Logic for Communicating Automata with Parameterized Topology

Benedikt Bollig

Laboratoire Spécification et Vérification

Séminaire Vérification LIAFA
27 janvier 2014
Introduction

Objective

A Büchi-Elgot-Trakhtenbrot theorem for communicating automata with parameterized topology.
Introduction

Objective

A Büchi-Elgot-Trakhtenbrot theorem for communicating automata with parameterized topology.

Bridges the gap between high-level specifications and system models.
Introduction

Objective

A Büchi-Elgot-Trakhtenbrot theorem for communicating automata with parameterized topology.

Bridges the gap between high-level specifications and system models.

Theorem (Büchi-Elgot-Trakhtenbrot ’60s)

Every MSO formula is equivalent to some finite automaton, and vice versa.
Introduction

Objective

A Büchi-Elgot-Trakhtenbrot theorem for communicating automata with parameterized topology.

Bridges the gap between high-level specifications and system models.

Theorem (Büchi-Elgot-Trakhtenbrot ’60s)

Every MSO formula is equivalent to some finite automaton, and vice versa.
Introduction

Objective
A Büchi-Elgot-Trakhtenbrot theorem for communicating automata with parameterized topology.

Bridges the gap between high-level specifications and system models.

Theorem (Büchi-Elgot-Trakhtenbrot ’60s)
Every MSO formula is equivalent to some finite automaton, and vice versa.

Has been extended to trees, graphs, weighted automata, …
... and communicating automata (CA)
... and communicating automata (CA)

Büchi-Elgot-Trakthenbrot theorems:
∀-bounded channels
[Henriksen-Mukund-Kumar-Sohoni-Thiagarajan 2000]
∃-bounded channels
[Genest-Kuske-Muscholl 2004]
unbounded channels (but weaker logic)
[B.-Leucker 2004]

But ... all results require communication topology to be fixed!
... and communicating automata (CA)

∀x(?b(x) → ∃y(x ⊳^*_{proc} y ∧ !b(y))
... and communicating automata (CA)

\[ \forall x (?b(x) \rightarrow \exists y (x \triangleleft_{\text{proc}}^* y \land !b(y)) \]

Büchi-Elgot-Trakthenbrot theorems:

- **∀-bounded channels** [Henriksen-Mukund-Kumar-Sohoni-Thiagarajan 2000]
- **∀-bounded channels** [Kuske 2002]
- **∃-bounded channels** [Genest-Kuske-Muscholl 2004]
- unbounded channels (but weaker logic) [B.-Leucker 2004]
... and communicating automata (CA)

∀x(?b(x) → ∃y(x ⊳^*_proc y ∧ !b(y))

Büchi-Elgot-Trakthenbrot theorems:
- ∀-bounded channels [Henriksen-Mukund-Kumar-Sohoni-Thiagarajan 2000]
- ∀-bounded channels [Kuske 2002]
- ∃-bounded channels [Genest-Kuske-Muscholl 2004]
- unbounded channels (but weaker logic) [B.-Leucker 2004]

But ...

... all results require communication topology to be fixed!
... and communicating automata (CA)

∀x(?b(x) → ∃y(x ≺^*_proc y ∧ !b(y))

Büchi-Elgot-Trakthenbrot theorems:

- ∀-bounded channels [Henriksen-Mukund-Kumar-Sohoni-Thiagarajan 2000]
- ∀-bounded channels [Kuske 2002]
- ∃-bounded channels [Genest-Kuske-Muscholl 2004]
- unbounded channels (but weaker logic) [B.-Leucker 2004]

But ...

... all results require communication topology to be fixed!
Towards a parameterized version

Parameterized realizability

Let $\varphi$ be a formula and $\mathcal{T}$ be a class of topologies.

Is there a CA that is equivalent to $\varphi$ on all topologies $\mathcal{T} \in \mathcal{T}$?
Towards a parameterized version

Parameterized realizability

Let $\varphi$ be a formula and $\mathcal{T}$ be a class of topologies.

Is there a CA that is equivalent to $\varphi$ on all topologies $T \in \mathcal{T}$?

More precisely:

- $\varphi$ is an MSO formula over MSCs (directed acyclic graphs)
- $\mathcal{T}$ is a class of topologies of bounded degree (such as pipelines, trees, grids, and rings)
Towards a parameterized version

Parameterized realizability

Let $\varphi$ be a formula and $\mathcal{X}$ be a class of topologies.

Is there a CA that is equivalent to $\varphi$ on all topologies $\mathcal{T} \in \mathcal{X}$?

More precisely:

- $\varphi$ is an MSO formula over MSCs (directed acyclic graphs)
- $\mathcal{X}$ is a class of topologies of bounded degree
  (such as pipelines, trees, grids, and rings)

Need for new notions

- Topologies (of bounded degree)
- Parameterized communicating automata (PCA)
Outline

- Topologies and MSCs
Outline

- Topologies and MSCs
- Parameterized communicating automata
Negative results: There is a formula $\varphi \in C$ that is not realizable for $T$.

Positive results: All formulas $\varphi \in C$ are realizable for $T$. 

Outline

- Topologies and MSCs
- Parameterized communicating automata
- MSO logic
Outline

- Topologies and MSCs
- Parameterized communicating automata
- MSO logic
- Negative results:
  There is a formula $\varphi \in C$ that is not realizable for $\mathcal{T}$. 
Outline

- Topologies and MSCs
- Parameterized communicating automata
- MSO logic

**Negative results:**

There is a formula $\varphi \in C$ that is not realizable for $\mathcal{I}$.

