science. Springer, 2021. To appear.

21 Jesper G. Henriksen, Madhavan Mukund, K. Narayan Kumar, Milind Sohoni, and P.S. Thiagarajan. A theory of regular msc languages. *Information and Computation*, 202(1):1–38, 2005.

22 Dietrich Kuske and Anca Muscholl. Communicating automata, 2014.

23 Julien Lange and Nobuko Yoshida. Verifying asynchronous interactions via communicating session automata. *CoRR*, abs/1901.09606, 2019. URL: http://arxiv.org/abs/1901.09606, arXiv:1901.09606.

24 Markus Lohrey and Anca Muscholl. Bounded MSC communication. In Mogens Nielsen and Uffe Engberg, editors, *Foundations of Software Science and Computation Structures, 5th International Conference, FOSSACS 2002. Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2002 Grenoble, France, April 8-12, 2002, Proceedings*, volume 2303 of *Lecture Notes in Computer Science*, pages 295–309. Springer, 2002. doi:10.1007/3-540-45931-6_21.

25 Markus Lohrey and Anca Muscholl. Bounded MSC communication. *Inf. Comput.*, 189(2):160–181, 2004. doi:10.1016/j.ic.2003.10.002.

26 P. Madhusudan and Gennaro Parlato. The tree width of auxiliary storage. In Thomas Ball and Mooly Sagiv, editors, *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2011, Austin, TX, USA, January 26-28, 2011*, pages 283–294. ACM, 2011.

27 Robert S. Streett. Propositional dynamic logic of looping and converse. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing, May 11-13, 1981, Milwaukee, Wisconsin, USA*, pages 375–383. ACM, 1981.

28 Gregor von Bochmann. Communication protocols and error recovery procedures. *Operating Systems Review*, 9(3):45–50, 1975.

A Proof for Section 2

A.1 Proof of Lemma 3

▶ **Lemma 3.** *Every prefix of a mailbox MSC is a mailbox MSC.*

Proof. Let $M = (\mathcal{E}, \rightarrow, \lhd, \lambda) \in \mathsf{MSC}_{\mathsf{mb}}$ and $M_0 = (\mathcal{E}_0, \rightarrow_0, \lhd_0, \lambda_0)$ be a prefix of M, i.e., $\mathcal{E}_0 \subseteq \mathcal{E}$. By contradiction, suppose that M_0 is not a mailbox MSC. Then, there are distinct $e, f \in \mathcal{E}_0$ such that $e \preceq_{M_0} f \preceq_{M_0} e$ with $\preceq_{M_0} = (\rightarrow_0 \cup \lhd_0 \cup \sqsubset_{M_0})^*$. As $\mathcal{E}_0 \subseteq \mathcal{E}$, we have that $\rightarrow_0 \subseteq \rightarrow$, $\lhd_0 \subseteq \lhd$, and $\sqsubset_{M_0} \subseteq \sqsubset_M$. Finally, $\preceq_{M_0} \subseteq \preceq_M$ and M is not a mailbox MSC, which is a contradiction. ◀

B Proofs for Section 3

B.1 Proof of Theorem 8

▶ **Theorem 8.** *The bounded model-checking problem for* com = mb *is decidable. When the formulas φ are from LCPDL, then the problem is solvable in exponential time.*

Proof. Using the mailbox semantics and the MSO formula φ_{mb}, we get

$$\begin{aligned} & L_{\mathsf{mb}}(\mathcal{S}) \cap \mathsf{MSC}^{k\text{-}\mathsf{stw}} \subseteq L(\varphi) \\ \iff\ & L_{\mathsf{p2p}}(\mathcal{S}) \cap \mathsf{MSC}^{k\text{-}\mathsf{stw}} \cap L(\varphi_{\mathsf{mb}}) \subseteq L(\varphi) \\ \iff\ & L_{\mathsf{p2p}}(\mathcal{S}) \cap \mathsf{MSC}^{k\text{-}\mathsf{stw}} \subseteq L(\varphi \vee \neg\varphi_{\mathsf{mb}})\,. \end{aligned}$$

The latter is decidable due to Fact 2. Similarly, we can use the LCPDL formula Φ_{mb}, whose size is polynomial in the number of processes and messages. ◀

B.2 Proof of Lemma 10

▶ **Lemma 10.** *Let $k \in \mathbb{N}$ and $\mathcal{C} \subseteq \mathsf{MSC}^{k\text{-}\mathsf{stw}}$. For all $M \in \mathsf{MSC} \setminus \mathcal{C}$, we have $(\mathit{Pref}(M) \cap \mathsf{MSC}^{(k+2)\text{-}\mathsf{stw}}) \setminus \mathcal{C} \neq \emptyset$.*

Proof. Let k and $\mathcal{C}$ be fixed, and let $M \in \mathsf{MSC} \setminus \mathcal{C}$. Let $M' \in \mathit{Pref}(M) \setminus \mathcal{C}$ such that, for all $\leq_{M'}$-maximal events e of M', removing e (and its adjacent edge(s)) creates an MSC in $\mathcal{C}$. We obtain such an MSC by successively removing maximal events. If M' is the empty MSC, we are done, since then $M' \in (\mathit{Pref}(M) \cap \mathsf{MSC}^{k+2\text{-}\mathsf{stw}}) \setminus \mathcal{C}$. Otherwise, let e be $\leq_{M'}$-maximal and let $M'' = M' \setminus \{e\}$.

Since M' was taken minimal in terms of number of events, $M'' \in \mathcal{C}$. So Eve has a winning strategy with $k+1$ pebbles for M''. Let us design a winning strategy with $k+3$ pebbles for Eve for M', which will show the claim.

Observe that the event e occurs at the end of the timeline of a process (say p), and it is part of at most two edges:

- one with the previous p-event (if any)
- one with the corresponding send event (if e is a receive event)

Let e_1, e_2 be the two neighbors of e. The strategy of Eve is the following: in the first round, mark e, e_1, e_2, then erase the edges (e_1, e) and (e_2, e), then split the remaining graph in two parts: M'' on the one side, and the single node graph $\{e\}$ on the other side. Then Eve applies its winning strategy for M'', except that initially the two events e_1, e_2 are marked (so she may need up to $k+3$ pebbles). ◀

C Proofs for Section 4

C.1 Proof of Proposition 16

▶ **Proposition 16.** *Let M be an MSC. Then, M is weakly synchronous iff no RS edge occurs on any cyclic path in the conflict graph* $\mathsf{CG}(M)$*.*

Proof. $\Longrightarrow$ Let M be an MSC. If M is weakly synchronous, then $M = M_1 \cdot \ldots \cdot M_n$ such that every M_i is an exchange. Hence, for every vertex v of the conflict graph, there is exactly one index $\iota(v) \in \{1, \ldots, n\}$ such that $\exists \lambda^{-1}(e) \in M_{\iota(v)}$, where $e \in \mathit{Send}(\_, \_, v)$. Note that if there is an edge from v to v' in the conflict graph, some action of v must happen before some action of v', i.e., $\iota(v) \leq \iota(v')$. Furthermore, note that if $v \xrightarrow{RS} v'$, then $\iota(v) < \iota(v')$, since within an exchange all the sends precede all the receives. So an RS edge cannot occur on a cyclic path.

