Logic for Communicating Automata with Parameterized Topology

Benedikt Bollig
LSV, ENS Cachan, CNRS & Inria, France
bollig@lsv.ens-cachan.fr

Abstract

We introduce parameterized communicating automata (PCA) as a model of systems where finite-state processes communicate through FIFO channels. Unlike classical communicating automata, a given PCA can be run on any network topology of bounded degree. The topology is thus a parameter of the system. We provide various Büchi-Elgot-Trakhtenbrot theorems for PCA, which roughly read as follows: Given a logical specification φ and a class of topologies $\mathfrak{T}$, there is a PCA that is equivalent to φ on all topologies from $\mathfrak{T}$. We give uniform constructions which allow us to instantiate $\mathfrak{T}$ with concrete classes such as pipelines, ranked trees, grids, rings, etc. The proofs build on a locality theorem for first-order logic due to Schwentick and Barthelmann, and they exploit concepts from the non-parameterized case, notably a result by Genest, Kuske, and Muscholl.

Categories and Subject Descriptors F.1.1 [*Computation by Abstract Devices*]: Models of Computation; F.3.1 [*Logics and Meanings of Programs*]: Specifying and Verifying and Reasoning about Programs

Keywords communicating automata, parameterized topology, message sequence charts, monadic second-order logic, realizability

1. Introduction

The Büchi-Elgot-Trakhtenbrot theorem states that finite automata and monadic second-order (MSO) logic over words are expressively equivalent [9, 14, 31]. This connection between automata and logic constitutes one of the cornerstones in theoretical computer science, as it bridges the gap between high-level specifications and operational system models. Various extensions of that result followed, providing logical characterizations of tree automata [29], asynchronous automata [32], and graph acceptors [30], to mention just a few.

In recent years, an analogous question has been studied for communicating automata (CA). A CA consists of several finite-state processes that can exchange messages through FIFO channels by performing send and receive actions. A single execution of a CA is captured by a message sequence chart (MSC), a finite directed acyclic graph, whose nodes represent the *events* that are observed during an execution. Its edge relation $\lhd = \lhd_{\mathsf{proc}} \cup \lhd_{\mathsf{msg}}$ visualizes causal dependencies between events. Edges from $\lhd_{\mathsf{proc}}$ connect consecutive events performed by a process, and edges from $\lhd_{\mathsf{msg}}$ connect send events with their corresponding receives. Logical characterizations have been established for unrestricted CA [6] and channel-bounded CA [19, 23, 26]. All these results require the underlying communication topology, which provides a set of processes and channels between them, to be *fixed*.

Now, it is a natural question to ask for an automaton that realizes a given logical specification for a *class* of topologies (for example, all grid topologies, no matter what the size of the grid is). This is what this paper is about, i.e., we aim for Büchi-Elgot-Trakhtenbrot theorems in a setting with non-fixed, *parameterized* topology.

In a first step, we introduce parameterized communicating automata (PCA). Unlike classical CA, a given PCA can be run on any network topology of *bounded degree* (such as pipelines, ranked trees, grids, or rings). PCA are a conservative extension of CA and, as such, also recognize sets of MSCs. Our study is centered around the following question, which depends on a given logical specification φ and a given class $\mathfrak{T}$ of topologies:

> *Is there a PCA $\mathcal{A}$ that is equivalent to φ on all topologies $\mathcal{T} \in \mathfrak{T}$ (when $\mathcal{A}$ is run on $\mathcal{T}$, it accepts precisely the MSCs over $\mathcal{T}$ that satisfy φ)?*

If the answer is affirmative, then we say that formula φ is *realizable* for $\mathfrak{T}$. This paper investigates realizability wrt. several logics and instances of $\mathfrak{T}$ in a unifying framework. We consider standard first-order and existential MSO logic, FO[σ] and EMSO[σ], respectively. Here, $\sigma \subseteq \{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$ is the collection of binary relation symbols that are available in the logic. All symbols are self-explanatory apart from $\sim$, which allows us to say that two events are executed by the same process.

Our first results settle the limits of what we can hope for:

(i) There is an FO[$\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}$]-formula that is *not* realizable for the class of "ring forests" (unions of ring topologies).

(ii) There is an FO[$\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*$]-formula that is *not* realizable already for the class of binary trees.

This shows that we have to restrict both, the topologies and the logic, and that we are left with only a small margin for positive results. However, we are able to provide various Büchi-Elgot-Trakhtenbrot theorems:

(iii) Every EMSO[$\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \sim$]-formula is realizable for the classes of pipelines, ranked trees, grids, and rings.

(iv) When we suppose that the channels of a PCA are bounded, then every EMSO[$\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}$]-formula is realizable for the classes of pipelines, ranked trees, grids, and rings.

CSL-LICS 2014, July 14–18, 2014, Vienna, Austria.

http://dx.doi.org/10.1145/2603088.2603093

Note that the logic used in (iii) is a priori weaker than the logic from (iv). The predicate $\sim$ allows one to say that two events are located on the same process. However, unlike $\lhd^*_{\mathsf{proc}}$, it cannot impose a particular order on them.

In fact, we obtain (iii) and (iv) as *corollaries* of more general, uniform statements: it is shown that every EMSO formula is realizable for $\mathfrak{T}$ whenever $\mathfrak{T}$ is *unambiguous*. Intuitively, this rules out cycle patterns that a PCA is not able to detect on its own. Indeed, the classes of pipelines, ranked trees, and grids are all unambiguous so that we get realizability as a direct corollary. To capture also the class of rings, some additional arguments are needed.

Note that, for (iii) and (iv) to apply, a class $\mathfrak{T}$ of topologies has to be *fixed in advance*. The construction of a PCA $\mathcal{A}$ from a formula φ is uniform, but it crucially depends on $\mathfrak{T}$. That is, though $\mathcal{A}$ can still be run on any other topology of bounded degree, it is only guaranteed to be equivalent to φ when it is applied to a topology from $\mathfrak{T}$. Now, when we fix $\mathfrak{T}$, we have in mind that we run $\mathcal{A}$ only on topologies $\mathcal{T} \in \mathfrak{T}$. For that reason, $\mathcal{A}$ does not have to check membership of $\mathcal{T}$ in $\mathfrak{T}$. However, it will have to collect some topological information to identify bounded subtopologies among those from $\mathfrak{T}$. In fact, the translation from logic to PCA builds on a locality theorem for first-order logic due to Schwentick and Barthelmann, which states that satisfaction of a formula in a structure can be reduced to satisfaction of a normal-form formula in bounded portions of the same structure [28]. This allows us to apply notions from the setting with fixed topologies, notably a result by Genest, Kuske, and Muscholl [19]. Hereby, the assumption that topologies are of bounded degree is crucial.

For the logic EMSO$[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$, we provide a variation of the theme: Every formula φ can be translated into a PCA that is equivalent to φ on all *prime* topologies. In that case, the construction is *independent* of a concrete class of topologies (once the bound on the degree has been fixed). Intuitively, a topology is prime if none of its cycles has a periodic labeling. For example, a ring is not prime, while pipelines, trees, and grids are all prime.[1]

Finally, every PCA $\mathcal{A}$ can be transformed into a formula from EMSO$[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ that is equivalent to $\mathcal{A}$ on *all* topologies of bounded degree. Thus, overall, we indeed establish a variety of Büchi-Elgot-Trakhtenbrot theorems for PCA.

Related Work It seems that neither PCA nor expressiveness of parameterized systems in general in terms of logic have been considered in the literature.

For classical CA, the term *realizability* was coined by Alur, Etessami, and Yannakakis [2], who formulated it as a decision problem: Given an MSC graph (a sort of regular expression over MSCs), is there a CA that generates the same set of MSCs? Note that we do not study a decision problem here, and our model of (parameterized) CA is different in that messages can carry information that is abstracted away in the observable behavior. Rather, our work is in line with [6, 19, 23, 26], which aim at a logical characterization of CA. Note that, like [6, 19, 23], we assume a semantics in terms of *finite* MSCs, while [26] considers infinite behaviors.

In [24], Jacobs and Bloem study parameterized synthesis, where a temporal-logic specification is transformed into a system of processes that are arranged in a token ring of arbitrary size. Building on [15], the idea is to reduce parameterized synthesis to distributed synthesis over a bounded architecture. Though we also use a reduction to a bounded case, our framework differs from [24] in the model (asynchronous rather than token communication), in the topologies, and in the logic.

In parameterized verification, one aims at showing that a *given* system is correct independently of the number of processes or the communication topology [1, 7, 8, 11–13, 21]. Our approach is different, since we *generate* a system model from a high-level specification.

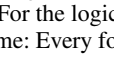
There have been a variety of automata constructions that exploit normal forms of first-order logic [17, 28, 30]. We actually borrow a technique from [17], but the overall framework is quite different.

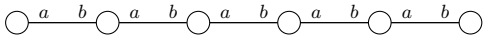
Finally, our contribution intersects the area of distributed algorithms. Indeed, the way a PCA evaluates a (sub)topology is similar to constructing a map of an *anonymous graph* [10]. In particular, our notion of unambiguous classes of topologies is in the spirit of universal sequences. There are also methods to evaluate graphs versus logical specifications [22]. Though all those techniques do not seem to be directly applicable, it will be worthwhile to explore possible connections further.

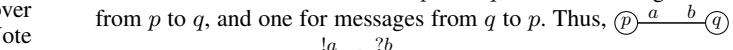
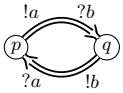
Outline Sections 2–4 settle basic notions such as topologies, MSCs, PCA, and MSO logic. In Section 5, we argue that we will have to restrict both topologies and logic. Sections 6 and 7 present the above-mentioned Büchi-Elgot-Trakhtenbrot theorems, respectively. We conclude in Section 8. Due to space constraints, most proofs are only sketched or omitted. All details can be found in the technical report available at the following link: `http://hal.archives-ouvertes.fr/hal-00872807/`

2. Preliminaries

2.1 Communication Topologies

A (communication) topology[2] is made up of single entities such as  . Here, a process (represented by the circle) is equipped with two interfaces, a and b. The interfaces allow the process to communicate with its environment. When they are connected to interfaces of other processes, we obtain a topology. A simple pipeline topology is depicted below.