**Positive results:**

All formulas $\varphi \in C$ are realizable for $\mathcal{I}$. 
Topologies and MSCs
Topologies

Topology

\[
b \quad a
\]
Topologies

Topology

\[ a \quad b \quad a \quad b \quad a \quad b \quad a \quad b \]
Topologies

Topology

$\begin{array}{c}
a \quad b \quad a \quad b \quad a \quad b \quad a \quad b
\end{array}$
Topologies and MSCs
Topologies

Pipeline

\[
\begin{align*}
& a \quad b \\
& a \quad b \\
& a \quad b \\
& a \quad b \\
& a \quad b
\end{align*}
\]
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

**Definition**

A topology over $\mathcal{N}$ is a pair $\mathcal{T} = (P, \rightarrow)$ where

- $P$ is the nonempty finite set of processes
- $\rightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

---

Topologies and MSCs
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

**Definition**

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

- $P$ is the nonempty finite set of processes
- $\rightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

Whenever $p \xrightarrow{a \ b} q$, the following hold:
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

**Definition**

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

- $P$ is the nonempty finite set of processes
- $\longrightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

Whenever $p \xleftarrow{a \ b} q$, the following hold:

1. $p \neq q$
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

**Definition**

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

- $P$ is the nonempty finite set of processes
- $\rightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

Whenever $p \xrightarrow{a\ b} q$, the following hold:

1. $p \neq q$
2. $q \xrightarrow{b\ a} p$
Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

**Definition**

A topology over $\mathcal{N}$ is a pair $\mathcal{T} = (P, \rightarrow)$ where

- $P$ is the nonempty finite set of processes
- $\rightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

Whenever $p \xrightarrow{a \ b} q$, the following hold:

1. $p \neq q$
2. $q \xrightarrow{b \ a} p$
3. $p \xrightarrow{a' \ b'} q'$ implies $(a = a' \iff q = q')$
Topologies

Pipeline

Fix finite set $\mathcal{N} = \{a, b, c, \ldots\}$ of interface names.

Definition

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

- $P$ is the nonempty finite set of processes
- $\rightarrow \subseteq P \times \mathcal{N} \times \mathcal{N} \times P$ is the edge relation

Whenever $p \xrightarrow{a \ b} q$, the following hold:

1. $p \neq q$
2. $q \xleftarrow{b \ a} p$
3. $p \xrightarrow{a' \ b'} q'$ implies $(a = a' \iff q = q')$
Topologies

Pipeline $\mathcal{T}_{\text{lin}}^5$

Diagram:

- A linear pipeline with five nodes labeled $a$, $b$, $a$, $b$, $a$, $b$. Each node is connected to the next in a linear sequence.
Topologies

Pipeline $\mathcal{T}_{\text{lin}}^5$

Tree

Topologies and MSCs
**Topologies**

**Pipeline** $\mathcal{T}_{\text{lin}}^5$

```
  a  b  a  b  a  b  a  b
```

**Tree**

```
  a  
 /   
 b  c  d  
   |   |   
   a  b  a
```

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

```
  a  b  a  b  a  b  a  b
  c  c  c  c  c  c  c  c
da  d  d  d  d  d  d  d
  a  b  a  b  a  b  a  b
  c  c  c  c  c  c  c  c
da  d  d  d  d  d  d  d
  a  b  a  b  a  b  a  b
  c  c  c  c  c  c  c  c
```

---

Topologies and MSCs  
8 / 34
Topologies

Pipeline $\mathcal{T}_{\text{lin}}^5$

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

Tree

Ring $\mathcal{T}_{\text{ring}}^5$
Message Sequence Charts (MSCs)

MSC

Definition
An MSC over $\mathcal{T} = (\mathcal{P},\mathcal{E})$ is a triple $\mathcal{M} = (\mathcal{E}, \prec, \ell)$ where

- $\mathcal{E}$ is the nonempty finite set of events
- $\prec = \prec_{\text{proc}} \sqcup \prec_{\text{msg}} \subseteq \mathcal{E} \times \mathcal{E}$ is acyclic
- $\ell : \mathcal{E} \to \mathcal{P}^+$ some extra conditions
Message Sequence Charts (MSCs)

**Definition**

An **MSC** over $\mathcal{T} = (P, \rightarrow)$ is a triple $M = (E, \prec, \ell)$ where
Definition

An **MSC** over $\mathcal{T} = (P, \rightarrow)$ is a triple $M = (E, \triangleleft, \ell)$ where

- $E$ is the nonempty finite set of events
Message Sequence Charts (MSCs)

MSC

Definition

An MSC over $T = (P, \longrightarrow)$ is a triple $M = (E, \triangleleft, \ell)$ where

- $E$ is the nonempty finite set of events
- $\triangleleft = \triangleleft_{\text{proc}} \cup \triangleleft_{\text{msg}} \subseteq E \times E$ acyclic
Message Sequence Charts (MSCs)

**Definition**

An MSC over $\mathcal{T} = (P, \rightarrow)$ is a triple $M = (E, \triangleleft, \ell)$ where

- $E$ is the nonempty finite set of events
- $\triangleleft = \triangleleft_{\text{proc}} \cup \triangleleft_{\text{msg}} \subseteq E \times E$ acyclic
- $\ell : E \rightarrow P$
Message Sequence Charts (MSCs)

Definition

An MSC over $\mathcal{T} = (P, \rightarrow)$ is a triple $M = (E, \triangleleft, \ell)$ where

- $E$ is the nonempty finite set of events
- $\triangleleft = \triangleleft_{\text{proc}} \cup \triangleleft_{\text{msg}} \subseteq E \times E$ acyclic
- $\ell : E \rightarrow P$
Message Sequence Charts (MSCs)

**Definition**

An **MSC** over \( \mathcal{T} = (P, \longrightarrow) \) is a triple \( M = (E, \triangleleft, \ell) \) where

- \( E \) is the nonempty finite set of events
- \( \triangleleft = \triangleleft_{\text{proc}} \cup \triangleleft_{\text{msg}} \subseteq E \times E \) acyclic
- \( \ell : E \rightarrow P \)

+ some extra conditions
Parameterized Communicating Automata (PCA)
Parameterized communicating automata (PCA)

Parameters communicating automata \( A \) over \( \{a, b\} \)

\[
\begin{align*}
\{a\} & \xrightarrow{!\text{req}\, a} s_1 & \xrightarrow{?\text{ack}\, a} s_2 & \xrightarrow{!\text{req}\, a} t_4 & \xrightarrow{?\text{ack}\, b} t_3 & \xrightarrow{!\text{ack}\, b} u_2 \\
\{a, b\} & \xrightarrow{?\text{req}\, b} t_0 & \xrightarrow{!\text{req}\, a} t_1 & \xrightarrow{?\text{req}\, b} u_0 & \xrightarrow{?\text{ack}\, a} t_2 & \xrightarrow{?\text{ack}\, b} u_1 \\
\{b\} & \xrightarrow{?\text{req}\, b} u_1 & \xrightarrow{?\text{ack}\, b} \end{align*}
\]
Parameterized Communicating Automata (PCA)