$\Longleftarrow$ Let M be an MSC. We assume now that the conflict graph of M does not contain a cyclic path with an RS edge. Let $V_1, \ldots, V_n$ be the set of maximal SCCs of the conflict graph, listed in some topological order. For a fixed i, let $M_i = s_1 \ldots s_m r_1 \ldots r_{m'}$ be the enumeration of the actions of the message exchanges of V_i defined by first taking all send actions of V_i obeying the relation $\leq_M$, and then all the receive actions of V_i in the same order as in $\leq_M$. Let $M' = M_1 \ldots M_n$. Then the conflict graph of M' is the same as that of M, as the permutation of actions we defined could only postpone a receive after a send of a same SCC, therefore it could only replace some $v \xrightarrow{RS} v'$ edge with an $v \xrightarrow{SR} v'$ edge between two vertices v, v' of a same SCC. However, since we assumed that the cycles (hence by extensions SCCs) do not contain RS edges, this cannot happen. Therefore M and M' have the same conflict graph, and correspond to the same MSC. Furthermore, since each M_i is a downward-closed set, M' is weakly synchronous, and so is M. ◀

C.2 Proof of Corollary 17

▶ **Corollary 17.** *The sets of weakly synchronous MSCs and weakly synchronous* mailbox *MSCs are LCPDL-definable. Both formulas have polynomial size.*

Proof. LCPDL can be used to express the graphical characterization of weakly synchronous MSCs. This follows from the formulas below. Here, we let $R = \bigvee_{a \in \mathit{Rec}(\_,\_,\_)} a$.

$$\xrightarrow{SS} = \mathsf{test}(\neg R) \cdot \rightarrow^{+} \cdot \mathsf{test}(\neg R)$$
$$\xrightarrow{RR} = \mathsf{test}(\neg R) \cdot \lhd \cdot \rightarrow^{+} \cdot \lhd^{-1} \cdot \mathsf{test}(\neg R)$$
$$\xrightarrow{RS} = \mathsf{test}(\neg R) \cdot \lhd \cdot \rightarrow^{+} \cdot \mathsf{test}(\neg R)$$
$$\xrightarrow{SR} = \mathsf{test}(\neg R) \cdot \rightarrow^{+} \cdot \lhd^{-1} \cdot \mathsf{test}(\neg R)$$
$$\xrightarrow{CG} = (\xrightarrow{SS} + \xrightarrow{RR} + \xrightarrow{RS} + \xrightarrow{SR})$$

The absence of RS edges in any cycle in the conflict graph can be expressed by the following formula:

$$\Phi_{wsync} = \neg \mathsf{Loop}\langle (\xrightarrow{CG})^{*} \cdot \xrightarrow{RS} \cdot (\xrightarrow{CG})^{*} \rangle$$ ◀

C.3 Proof of Theorem 20

▶ **Theorem 20.** *The following problem is undecidable: Given finite sets $\mathbb{P}$ and $\mathbb{M}$ as well as a communicating system $\mathcal{S}$, is every MSC in $L_{\mathsf{p2p}}(\mathcal{S})$ weakly synchronous?*

Proof. We show that the control state reachability problem for p2p weakly synchronizable systems is not decidable. This immediately shows that the model-checking problem for p2p weak synchronizable systems is not decidable. With some extra coding, it also shows that the membership problem (decide whether a given system is p2p weakly synchronizable) also is undecidable: indeed, it is enough to add a non weak synchronizable behavior after the control states for which reachability is undecidable: the system will be not weakly synchronizable iff the control states are reached.

We reduce from Post correspondence problem (PCP). Let us recall that a PCP instance consists of N pairs (u_i, v_i) of finite words over an alphabet A, and that PCP undecidability holds already for $N = 7$ and $A = \{0, 1\}$. We let the set of messages be $\{1, \ldots, N\} \uplus A \uplus \{\sharp\}$, and we consider a system with four machines: Prover1, Prover2, Verifier1, and Verifier2. We have unidirectional communication channels from provers to verifiers, so the system is weakly synchronous by construction.

Informally, the system works as follows:

- Prover1 guesses a solution $u_{i_1} \ldots u_{i_m}$ of the PCP instance, and Prover2 also guesses the same solution $v_{i_1} ... v_{i_m}$.
- Prover1 sends $u_{i_1} \ldots u_{i_n}$ to Verifier1 and sends simultaneously $i_1 \ldots i_m$ to Verifier2
- Prover2 sends $v_{i_1} \ldots v_{i_m}$ to Verifier1 and sends simultaneously $i_1 \ldots i_m$ to Verifier 2
- Verifier1 checks that the two words are equal and Verifier2 checks that the sequences of indices are equal.

Let us now formally define these machines. We describe them with regular expressions. For $w = a_1 \cdots a_n$, we write $\mathit{send}^*(p, q, w)$ (resp $\mathit{rec}^*(p, q, w)$) for $\mathit{send}(p, q, a_1) \cdots \mathit{send}(p, q, a_n)$ (resp $\mathit{rec}(p, q, a_1) \cdots \mathit{rec}(p, q, a_n)$). We abbreviate Prover1 as P1, Prover2 as P2, Verifier1 as V1, and Verifier2 as V2

- Prover1 is

$$\Big(\sum_{i=1}^{N} \mathit{send}(P_1, V_1, i)\mathit{send}^*(P_1, V_2, u_i)\Big)^+ \mathit{send}(P_1, V_1, \sharp)\mathit{send}(P_1, V_2, \sharp)$$

- Prover2 is

$$\Big(\sum_{i=1}^{N} \mathit{send}(P_2, V_1, i)\mathit{send}^*(P_2, V_2, v_i)\Big)^+ \mathit{send}(P_2, V_1, \sharp)\mathit{send}(P_2, V_2, \sharp)$$

- Verifier1 is

$$\Big(\sum_{i=1}^{N} \mathit{rec}(P_1, V_1, i)\mathit{rec}(P_2, V_1, i)\Big)^* \mathit{rec}(P_1, V_1, \sharp)\mathit{rec}(P_2, V_1, \sharp)$$

- Verifier2 is

$$\Big(\sum_{a \in \Sigma} \mathit{rec}(P_1, V_2, a)\mathit{rec}(P_2, V_2, a)\Big)^* \mathit{rec}(P_1, V_2, \sharp)\mathit{rec}(P_2, V_2, \sharp)$$

It can be checked that all machines reach their own final state if and only if the PCP instance has a solution. ◀

C.4 Proof of Proposition 24

▶ **Proposition 24.** *Let $k \in \mathbb{N}$. The set of weakly k-synchronous p2p (mailbox, respectively) MSCs is effectively MSO-definable.*

Proof. The formula for the property that there is no strongly connected component of size greater than k in the conflict graph is as follows:

$$\nexists e_1, \ldots, e_{k+1}. \, [\bigwedge_{i \neq j} \mathsf{CG}^*(e_i, e_j)]$$ ◀

C.5 Proof of Proposition 26

▶ **Proposition 26.** *Let $k \in \mathbb{N}$. The set of MSCs that are weakly k-synchronous have special tree-width bounded by $2k + |\mathbb{P}|$.*

Proof. Let M be a k-synchronous MSC. By definition, we know that $M = M_1 \cdot \ldots \cdot M_n$ such that every M_i is a k-exchange.

Eve's strategy is to mark the vertices belonging to the set M_1. Hence, she marks at most $2k$ vertices. We can remove the edges between these vertices. Let the new marked MSC fragment be (G, U), where G is the new MSC fragment (with the edges between marked vertices removed), and U the set of marked vertices.

Notice that $|U| \leq 2k$. Furthermore, since every vertex corresponding to a send message in U is either unmatched or matched with a reception in U (by definition), we can be sure that there are no message edges between vertices of U and any other vertex. Moreover, we can also be sure that there are at most $|\mathbb{P}|$ process edges between vertices in U and vertices outside this set. Let us mark these $|\mathbb{P}|$ vertices. We call these vertices U'. Let the new marked MSC fragment be $(G', U \cup U')$, where G' is the new MSC fragment (with the edges between all vertices in $U \cup U'$ removed). Now, we see that there are no edges between any of the vertices in U and any other vertex, i.e. all the vertices in U are isolated. We can divide the MSC fragment to consist of the vertices U and $V \setminus U$ and the corresponding edges.