Thus, a topology is essentially a graph, whose nodes are processes that can communicate with adjacent processes via their interfaces. The pipeline, for example, will allow a process to execute actions $!a$ and $?a$ in order to send a message to (receive a message from, respectively) its right neighbor, if it exists. Accordingly, $!b$ and $?b$ refer to the left neighbor. Actually, we assume that any two processes p and q that are adjacent in the topology communicate through (a priori unbounded) FIFO channels. More precisely, there are two FIFO channels between p and q, one for messages sent from p to q, and one for messages from q to p. Thus,  can be understood as  .

Let us define topologies formally. Throughout the paper, unless stated otherwise, we fix a nonempty finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of *(interface) names*. When we talk about a concrete process, we may also say *interface* instead of name.

Definition 1. *A* topology *over $\mathcal{N}$ is a pair $\mathcal{T} = (P, \longmapsto)$ where*

- *P is the nonempty finite set of* processes*, and*
- *$\longmapsto \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the* edge relation*.*

[1] "Prime" is a property of *single* topologies, while "unambiguous" refers to *sets* of topologies.

[2] What we call topology is sometimes termed *architecture*. It seems that, however, in a parameterized setting, the term topology is more custom. Actually, our definition does not quite correspond to architectures from the literature, which often assume an explicit set of *channels*. In our work, channels arise implicitly when processes are connected via their *interfaces*. Interfaces support the view that a process is specified independently of a concrete topology.

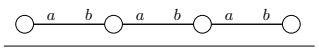


Figure 1. Pipeline $\mathcal{T}^4_{\text{lin}}$

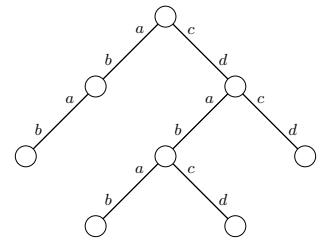


Figure 2. Tree topology

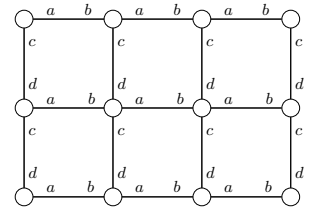


Figure 3. Grid $\mathcal{T}^{3,4}_{\text{grid}}$

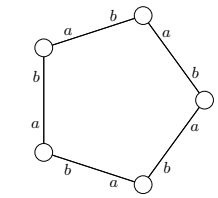


Figure 4. Ring $\mathcal{T}^5_{\text{ring}}$

We write $p \overset{a\ b}{\longmapsto} q$ *for* $(p, a, b, q) \in \longmapsto$*, which signifies that the a-interface of p points to q, and the b-interface of q points to p. We require that, whenever* $p \overset{a\ b}{\longmapsto} q$*, the following hold:*

(a) $p \neq q$,
(b) $q \overset{b\ a}{\longmapsto}$ *p, and*
(c) *for all* $a', b' \in \mathcal{N}$ *and* $q' \in P$ *such that* $p \overset{a'\ b'}{\longmapsto} q'$*, we have* $a = a'$ *iff* $q = q'$.

By (a), a topology does not contain self-loops. Condition (b) says that two adjacent processes are *mutually* connected (in other words, a topology is "undirected"). By (c), a name points to at most one process, and two distinct names point to distinct processes.

We usually consider topologies up to isomorphism. The set of all topologies over $\mathcal{N}$ is denoted by $\mathbb{T}_{\mathcal{N}}$.

Given a topology $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$, a PCA will run identical subautomata on processes of the same *type*. We define $type_{\mathcal{T}} : P \to 2^{\mathcal{N}}$ by $type_{\mathcal{T}}(p) := \{a \in \mathcal{N} \mid \text{there are } b \in \mathcal{N} \text{ and } q \in P \text{ such that } p \overset{a\ b}{\longmapsto} q\}$. Thus, $type_{\mathcal{T}}(p)$ contains those interfaces of p that are connected to some other process.

Remark 1. One can also define types independently of $\mathcal{N}$, in terms of an extra finite set $Types$, and include a mapping $type : P \to Types$ in the topology. In our setting, this can be encoded by choosing $\mathcal{N} \times Types$ as new set of names. In a topology, an edge $p \overset{a\ b}{\longmapsto} q$ is then replaced by an edge with names $(a, type(p))$ and $(b, type(q))$. All definitions can be adapted easily, and all results hold verbatim in this alternative setting.

Example 1. Let us identify some typical topology classes. We give an informal description. The precise definitions are as expected.

Pipelines: A pipeline is a topology over $\{a, b\}$, as already indicated above. Recall that interface a points to the right neighbor of a process (if it exists), while b is connected to the left neighbor. Accordingly, the leftmost process has type $\{a\}$, the rightmost process has type $\{b\}$, and all inner processes have type $\{a, b\}$. The pipeline with $n \geq 2$ processes is denoted by $\mathcal{T}^n_{\text{lin}}$. Figure 1 depicts $\mathcal{T}^4_{\text{lin}}$. We let $\mathfrak{T}_{\text{lin}} = \{\mathcal{T}^n_{\text{lin}} \mid n \geq 2\} \subseteq \mathbb{T}_{\{a,b\}}$ denote the set of all pipelines.

Trees: We suppose that trees are binary, but we could consider arbitrary ranked alphabets. An example tree is depicted in Figure 2. Hence, a tree is a topology over $\{a, b, c, d\}$ where interface a points to the left son, and c to the right son of a process, while b and d are their respective "dual" interfaces. We suppose that a tree has at least two processes. The type of a leaf is either $\{b\}$ or $\{d\}$. The type of the root is either $\{a\}$, $\{c\}$, or, as is the case in Figure 2, $\{a, c\}$. The set of all tree topologies is denoted by $\mathfrak{T}_{\text{tree}} \subseteq \mathbb{T}_{\{a,b,c,d\}}$. Note that pipelines can be seen as a special case of trees.

Grids: A grid is a topology over $\{a, b, c, d\}$ where processes are arranged in a matrix. It is uniquely given by its number $m \geq 1$ of rows and its number $n \geq 1$ of columns. Again, there should be at least two processes so that we suppose $\max\{m, n\} \geq 2$. A process that is not located on the border has a right and a left neighbor (following a and b, respectively), but also adjacent nodes below and above (following c and d). Let $\mathcal{T}^{m,n}_{\text{grid}}$ denote the grid with m rows and n columns. An example is illustrated in Figure 3. By $\mathfrak{T}_{\text{grid}} = \{\mathcal{T}^{m,n}_{\text{grid}} \mid m, n \geq 1 \text{ with } \max\{m, n\} \geq 2\} \subseteq \mathbb{T}_{\{a,b,c,d\}}$, we denote the set of all grids. Again, a pipeline is a special case of a grid.

Rings: A ring can be seen as a pipeline where the endpoints are glued together. Thus, it is a topology over $\{a, b\}$ in which every process has type $\{a, b\}$. The ring with $n \geq 3$ processes is denoted by $\mathcal{T}^n_{\text{ring}}$. Figure 4 illustrates $\mathcal{T}^5_{\text{ring}}$. We denote the set of all rings by $\mathfrak{T}_{\text{ring}} = \{\mathcal{T}^n_{\text{ring}} \mid n \geq 3\} \subseteq \mathbb{T}_{\{a,b\}}$. □

Remark 2. For many concrete topology classes of bounded degree such as pipelines, grids, or rings, the names are canonical so that fixing them in advance is not a restriction. Moreover, in most cases considered in the literature, a few (sometimes even one) process types will do. In particular, it is a common assumption that processes in a ring are indistinguishable [8, 15, 24]. However, one could also assume a distinguished leader process and add another interface name just for the purpose of identifying the leader; cf. also Remark 1.

2.2 Message Sequence Charts

The semantics of both an automaton and a logic formula will be defined as a set of *message sequence charts (MSCs)*. Each MSC depicts a single execution of a system. It is formalized as a labeled finite directed acyclic graph whose nodes, the *events*, are associated with processes from a given communication topology. Events are linked by process edges $\lhd_{\mathsf{proc}}$ and message edges $\lhd_{\mathsf{msg}}$. The process edges connect consecutive events of one process, and message edges connect send events with their corresponding receives according to a FIFO policy.

Definition 2. *An MSC over* $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$ *is a triple* $M = (E, \lhd, \ell)$ *where*

- E *is the nonempty finite set of* events*,*
- $\lhd \subseteq E \times E$ *is the acyclic edge relation, which is partitioned into* $\lhd_{\mathsf{proc}}$ *and* $\lhd_{\mathsf{msg}}$*, and*
- $\ell : E \to P$ *determines the location of an event in the topology; for* $p \in P$*, we let* $E_p := \{e \in E \mid \ell(e) = p\}$.

We require that the following hold:

- $\lhd_{\mathsf{proc}}$ *is a union* $\bigcup_{p \in P} \lhd_p$ *where each* $\lhd_p \subseteq E_p \times E_p$ *is the direct-successor relation of some total order on* E_p*,*
- *there are a partition* $E = E_! \uplus E_?$ *and a bijection* $\mu : E_! \to E_?$ *such that* $\lhd_{\mathsf{msg}} = \{(e, \mu(e)) \mid e \in E_!\}$*,*
- *for all* $(e, f) \in \lhd_{\mathsf{msg}}$*, there are* $a, b \in \mathcal{N}$ *such that* $\ell(e) \overset{a\ b}{\longmapsto} \ell(f)$ *(communication is restricted to adjacent processes), and*
- *for all* $(e, f), (e', f') \in \lhd_{\mathsf{msg}}$ *such that* $\ell(e) = \ell(e')$ *and* $\ell(f) = \ell(f')$*, we have* $e \lhd^*_{\mathsf{proc}} e'$ *iff* $f \lhd^*_{\mathsf{proc}} f'$ *(FIFO).*

We do not distinguish isomorphic MSCs over $\mathcal{T}$.