PCA $\mathcal{A}$ over $\{a, b\}$

$\{a\} \quad \{a, b\} \quad \{b\}$

- $s_0 \xrightarrow{!\text{req}a} s_1 \xrightarrow{?\text{ack}a} s_2 \xrightarrow{!\text{ack}a} t_3 \xrightarrow{!\text{ack}b} t_4$
- $t_0 \xrightarrow{?\text{req}b} t_1 \xrightarrow{!\text{req}a} t_2 \xrightarrow{?\text{req}b} u_0 \xrightarrow{!\text{ack}b} u_1 \xrightarrow{!\text{ack}b} u_2$

PCA $\mathcal{A}$ running on $\mathcal{T}_{\text{lin}}^5$

- $a \quad b \quad a \quad b \quad a \quad b \quad a \quad b$
Parameterized communicating automata (PCA)

PCA $\mathcal{A}$ over $\{a, b\}$

- $\{a, b\}$
- $\{b\}$

$\mathcal{A}$ running on $T_{\text{lin}}^5$

- $a \rightarrow b \rightarrow a \rightarrow b \rightarrow a \rightarrow b$
- $s_0 \rightarrow s_1 \rightarrow s_2$
Parameterized communicating automata (PCA)

PCA $\mathcal{A}$ over $\{a, b\}$

PCA $\mathcal{A}$ running on $\mathcal{T}_{\text{lin}}^5$
Parameterized communicating automata (PCA)

**PCA \( A \) over \( \{a, b\} \)**

- \[ \{a, b\} \]
- \( t_0 \)
- \(?_{\text{req}}b\)
- \( t_1 \)
- \( !_{\text{req}}a\)
- \( t_2 \)
- \(?_{\text{ack}}a\)
- \( t_3 \)
- \( !_{\text{ack}}b\)
- \( t_4 \)

**PCA \( A \) running on \( \mathcal{T}_{\text{lin}}^5 \)**

- \( a \) \( b \)
- \( s_0 \)
- \( t_0 \)
- \(?_{\text{req}}b\)
- \( s_1 \)
- \( t_1 \)
- \( !_{\text{req}}a\)
- \( s_2 \)
- \( t_2 \)
- \(?_{\text{ack}}a\)
- \( s_3 \)
- \( t_3 \)
- \( !_{\text{ack}}b\)
- \( s_4 \)
- \( t_4 \)

- \( u_0 \)
- \(?_{\text{req}}b\)
- \( u_1 \)
- \( !_{\text{ack}}b\)
- \( u_2 \)

Accepted language \( L_{\mathcal{T}_{\text{lin}}^n}(A) = \{M_{\text{lin}}^n\} \) for all \( n \geq 2 \)
Parameterized communicating automata (PCA)

PCA $\mathcal{A}$ over $\{a, b\}$

PCA $\mathcal{A}$ running on $T_{\text{lin}}^5$

Accepted language $L_{T_{\text{lin}}}^n(\mathcal{A}) = \{M_{\text{lin}}^n\}$ for all $n \geq 2$.
Parameterized communicating automata (PCA)

Parameterized Communicating Automata (PCA)

\[ \text{PCA } A \text{ running on } \mathcal{T}_{\text{lin}}^5 \]

![Diagram of PCA A running on T_{lin}^5](image-url)
Parameterized communicating automata (PCA)

**PCA \( A \) running on \( \mathcal{T}_{\text{lin}}^5 \)**

![Diagram of a parameterized communicating automaton running on a linear trace of length 5](image)

- States: \( s_0, s_1, s_2, u_0, u_1, u_2 \)
- Transitions:
  - \( s_0 \to t_0 \): \( a \), \( b \)
  - \( t_0 \to t_1 \): \( a \), \( b \)
  - \( t_1 \to t_2 \): \( a \), \( b \)
  - \( t_2 \to t_3 \): \( a \), \( b \)
  - \( t_3 \to t_4 \): \( a \), \( b \)
  - \( t_4 \to u_0, u_1, u_2 \): \( a \), \( b \)

- Inputs:
  - \( !\text{req} a, !\text{req} b \)
  - \( ?\text{ack} a, ?\text{ack} b \)

- Accepted language:
  - \( L_{\text{lin}}(A) = \{ M_{\text{lin}} \} \) for all \( n \geq 2 \)
Parameterized communicating automata (PCA)

PCA $\mathcal{A}$ running on $\mathcal{T}_{\text{lin}}^5$
Parameterized communicating automata (PCA)

\[ M_{\text{lin}}^5 = \in L_{T_{\text{lin}}^5}(A) \]
Parameterized communicating automata (PCA)

Accepted language

\[ L_{\mathcal{T}_{\text{lin}}}^n (A) = \{ M_{\text{lin}}^n \} \]
for all \( n \geq 2 \)

\[ M_{\text{lin}}^5 = \]

\[ \in L_{\mathcal{T}_{\text{lin}}}^5 (A) \]
Parameterized communicating automata (PCA)

**PCA \( \mathcal{A} \) over \( \{a, b\} \)**

- **\( s_0 \)**
  - \( \text{!req } a \)
  - \( \text{?ack } a \)

- **\( s_1 \)**
  - \( \text{?req } b \)
  - \( \text{!req } a \)

- **\( s_2 \)**
  - \( \text{?ack } a \)
  - \( \text{?req } b \)

- **\( t_0 \)**
  - \( \text{?req } b \)

- **\( t_1 \)**
  - \( \text{!req } a \)
  - \( \text{!ack } a \)

- **\( t_2 \)**
  - \( \text{?ack } a \)

- **\( t_3 \)**
  - \( \text{!ack } b \)

- **\( t_4 \)**

- **\( u_0 \)**
  - \( \text{?req } b \)

- **\( u_1 \)**
  - \( \text{!req } a \)

- **\( u_2 \)**

\( S \) finite set of states
\( \text{Msg} \) finite set of messages
\( I : (2^N \backslash \{\emptyset\}) \rightarrow 2^S \) initial states
\( \Delta \) the set of transitions
\( F \) a boolean combination of statements

\[ \langle \#(s) \geq k \rangle \] with \( s \in S \) and \( k \in \mathbb{N} \)

\( s \) occurs at least \( k \) times as the terminal state of an active process.
Parameterized communicating automata (PCA)

Definition

A PCA over \( \mathcal{N} \) is a tuple \((S, \text{Msg}, \Delta, I, F)\):

- \( S \) finite set of states
Parameterized communicating automata (PCA)

**Definition**

A PCA over \( N \) is a tuple \((S, \text{Msg}, \Delta, I, F)\):