Let the MSC fragment with vertices in U be (G_1, U_1). It consists of at most $2k$ isolated colored vertices. Let the MSC fragment with vertices in $V \setminus U$ be (G_2, U_2). We observe that $|U_2| = n$. Adam trivially loses if he chooses (G_1, U_1), hence, he has to choose (G_2, U_2). Now, we mark the vertices corresponding to M_2, which are again, at most $2k$. We have two possibilities for each vertex in U_2, either they belong to the set M_2 or belong to another set M_p where $p > 2$. However, if they belong to M_p, we can be sure that there is no other event on the same process that belongs to M_2 - this is because it was the successor of some event in M_1. Hence, we see once again, that marking all the vertices in M_2 and the immediate successors along each process will result in marked vertices of size at most $2k + |\mathbb{P}|$. And once again, we see that we can separate into MSC fragments (G_1', U_1') and (G_2', U_2') such that every vertex in U_1' is isolated, and $|U_1'| \leq 2k$. We do this for all $i \in [n]$, and hence, we can effectively use $2k + |\mathbb{P}|$ colors. Therefore, set of MSCs over $|\mathbb{P}|$ processes which are k-synchronous have bounded special tree-width. ◀

C.6 Proof of Proposition 32

As was shown in [14], in order to capture the mailbox semantics, we need extended edges. We recall from [14] the extended edge relation $\xrightarrow{XY}$ with $X, Y \in \{S, R\}$ in Figure 10. We call the conflict graph along with the new extended edges the *extended conflict graph* (ECG). This graph is also used to characterize some classes of MSCs in Section 4.3.

$$\frac{v_1 \xrightarrow{XY} v_2}{v_1 \overset{XY}{\dashrightarrow} v_2}\ \text{(Rule 1)} \qquad \frac{v \in \lhd}{v \overset{SR}{\dashrightarrow} v}\ \text{(Rule 2)} \qquad \frac{v_1 \xrightarrow{RR} v_2}{v_1 \overset{SS}{\dashrightarrow} v_2}\ \text{(Rule 3)}$$

$$\frac{v_1 \overset{XY}{\dashrightarrow} \overset{YZ}{\dashrightarrow} v_2}{v_1 \overset{XZ}{\dashrightarrow} v_2}\ \text{(Rule 4)} \qquad \frac{\begin{array}{c} e_1 \in \mathit{Matched}(M) \quad e_2 \in \mathit{Unm}(M) \\ e_1 \in \mathit{Send}(p_1, q, \_),\ e_2 \in \mathit{Send}(p_2, q, \_),\ p_1, p_2, q \in \mathbb{P} \end{array}}{\mu(e_1) \overset{SS}{\dashrightarrow} e_2}\ \text{(Rule 5)}$$

Figure 10 Additional rules for extended conflict graph; $\xrightarrow{XY}$ refers to an edge in the conflict graph

▶ **Proposition 32.** *For all $k \in \mathbb{N}$, the set of strongly k-synchronous mailbox MSCs is MSO-definable and STW-bounded.*

Similar to Theorem 25, we now show the graphical characterization of strong synchronizability.

▶ **Theorem 41** (Graphical Characterization of strongly k-synchronous MSCs)**.** *Let $M \in \mathsf{MSC}_{\mathsf{mb}}$. M is strongly k-synchronous iff every strongly connected component (SCC) in the ECG is of size at most k and no RS edge occurs on any cycle in the ECG.*

Proof. ($\implies$) Assume that we have an MSC M that is strongly k-synchronous. Hence, we can divide $M = M_1 \dots M_n$ such that each M_i is a k-exchange. By contradiction, suppose that there is an SCC of size $k' > k$ in $\mathsf{ECG}(M)$. As there are at most k messages in each k-exchange, there are v, v' which belong to the SCC such that $v \in M_i$ and $v' \in M_j$, $1 \leq i < j \leq n$. Then, we have $v \dashrightarrow^* v' \dashrightarrow^* v$.

By induction, we prove that $v' \dashrightarrow^* v$ implies that $j \leq i$.

Base There are two cases.

- Suppose that $v' \xrightarrow{XY} v$ then an action of v' is done by the same process than an action of v and it is done before it. Then, $j \leq i$.
- Suppose that $v' \overset{SS}{\dashrightarrow} v$, built by Rule 5 (because others rules do not add any edges between vertices that are not already connected), then, v' is matched and v is unmatched, such that, $v \in \mathit{Send}(q, p, v)$ and $v' \in \mathit{Send}(q', p, v')$, $q, q' \in \mathbb{P}$. Then, the send of v' has to be done before the send of v and so $j \leq i$.

Step By hypothesis, there is $v' \dashrightarrow^* v_1 \dashrightarrow v$ such that $v_1 \in M_l$, $j \leq l \leq n$. There are also two cases.

- Either $v_1 \xrightarrow{XY} v$. Then, an action of v_1 is done before and by the same process than an action of v. Then, $l \leq i$ and so $j \leq i$.
- Or $v_1 \overset{SS}{\dashrightarrow} v$. Then, similarly as before, v_1 has to be sent before v and so $l \leq i$. Therefore, $j \leq i$.

Finally, we have that $v' \dashrightarrow^* v$ implies that $j \leq i$ and so there is a contradiction.

Now, we show that there is no RS edge in any SCC. By contradiction, suppose that we have $v \xrightarrow{RS} v' \dashrightarrow^* v$ in the extended conflict graph. Then, as proved before, v and v' have to be in the same M_i, $1 \leq i \leq n$. However, $v \xrightarrow{RS} v'$ implies that the reception of v has to be done before the send of v', but a k-exchange can, by definition, be linearized with all the sends followed by all the receptions. So we have a contradiction.

($\Longleftarrow$) Conversely, assume that every SCC in the extended conflict graph of M is of size at most k and no RS edge occurs on any cyclic path in the ECG. Then, we first show that every SCC in the extended conflict graph is k-synchronous. Let C be an SCC formed of a set of nodes $v_1, \cdots, v_n$, for some $1 \leq n \leq k$ such that $s_i \in \mathit{Send}(\_,\_,v_i)$, for all $1 \leq i \leq n$. W.l.o.g., assume that the indexing of the nodes in C is consistent with the edges labeled by SS (note that there is no cycle formed only of edges labeled by SS), i.e., for every $1 \leq i_1 < i_2 \leq n$, C doesn't contain an edge labeled by SS from i_2 to i_1, and for every $1 \leq i < j < k \leq n$, if $s_i, s_k \in \mathit{Send}(p,\_,\_)$ for $p \in \mathbb{P}$ then $s_j \in \mathit{Send}(p,\_,\_)$. Let $i_1, \cdots, i_m$ be the maximal subsequence of $1, \ldots, n$ such that $r_\ell \in \mathit{Rec}(\_,\_,v_i)$ for every $\ell = i_j$ where $1 \leq j \leq m$. We have that C is the graph of the execution $e = s_{i_1} \cdots s_{i_n} r_{i_1} \cdots r_{i_m}$. The fact that all sends can be executed before the receives is a consequence of the fact that C doesn't contain edges labeled by RS. Then, the order between receives is consistent with the one between sends because C satisfies causal delivery. By definition, e is the label of an n-exchange transition, and therefore, C is strongly k-synchronous.