Given an MSC $M = (E, \lhd, \ell)$, we define a mapping $act_M : E \to \{!a, ?a \mid a \in \mathcal{N}\}$ that associates with an event the

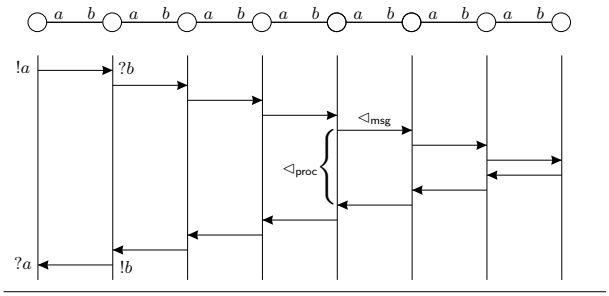


Figure 5. MSC M^8_{lin} over topology $\mathcal{T}^8_{\text{lin}}$

action that it executes: for $(e, f) \in \lhd_{\mathsf{msg}}$ and $a, b \in \mathcal{N}$ such that $\ell(e) \overset{a\ b}{\longmapsto} \ell(f)$, we let $act_M(e) = !a$ and $act_M(f) = ?b$.

Example 2. Figure 5 illustrates an MSC, call it M^8_{lin}, over topology $\mathcal{T}^8_{\text{lin}} \in \mathfrak{T}_{\text{lin}}$. The behavior of each process is represented by a top-down process line. Arrows between process lines determine the relation $\lhd_{\mathsf{msg}}$, connecting send events with their receive events. For illustration, some events are labeled with the actions that they execute. We may consider M^n_{lin} as the execution of a P2P protocol: a request from the leftmost process is forwarded by $n-2$ inner processes of type $\{a, b\}$, until it reaches the rightmost process. An acknowledgement is then relayed back to the first process along the same way backwards. Figure 10 depicts an example MSC over $\mathcal{T}^{2,5}_{\text{grid}}$. □

Our main result will deal with systems that have (existentially) B-bounded channels, for some $B \geq 1$ [19, 20]. Intuitively, an MSC is B-bounded if it can be scheduled in such a way that, along the execution, there are never more than B messages in each channel. Formally, we define boundedness via linearizations. A linearization of an MSC $M = (E, \lhd, \ell)$ over $\mathcal{T} = (P, \longmapsto)$ is any total order $\preceq \subseteq E \times E$ satisfying $\lhd^* \subseteq \preceq$. Then, $\preceq$ is called *B-bounded* if, for all $f \in E$, $p, q \in P$, and $a, b \in \mathcal{N}$ such that $p \overset{a\ b}{\longmapsto} q$, we have $|\{e \in E \mid e \preceq f, \ell(e) = p, \text{and } act_M(e) = !a\}| - |\{e \in E \mid e \preceq f, \ell(e) = q, \text{and } act_M(e) = ?b\}| \leq B$. In other words, in any prefix of $\preceq$, there are no more than B pending messages, in every "channel" (p, q). Now, we say that MSC M is B-bounded if it has some B-bounded linearization. For example, for all $n \geq 2$, the MSC M^n_{lin} (cf. Figure 5) is 1-bounded, because its (only) linearization is 1-bounded.

3. Parameterized Communicating Automata

Next, we introduce PCA. Their definition does not depend on a topology, but only on $\mathcal{N}$. The language of a PCA, a set of MSCs, is then parameterized by a topology.

Definition 3. *A* parameterized communicating automaton (PCA) *over $\mathcal{N}$ is a tuple $\mathcal{A} = (S, Msg, \Delta, I, F)$ where*

- *S is the finite set of* states*,*
- *Msg is the finite set of* messages*,*
- *$I : (2^{\mathcal{N}} \setminus \{\emptyset\}) \to 2^S$ assigns to each nonempty process type its* initial states*,*
- *F is the* acceptance condition*: a finite boolean combination of statements $\langle \#(s) \geq k \rangle$ with $s \in S$ and $k \in \mathbb{N}$ (to be read as "s occurs at least k times as the terminal state of a process"), and*
- *$\Delta \subseteq S \times \Sigma_{\mathcal{A}} \times S$ is the set of* transitions*.*

Here, $\Sigma_{\mathcal{A}} := \{!_m a, ?_m a \mid a \in \mathcal{N} \text{ and } m \in Msg\}$ contains send actions $!_m a$ and receive actions $?_m a$. A transition $(s, \eta, s') \in \Delta$ is also written $s \overset{\eta}{\Rightarrow} s'$.

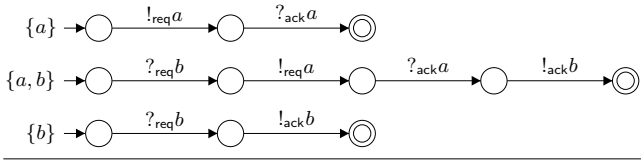


Figure 6. PCA $\mathcal{A}$ over $\{a, b\}$

The class of PCA over $\mathcal{N}$ is denoted by $\mathbb{PCA}_{\mathcal{N}}$. A PCA over $\mathcal{N}$ can be run on any topology $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$. The idea is that every process $p \in P$ executes a copy of the transition system (S, Δ), starting in some state from $I(type_{\mathcal{T}}(p))$. Thus, one could define a PCA (equivalently) as a collection of finite automata, each describing the local behavior of a particular process type. Suppose $p \overset{a\ b}{\longmapsto} q$ for processes $p, q \in P$ and names $a, b \in \mathcal{N}$. When p executes a transition $(s, !_m a, s') \in \Delta$, it changes its local state from s to s' and writes m into the FIFO channel (p, q). The message m can then be received by process q executing a transition with action $?_m b$. However, messages are abstracted away in the observable MSC behavior (they are in the spirit of stack symbols in visibly pushdown automata [3]).

Formally, we define the semantics of a PCA directly on MSCs. This is equivalent to an operational semantics in terms of an infinite transition system, but closer to the logical approach where formulas are evaluated over MSCs (see Section 4). Let $\mathcal{A} = (S, Msg, \Delta, I, F) \in \mathbb{PCA}_{\mathcal{N}}$ be a PCA, $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$ be a topology, and $M = (E, \lhd, \ell)$ be an MSC over $\mathcal{T}$. A run of $\mathcal{A}$ on M will be a mapping $\rho : E \to S$. Intuitively, $\rho(e)$ is the state that process $\ell(e)$ reaches after executing $e \in E$. To define when ρ is a run, we will need some more notation.

Set $P_M := \{p \in P \mid E_p \neq \emptyset\}$, which is the set of *active processes* of M. A *(global) initial state* of $\mathcal{A}$ for M is a tuple $\iota = (\iota_p)_{p \in P_M}$ where $\iota_p \in I(type_{\mathcal{T}}(p))$ for all $p \in P_M$. Given ι and $\rho : E \to S$ (a possible MSC run), we define another mapping $\rho^-_\iota : E \to S$, which returns the source state of a transition. For $(f, e) \in \lhd_{\mathsf{proc}}$, we let $\rho^-_\iota(e) = \rho(f)$; for a $\lhd_{\mathsf{proc}}$-minimal event $e \in E$, we let $\rho^-_\iota(e) = \iota_{\ell(e)}$.

Now, a mapping $\rho : E \to S$ is called a *run* of $\mathcal{A}$ on M if there is an initial state $\iota = (\iota_p)_{p \in P_M}$ for M such that, for all $(e, f) \in \lhd_{\mathsf{msg}}$, there are $a, b \in \mathcal{N}$ and $m \in Msg$ satisfying $\ell(e) \overset{a\ b}{\longmapsto} \ell(f)$, $\rho^-_\iota(e) \overset{!_m a}{\Longrightarrow} \rho(e)$, and $\rho^-_\iota(f) \overset{?_m b}{\Longrightarrow} \rho(f)$.

To determine if ρ is accepting, we define a multiset $h_\rho : S \to \mathbb{N}$ over S that counts how often each state occurs as the terminal state of an *active* process. For $s \in S$, we let $h_\rho(s) = |\{e \in E \mid e \text{ is } \lhd_{\mathsf{proc}}\text{-maximal and } \rho(e) = s\}|$. We say that ρ is *accepting* if h_ρ satisfies F in the expected manner; in particular, h_ρ satisfies $\langle \#(s) \geq k \rangle$ if $h_\rho(s) \geq k$. The MSC M is accepted by $\mathcal{A}$ if it admits an accepting run of $\mathcal{A}$. For a topology $\mathcal{T}$, the set of MSCs over $\mathcal{T}$ that are accepted by $\mathcal{A}$ is denoted by $L_{\mathcal{T}}(\mathcal{A})$. Finally, we let $L^B_{\mathcal{T}}(\mathcal{A})$ be the restriction of $L_{\mathcal{T}}(\mathcal{A})$ to B-bounded MSCs.

Example 3. Consider the PCA $\mathcal{A}$ over $\{a, b\}$ from Figure 6. The acceptance condition F is simply the conjunction of formulas $\neg\langle \#(s) \geq 1 \rangle$ with s ranging over the states *without* double circle. Recall that the messages req and ack do not occur in the accepted MSCs. In this example, we could actually do with just one message ($|Msg| = 1$). In general, however, message contents increase the expressive power of PCA. Note that MSC M^8_{lin} is the only MSC that is accepted by $\mathcal{A}$ over $\mathcal{T}^8_{\text{lin}}$ (cf. Example 2). We actually have $L_{\mathcal{T}^n_{\text{lin}}}(\mathcal{A}) = L^1_{\mathcal{T}^n_{\text{lin}}}(\mathcal{A}) = \{M^n_{\text{lin}}\}$ for all $n \geq 2$. □

Remark 3. The multiset h_ρ defined to evaluate the acceptance condition of a PCA does not include any states of non-active (i.e., idle) processes. So, a PCA cannot express "the topology has $\bowtie 5$ processes" where $\bowtie \in \{\geq, \leq, =\}$, but only "$\bowtie 5$ processes are

active". In principle, one could include idle processes as well. However, this has to be reflected in the logic (cf. Theorem 9). One possibility is to consider processes as single events. But, apart from involving a more technical presentation, this does not seem to be natural. Alternatively, one could consider a two-sorted logic to reason about both events and processes. In that case, one very quickly exceeds the capability of PCA to evaluate a topology, as their runs rely on the messages that occur in an MSC. A two-sorted logic also goes against the intuition that PCA accept behaviors rather than topologies.

The section concludes with some closure properties of PCA.