- \( S \) finite set of states
- \( \text{Msg} \) finite set of messages

\[ F = \bigwedge_{s \in S \setminus \{s_2, t_4, u_2\}} \neg \langle #(s) \geq 1 \rangle \]

\( s \) occurs at least \( k \) times as the terminal state of an active process

**PCA \( A \) over \( \{a, b\} \)**

- **States**
  - \( s_0 \)
  - \( s_1 \)
  - \( s_2 \)
  - \( t_0 \)
  - \( t_1 \)
  - \( t_2 \)
  - \( t_3 \)
  - \( t_4 \)
  - \( u_0 \)
  - \( u_1 \)
  - \( u_2 \)

- **Transitions**
  - \( !\text{req} a \) from \( s_0 \) to \( s_1 \)
  - \( ?\text{ack} a \) from \( s_1 \) to \( s_2 \)
  - \( ?\text{req} b \) from \( s_0 \) to \( t_0 \)
  - \( ?\text{ack} a \) from \( t_0 \) to \( t_1 \)
  - \( !\text{req} a \) from \( t_1 \) to \( t_2 \)
  - \( ?\text{ack} b \) from \( t_2 \) to \( t_3 \)
  - \( !\text{req} b \) from \( s_1 \) to \( u_0 \)
  - \( ?\text{ack} b \) from \( u_0 \) to \( u_1 \)
  - \( !\text{ack} b \) from \( u_1 \) to \( u_2 \)
  - \( !\text{ack} b \) from \( u_2 \) to \( t_4 \)
Parameterized communicating automata (PCA)

**Definition**

A PCA over \( \mathcal{N} \) is a tuple \((S, \text{Msg}, \Delta, I, F)\):

- **S** finite set of states
- **\text{Msg}** finite set of messages
- **I** : \((2^\mathcal{N} \setminus \{\emptyset\}) \rightarrow 2^S\) initial states

\[ I = \bigwedge_{s \in S \setminus \{s_2, t_4, u_2\}} \neg \langle \#(s) \geq 1 \rangle \]
Parameterized communicating automata (PCA)

**Definition**

A PCA over $\mathcal{N}$ is a tuple $(S, \text{Msg}, \Delta, I, F)$:
- $S$ finite set of states
- $\text{Msg}$ finite set of messages
- $I : (2^\mathcal{N} \setminus \{\emptyset\}) \rightarrow 2^S$ initial states
- $\Delta$ the set of transitions
Parameterized communicating automata (PCA)

Definition

A PCA over $\mathcal{N}$ is a tuple $(S, Msg, \Delta, I, F)$:

- $S$ finite set of states
- $Msg$ finite set of messages
- $I : (2^\mathcal{N} \setminus \{\emptyset\}) \to 2^S$ initial states
- $\Delta$ the set of transitions
- $F$ a boolean combination of statements
  $\langle \#(s) \geq k \rangle$ with $s \in S$ and $k \in \mathbb{N}$
### Parameterized communicating automata (PCA)

**PCA $\mathcal{A}$ over $\{a, b\}$**

- $\{a\} \rightarrow s_0 \rightarrow t_0 \rightarrow u_0 \rightarrow \{b\}$  
  - $s_0$  
  - $t_0$  
  - $u_0$

- $\{a, b\} \rightarrow t_1 \rightarrow u_1 \rightarrow \{a\}$  
  - $t_1$  
  - $u_1$

- $\{a\} \rightarrow s_1 \rightarrow t_2 \rightarrow u_2 \rightarrow \{b\}$  
  - $s_1$  
  - $t_2$  
  - $u_2$

- $\{b\} \rightarrow s_2 \rightarrow t_3 \rightarrow u_3 \rightarrow \{a\}$  
  - $s_2$  
  - $t_3$  
  - $u_3$

**Definition**

A PCA over $\mathcal{N}$ is a tuple $(S, \text{Msg}, \Delta, I, F)$:

- $S$ finite set of states
- $\text{Msg}$ finite set of messages
- $I : (2^N \setminus \{\emptyset\}) \rightarrow 2^S$ initial states
- $\Delta$ the set of transitions
- $F$ a boolean combination of statements $\langle \#(s) \geq k \rangle$ with $s \in S$ and $k \in \mathbb{N}$
  - “$s$ occurs at least $k$ times as the terminal state of an active process”
Parameterized communicating automata (PCA)

**Definition**

A PCA over $\mathcal{N}$ is a tuple $(S, \text{Msg}, \Delta, I, F)$:

- $S$ finite set of states
- $\text{Msg}$ finite set of messages
- $I : (2^\mathcal{N} \setminus \{\emptyset\}) \rightarrow 2^S$ initial states
- $\Delta$ the set of transitions
- $F$ a boolean combination of statements $\langle \#(s) \geq k \rangle$ with $s \in S$ and $k \in \mathbb{N}$
  
  “$s$ occurs at least $k$ times as the terminal state of an active process”

$$F = \bigwedge_{s \in S \setminus \{s_2, t_4, u_2\}} \neg \langle \#(s) \geq 1 \rangle$$
Parameterized communicating automata (PCA)

A PCA cannot say “the topology has at least 5 processes”
Parameterized communicating automata (PCA)

A PCA cannot say “the topology has at least 5 processes”

A PCA can say “at least 5 processes of type \{a, b\} are active”
A PCA cannot say “the topology has at least 5 processes”

A PCA can say “at least 5 processes of type \{a, b\} are active”

A PCA cannot distinguish between:

\[ \begin{array}{c}
\text{a} \quad \text{b} \\
\downarrow \quad \downarrow \\
\text{a} \quad \text{b} \\
\downarrow \quad \downarrow \\
\text{a} \quad \text{b} \\
\downarrow \quad \downarrow \\
\text{a} \quad \text{b} \\
\downarrow \quad \downarrow \\
\text{a} \quad \text{b} \\
\end{array} \]
Parameterized communicating automata (PCA)

A PCA cannot say “the topology has at least 5 processes”

A PCA can say “at least 5 processes of type \{a, b\} are active”

A PCA cannot distinguish between:
Parameterized communicating automata (PCA)

A PCA cannot say “the topology has at least 5 processes”

A PCA can say “at least 5 processes of type \{a, b\} are active”

A PCA cannot distinguish between:

```
a  b  a  b  a  b  a  b  a  b  a  b
```

```
a  b  a  b  a  b  a  b  a  b  a  b
```

```
a  b  a  b  a  b  a  b  a  b  a  b
```

```
a  b  a  b  a  b  a  b  a  b  a  b
```
Parameterized communicating automata (PCA)