To complete the proof we proceed by induction on the number of strongly connected components of the extended conflict graph. The base case is for an MSC with a single SCC, which can be deduced from above. For the induction step, assume that the claim holds for every MSC whose extended conflict graph has at most n strongly connected components, and let M be a MSC with $n+1$ strongly connected components. Let C be a strongly connected component of M such that C has no outgoing edges towards another strongly connected component of M. By the definition of the extended conflict-graph, $M = M' \cdot M''$ is the MSC corresponding to the nodes of C. We have shown above that M'' is k-synchronous, and by the induction hypothesis, M' is also k-synchronous. As there is no outgoing edges from M'', we know that all messages in it have not to be done before a message of M'. Therefore, M is strongly k-synchronous. ◀

For the extended conflict graph, we use the following MSO formulas to express the edge relation. For instance, the extended SR edge relation includes all SR edges along with the set of self loops around each message that ensures that the sends are before the corresponding receives.

$$\mathsf{ESR}(e_1,e_2) = \mathsf{SR}(e_1,e_2) \ \vee \ (\exists f_1.\ [e_1 \lhd f_1 \ \wedge \ (e_1 = e_2)])$$

Similarly, the extended send edge relation includes the SS edges along with the edges produced from Rule 3 and Rule 5.

$$\mathsf{ESS}(e_1,e_2) = \mathsf{SS}(e_1,e_2) \ \vee \ \mathsf{RR}(e_1,e_2) \ \vee \ \Bigg(\exists f_1.\ [e_1 \lhd f_1 \ \wedge \ \nexists f_2.\ [e_2 \lhd f_2]] \\ \wedge \bigvee_{p,p',q \in \mathbb{P}} [(\lambda(e_1) = \mathit{Send}(p,q,\_) \ \wedge \ \lambda(e_2) = \mathit{Send}(p',q,\_))]\Bigg)$$

The extended RR and RS edges are the same as in the conflict graph.

$$\mathsf{ERR}(e_1,e_2) = \mathsf{RR}(e_1,e_2)$$
$$\mathsf{ERS}(e_1,e_2) = \mathsf{RS}(e_1,e_2)$$

The transitive closure of each of these formulas is defined as follows. It essentially takes care of Rule 4. For all $X, Y, Z \in \{R, S\}$, we have:

$$\mathsf{EXY}(e_1, e_2) \wedge \mathsf{EYZ}(e_2, e_3) \implies \mathsf{EXZ}^*(e_1, e_3)$$
$$\mathsf{EXZ}(e_1, e_2) \implies \mathsf{EXZ}^*(e_1, e_2)$$

We then extend the rest of the results, as in the case of the conflict graph in the previous section.

And finally, for the condition of the bounded STW, we observe that the set of strongly k-synchronizable MSCs are included in the set of weakly k-synchronizable MSCs. Hence, the decomposition strategy as used for the weakly k-synchronizable MSCs can be applied to the set of strongly k-synchronizable MSCs.

Therefore, the family $(\mathcal{C}_k)_{k\in\mathbb{N}}$ is MSO-definable and STW-bounded.

C.7 Proof of Proposition 38

We define the following relation in order to characterize k-mailbox-bounded MSCs.

Let $k \geq 1$, and let M be a fixed mailbox MSC. Let $\xrightarrow{\text{rev}}_k$ be the binary relation among events of M defined as follows: $r \xrightarrow{\text{rev}}_k s$ if

1. r is a receive event of a process p;
2. let r' be the k-th receive event of process p after r; then $s \lhd r'$.

▶ **Lemma 42.** *M is existential k-mailbox-bounded if and only if $\preceq_M \cup \xrightarrow{rev}_k$ is acyclic.*

Proof. Assume that M is existential k-mailbox-bounded. Let $\rightsquigarrow$ be a mailbox linearisation of M such that for all $e \in \mathit{Matched}(M)$, say with $\lambda(e) = \mathit{send}(p, q, m)$,

$$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, e) - \#_{\mathit{Rec}(-,q,\_)}(\rightsquigarrow, e) \leq k\,.$$

Then $\rightsquigarrow$ is also a linearisation of $(\preceq_M \cup \xrightarrow{\text{rev}}_k)^*$. Indeed, if it was not the case, there would be a pair of events r, s such that $r \xrightarrow{\text{rev}}_k s$ and $s \rightsquigarrow r$. But then we would have

$$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, s) - \#_{\mathit{Rec}(-,q,\_)}(\rightsquigarrow, s) > k\,,$$

and the contradiction. So $\rightsquigarrow$ is a linearisation of $(\preceq_M \cup \xrightarrow{\text{rev}}_k)^*$ and $\preceq_M \cup \xrightarrow{\text{rev}}_k$ is acyclic.

Conversely, assume that $\preceq_M \cup \xrightarrow{\text{rev}}_k$, and let $\rightsquigarrow$ be a linearisation of $(\preceq_M \cup \xrightarrow{\text{rev}}_k)^*$. In particular, $\rightsquigarrow$ is a mailbox linearisation of M. Let us show that for all $s \in \mathit{Matched}(M)$, say with $\lambda(s) = \mathit{send}(p, q, m)$,

$$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, s) - \#_{\mathit{Rec}(-,q,\_)}(\rightsquigarrow, s) \leq k\,.$$

Let $s \in \mathit{Matched}(M)$ be fixed, and let r' be such that $s \lhd r'$. There are two cases:

- $\#_{\mathit{Rec}(-,q,\_)}(\rightarrow, r') \leq k$. Then

  $$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, s) \leq k\,,$$

  because all sends before s are matched. So

  $$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, s) - \#_{\mathit{Rec}(-,q,\_)}(\rightsquigarrow, s) \leq k\,,$$

- $\#_{\mathit{Rec}(-,q,\_)}(\rightarrow, r') \leq k$. Then there is r on process q such that $r \xrightarrow{\text{rev}}_k s$. So $r \rightsquigarrow s$, and there are at most k messages in the buffer of q at the time of event s, or in other words,

  $$\#_{\mathit{Send}(-,q,\_)}(\rightsquigarrow, e) - \#_{\mathit{Rec}(-,q,\_)}(\rightsquigarrow, e) \leq k\,.$$

So $\rightsquigarrow$ is a mailbox linearisation with k bounded buffers, and M is existential k-mailbox-bounded. ◀

▶ **Proposition 38.** *For all $k \in \mathbb{N}$, the set of existentially k-mailbox-bounded MSCs is MSO-definable and STW-bounded.*

Proof. Let $k \geq 1$ be fixed. Since every existentially k-mailbox-bounded MSCs is also existentially k-p2p-bounded, and since the class of existentially k-p2p-bounded MSCs is STW bounded (cf Proposition 44), the class of existentially k-mailbox-bounded MSCs is also STW bounded.

Let us show that it is moreover MSO definable.