Theorem 1. *PCA are closed under union and intersection: For all $\mathcal{A}_1, \mathcal{A}_2 \in \mathbb{PCA}_{\mathcal{N}}$, there are PCA $\mathcal{A}$ and $\mathcal{B}$ over $\mathcal{N}$ such that, for all topologies $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$, we have $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\mathcal{A}_1) \cup L_{\mathcal{T}}(\mathcal{A}_2)$ and $L_{\mathcal{T}}(\mathcal{B}) = L_{\mathcal{T}}(\mathcal{A}_1) \cap L_{\mathcal{T}}(\mathcal{A}_2)$.*

The construction of $\mathcal{A}$ and $\mathcal{B}$ follows a standard scheme. The only (minor) subtle point is the acceptance condition.

Theorem 2. *PCA are not closed under complementation: There is $\mathcal{A} \in \mathbb{PCA}_{\{a,b\}}$ such that, for all $\mathcal{B} \in \mathbb{PCA}_{\{a,b\}}$, we have $L_{\mathcal{T}^2_{\mathsf{lin}}}(\mathcal{B}) \neq \{M \mid M \text{ is an MSC over } \mathcal{T}^2_{\mathsf{lin}}\} \setminus L_{\mathcal{T}^2_{\mathsf{lin}}}(\mathcal{A})$.*

Theorem 2 is an immediate consequence of the fact that fixed-topology CA over two processes are not complementable [6]. Finally, non-deterministic PCA are strictly more expressive than deterministic ones (we do not give the formal definitions). This already holds over 1-bounded MSCs, which follows from the case of fixed-topology CA and requires a topology with five processes and five interface names [20].

4. MSO Logic and Locality of FO logic

While PCA serve as a model of an implementation of a communicating system, we use monadic second-order (MSO) logic to *specify* properties of MSCs.

4.1 Monadic Second-Order Logic

The set $\mathrm{MSO}_{\mathcal{N}}$ of MSO formulas over $\mathcal{N}$ is given by the following grammar:

$$\begin{aligned}\varphi \ ::= \ & act(x) = !a \mid act(x) = ?a \mid a \in \mathit{type}(x) \mid \\ & x \lhd_{\mathsf{proc}} y \mid x \lhd^*_{\mathsf{proc}} y \mid x \lhd_{\mathsf{msg}} y \mid x \lhd^* y \mid x \sim y \mid \\ & x = y \mid x \in X \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\varphi \mid \exists X\varphi\end{aligned}$$

where $a \in \mathcal{N}$, 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), all taken from infinite supplies of variables. We use standard abbreviations such as $\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$, $\varphi \to \psi \equiv \neg\varphi \vee \psi$, and $\forall x\varphi \equiv \neg\exists x\neg\varphi$.

The set $\mathrm{FO}_{\mathcal{N}}$ of *first-order* formulas is the fragment of $\mathrm{MSO}_{\mathcal{N}}$ without second-order quantification $\exists X$. Moreover, $\mathrm{EMSO}_{\mathcal{N}}$ (*existential* MSO) is the set of formulas of the form $\exists X_1 \ldots \exists X_n \varphi$ with $\varphi \in \mathrm{FO}_{\mathcal{N}}$.

A formula is evaluated wrt. an MSC $M = (E, \lhd, \ell)$ over some topology $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$. Free variables x and X are interpreted by a mapping $\mathfrak{I}$ as an event $\mathfrak{I}(x) \in E$ and a set of events $\mathfrak{I}(X) \subseteq E$, respectively. For η of the form $!a$ or $?a$, the atomic formula $act(x) = \eta$ is true if $act_M(\mathfrak{I}(x)) = \eta$. Formula $a \in \mathit{type}(x)$ is true if $a \in \mathit{type}_{\mathcal{T}}(\ell(\mathfrak{I}(x)))$, i.e., a is contained in the type of the process where $\mathfrak{I}(x)$ is located. Formula, $x \lhd^*_{\mathsf{proc}} y$ is satisfied if $\mathfrak{I}(x) \lhd^*_{\mathsf{proc}} \mathfrak{I}(y)$. Moreover, $x \sim y$ holds true if $\ell(\mathfrak{I}(x)) = \ell(\mathfrak{I}(y))$, i.e., $\mathfrak{I}(x)$ and $\mathfrak{I}(y)$ are located on the same process. Other formulas are interpreted as expected.

Though, even in $\mathrm{FO}_{\mathcal{N}}$, some binary predicates are mutually expressible in terms of others (e.g., $\lhd_{\mathsf{proc}}$ and $\sim$ in terms of $\lhd^*_{\mathsf{proc}}$, and $\lhd^*_{\mathsf{proc}}$ in terms of $\lhd^*$ and $\sim$), we include all of them explicitly in the logic. They will be used in fragments in which they would no longer be expressible.

Let $\sigma \subseteq \{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$ be a nonempty set of relation symbols. The logics $\mathrm{FO}_{\mathcal{N}}[\sigma]$ and $\mathrm{EMSO}_{\mathcal{N}}[\sigma]$ restrict $\mathrm{FO}_{\mathcal{N}}$ and $\mathrm{EMSO}_{\mathcal{N}}$, respectively: instead of $\{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$, we can only access the relation symbols from σ. Our main (positive) result will concern the logic $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ (recall that $\lhd_{\mathsf{proc}}$ and $\sim$ can be expressed in terms of $\lhd^*_{\mathsf{proc}}$).

Let $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$ be a topology, and let $\varphi \in \mathrm{MSO}_{\mathcal{N}}$ be a sentence, i.e., a formula without free variables. The set of MSCs over $\mathcal{T}$ that satisfy φ is denoted by $L_{\mathcal{T}}(\varphi)$. Moreover, for $B \geq 1$, we let $L^B_{\mathcal{T}}(\varphi)$ denote the restriction of $L_{\mathcal{T}}(\varphi)$ to B-bounded MSCs.

Example 4. We will consider two $\mathrm{FO}_{\{a,b\}}$-sentences. First, formula $\varphi_1 = \forall x(act(x) = ?b \ \to \ \exists y(x \lhd^*_{\mathsf{proc}} y \wedge act(y) = !b))$ says that every process that receives a message from its b-interface, eventually sends a message through b. Note that $M^n_{\mathsf{lin}} \in L_{\mathcal{T}^n_{\mathsf{lin}}}(\varphi_1)$ for all $n \geq 2$ (cf. Example 2). Next, let $\varphi_2 = \exists x \exists y(b \notin \mathit{type}(x) \wedge a \notin \mathit{type}(y) \wedge x \lhd_{\mathsf{msg}} y)$. Interpreted over pipelines, φ_2 says that the leftmost process sends a message to the rightmost process. We have $M^n_{\mathsf{lin}} \in L_{\mathcal{T}^n_{\mathsf{lin}}}(\varphi_2)$ iff $n = 2$. □

4.2 Locality of FO Logic

Next, we state a locality theorem due to Schwentick and Barthelmann [28].[3] It formalizes the intuition that first-order logic can only reason about local *neighborhoods*, which include elements whose distance from a given center is bounded by a radius that depends on the formula.

Fix a nonempty set $\sigma \subseteq \{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$ of relation symbols. Let $M = (E, \lhd, \ell)$ be an MSC over some topology $\mathcal{T} = (P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$. The distance $dist^{\sigma}_M(e, f)$ between events $e, f \in E$ is the minimal length of a path between e and f in the graph of M with edges given by σ, in either direction (or ∞ if such a path does not exist). For example, let $\sigma = \{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$. Then, $dist^{\sigma}_M(e, f)$ refers to the distance in the (undirected) graph $(E, \lhd^*_{\mathsf{proc}} \cup (\lhd^*_{\mathsf{proc}})^{-1} \cup \lhd_{\mathsf{msg}} \cup \lhd^{-1}_{\mathsf{msg}})$ so that $dist^{\sigma}_M(e, f) \leq 1$ for all $e, f \in E$ such that $\ell(e) = \ell(f)$, and $dist^{\sigma}_M(e, f) = dist^{\sigma}_M(f, e) = 1$ for all $(e, f) \in \lhd_{\mathsf{msg}}$.

Example 5. Let M be the MSC over $\mathcal{T}^8_{\mathsf{lin}}$ from Figure 7 and let e be the distinguished event given by the white circle. The set of events f such that $dist^{\sigma}_M(e, f) \leq 3$ depends on σ, and is illustrated for $\{\lhd^*\}$, $\{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$ (inducing the same set as $\{\lhd_{\mathsf{msg}}, \sim\}$), and $\{\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$. □

Let $r \geq 1$ be a natural number. A formula $\chi \in \mathrm{FO}_{\mathcal{N}}[\sigma]$ is called (r, σ)-*local* around a first-order variable y if (i) y is not quantified in χ and (ii) χ is obtained from some $\mathrm{FO}_{\mathcal{N}}[\sigma]$-formula by replacing each subformula of the form $\exists z\psi$ with $\exists z(dist^{\sigma}(y, z) < r \wedge \psi)$, and each subformula of the form $\forall z\psi$ with $\forall z(dist^{\sigma}(y, z) < r \to \psi)$. Here, $dist^{\sigma}(y, z) < r$ denotes the obvious $\mathrm{FO}_{\mathcal{N}}[\sigma]$-formula. We use strict inequality for technical reasons (cf. [17]). Adapted to our setting, [28] yields the following:

Theorem 3 (Schwentick & Barthelmann, [28])**.** *For every sentence $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\sigma]$, there are $r \geq 1$ and a sentence $\varphi' = \exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \chi \in \mathrm{EMSO}_{\mathcal{N}}[\sigma]$ (with $m, n \geq 0$) such that $\chi \in \mathrm{FO}_{\mathcal{N}}[\sigma]$ is (r, σ)-local around y and, for all topologies $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$, we have $L_{\mathcal{T}}(\varphi) = L_{\mathcal{T}}(\varphi')$.*

[3] Gaifman's normal form [16] appears to be more difficult to deal with in our context.

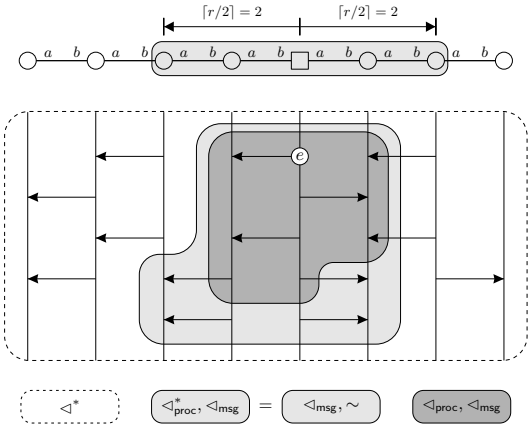


Figure 7. Distance $r = 3$ in an MSC depending on the signature

5. Negative Results

Recall that we are interested in realizability of formulas φ for a class $\mathfrak{T}$ of topologies: Is there a PCA $\mathcal{A}$ such that $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\varphi)$ for all $\mathcal{T} \in \mathfrak{T}$? In this section, we show that such a PCA does not always exist.