- A PCA cannot say “the topology has at least 5 processes”
- A PCA can say “at least 5 processes of type \{a, b\} are active”
- A PCA cannot distinguish between:

![Diagram showing the inability to distinguish between two similar topologies]
MSO Logic
MSO logic

\[ \varphi ::= !a(x) \mid ?a(x) \mid a \in \text{type}(x) \mid \]
\[ x \triangleleft_{\text{proc}} y \mid x \triangleleft_{\text{proc}}^{*} y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft^{*} y \mid x \sim y \mid \]
\[ x = y \mid x \in X \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \exists X \varphi \]

where \( a \in \mathcal{N} \)
MSO logic

\[ \varphi ::= !a(x) \mid ?a(x) \mid a \in \text{type}(x) \mid \]

\[ x \triangleleft_{\text{proc}} y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft_{\ast} y \mid x \sim y \mid \]

\[ x = y \mid x \in X \mid \neg \varphi \mid \varphi \land \varphi \mid \exists x \varphi \mid \exists X \varphi \]

where \( a \in \mathcal{N} \)

\( x \sim y \) says that \( x \) and \( y \) are located on the same process.
MSO logic

\[ \varphi ::= !a(x) \mid ?a(x) \mid a \in \text{type}(x) \mid \]
\[ x \preceq_{\text{proc}} y \mid x \preceq_{\text{proc}}^* y \mid x \preceq_{\text{msg}} y \mid x \preceq^* y \mid x \sim y \mid \]
\[ x = y \mid x \in X \mid \neg \varphi \mid \varphi \land \varphi \mid \exists x \varphi \mid \exists X \varphi \]

where \( a \in \mathcal{N} \)

\( x \sim y \) says that \( x \) and \( y \) are located on the same process.

Define fragments:
- FO: first-order logic, without \( \exists X \varphi \)
### MSO Logic

**MSO logic**

$$\varphi ::= !a(x) \mid ?a(x) \mid a \in \text{type}(x) \mid$$

$$x \triangleleft_{\text{proc}} y \mid x \triangleleft^*_{\text{proc}} y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft^* y \mid x \sim y \mid$$

$$x = y \mid x \in X \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \exists X \varphi$$

where \( a \in \mathcal{N} \)

\( x \sim y \) says that \( x \) and \( y \) are located on the same process.

Define fragments:

- **FO**: first-order logic, without \( \exists X \varphi \)
- **EMSO**: formulas of the form \( \exists X_1 \ldots \exists X_n \varphi \) with \( \varphi \in \text{FO} \)
MSO logic

\[ \varphi ::= !a(x) \mid ?a(x) \mid a \in \text{type}(x) \mid \]

\[ \sigma \subseteq x \triangleleft_{\text{proc}} y \mid x \triangleleft_{\text{proc}}^* y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft^* y \mid x \sim y \mid \]

\[ x = y \mid x \in X \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \exists X \varphi \]

where \( a \in \mathcal{N} \)

\( x \sim y \) says that \( x \) and \( y \) are located on the same process.

Define fragments:

- **FO**: first-order logic, without \( \exists X \varphi \)
- **EMSO**: formulas of the form \( \exists X_1 \ldots \exists X_n \varphi \) with \( \varphi \in \text{FO} \)
- **FO[\( \sigma \)]** and **EMSO[\( \sigma \)]** (e.g., FO[\( \triangleleft_{\text{proc}}, \triangleleft_{\text{msg}} \)])
MSO logic

MSO logic

\[ \varphi ::= \!a(x) \mid \?a(x) \mid a \in \text{type}(x) \mid \]

\[ \sigma \subseteq x \triangleleft_{\text{proc}} y \mid x \triangleleft_{\text{proc}}^* y \mid x \triangleleft_{\text{msg}} y \mid x \triangleleft^* y \mid x \sim y \mid \]

\[ x = y \mid x \in X \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \exists X \varphi \]

where \( a \in \mathcal{N} \)

\( x \sim y \) says that \( x \) and \( y \) are located on the same process.

Define fragments:

- FO: first-order logic, without \( \exists X \varphi \)
- EMSO: formulas of the form \( \exists X_1 \ldots \exists X_n \varphi \) with \( \varphi \in \text{FO} \)
- FO[\( \sigma \)] and EMSO[\( \sigma \)] (e.g., FO[\( \triangleleft_{\text{proc}}, \triangleleft_{\text{msg}} \)])

Let \( L_{\mathcal{T}}(\varphi) \) be the set of MSCs over \( \mathcal{T} \) that are a model of \( \varphi \).
MSO logic

MSC $M^6_{\text{lin}}$
MSC $M^6_{\text{lin}}$

$M^6_{\text{lin}} \models \forall x (?b(x) \rightarrow \exists y (x \triangleleft^*_\text{proc} y \land !b(y)))$
MSO logic

MSC $M_{\text{lin}}^6$

$M_{\text{lin}}^6 \models \forall x(\exists y(x \triangleleft^*_\text{proc} y \land \neg b(y))) \in FO[\triangleleft^*_\text{proc}]$
\( M_{\text{lin}}^6 \models \forall x (\neg b(x) \rightarrow \exists y (x \triangleleft^* y \land \neg b(y))) \in \text{FO}[\triangleleft^*_\text{proc}] \)

\( M_{\text{lin}}^6 \models \forall x \forall y (x \triangleleft^* y \lor y \triangleleft^* x) \)
MSC $M_{\text{lin}}^6$

- $M_{\text{lin}}^6 \models \forall x (?b(x) \rightarrow \exists y (x \triangleleft^*_{\text{proc}} y \land !b(y))) \in \text{FO}[\triangleleft^*_{\text{proc}}]$
- $M_{\text{lin}}^6 \models \forall x \forall y (x \triangleleft^* y \lor y \triangleleft^* x)$
- $M_{\text{lin}}^6 \not\models \exists x \exists y (b \not\in \text{type}(x) \land a \not\in \text{type}(y) \land x \triangleleft_{\text{msg}} y)$
MSC $M_{\text{lin}}^6$

$M_{\text{lin}}^6 \models \forall x (?b(x) \rightarrow \exists y (x \triangleleft^*_{\text{proc}} y \land !b(y))) \in \text{FO}[\triangleleft^*_{\text{proc}}]$,