By Lemma 42, it is enough to show that the acyclicity of $\preceq_M \cup \xrightarrow{\text{rev}}_k$ is MSO definable, and since $\preceq_M$ was already shown MSO definable and acyclicity is easily MSO definable, it is enough to show that $\xrightarrow{\text{rev}}_k$ is MSO definable. It is indeed the case, as demonstrated by this formula

$$\varphi(r,s) = \exists r_1, r_2, \ldots, r_n . r \to r_1 \to r_2 \to \ldots \to r_n \wedge s \lhd r_n.$$

Finally, let us show that existentially k-mailbox-bounded is also LCPDL definable. This follows from the following formulas:

$$\begin{aligned}
\prec_M = &\quad (\lhd + \to)^+ \\
R = &\quad \langle \lhd^{-1} \rangle \top \\
\xrightarrow{\text{next R}} = &\quad (\to \wedge \mathsf{test}(\neg R))^* \cdot (\to \wedge \mathsf{test}(R)) \\
\xrightarrow{\text{rev}}_k = &\quad (\xrightarrow{\text{next R}})^k . (\lhd)^{-1}. \\
\Phi_{\exists k \text{ mb-bounded}} = &\quad \neg \mathsf{ELoop}\langle (\prec_M + \xrightarrow{\text{rev}}_k)^+ \rangle
\end{aligned}$$

◀

C.8 Existentially k-p2p-bounded MSCs

▶ **Definition 43.** *Let $M = (\mathcal{E}, \to, \lhd, \lambda) \in \mathsf{MSC}$ and $k \in \mathbb{N}$. A linearization $\rightsquigarrow$ of M is called* k-p2p-bounded *if, for all $e \in Matched(M)$, say with $\lambda(e) = send(p,q,m)$,*
$\#_{Send(p,q,\_)}(\rightsquigarrow, e) - \#_{Rec(p,q,\_)}(\rightsquigarrow, e) \leq k$,

We call M existentially k-p2p-bounded *if it has some p2p linearization that is k-p2p-bounded,*

▶ **Proposition 44.** *For all $k \in \mathbb{N}$, the set of existentially k-p2p-bounded MSCs is MSO-definable and STW-bounded.*

Proof. The set of existentially k-p2p-bounded MSCs was shown to be MSO-definable (in fact, even FO-definable) in [25]. Note that there are minor differences in the definitions (in particular, the fact that we deal with unmatched messages), which, however, do not affect FO-definability. In [7, Proposition 5.4, page 163], it was shown that their special tree-width is bounded by $k|\mathbb{P}|^2 + |\mathbb{P}|$. ◀

▶ **Theorem 45.** *For $\mathsf{com} \in \{\mathsf{p2p}, \mathsf{mb}\}$, the following problem is decidable: Given finite sets $\mathbb{P}$ and $\mathbb{M}$, a communicating system $\mathcal{S}$, and $k \in \mathbb{N}$, is every MSC in $L_{\mathrm{com}}(\mathcal{S})$ existentially k-p2p-bounded?*

Proof. Again, the proof follows exactly the same lines as that or Theorem 27, now using Proposition 44. ◀

Note that this is similar to the problem considered in [18, 22], though there is a subtle difference: in [18, 22], there are a notion of deadlock and distinguished final configurations.

C.9 Definition of universally bounded MSCs

▶ **Definition 46** (Universally bounded MSC). *Let $M = (\mathcal{E}, \to, \lhd, \lambda)$ and $k \in \mathbb{N}$. We call M universally k-p2p-bounded (resp., universally k-mailbox-bounded) if every p2p (resp., mailbox) linearization $\leadsto \subseteq \mathcal{E} \times \mathcal{E}$ is k-p2p-bounded (resp., k-mailbox-bounded).*

D Additional Material for Section 5

D.1 Proofs

Proof of the property from Remark 34:

▶ **Proposition 47.** *Consider an MSC of the form $M = M_1 \cdot \ldots \cdot M_n$ such that every MSC $M_i = (\mathcal{E}_i, \to_i, \lhd_i, \lambda_i)$ is a (k-)exchange. Then, for all $(e, f) \in \leq_M$, there are $1 \leq i \leq j \leq n$ such that $e \in \mathcal{E}_i$ and $f \in \mathcal{E}_j$.*

Proof. If $e \leq_M f$, then there is a sequence of events $e = e_0 \bowtie_1 e_1 \bowtie_2 \ldots \bowtie_m e_m = f$ where $\bowtie$ is either $\to$ or $\lhd$. Clearly, for every $\ell \in \{0, \ldots, m-1\}$, there are $1 \leq i \leq j \leq n$ such that $e_\ell \in \mathcal{E}_i$ and $e_{\ell+1} \in \mathcal{E}_j$. By transitivity, this proves the statement. ◀

▶ **Proposition 39.** *Every weakly k-synchronous MSC is existentially k-p2p-bounded. Moreover, every strongly k-synchronous mailbox MSC is existentially k-mailbox-bounded.*

Proof. We begin by considering a p2p MSC. Let $M \in \mathsf{MSC}$ be such that M is strongly k-synchronous. Then, there is $M = M_1 \cdots M_n$, M_i which is a k-exchange, $1 \leq i \leq n$. By induction on M:

1. **Base** $M = M_1$ then M is a k-exchange and by definition $|\mathit{Matched}(M) \cup \mathit{Unm}(M)| \leq k$. Then, for all $e \in \mathit{Matched}(M)$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$, we have

$$\#_{\mathit{Send}(p,q,\_)}(\leadsto, e) - \#_{\mathit{Rec}(p,q,\_)}(\leadsto, e) \leq k.$$

   Then, M is k-p2p-bounded.

2. **Step** $M = M' \cdot M_n$ and we suppose that $M' = M_1 \cdots M_{n-1}$ is k-p2p-bounded. For all $1 \leq i \leq n$, M_i is a k-exchange, and so an MSC, and by definition we know that any reception belongs to the same MSC as its matched send. Let $\leadsto$ be a linearization and $f \in \mathit{Matched}(M')$ s.t. $\lambda(f) = \mathit{send}(p, q, m)$ and, for all $e \in \mathit{Matched}(M')$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$,

$$\#_{\mathit{Send}(p,q,\_)}(\leadsto, f) > \#_{\mathit{Send}(p,q,\_)}(\leadsto, e).$$

   Then, we have:

$$\#_{\mathit{Send}(p,q,\_)}(\leadsto, f) - \#_{\mathit{Rec}(p,q,\_)}(\leadsto, f) = 0.$$

   Note that there is no unmatched message sent to q before f as f is matched. As M_n is a k-exchange, we have for all $e \in \mathit{Matched}(M_n)$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$

$$\#_{\mathit{Send}(p,q,\_)}(\leadsto, e) - \#_{\mathit{Rec}(p,q,\_)}(\leadsto, e) \leq k.$$

   Finally, for all $e' \in \mathit{Matched}(M)$ s.t. $\lambda(e') = \mathit{send}(p, q, m)$, there is $e \in \mathit{Matched}(M_n)$ such that we can decompose:

$$\#_{\mathit{Send}(p,q,\_)}(\leadsto, e') = \#_{\mathit{Send}(p,q,\_)}(\leadsto, f) + \#_{\mathit{Send}(p,q,\_)}(\leadsto, e)$$

and

$$\#_{Rec(p,q,\_)}(\rightsquigarrow, e') = \#_{Rec(p,q,\_)}(\rightsquigarrow, f) + \#_{Rec(p,q,\_)}(\rightsquigarrow, e).$$

Therefore

$$\#_{Send(p,q,\_)}(\rightsquigarrow, e') - \#_{Rec(p,q,\_)}(\rightsquigarrow, e') \leq k.$$

Then, M is k-p2p-bounded.

Now, we move to mailbox MSCs. Let $M \in \mathsf{MSC}_{\mathsf{mb}}$ be a strongly k-synchronous MSC. By definition, $M = M_1 \cdots M_n$ such that every $M_i = (\mathcal{E}_i, \rightarrow_i, \lhd_i, \lambda_i)$ is a k-exchange and, for all $(e,f) \in \sqsubset_M$, there are indices $1 \leq i < j \leq n$ such that $e \in \mathcal{E}_i$ and $f \in \mathcal{E}_j$.

By induction of M, we show that M is k-mailbox-bounded.