5.1 Restrictions on Topologies Are Necessary

In fact, there is a sentence from $\mathrm{FO}_{\{a,b\}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ that is not realizable for the class of *all* topologies over $\{a, b\}$. This even holds for the class $\mathfrak{T}^*_{\text{ring}} \subseteq \mathbb{T}_{\{a,b\}}$ of *ring forests* and when we restrict to 1-bounded MSCs. A ring forest is a disjoint union of an arbitrary number of rings (possibly containing several copies of one and the same ring).

Theorem 4. *There exists a sentence $\varphi \in \mathrm{FO}_{\{a,b\}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ such that, for all PCA $\mathcal{A} \in \mathbb{PCA}_{\{a,b\}}$, there is $\mathcal{T} \in \mathfrak{T}^*_{\text{ring}}$ with $L^1_{\mathcal{T}}(\mathcal{A}) \neq L^1_{\mathcal{T}}(\varphi)$.*

Proof. The sentence φ will say that every event is part of the cycle pattern that is depicted in Figure 8:

$$\varphi = \forall x \exists x_1, \ldots, x_6 (x \in \{x_1, \ldots, x_6\} \wedge \mathit{cycle}(x_1, \ldots, x_6))$$

where $\mathit{cycle}(x_1, \ldots, x_6)$ is defined as

$$\begin{aligned} & x_1 \lhd_{\mathsf{msg}} x_2 \lhd_{\mathsf{proc}} x_3 \lhd_{\mathsf{msg}} x_4 \lhd_{\mathsf{proc}} x_5 \lhd_{\mathsf{msg}} x_6 \\ \wedge\ & x_1 \lhd_{\mathsf{proc}} x_6 \wedge \textstyle\bigwedge_{i \in \{1,3,5\}} \mathit{act}(x_i) = !a \end{aligned}$$

Towards a contradiction, suppose there is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathfrak{T}^*_{\text{ring}}$, $L_{\mathcal{T}}(\mathcal{A})$ and $L_{\mathcal{T}}(\varphi)$ agree on all 1-bounded MSCs.

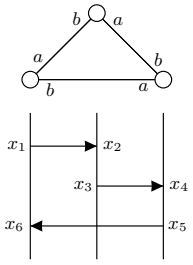


Figure 8. Cycle

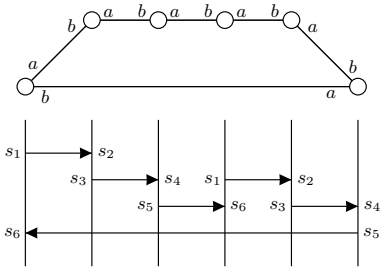


Figure 9. Fusing two cycles

Consider the MSC M_n that consists of $n \geq 1$ disjoint copies of the "atomic" MSC from Figure 8. That is, M_n is an MSC over the topology $\mathcal{T}_n = \mathcal{T}^3_{\text{ring}} \uplus \ldots \uplus \mathcal{T}^3_{\text{ring}} \in \mathfrak{T}^*_{\text{ring}}$ with n disjoint copies of $\mathcal{T}^3_{\text{ring}}$. Obviously, $M_n \in L^1_{\mathcal{T}_n}(\varphi) = L^1_{\mathcal{T}_n}(\mathcal{A})$ for all $n \geq 1$. When we choose n large enough, then there is an accepting run ρ of $\mathcal{A}$ on M_n that behaves the same on at least two disjoint copies of atomic MSCs. More precisely, M_n is an MSC over $\mathcal{T} \uplus \mathcal{T}^3_{\text{ring}} \uplus \mathcal{T}^3_{\text{ring}} \in \mathfrak{T}^*_{\text{ring}}$, for some $\mathcal{T}$, such that ρ assigns states $s_1, \ldots, s_6$ to the events $x_1, \ldots, x_6$ of the atomic MSCs over the last two copies of $\mathcal{T}^3_{\text{ring}}$, respectively. We will now replace these two atomic MSCs with the larger one from Figure 9, over $\mathcal{T}^6_{\text{ring}}$. The resulting MSC, call it M'_n, is a 1-bounded MSC over $\mathcal{T} \uplus \mathcal{T}^6_{\text{ring}}$. There is still a run on M'_n, using the assignment shown in Figure 9. As the multiset of terminal states does not change, the run is accepting. But this is a contradiction, since M'_n does not satisfy φ. □

The proof of Theorem 4 reveals that PCA have limited ability to "detect" cycles in an MSC and in a topology. In the following sections, we will, therefore, restrict the "cyclic behavior" of a class of topologies (of a single topology, respectively).

In Section 6, the construction of a PCA indeed requires a class $\mathfrak{T}$ of topologies to be given in advance. We show that certain formulas are realizable for all classes $\mathfrak{T}$ that are *unambiguous*: one can tell by looking at a sequence $w \in (\mathcal{N} \times \mathcal{N})^*$ of edge labels whether w produces a cycle or not, in any topology of $\mathfrak{T}$. This notion exludes the set of ring forests exploited in Theorem 4, but it captures the classes of pipelines, trees, and grids, as well as singleton rings and the class of "almost all" rings (which will finally allows us to cover the class of all rings with one single PCA).

In Section 7, the construction of a PCA from a given formula does not depend on a class of topologies, but only on $\mathcal{N}$. It makes sure that the synthesized PCA agrees with the formula on all *prime* topologies. This forbids cycles with a periodic labeling (as they occur in rings), but includes all pipelines, trees, and grids.

5.2 Restrictions on Logic Are Necessary

Next, we argue that we have to restrict the logic, too. In fact, when we take $\mathrm{FO}_{\mathcal{N}}$ (i.e., first-order logic with all binary predicates), then the negative result even holds for the class of trees (actually, for simple bus topologies). This has to be contrasted with the expressive equivalence of MSO and CA over *fixed* topologies when imposing any existential bound on the channels [19].

Theorem 5. *There exists a sentence $\varphi \in \mathrm{FO}_{\{a,b,c,d\}}$ such that, for all PCA $\mathcal{A} \in \mathbb{PCA}_{\{a,b,c,d\}}$, there is $\mathcal{T} \in \mathfrak{T}_{\text{tree}}$ with $L^1_{\mathcal{T}}(\mathcal{A}) \neq L^1_{\mathcal{T}}(\varphi)$.*

The proof uses a technique from [30], which was employed to show that FO (with reflexive transitive closure relations) and a local variant of EMSO are incomparable over *pictures*.

We briefly discuss Theorem 5. Let $\varphi \in \mathrm{FO}_{\mathcal{N}}[\sigma]$, for some $\sigma \subseteq \{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$, and suppose r is the radius associated with φ according to Theorem 3. Satisfaction of φ in an MSC M essentially depends on the σ-neighborhoods of the latter (informally, the σ-neighborhood of an event e is the substructure of M, including the *actions*, induced by all elements f such that $\mathit{dist}^{\sigma}_{M}(e, f) \leq r$). When σ contains $\lhd^*$, there is no a priori bound on the size of the σ-neighborhood of e: it may feature events f that are far from e in terms of the number of messages that separate them. In other words, $\mathit{dist}^{\sigma}_{M}(e, f)$ can be small for $\sigma = \{\lhd^*\}$, but large for $\sigma = \{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$. This is exemplified in Figure 7. On the other hand, the exchange of information in a PCA relies on messages and is restricted to processes that are adjacent in the topology. This intuitively explains the mismatch between automata and logic that Theorem 5 reveals.

The size of a $\{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$-neighborhood is a priori unbounded as well. However, unlike a $\{\lhd^*\}$-neighborhood, it has "bounded

width", since it spans over a bounded area of the *underlying topology*: Figure 7 illustrates that, provided $\lhd^* \notin \sigma$, a *topology* neighborhood of radius $\lceil r/2 \rceil$ is indeed sufficient to cover all events f with $dist^{\sigma}_M(e,f) \leq r$. As our topologies are structures of bounded degree, there are only a bounded number of such topology neighborhoods. This allows us to take advantage of results on CA over fixed topologies. We pursue this idea in Section 6 to show realizability of $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$- and $\mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \sim]$-formulas for unambiguous classes of topologies (as discussed in the previous subsection).

Any $\{\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$-neighborhood in an MSC also spans over a bounded topology (cf. Figure 7). But here, we can go one step further: up to isomorphism, there are only a bounded number of $\{\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$-neighborhoods. Using this, we will build, in Section 7, a PCA that evaluates a formula by "computing" such neighborhoods in an MSC. The construction is independent of a class of topologies. However, the resulting PCA is only guaranteed to be equivalent to the given formula when it is applied to *prime* topologies, in which no cycle has a periodic labeling (cf. Definition 5).

6. EMSO vs. PCA over Unambiguous Topology Classes

6.1 Main Result and Consequences

Following the discussion in the previous section, we will consider logics that discard $\lhd^*$. Particular attention is paid to the logic $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$, containing the process-order relation.

Recall that the $\{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$-neighborhood of an event is covered by a bounded area in the underlying topology. We exploit this to translate a formula into a PCA by simulating several fixed-topology CA in parallel. In fact, in our construction of a PCA, a process will have to determine its (bounded) topology neighborhood, so that it can launch a corresponding fixed-topology CA. To some extent, it can verify whether it is the source of a w-labeled path, for a given $w \in (\mathcal{N} \times \mathcal{N})^*$. In general, however, there is no means to discover whether w forms a cycle (cf. Theorem 4). We will, therefore, consider topology classes $\mathfrak{T}$ in which some words are *unambiguous*, i.e., they always form a cycle, or never form a cycle in $\mathfrak{T}$. For example, $(a,b)(c,d)(b,a)(d,c)$ is unambiguous for the class of grids: it gives rise to a cycle whenever it is applicable. As we deal with bounded topology neighborhoods, we have to consider only words up to a certain length.