$M_{\text{lin}}^6 \models \forall x \forall y (x \triangleleft^* y \lor y \triangleleft^* x)$,

$M_{\text{lin}}^6 \not\models \exists x \exists y (b \not\in \text{type}(x) \land a \not\in \text{type}(y) \land x \triangleleft_{\text{msg}} y) =: \varphi$
**MSC $M^6_{\text{lin}}$**

- $M^6_{\text{lin}} \models \forall x (\exists b(x) \rightarrow \exists y (x \triangleleft^*_{\text{proc}} y \land \neg b(y))) \in \text{FO}[\triangleleft^*_{\text{proc}}]$
- $M^6_{\text{lin}} \models \forall x \forall y (x \triangleleft^* y \lor y \triangleleft^* x)$
- $M^6_{\text{lin}} \not\models \exists x \exists y (b \notin \text{type}(x) \land a \notin \text{type}(y) \land x \triangleleft_{\text{msg}} y) =: \varphi$
- $M^6_{\text{lin}} \models \varphi \iff n = 2$
Positive results

Theorem

For every PCA $\mathcal{A}$, there is a formula $\varphi \in \text{EMSO}[^{\ll}_{\text{proc}}, ^{\ll}_{\text{msg}}]$ that is equivalent to $\mathcal{A}$ on all topologies.
Positive results

**Theorem**

For every PCA $\mathcal{A}$, there is a formula $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$ that is equivalent to $\mathcal{A}$ on all topologies.

**Proof**

Standard.
Negative Results
Restrictions on topologies are necessary

Theorem

There exists a sentence \( \varphi \in \text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}] \) over \( \{a, b\} \) such that, for all PCA \( \mathcal{A} \), there is a ring forest \( \mathcal{T} \) with \( L_{\mathcal{T}}(\mathcal{A}) \neq L_{\mathcal{T}}(\varphi) \).
Restrictions on topologies are necessary

**Theorem**
There exists a sentence $\varphi \in \text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}]$ over $\{a, b\}$ such that, for all PCA $\mathcal{A}$, there is a ring forest $\mathcal{T}$ with $L_{\mathcal{T}}(\mathcal{A}) \neq L_{\mathcal{T}}(\varphi)$.

**Proof**

$$\varphi = \forall x \exists x_1, \ldots, x_6 (x \in \{x_1, \ldots, x_6\} \land \text{cycle}(x_1, \ldots, x_6))$$
Restrictions on topologies are necessary

Suppose there is $A$ such that $L_T(A) = L_T(\phi)$ for all ring forests $T$. We have $M = \ldots M \sqcup \ldots \sqcup M$ for all $n \in L_T$ ring $\ldots \sqcup T \ldots$ ring ($A$) for all $n \geq 1$. For large enough $n$, there is a run of $A$ that behaves the same on two copies of $M$. Replace the two copies with MSc over $T$. Contradiction. □

Lesson learned: PCA have limited ability to "detect" cycles.
Restrictions on topologies are necessary

Suppose there is $\mathcal{A}$ such that $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\varphi)$ for all ring forests $\mathcal{T}$. 

\[ M = \]

- Lesson learned: PCA have limited ability to "detect" cycles.
Restrictions on topologies are necessary

\[ M = \]

- Suppose there is \( \mathcal{A} \) such that \( L_T(\mathcal{A}) = L_T(\varphi) \) for all ring forests \( T \).
- We have \( \biguplus_{n} M \in \mathcal{L}(\mathcal{T}^{3}_{\text{ring}} \uplus \ldots \uplus \mathcal{T}^{3}_{\text{ring}}(\mathcal{A})) \) for all \( n \geq 1 \).
Restrictions on topologies are necessary

\[ M = \]

- Suppose there is \( \mathcal{A} \) such that \( L_T(\mathcal{A}) = L_T(\varphi) \) for all ring forests \( T \).
- We have \( M \uplus \ldots \uplus M \in L_{T_{\text{ring}}}^3 \uplus \ldots \uplus T_{\text{ring}}^3 (\mathcal{A}) \) for all \( n \geq 1 \).
- For large enough \( n \), there is a run of \( \mathcal{A} \) that behaves the same on two copies of \( M \).
Restrictions on topologies are necessary

Suppose there is $\mathcal{A}$ such that $L_{\mathcal{T}}(\mathcal{A}) = L_{\mathcal{T}}(\varphi)$ for all ring forests $\mathcal{T}$.

We have $M \uplus \ldots \uplus M \in L_{\mathcal{T}_{\text{ring}}^3 \uplus \ldots \uplus \mathcal{T}_{\text{ring}}^3}(\mathcal{A})$ for all $n \geq 1$.

For large enough $n$, there is a run of $\mathcal{A}$ that behaves the same on two copies of $M$.

Replace the two copies with MSC over $\mathcal{T}_{\text{ring}}^6$. Contradiction. □
Restrictions on topologies are necessary

Suppose there is $\mathcal{A}$ such that $L_T(\mathcal{A}) = L_T(\varphi)$ for all ring forests $T$.

We have $M \uplus \ldots \uplus M \in L_{T^{3}}_{\text{ring}} \uplus \ldots \uplus T_{\text{ring}}^{3}(\mathcal{A})$ for all $n \geq 1$.

For large enough $n$, there is a run of $\mathcal{A}$ that behaves the same on two copies of $M$.

Replace the two copies with MSC over $T^{6}_{\text{ring}}$. Contradiction.

Lesson learned

PCA have limited ability to “detect” cycles.
Restrictions on logic are necessary

Theorem

There exists a sentence $\varphi \in \text{FO}[\triangleleft^*_{\text{proc}}, \triangleleft_{\text{msg}}, \triangleleft^*]$ over $\{a, b, c, d\}$ such that, for all PCA $A$, there is a tree $T$ with $L_T(A) \neq L_T(\varphi)$.
Restrictions on logic are necessary

**Theorem**

There exists a sentence \( \varphi \in \text{FO}[\langle \mathsf{proc}^*, \mathsf{msg}^*, \mathsf{msg}^* \rangle] \) over \( \{a, b, c, d\} \) such that, for all PCA \( A \), there is a tree \( T \) with \( L_T(A) \neq L_T(\varphi) \).