1. **Base** $M = M_1$ then M is a k-exchange and $|Matched(M) \cup Unm(M)| \leq k$. Let $\rightsquigarrow$ be any linearization of M and so, for all $e \in Matched(M)$ s.t. $\lambda(e) = send(p,q,m)$, we have $\#_{Send(\_,q,\_)}(\rightsquigarrow, e) - \#_{Rec(\_,q,\_)}(\rightsquigarrow, e) \leq k$. Then, M is k-mailbox-bounded.
2. **Step** $M = M' \cdot M_n$ and we suppose that $M' = M_1 \cdots M_{n-1}$ is k-mailbox-bounded. For all $1 \leq i \leq n$, M_i is a k-exchange, and so an MSC, and by definition we know that any reception belongs to the same MSC as its matched send. Let $\rightsquigarrow$ be a linearization and $f \in Matched(M')$ s.t. $\lambda(f) = send(p,q,m)$ and, for all $e \in Matched(M')$ s.t. $\lambda(e) = send(p,q,m)$,

   $$\#_{Send(\_,q,\_)}(\rightsquigarrow, f) > \#_{Send(\_,q,\_)}(\rightsquigarrow, e).$$

   Then, we have:

   $$\#_{Send(\_,q,\_)}(\rightsquigarrow, f) - \#_{Rec(\_,q,\_)}(\rightsquigarrow, f) = 0.$$

   Note that there is no unmatched message sent to q before f as f is matched. As M_n is a k-exchange, we have for all $e \in Matched(M_n)$ s.t. $\lambda(e) = send(p,q,m)$

   $$\#_{Send(\_,q,\_)}(\rightsquigarrow, e) - \#_{Rec(\_,q,\_)}(\rightsquigarrow, e) \leq k.$$

   Finally, for all $e' \in Matched(M)$ s.t. $\lambda(e') = send(p,q,m)$, there is $e \in Matched(M_n)$ such that we can decompose:

   $$\#_{Send(\_,q,\_)}(\rightsquigarrow, e') = \#_{Send(\_,q,\_)}(\rightsquigarrow, f) + \#_{Send(\_,q,\_)}(\rightsquigarrow, e)$$

   and

   $$\#_{Rec(\_,q,\_)}(\rightsquigarrow, e') = \#_{Rec(\_,q,\_)}(\rightsquigarrow, f) + \#_{Rec(\_,q,\_)}(\rightsquigarrow, e).$$

   Therefore

   $$\#_{Send(\_,q,\_)}(\rightsquigarrow, e') - \#_{Rec(\_,q,\_)}(\rightsquigarrow, e') \leq k.$$

   Then, M is k-mailbox-bounded.

◀

▶ **Proposition 40.** *Every weakly (resp. strongly) synchronizable and universally k-bounded system is weakly (resp. strongly) k'-synchronizable for a k'.*

Proof. Let $\mathcal{S}$ be a system such that, for all $M \in \mathsf{MSC}$, $M = M_1 \cdots M_n$ where M_i is an exchange, $1 \leq i \leq n$. Moreover, for all $e \in \mathit{Matched}(M)$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$,

- if $M \in \mathsf{MSC} \setminus \mathsf{MSC}_{\mathsf{mb}}$, for all $\rightsquigarrow \subseteq \leq_M$,

$$\#_{Send(p,q,\_)}(\rightsquigarrow, e) - \#_{Rec(p,q,\_)}(\rightsquigarrow, e) \leq k.$$

- if $M \in \mathsf{MSC}_{\mathsf{mb}}$, for all $\rightsquigarrow \subseteq \preceq_M$,

$$\#_{Send(\_,q,\_)}(\rightsquigarrow, e) - \#_{Rec(\_,q,\_)}(\rightsquigarrow, e) \leq k.$$

1. **Base** Suppose that $M = M_1$ so M is an exchange.
   - Either $M \in \mathsf{MSC} \setminus \mathsf{MSC}_{\mathsf{mb}}$ and, as M is k-p2p-bounded, $\mid M \mid = k_1 \leq k \times \mid \mathbb{P} \mid^2$. So M is weakly k_1-synchronous.
   - Or $M \in \mathsf{MSC}_{\mathsf{mb}}$ and, as M is k-mailbox-bounded, $\mid M \mid = k_2 \leq k \times \mid \mathbb{P} \mid$. So M is strongly k_2-synchronous.
2. **Step** Suppose now that $M = M' \cdot M''$ such that M' is weakly k-synchronous for a $k' \in \mathbb{N}$. Then,
   - if $M \in \mathsf{MSC} \setminus \mathsf{MSC}_{\mathsf{mb}}$, for all linearizations $\rightsquigarrow \subseteq \leq_M$, let $f \in \mathit{Matched}(M')$ s.t. $\lambda(f) = \mathit{send}(p, q, m)$ and, for all $e \in \mathit{Matched}(M')$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$,

     $$\#_{Send(p,q,\_)}(\rightsquigarrow, f) > \#_{Send(p,q,\_)}(\rightsquigarrow, e).$$

     Then, we have:

     $$\#_{Send(p,q,\_)}(\rightsquigarrow, f) - \#_{Rec(p,q,\_)}(\rightsquigarrow, f) = 0.$$

     As $\mathcal{S}$ is universally k-bounded, $\mid M'' \mid \leq k_1$, and as M'' is an exchange, we know that M'' is a k_1-exchange.
   - if $M \in \mathsf{MSC}_{\mathsf{mb}}$, for all linearizations $\rightsquigarrow \subseteq \preceq_M$, let $f \in \mathit{Matched}(M')$ s.t. $\lambda(f) = \mathit{send}(p, q, m)$ and, for all $e \in \mathit{Matched}(M')$ s.t. $\lambda(e) = \mathit{send}(p, q, m)$,

     $$\#_{Send(\_,q,\_)}(\rightsquigarrow, f) > \#_{Send(\_,q,\_)}(\rightsquigarrow, e).$$

     Then, we have:

     $$\#_{Send(\_,q,\_)}(\rightsquigarrow, f) - \#_{Rec(\_,q,\_)}(\rightsquigarrow, f) = 0.$$

     As $\mathcal{S}$ is universally k-bounded, $\mid M'' \mid \leq k_2$, and as M'' is an exchange, we know that M'' is a k_2-exchange.

   Then, M is at least weakly k_1-synchronous ($k_1 > k_2$).

Finally, as all MSCs are weakly k_1-synchronous, $\mathcal{S}$ is weakly k_1-synchronizable.

The equivalent proposition for strong properties can be shown in the same way. As an MSC is strongly synchronizable, it can be divided while maintaining the mailbox order. In a recursive way, as MSC is universally k-bounded, we have that each exchange of the MSC is bounded. Finally, each MSC is strongly k'-synchronous for a k' depending of k and the number of channels, and so the system is strongly k'-synchronizable.

◀

D.2 Examples

D.2.1 p2p systems

For p2p semantics, weakly synchronizable and strongly synchronizable systems form a single class, as strongly k-synchronizable and weakly k-synchronizable form a single class too. Indeed, as a consequence of Proposition 47, weakly k-synchronizable systems are strongly k-synchronizable, and weakly synchronizable are strongly synchronizable. Therefore, for this p2p part, we will only talk about weak classes. We will see that this is not the case for mailbox systems.

However, universally bounded and existentially bounded classes are not equal, an universally bounded system is existentially bounded by definition but we can find a existentially bounded system, as system $\mathcal{S}_5$ in Fig. 11 where, as we can see in a corresponding MSC in Fig. 7, an unbounded number of m_3 can be sent before be read by r, which prevent the system to be universally bounded.