Let $w = (a_1,b_1)\dots(a_n,b_n) \in (\mathcal{N} \times \mathcal{N})^*$. The length n of w is denoted by $|w|$. For a topology $(P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$ and processes $p, q \in P$, we write $p \overset{w}{\longmapsto} q$ if there is a w-labeled path from p to q, i.e., there are $p_0, \dots, p_n \in P$ such that $p = p_0 \overset{a_1\ b_1}{\longmapsto} p_1 \overset{a_2\ b_2}{\longmapsto} \dots \overset{a_n\ b_n}{\longmapsto} p_n = q$.

Definition 4. *Let $k \in \mathbb{N}$ and let $\mathfrak{T} \subseteq \mathbb{T}_{\mathcal{N}}$ be a* class *of topologies. We say that $\mathfrak{T}$ is* k-unambiguous *if, for all $w \in (\mathcal{N} \times \mathcal{N})^*$ with $|w| \leq k$, all topologies $(P, \longmapsto), (P', \longmapsto') \in \mathfrak{T}$, and all processes $p, q \in P$ and $p', q' \in P'$ such that $p \overset{w}{\longmapsto} q$ and $p' \overset{w}{\longmapsto}' q'$, we have $p = q$ iff $p' = q'$.*

In other words, if a topology from $\mathfrak{T}$ admits a cycle of length $\leq k$ with label w, then following w (if possible) will always form a cycle, in any topology from $\mathfrak{T}$. Note that "unambiguous" is a property of a *class* of topologies. It captures the classes of pipelines, trees, and grids. Moreover, it will allow us to apply our results to ring topologies.

Lemma 1. *The classes $\mathfrak{T}_{\mathrm{lin}}$, $\mathfrak{T}_{\mathrm{tree}}$, $\mathfrak{T}_{\mathrm{grid}}$, and $\{\mathcal{T}^n_{\mathrm{ring}} \mid n \geq \max\{3, k+1\}\}$ are all k-unambiguous, for every $k \in \mathbb{N}$. Moreover, $\{\mathcal{T}\}$ is k-unambiguous for all $\mathcal{T} \in \mathfrak{T}_{\mathrm{ring}}$ and $k \in \mathbb{N}$.*

For a nonempty set $\sigma \subseteq \{\lhd_{\mathsf{proc}}, \lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \lhd^*, \sim\}$ and a sentence $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\sigma]$, let $r_\varphi \geq 1$ denote the radius associated with φ according to Theorem 3. Note that an exponential upper bound for r_φ was given in [25]. We now present our main result:

Theorem 6. *Let $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ be a sentence, $B \geq 1$, and $\mathfrak{T} \subseteq \mathbb{T}_{\mathcal{N}}$ be an $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A} \in \mathbb{PCA}_{\mathcal{N}}$ such that, for all $\mathcal{T} \in \mathfrak{T}$, we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\varphi)$.*

Before we prove Theorem 6, let us discuss it and state some consequences. First, note that we have to commit to a class $\mathfrak{T}$ of topologies before constructing the PCA. This is also what usually happens in practice: one has a "concrete" class of topologies in mind when writing a formula. For example, a specifier may want to synthesize a PCA that is equivalent to the given formula on all grids.[4] Though, a priori, different topology classes give rise to different PCA, we give a uniform construction and proof. By Lemma 1, we can then instantiate $\mathfrak{T}$ in Theorem 6 with various classes so that we obtain the following corollary (for simplicity, we consider pipelines and rings as topologies over $\{a,b,c,d\}$):

Corollary 1. *Let $\varphi \in \mathrm{EMSO}_{\{a,b,c,d\}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ be a sentence, $B \geq 1$, and $\mathfrak{T}$ be any of the following:*

- *the set of pipeline topologies,*
- *the set of grid topologies,*
- *the set of tree topologies,*
- *the set of ring topologies with at least $r_\varphi + 3$ processes, or*
- *the singleton set $\{\mathcal{T}\}$ where $\mathcal{T}$ is any ring topology.*

Then, there is a PCA $\mathcal{A} \in \mathbb{PCA}_{\{a,b,c,d\}}$ such that, for all $\mathcal{T} \in \mathfrak{T}$, we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\varphi)$.

The construction where one single ring topology is given is *not* an immediate consequence of a corresponding result from the setting with fixed topologies [19]. The reason is that CA over fixed topologies have an initial state per *process*, while PCA have initial states per *process type*, which is a priori weaker. Now, given a formula φ, Corollary 1 gives us a way of "covering" *all* rings in terms of a *finite* collection of PCA: For those rings with at most $r_\varphi + 2$ processes, we can construct tailor-made PCA, i.e., one PCA for each ring. All other rings can be covered by one single PCA. Using this observation, we can even do better than that and show that one single PCA is enough. The idea is to "approximate" the size of the given ring in terms of the number of active processes and to launch a corresponding PCA according to Corollary 1. As we already know that we will run the PCA on a *ring*, we in fact only have to "determine" its size.

Theorem 7. *Let $\varphi \in \mathrm{EMSO}_{\{a,b\}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ be a sentence and $B \geq 1$. There is a PCA $\mathcal{A} \in \mathbb{PCA}_{\{a,b\}}$ such that, for all $\mathcal{T} \in \mathfrak{T}_{\mathrm{ring}}$, we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\varphi)$.*

Let us come back to the generic result of Theorem 6. Its proof uses the logical characterization of fixed-topology CA [19]. But it works similarly for $\mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \sim]$ when we take [6] instead of [19]. In the setting of fixed topologies, $\sim$ reduces to a local comparison of event labels so that [6] is indeed applicable. The logic $\mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \sim]$ is a priori weaker than $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$, but allows us to drop the channel restriction.

Theorem 8. *Let $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}, \sim]$ be a sentence and $\mathfrak{T} \subseteq \mathbb{T}_{\mathcal{N}}$ be an $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A} \in \mathbb{PCA}_{\mathcal{N}}$ such that, for all $\mathcal{T} \in \mathfrak{T}$, we have $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\varphi)$.*

[4] The synthesized PCA can still be run on any other topology (over the same set of names). However, it is sufficient to know that it is *equivalent to the formula when we run it on a grid (respectively, topology from $\mathfrak{T}$).*

From Theorem 8 and Lemma 1, we can derive statements analogous to Corollary 1 and Theorem 7 (which we omit).

We do not know whether Theorem 8 holds for the logic $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ (or, equivalently, whether Theorem 6 holds when we drop the channel bound). The answer is affirmative if $\mathrm{EMSO}_{\mathcal{N}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ is equivalent to unbounded CA over *fixed* topologies. But this is an open problem.

The translation of PCA to $\mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ is not restricted to topologies of a particular form.

Theorem 9. *Let $\mathcal{A} \in \mathbb{PCA}_{\mathcal{N}}$ be a PCA over $\mathcal{N}$. There is a sentence $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ such that, for all topologies $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$, we have $L_{\mathcal{T}}(\varphi) = L_{\mathcal{T}}(\mathcal{A})$.*

Proof. The proof is by a standard construction, and we give only a sketch of it. Suppose the set of states of $\mathcal{A}$ is S. Using a block $\exists(X_s)_{s\in S}$ of existentially quantified second-order variables, the formula will guess an assignment of states to events, which is checked by the first-order part for being an accepting run. As the requirements of a run can be verified locally, the predicates from the logic $\mathrm{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ are indeed sufficient. To verify the existence of an appropriate initial state, we have to determine the type of a process, which is done using formulas $a \in \mathit{type}(x)$. Moreover, to simulate an acceptance constraint "s occurs at least k times as the terminal state of a process", there will be a subformula that asks for k pairwise distinct $\lhd_{\mathsf{proc}}$-maximal events $x_1, \ldots, x_k$ such that $\bigwedge_{i\in\{1,\ldots,k\}} x_i \in X_s$. □

6.2 Proof Sketch for Main Result (Theorem 6)

The rest of this section is devoted to the proof of Theorem 6 (which works for Theorem 8 with only minor changes).

Set $\sigma^* = \{\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$ and let $\varphi \in \mathrm{EMSO}_{\mathcal{N}}[\sigma^*]$ be a sentence. According to Theorem 3, there are a radius $r = r_\varphi \geq 1$ and a sentence $\varphi' = \exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \chi \in \mathrm{EMSO}_{\mathcal{N}}[\sigma^*]$ such that $\chi \in \mathrm{FO}_{\mathcal{N}}[\sigma^*]$ is (r, σ^*)-local around y and, for all $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$, we have $L_{\mathcal{T}}(\varphi) = L_{\mathcal{T}}(\varphi')$. The free variables of $\forall y\chi$ can be considered as unary predicates and are dealt with by projection from an extended alphabet. By means of the acceptance condition of a PCA, one can make sure that variables x_i are indeed interpreted as exactly one event. So, it essentially remains to translate the formula $\forall y\chi$ into a PCA.

There is, however, another subtlety. Satisfaction of χ in an MSC depends on the neighborhood of y of radius r but also on the truth values of propositions involving only the free variables of $\forall y\chi$. Following [17, page 806], the PCA will guess and verify these truth values. By means of the acceptance condition, we can make sure that the guess is consistent throughout a run.

For simplicity, we henceforth suppose that y is the only free variable of χ (and write $\chi(y)$). Thus, for the rest of the proof, we fix $r, B \geq 1$, an $(r+2)$-unambiguous set $\mathfrak{T} \subseteq \mathbb{T}_{\mathcal{N}}$ of topologies, and a sentence $\forall y\chi(y) \in \mathrm{FO}_{\mathcal{N}}[\sigma^*]$ such that $\chi(y)$ is (r, σ^*)-local around y. We will build a PCA $\mathcal{A} \in \mathbb{PCA}_{\mathcal{N}}$ such that, for all $\mathcal{T} \in \mathfrak{T}$, we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\forall y\chi(y))$. We sketch the construction and try to give some intuition. The formal definition of $\mathcal{A}$ is technical and requires a lot of additional notation.

We exploit locality of $\chi(y)$: to know whether $M, e \models \chi(y)$, i.e., MSC M satisfies $\chi(y)$ when y is interpreted as e, it is sufficient to look at the neighborhood of e with radius r (as y is the only free variable, $r-1$ actually would be enough).