**Proof (idea goes back to [Thomas 1996])**

\[
\begin{pmatrix}
\bullet & \circ & \circ & \circ \\
\circ & \circ & \bullet & \circ
\end{pmatrix}
\]
Restrictions on logic are necessary

Theorem

There exists a sentence \( \varphi \in FO[\preceq^\ast_{\text{proc}}, \preceq_{\text{msg}}, \preceq^\ast]\) over \{a, b, c, d\} such that, for all PCA \( A \), there is a tree \( T \) with \( L_T(A) \neq L_T(\varphi) \).

Proof (idea goes back to [Thomas 1996])

Lesson learned

Look at more “local” logics.
Positive Results
Locality of FO logic

**Theorem** [Schwentick-Barthelmann 1999]

Every formula $\varphi \in \text{FO}[\sigma]$ is equivalent to a formula of the form $\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]$ where $\psi$ is $r$-local around $y$, for some $r \geq 1$ (quantification is restricted to elements of distance $\leq r$ from $y$).
Locality of FO logic

**Theorem [Schwentick-Barthelmann 1999]**

Every formula \( \varphi \in \text{FO}[\sigma] \) is equivalent to a formula of the form
\[
\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]
\]
where \( \psi \) is \( r \)-local around \( y \), for some \( r \geq 1 \) (quantification is restricted to elements of distance \( \leq r \) from \( y \)).
Locality of FO logic

**Theorem** [Schwentick-Barthelmann 1999]

Every formula $\varphi \in \text{FO}[\sigma]$ is equivalent to a formula of the form $\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]$ where $\psi$ is $r$-local around $y$, for some $r \geq 1$ (quantification is restricted to elements of distance $\leq r$ from $y$).
Locality of FO logic

Theorem [Schwentick-Barthelmann 1999]

Every formula $\varphi \in \text{FO}[\sigma]$ is equivalent to a formula of the form $\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]$ where $\psi$ is $r$-local around $y$, for some $r \geq 1$ (quantification is restricted to elements of distance $\leq r$ from $y$).

$r = 3$

---

Positive Results
Locality of FO logic

**Theorem** [Schwentick-Barthelmann 1999]

Every formula $\varphi \in \text{FO}[\sigma]$ is equivalent to a formula of the form $\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]$ where $\psi$ is $r$-local around $y$, for some $r \geq 1$ (quantification is restricted to elements of distance $\leq r$ from $y$).
Locality of FO logic

**Theorem** [Schwentick-Barthelmann 1999]

Every formula $\varphi \in \text{FO}[\sigma]$ is equivalent to a formula of the form

$\exists x_1 \ldots \exists x_n \forall y \psi \in \text{FO}[\sigma]$ where $\psi$ is $r$-local around $y$, for some $r \geq 1$ (quantification is restricted to elements of distance $\leq r$ from $y$).

\[
r = 3 \\
\lceil r/2 \rceil = 2 \ \ \ \ \ \ \ [r/2] = 2
\]
Positive results

Theorem

Let \( \varphi \in \text{EMSO}[\prec^*_\text{proc}, \prec_\text{msg}], \ B \geq 1, \) and \( \mathcal{T} \) 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.

There is a PCA \( A \) such that, for all \( \mathcal{T} \in \mathcal{T} \), we have \( L_B^\mathcal{T}(A) = L_B^\mathcal{T}(\varphi) \).

Here, \( L_B^\mathcal{T}(A) \) is the restriction of \( L_T^A \) to \( B \)-bounded MSCs.

Theorem

Let \( \varphi \in \text{EMSO}[\prec^*_\text{proc}, \prec_\text{msg}], \ B \geq 1, \) and \( \mathcal{T} \) be a \((r_\varphi + 2)\)-unambiguous set of topologies. There is a PCA \( A \) such that, for all \( \mathcal{T} \in \mathcal{T} \), we have \( L_B^\mathcal{T}(A) = L_B^\mathcal{T}(\varphi) \).

Here, \( r_\varphi \) is the radius associated with the first-order kernel of \( \varphi \).
Positive results

Theorem

Let $\varphi \in \text{EMSO}[\langle^*_\text{proc}, \langle_\text{msg}\rangle], B \geq 1$, and $\mathcal{T}$ 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.

There is a PCA $A$ such that, for all $T \in \mathcal{T}$, we have $L_B^T(A) = L_B^T(\varphi)$.
Here, $L_B^T(A)$ is the restriction of $L_T(A)$ to $B$-bounded MSCs.
Positive results

Theorem

Let \( \varphi \in \text{EMSO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}] \), \( B \geq 1 \), and \( \mathcal{T} \) 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.

There is a PCA \( A \) such that, for all \( T \in \mathcal{T} \), we have \( L_B^T(A) = L_B^T(\varphi) \).

Here, \( r_\varphi \) is the radius associated with the first-order kernel of \( \varphi \).
### Positive results

#### Theorem

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$, $B \geq 1$, and $\mathcal{T}$ 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.

There is a PCA $A$ such that, for all $\mathcal{T} \in \mathcal{S}$, we have $L^B_\mathcal{T}(A) = L^B_\mathcal{T}(\varphi)$.

Here, $L^B_\mathcal{T}(A)$ is the restriction of $L_\mathcal{T}(A)$ to $B$-bounded MSCs.
Positive results

Corollary

Let $\varphi \in \text{EMSO}[\triangleleft^*_\text{proc}, \triangleleft_\text{msg}], \ B \geq 1,$ and $\mathcal{T}$ 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.

There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{Z},$ we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\varphi).$

Here, $L^B_{\mathcal{T}}(\mathcal{A})$ is the restriction of $L_{\mathcal{T}}(\mathcal{A})$ to $B$-bounded MSCs.
Positive results

Corollary

Let $\varphi \in \text{EMSO}[\prec^*_{\text{proc}}, \prec_{\text{msg}}]$, $B \geq 1$, and $\mathcal{I}$ 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.

There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{I}$, we have $L_B^\mathcal{T}(\mathcal{A}) = L_B^\mathcal{T}(\varphi)$.

Here, $L_B^\mathcal{T}(\mathcal{A})$ is the restriction of $L_\mathcal{T}(\mathcal{A})$ to $B$-bounded MSCs.

Theorem

Let $\varphi \in \text{EMSO}[\prec^*_{\text{proc}}, \prec_{\text{msg}}]$, $B \geq 1$, and $\mathcal{I}$ be a $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{I}$, we have $L_B^\mathcal{T}(\mathcal{A}) = L_B^\mathcal{T}(\varphi)$.