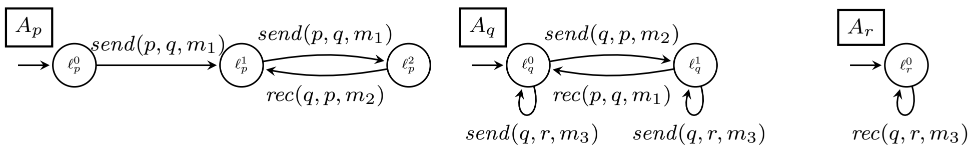


Figure 11 System $\mathcal{S}_5$

By definition, weakly k-synchronizable systems are weakly synchronizable. Also, strongly k-synchronizable systems are included into existentially k-bounded systems, as proved by Proposition 39, so weakly k-synchronizable are included into existentially k-bounded.

We can see that weakly synchronizable systems and existentially bounded systems are incomparable. System $\mathcal{S}_6$ in Fig. 12 is weakly synchronizable because we can send all messages before read them, and existentially 1-bounded because each MSC of $L_{\mathsf{p2p}}(\mathcal{S}_6)$ has a linearization of the form $send(q,p,m_2) \cdot (send(p,q,m_1) \cdot rec(p,q,m_1))^* rec(q,p,m_2)$, allowing to have in each channel only one pending message.

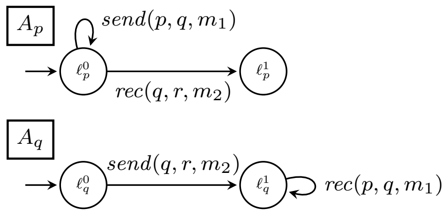


Figure 12 System $\mathcal{S}_6$

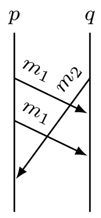


Figure 13 MSC M_6

But, system $\mathcal{S}_7$ in Fig.14 is only weakly synchronizable. Indeed, for each execution we can add an iteration of message m_1 or m_2, or both, and, as we can see in Fig. 15, we need to send all messages before begin to read, and so each execution need a bigger channel than the previous one.

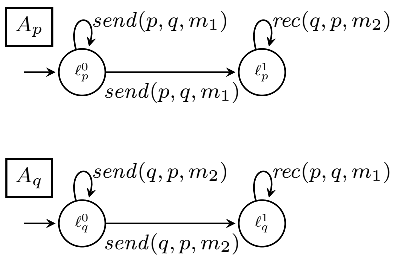


Figure 14 System $\mathcal{S}_7$

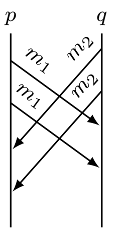


Figure 15 MSC M_7

For system $\mathcal{S}_8$ in Fig. 16, we can see an example of MSC in Fig. 17 which cannot be divided into exchanges as send and receptions are intertwined and so $\mathcal{S}_8$ is not weakly synchronizable.

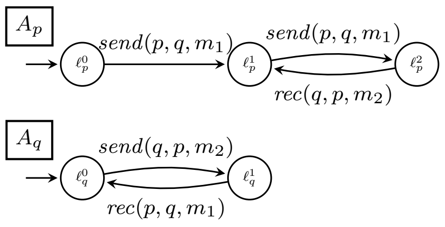


Figure 16 System $\mathcal{S}_8$

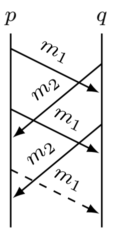


Figure 17 MSC M_8

We can observe that $\mathcal{S}_8$ is universally 3-bounded as each time a message is sent, one is received and the maximal number of messages in a buffer is 3. As we said, $\mathcal{S}_7$ is weakly synchronizable but not existentially bounded and so not universally bounded, so weakly synchronizable and universally bounded systems are incomparable. However, as proved in Proposition 40, a system which is both is also weakly k-synchronizable (not necessary for the same k).

Finally, weakly k-synchronizable and universally k-bounded systems are incomparable. System $\mathcal{S}_1$ in Fig. 2 is both weakly 1-synchronizable and universally 1-bounded. But, we have, for example, system $\mathcal{S}_3$ below which is weakly 1-synchronizable but not universally k-bounded for any k. Indeed, each execution can be rescheduled to have all the receptions just after the respective sends. Then each MSC can be divided into 1-exchanges (for an example, see MSC M_3 in Fig. 5). However, as we can send an unbounded number of m_2 messages before reading them, the size of channel c_p can also be unbounded thus the system is not universally bounded. Conversely, as we have seen, system $\mathcal{S}_8$ is universally 3-bounded but not weakly k-synchronizable for any k.

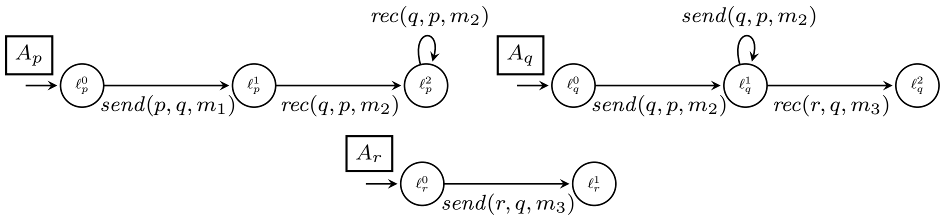


Figure 18 System $\mathcal{S}_3$

D.2.2 Mailbox systems

Now consider mailbox semantics. As depicted in Fig. 9, we can now distinguish between weakly and strongly synchronizable systems.

Some inclusions are obvious, by definition of the classes:

- universally bounded systems are existentially bounded, but it is a proper inclusion. Indeed, a system, as system $\mathcal{S}_5$ in Fig. 11, can be existentially 1-bounded, but not universally bounded. In this case, as in p2p semantics, an unbounded number of message m_3 can be sent before be read, as we can see in MSC M_5 in Fig. 7.
- strongly k-synchronizable systems are strongly synchronizable. As well, a system can be strongly synchronizable without have a bound on the size of its exchange, as system $\mathcal{S}_7$ in Fig 14. See an example of MSC of $\mathcal{S}_7$ in Fig 15, where we can always build a bigger exchange adding iterations of messages m_1 or m_2.
- weakly k-synchronizable systems are weakly synchronizable. We can also find a weakly synchronizable system without bound on the size of its exchange. Let see $\mathcal{S}_2$ in Fig. 19 with MSC M_2 in Fig. 4, which can be divided into exchanges, each message can be in a separate exchange, except messages m_2 and m_3 that have to be all in the same exchange. As we can have as many repetitions of them, this exchange no have bound on its size and prevent the system to be weakly k-synchronizable for a precise k.

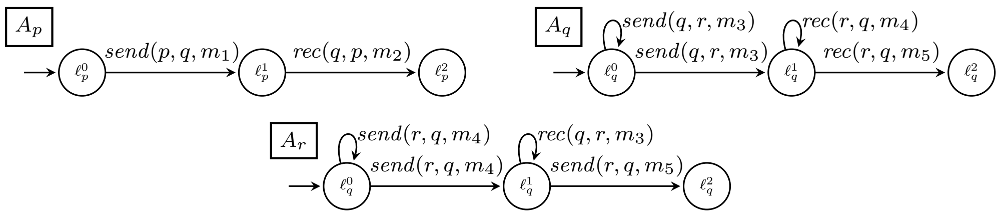


Figure 19 System $\mathcal{S}_2$

The classes of weakly synchronizable and weakly k-synchronizable systems are incomparable with the classes of existentially bounded and universally bounded systems.

Indeed, let see system $\mathcal{S}_8$ in Fig. 16 which is universally 3-bounded. It cannot be weakly synchronizable (and so weakly k-synchronizable for a k) as we cannot divide a corresponding MSC, as M_8 in Fig. 17, into exchanges, where all sends have to be before all receptions, as in p2p semantics. We see a difference with the p2p semantics looking at $\mathcal{S}_9$ in Fig. 20 which is weakly synchronizable but not existentially bounded. Indeed, we can see in Fig. 21 that M_9 can be divided easily into 1-exchange but, as m_1 has to be sent before m_4, and m_2 has to be sent to send m_4, it means that all messages m_2 have to wait the send of m_4 to be read. Then, each execution is x-mailbox-bounded, where x is the number of repetitions of m_2. Then, there is no bound on the buffers and $\mathcal{S}_9$ cannot be existentially bounded (and so universally bounded).