Example 6. Consider the MSC M over $\mathcal{T} = \mathcal{T}^{2,5}_{\mathrm{grid}}$ depicted in Figure 10. Take any event e that is located on process p. All events f such that $\mathit{dist}^{\sigma^*}_M(e, f) \leq r = 3$ lie on a process in the gray-shaded *topology neighborhood* of p with radius $R = \lceil r/2 \rceil = 2$, which has p as a distinguished center. We call this neighborhood a *sphere* and denote it by $R\text{-Sph}(\mathcal{T}, p)$. One major task of $\mathcal{A}$ is to identify spheres in the topology it is run on. But it has to rely on the messages that are predetermined by M. Therefore, $\mathcal{A}$ can actually only detect a substructure of $R\text{-Sph}(\mathcal{T}, p)$. Figure 11 depicts its restriction $R\text{-Sph}(\mathcal{T}, p) \restriction M$ (gray-shaded) to those edges that are "covered" by a message of M and to those nodes that one can reach from p with at most R such edges. However, every process preserves its complete type information. Observe that process $(2, 5)$ is not part of $R\text{-Sph}(\mathcal{T}, p) \restriction M$ anymore. Moreover, the edge between $(1, 3)$ and $(1, 4)$ is removed, since it is not covered by a message. Let $R\text{-Sph}(M, p)$ be the restriction of M to $R\text{-Sph}(\mathcal{T}, p) \restriction M$ (cf. again Figure 11). We call $R\text{-Sph}(M, p)$ a *partial* MSC, since it has some unmatched events. In fact, satisfaction $M, e \models \chi(y)$ only depends on $R\text{-Sph}(M, p)$, for all events e on p. □

The example illustrates that, essentially, we have to cope with spheres and partial MSCs, i.e., structures of bounded "width". For a fixed sphere, every $\mathrm{MSO}_{\mathcal{N}}$-formula can be translated to a (fixed-topology) CA that accepts exactly the partial MSCs that are a model of the formula [19]. For [19] to apply, we need to restrict to B-bounded MSCs (unless we prove Theorem 8, which is based on [6]). Thus, given a sphere θ, we can construct a CA $\mathcal{B}_\theta$ that recognizes the partial MSCs over θ that satisfy $\chi(y)$ for all events e located on the sphere center. Up to isomorphism, there are only finitely many spheres θ of radius R and, thus, finitely many CA $\mathcal{B}_\theta$. To obtain the PCA $\mathcal{A}$ running on MSCs over arbitrarily large topologies, we will *glue* these CA together. Note that, to exploit unambiguousness, it is crucial that we restrict to those spheres that arise from topologies in $\mathfrak{T}$.

We proceed as follows. Every process p guesses a sphere θ, supposing that its topology neighborhood looks like θ, and runs a copy of $\mathcal{B}_\theta$ to make sure that the partial MSC in its neighborhood is accepted by $\mathcal{B}_\theta$. Whenever p communicates with neighboring processes, the guess is forwarded in terms of messages. Processes receiving the guess also have to simulate $\mathcal{B}_\theta$. Since neighboring processes have to verify their own guess as well, a process will actually have to run several CA simultaneously. Note that processes reach an agreement only if their non-deterministic guesses are consistent. That is, at some point, they will have to guess a sphere and may realize only later whether these spheres are compatible. The main difficulty, however, is to verify that a (consistent) guess is correct and reflects a neighborhood in the underlying MSC so that the right CA is applied. The procedure of guessing and forwarding spheres is not able to check by itself whether a cycle in a sphere is correctly simulated in an MSC, and vice versa. It is only correct by the fact that the underlying set $\mathfrak{T}$ of topologies is $(r+2)$-unambiguous. Indeed, $2R + 1 = 2\lceil r/2 \rceil + 1 \leq r + 2$ is the maximal length of a cycle through a sphere center that is needed to cover a given edge in the sphere.

Example 7. We resume Example 6 to illustrate the functioning of $\mathcal{A}$. So, assume $\mathcal{T} = \mathcal{T}^{2,5}_{\mathrm{grid}} \in \mathfrak{T}$, $r = 3$, and $R = \lceil r/2 \rceil = 2$. In an accepting run of $\mathcal{A}$ on the MSC M from Figure 10, process $p = (2, 3)$ will guess the sphere $\theta = R\text{-Sph}(\mathcal{T}, p) \restriction M$ illustrated in Figure 11. Accordingly, it will launch the fixed-topology CA $\mathcal{B}_\theta$, which accepts the partial MSC $R\text{-Sph}(M, p)$ (cf. again Figure 11). Actually, $\mathcal{B}_\theta$ consists of several local automata $\mathcal{B}_\theta[q]$, one per sphere process q (rather than process type). Thus, p simulates the local automaton $\mathcal{B}_\theta[(2', 3')]$. In doing so, it eventually sends θ to process $(2, 2)$, together with its current position $(2', 3')$ in θ. Receiving the message through interface a, $(2, 2)$ can infer its own position $(2', 2')$ in θ, and so it learns that it has to simulate $\mathcal{B}_\theta[(2', 2')]$. Similarly, $(2, 2)$ has to receive a message over interface d that confirms that $(1, 2)$ has launched $\mathcal{B}_\theta[(1', 2')]$, and so on. There are subtle arguments and technical issues in $\mathcal{A}$ that guarantee that $\mathcal{T}$ and θ simulate each other. We sketch only the direction "$\mathcal{T}$ simulates θ". Let $w = (b, a)(d, c)(a, b)(c, d)$. Our construc-

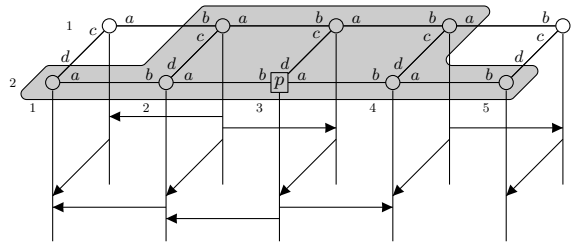


Figure 10. MSC M over $\mathcal{T}^{2,5}_{\text{grid}}$, and 2-Sph$(\mathcal{T}^{2,5}_{\text{grid}}, p)$

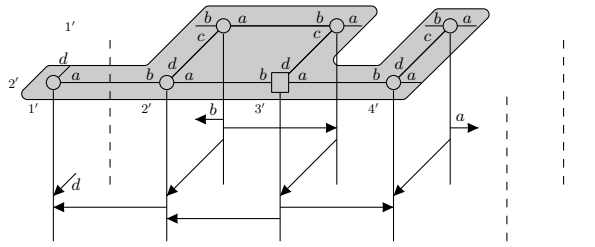


Figure 11. MSC 2-Sph(M, p) over 2-Sph$(\mathcal{T}^{2,5}_{\text{grid}}, p) \restriction M$

tion makes sure that, starting from p, $\mathcal{T}$ exhibits a w-labeled path. A priori, this does not imply that the path returns to p. But as θ arises from the $(r+2)$-unambiguous class $\mathfrak{T}$, $|w| = 4 \leq r+2$, and w forms a cycle in θ, w forms a cycle in $\mathcal{T}$ as well. Recall that every process has to run several CA simultaneously, which we did not take into account in the example. □

Remark 4. The normal form stated in Theorem 3 can be computed effectively (it builds on Gaifman's effective normal form [16]). Thus, Theorem 6 is constructive if the spheres R-Sph$(\mathcal{T}, p) \restriction M$ with $\mathcal{T} \in \mathfrak{T}$ are effectively representable (which holds for all standard classes). It follows from the word case that the PCA cannot be computed in elementary time.

Remark 5. We cannot exploit [23, 26], dealing with universally bounded CA, instead of [19], even if we restrict to universally B-bounded MSCs M (*all* linearizations are B-bounded): while R-Sph(M, p) is guaranteed to be (existentially) B-bounded, it is not necessarily universally B-bounded.

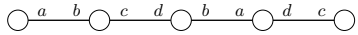
7. EMSO vs. PCA over Prime Topologies

Next, we present an orthogonal approach to realizability, where the construction of a PCA is independent of a concrete class of topologies. For this, we will restrict the logic further.

Recall that satisfaction of an FO$_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$-formula essentially depends on the $\{\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\}$-neighborhoods that occur in an MSC[5] (Theorem 3). Up to isomorphism, there are only finitely many such neighborhoods, for a fixed radius. We slightly *modify* the construction from [6] (which we had used as a *black-box* for Theorem 8) and define a PCA that, when running on an MSC, outputs the neighborhood of each event. However, this automaton can, a priori, not detect cycles (cf. Theorem 4) and "needs some help" from the underlying topology. It is only guaranteed to compute neighborhoods correctly when it is run on *prime* topologies. "Prime" is in the spirit of "unambiguous", but on a lower level, since it is tailored to detecting neighborhoods in MSCs rather than in topologies. While "prime" discards all ring topologies and ring forests, it includes all pipelines, trees, and grids. We give more intuition in the proof sketch for Theorem 10 below.

Definition 5. *A topology $(P, \longmapsto) \in \mathbb{T}_{\mathcal{N}}$ is called* prime *if, for all $p \in P$, $w \in (\mathcal{N} \times \mathcal{N})^*$, and $n \geq 1$, we have that $p \overset{w^n}{\longmapsto} p$ implies $p \overset{w}{\longmapsto} p$.*

In other words, a prime topology satisfies the following monotonicity property: If $p \overset{w}{\longmapsto} q$ with $p \neq q$, then starting from p and "applying" w several times will never lead back to p. Note that "prime" is a property of a *single* topology, while "unambiguous" refers to a *class* of topologies. Both notions are, in general, incomparable. For example, for every $n \geq 3$, the ring topology $\mathcal{T}^n_{\text{ring}}$ is not prime, while the class $\{\mathcal{T}^n_{\text{ring}}\}$ is k-unambiguous, for all $k \in \mathbb{N}$. Conversely, consider $\mathcal{T}^{2,2}_{\text{grid}}$ and the topology $\mathcal{T}$ given as follows:



Both, $\mathcal{T}^{2,2}_{\text{grid}}$ and $\mathcal{T}$, are prime. However, $\{\mathcal{T}^{2,2}_{\text{grid}}, \mathcal{T}\}$ is not 4-unambiguous, as $(a,b)(c,d)(b,a)(d,c)$ gives rise to a cycle in the grid, while it does not form a cycle in $\mathcal{T}$.