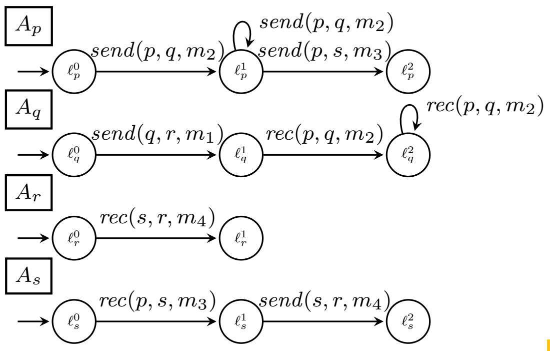


Figure 20 System $\mathcal{S}_9$

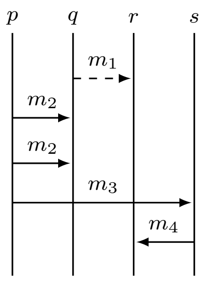


Figure 21 MSC M_9

Finally, as proved in Proposition 40, if a system is weakly synchronizable and universally bounded system, there is a k such that it is also strongly k-synchronizable. In all the other intersections, we can find a system:

- weakly synchronizable and existentially bounded (but not weakly k-synchronizable or universally bounded): $\mathcal{S}_{10}$ in Fig. 22 is existentially 1-bounded but there is no bound on the size of the exchange containing messages m_3 and m_4 and so $\mathcal{S}_{10}$ cannot be weakly k-synchronizable, and repetitions of m_3 prevent it to be universally bounded;

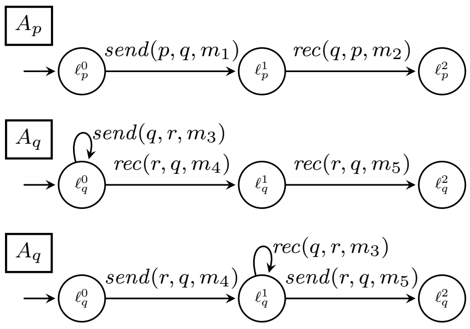


Figure 22 System $\mathcal{S}_{10}$

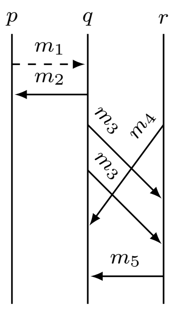


Figure 23 MSC M_{10}

- weakly k-synchronizable and existentially bounded (but not universally bounded): $\mathcal{S}_{11}$ in Fig. 24 is weakly 1-synchronizable and existentially 1-bounded but repetitions of m_3 prevent it from being universally bounded;

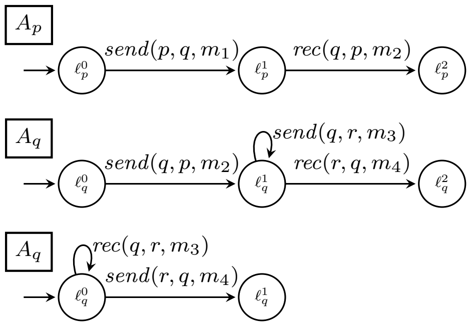


Figure 24 System $\mathcal{S}_{11}$

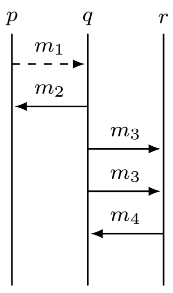


Figure 25 MSC M_{11}

- weakly k-synchronizable and universally bounded as $\mathcal{S}_8$ in Fig. 2.

Now, focus on the strong classes. We know by Proposition 39 that a strongly k-synchronizable is existentially bounded. So let compare strongly k-synchronizable systems with universally bounded ones. We can see that there are incomparable. $\mathcal{S}_4$ in Fig. 26 is strongly 1-synchronizable, as we can see with M_4 in Fig. 6, but cannot be universally bounded as the unbounded iterations of m_1 can be stored before begin to read. As we see before $\mathcal{S}_7$ is only universally bounded and cannot have any synchronizable property. But, $\mathcal{S}_{12}$ in Fig. 27 is both strongly 1-synchronizable and universally 1-bounded.

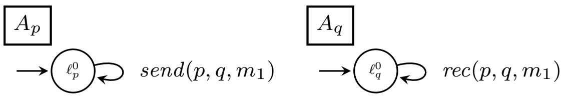


Figure 26 System $\mathcal{S}_4$

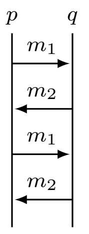


Figure 28 MSC M_{12}

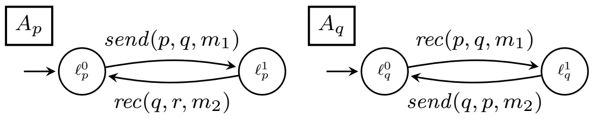


Figure 27 System $\mathcal{S}_{12}$

Similarly, strongly synchronizable systems are incomparable with existentially bounded ones. $\mathcal{S}_{13}$ in Fig. 29 is strongly synchronizable (and also weakly 1-synchronizable) but, to get an execution from M_{13} we need to send message m_5 before m_1 and, as we see with M_6 previously, the number of iterations of m_2 becomes the bound of the channels for this execution. As it is unbounded, the system cannot be existentially bounded.

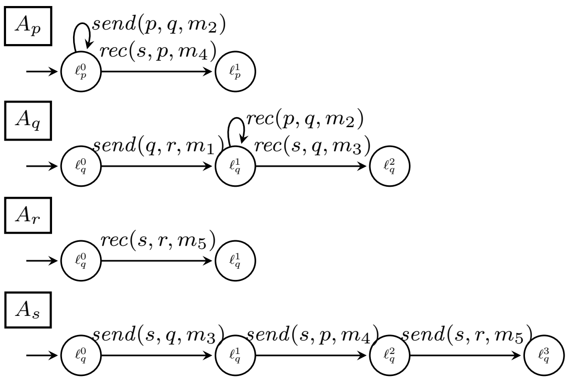


Figure 29 System $\mathcal{S}_{13}$

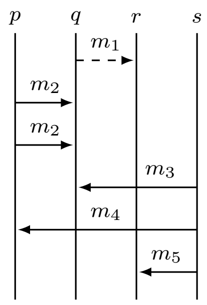


Figure 30 MSC M_{13}

$\mathcal{S}_{10}$ in Fig. 22 is existentially bounded and weakly synchronizable, but, again, message m_5 have to be sent before m_1, so all messages in M_{10} have to be in the same exchange. However, on process r, receptions of m_3 precedes send of m_5 and so this is not an exchange, and the system is not strongly synchronizable.

For the intersection, as with weak, a strongly synchronizable and existentially bounded systems is always strongly k-synchronizable for a k, by Proposition 40. We can have for example, $\mathcal{S}_2$ which is strongly synchronizable, universally bounded, weakly 1-synchronizable but, as an exchange have to contain all messages that we have in M_3, and we can repeat m_2 as many times we want, an exchange in mailbox is no bounded, and the system is not strongly k-synchronizable for any k. We can also have $\mathcal{S}_6$ in Fig. 12 which is strongly synchronizable and existentially bounded but, as one execution is always an exchange, and it can grow without limits, we cannot be neither strongly k-synchronizable nor weakly k-synchronizable for any k.