Lemma 2. *All topologies in $\mathfrak{T}_{\text{lin}}$, $\mathfrak{T}_{\text{tree}}$, and $\mathfrak{T}_{\text{grid}}$ are prime, while none of the topologies in $\mathfrak{T}_{\text{ring}}$ is prime.*

We build a PCA that is equivalent to a given formula φ on all prime topologies. Unlike in Theorem 6, it does not depend on a class of topologies, but only on φ and $\mathcal{N}$.

Theorem 10. *Let $\varphi \in \text{EMSO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ be a sentence. There is a PCA $\mathcal{A} \in \mathbb{PCA}_{\mathcal{N}}$ such that, for all prime topologies $\mathcal{T} \in \mathbb{T}_{\mathcal{N}}$, we have $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\varphi)$.*

Proof. We will sketch the idea. Thanks to Theorem 3, the problem reduces to constructing a PCA from a formula $\forall y \chi \in \text{FO}_{\mathcal{N}}[\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$ where χ is $(r, \{\lhd_{\mathsf{proc}}, \lhd_{\mathsf{msg}}\})$-local around y, for some $r \geq 1$ (cf. proof of Theorem 6). To translate $\forall y \chi$ into a PCA, we use the *sphere automaton*, a PCA that "detects" neighborhoods of radius r in an input MSC (including possible interpretations of free variables). More precisely, it accepts any MSC, over any given *prime* topology. Moreover, in any accepting run, the state assigned to event e tells us whether χ holds, or not, when y is interpreted as e. A sphere automaton is presented in [6] for a fixed, known topology, but it is actually independent of that topology. In the proof, it is only needed that MSCs are prime, essentially in the same sense as for topologies. But MSCs over prime topologies are indeed prime so that we obtain the desired sphere automaton. As a last step, the latter is restricted to states that signal that χ holds.

To compute neighborhoods, the PCA has to guarantee that certain sequences w of "directions" form a cycle in an MSC. While there is no direct way to enforce that w forms a cycle, a PCA can make sure that, for all $n \in \mathbb{N}$, an event admits a w^n-labeled path. As MSCs inherit the prime-property from topologies, this implies that w indeed gives rise to a cycle. □

Remark 6. Using Hanf's normal form (which we did not do to avoid additional notation), the PCA can be built in elementary time [5]. This is a priori not guaranteed using Theorem 3.

8. Conclusion

In this paper, we developed a framework for communicating systems with parameterized network topology. In particular, we provided various characterizations of PCA in terms of EMSO logic. Our constructions and proofs are uniform and capture typical cases such as pipelines, trees, grids, and rings. Table 1 gives a rough summary of our positive results.

[5] In a neighborhood, we only keep the process types, not the processes.

	$\lhd_{\mathsf{proc}}$	$\lhd_{\mathsf{msg}}$	$\sim$	$\lhd^*_{\mathsf{proc}}$
unambiguous, bounded (Thm. 6)	•	•	•	•
unambiguous, unbounded (Thm. 8)	•	•	•	
prime, unbounded (Thm. 10)	•	•		

Table 1. Summary

By Theorem 7, the notion "unambiguous" is not optimal: all $\mathrm{EMSO}_{\{a,b\}}[\lhd^*_{\mathsf{proc}}, \lhd_{\mathsf{msg}}]$-formulas are realizable for $\mathfrak{T}_{\mathrm{ring}}$, which is not k-unambiguous, for all $k \geq 3$. It seems difficult to characterize exactly those classes for which all formulas from a given logic are realizable, but it will be worthwhile to search for classes with comparably simple characterizations that generalize our results.

Our constructions crucially rely on the bounded-degree property. An obvious question would be a framework including topologies of unbounded degree such as star topologies or, more generally, unranked trees, possibly under restrictions like bounded-depth processes [27]. However, it is not clear how a parameterized automaton should look like in that case. One possibility is to employ registers so that, at any time, a process can remember "some" of its neighbors [4, 11]. One may also consider logical characterizations of systems with broadcast and/or FIFO communication [13].

Our framework may carry over to Zielonka's asynchronous automata [32] with binary actions. These automata have been considered in [18] over tree architectures to get decidability of the controller-synthesis problem. This also raises the question about a parameterized formulation of the control problem.

It is important to study also parameterized *verification*: Given a PCA $\mathcal{A}$, is there a topology $\mathcal{T}$ such that $L_{\mathcal{T}}(\mathcal{A}) \neq \emptyset$? Since those questions are undecidable in general, one has to impose restrictions, on PCA and/or on the topologies.

Acknowledgments

The author thanks Dietrich Kuske for stimulating discussions, which contributed substantially to this work. He thanks Nicolas Baudru for the pointer to [10]. The anonymous reviewers gave insightful comments and interesting suggestions for future work. The author is particularly indebted to a reviewer whose remarks led to a considerably improved presentation and further results. This work was partially supported by EGIDE/DAAD-Procope (TAMTV).

References

[1] P. A. Abdulla, A. Bouajjani, B. Jonsson, and M. Nilsson. Handling global conditions in parameterized system verification. In *Proc. of CAV'99*, volume 1633 of *LNCS*, pages 134–145. Springer, 1999.

[2] R. Alur, K. Etessami, and M. Yannakakis. Realizability and verification of MSC graphs. *Theor. Comput. Sci.*, 331(1):97–114, 2005.

[3] R. Alur and P. Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3):1–43, 2009.

[4] B. Bollig, A. Cyriac, L. Hélouët, A. Kara, and T. Schwentick. Dynamic communicating automata and branching high-level MSCs. In *Proc. of LATA'13*, volume 7810 of *LNCS*, pages 177–189. Springer, 2013.

[5] B. Bollig and D. Kuske. An optimal construction of Hanf sentences. *J. Appl. Log.*, 10(2):179–186, 2012.

[6] B. Bollig and M. Leucker. Message-passing automata are expressively equivalent to EMSO logic. *Theor. Comput. Sci.*, 358(2-3):150–172, 2006.

[7] A. Bouajjani, P. Habermehl, and T. Vojnar. Verification of parametric concurrent systems with prioritised FIFO resource management. *Form. Method. Syst. Des.*, 32(2):129–172, 2008.

[8] M. C. Browne, E. M. Clarke, and O. Grumberg. Reasoning about networks with many identical finite state processes. *Inf. Comput.*, 81(1):13–31, 1989.

[9] J. Büchi. Weak second-order arithmetic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.

[10] J. Chalopin, S. Das, and A. Kosowski. Constructing a map of an anonymous graph: Applications of universal sequences. In *Proc. of OPODIS'10*, volume 6490 of *LNCS*, pages 119–134. Springer, 2010.

[11] G. Delzanno, A. Sangnier, and R. Traverso. Parameterized verification of broadcast networks of register automata. In *Proc. of RP'13*, volume 8169 of *LNCS*, pages 109–121. Springer, 2013.

[12] G. Delzanno, A. Sangnier, and G. Zavattaro. Parameterized verification of ad hoc networks. In *Proc. of CONCUR'10*, volume 6269 of *LNCS*. Springer, 2010.

[13] G. Delzanno and R. Traverso. Decidability and complexity results for verification of asynchronous broadcast networks. In *Proc. of LATA'13*, volume 7810 of *LNCS*, pages 238–249. Springer, 2013.

[14] C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.

[15] E. A. Emerson and K. S. Namjoshi. On reasoning about rings. *Int. J. Found. Comput. Sci.*, 14(4):527–550, 2003.

[16] H. Gaifman. On local and nonlocal properties. In J. Stern, editor, *Logic Colloquium '81*, pages 105–135. North-Holland, 1982.

[17] P. Gastin and D. Kuske. Uniform satisfiability problem for local temporal logics over Mazurkiewicz traces. *Inf. Comput.*, 208(7):797–816, 2010.

[18] B. Genest, H. Gimbert, A. Muscholl, and I. Walukiewicz. Asynchronous games over tree architectures. In *Proc. of ICALP'13*, volume 7966 of *LNCS*, pages 275–286. Springer, 2013.

[19] B. Genest, D. Kuske, and A. Muscholl. A Kleene theorem and model checking algorithms for existentially bounded communicating automata. *Inf. Comput.*, 204(6):920–956, 2006.

[20] B. Genest, D. Kuske, and A. Muscholl. On communicating automata with bounded channels. *Fundam. Inform.*, 80(1-3):147–167, 2007.

[21] S. M. German and A. P. Sistla. Reasoning about systems with many processes. *J. ACM*, 39(3):675–735, 1992.

[22] S. Grumbach and Z. Wu. Logical locality entails frugal distributed computation over graphs (extended abstract). In *Proc. of WG'09*, volume 5911 of *LNCS*, pages 154–165. Springer, 2010.

[23] J. G. Henriksen, M. Mukund, K. Narayan Kumar, M. Sohoni, and P. S. Thiagarajan. A theory of regular MSC languages. *Inf. Comput.*, 202(1):1–38, 2005.

[24] S. Jacobs and R. Bloem. Parameterized synthesis. In *Proc. of TACAS'12*, volume 7214 of *LNCS*, pages 362–376. Springer, 2012.

[25] H. J. Keisler and W. B. Lotfallah. Shrinking games and local formulas. *Ann. Pure Appl. Logic*, 128(1-3):215–225, 2004.

[26] D. Kuske. Regular sets of infinite message sequence charts. *Inf. Comput.*, 187:80–109, 2003.

[27] R. Meyer. On boundedness in depth in the pi-calculus. In *Proc. of IFIP-TCS'08*, volume 273 of *IFIP*, pages 477–489. Springer, 2008.

[28] T. Schwentick and K. Barthelmann. Local normal forms for first-order logic with applications to games and automata. *Discrete Math. Theor. Comput. Sci.*, 3(3):109–124, 1999.

[29] J. W. Thatcher and J. B. Wright. Generalized finite automata theory with application to a decision problem of second-order logic. *Math. Syst. Theory*, 2(1):57–81, 1968.

[30] W. Thomas. Elements of an automata theory over partial orders. In *Proc. of POMIV'96*, volume 29 of *DIMACS*. AMS, 1996.

[31] B. A. Trakhtenbrot. Finite automata and monadic second order logic. *Siberian Math. J*, 3:103–131, 1962. In Russian; English translation in *Amer. Math. Soc. Transl.* 59, 1966, 23–55.

[32] W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21:99–135, 1987.