Here, $r_\varphi$ is the radius associated with the first-order kernel of $\varphi$. 
Unambiguous topology classes

**Definition**

Let \( k \in \mathbb{N} \). A class \( \mathcal{F} \) of topologies is *\( k \)-unambiguous* if, for all \( w \in (\mathcal{N} \times \mathcal{N})^* \) with \(|w| \leq k\), all \((P, \rightarrow), (P', \rightarrow') \in \mathcal{F}\), and all processes \( p, q \in P \) and \( p', q' \in P' \) such that \( p \xrightarrow{w} q \) and \( p' \xrightarrow{w'} q' \), we have \( p = q \) iff \( p' = q' \).
Unambiguous topology classes

Definition
Let $k \in \mathbb{N}$. A class $\mathcal{T}$ of topologies is $k$-unambiguous if, for all $w \in (\mathcal{N} \times \mathcal{N})^*$ with $|w| \leq k$, all $(P, \rightarrow), (P', \rightarrow') \in \mathcal{T}$, and all processes $p, q \in P$ and $p', q' \in P'$ such that $p \xrightarrow{w} q$ and $p' \xrightarrow{w'} q'$, we have $p = q$ iff $p' = q'$.

In other words:
If $w$ forms a cycle in a topology from $\mathcal{T}$, then it forms a cycle anywhere, in any topology of $\mathcal{T}$ (if it is applicable).
Unambiguous topology classes

- **Pipelines** $k$-unambiguous for all $k \in \mathbb{N}$

- **Trees** $k$-unambiguous for all $k \in \mathbb{N}$

- **Grids** $k$-unambiguous for all $k \in \mathbb{N}$

- **Rings** not $k$-unambiguous for all $k \geq 3$

But:
The class of rings of size $\geq k + 1$ is $k$-unambiguous, for all $k \in \mathbb{N}$.

Every single ring is $k$-unambiguous, for all $k \in \mathbb{N}$.
Unambiguous topology classes

Pipelines

- $k$-unambiguous for all $k \in \mathbb{N}$

Trees

- $k$-unambiguous for all $k \in \mathbb{N}$

Rings

- Not $k$-unambiguous for all $k \geq 3$

But:
The class of rings of size $\geq k + 1$ is $k$-unambiguous, for all $k \in \mathbb{N}$.

Every single ring is $k$-unambiguous, for all $k \in \mathbb{N}$.
Unambiguous topology classes

Pipelines $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

Trees $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

Grids $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

$w = (a, b)(c, d)(b, a)(d, c)$
Unambiguous topology classes

Pipelines  $\checkmark$  $k$-unambiguous for all $k \in \mathbb{N}$

Trees  $\checkmark$  $k$-unambiguous for all $k \in \mathbb{N}$

Grids  $\checkmark$  $k$-unambiguous for all $k \in \mathbb{N}$

Rings  $\times$  not $k$-unambiguous for all $k \geq 3$

$w = (a, b)(c, d)(b, a)(d, c)$
Unambiguous topology classes

**Pipelines** ✓ $k$-unambiguous for all $k \in \mathbb{N}$

![Diagram of pipelines]

**Trees** ✓ $k$-unambiguous for all $k \in \mathbb{N}$

![Diagram of trees]

**Grids** ✓ $k$-unambiguous for all $k \in \mathbb{N}$

![Diagram of grids]

**Rings** ✗ not $k$-unambiguous for all $k \geq 3$

![Diagram of rings]

\[ w = (a, b)(c, d)(b, a)(d, c) \]

But:

- The class of rings of size $\geq k + 1$ is $k$-unambiguous, for all $k \in \mathbb{N}$.
Unambiguous topology classes

Pipelines $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

Trees $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

Grids $\checkmark$ $k$-unambiguous for all $k \in \mathbb{N}$

Rings $\times$ not $k$-unambiguous for all $k \geq 3$

$w = (a, b)(c, d)(b, a)(d, c)$

But:

- The class of rings of size $\geq k + 1$ is $k$-unambiguous, for all $k \in \mathbb{N}$.
- Every single ring is $k$-unambiguous, for all $k \in \mathbb{N}$. 
Unambiguous topology classes

Theorem

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}], B \geq 1$, and $\mathcal{T}$ be a $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{T}$, we have $L^B_{\mathcal{T}}(\mathcal{A}) = L^B_{\mathcal{T}}(\varphi)$. 

Proof

Translate $\varphi$ into normal form

$\exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi$

where $\psi$ is $r_\varphi$-local around $y$.

Existential quantification $\Rightarrow$ projection & guessing of truth values for propositions involving only free variables of $\forall y \psi$.[Gastin-Kuske 2010]

Construct fixed-topology CA $\mathcal{A}^{\theta}$ for formula $\forall y \psi$ over all topology neighborhoods $\theta$ of radius $\lceil r_\varphi / 2 \rceil$.[Genest-Kuske-Muscholl 2004]

Glue fixed-topology CA together to obtain a PCA for $\forall y \psi$. 
Unambiguous topology classes

**Theorem**

Let $\varphi \in \text{EMSO}[\preceq^*_\text{proc}, \preceq_{\text{msg}}]$, $B \geq 1$, and $\mathcal{S}$ be a $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $A$ such that, for all $T \in \mathcal{S}$, we have $L^B_T(A) = L^B_T(\varphi)$.

**Proof**

- Translate $\varphi$ into normal form

  $$\exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi$$

  where $\psi$ is $r_\varphi$-local around $y$.  

Positive Results
**Unambiguous topology classes**

**Theorem**

Let \( \varphi \in \text{EMSO}[\prec_\text{proc}, \preceq_\text{msg}] \), \( B \geq 1 \), and \( \mathcal{T} \) be a \((r_\varphi + 2)\)-unambiguous set of topologies. There is a PCA \( A \) such that, for all \( \mathcal{T} \in \mathcal{T} \), we have \( L^B_T(A) = L^B_T(\varphi) \).

**Proof**

- **Translate \( \varphi \) into normal form**

  \[
  \exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi
  \]

  where \( \psi \) is \( r_\varphi \)-local around \( y \).

- **Existential quantification \( \Rightarrow \) projection & guessing of truth values for propositions involving only free variables of \( \forall y \psi \)** [Gastin-Kuske 2010].
Unambiguous topology classes

Theorem

Let \( \varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}] \), \( B \geq 1 \), and \( \mathcal{S} \) be a \((r_\varphi + 2)\)-unambiguous set of topologies. There is a PCA \( A \) such that, for all \( T \in \mathcal{S} \), we have \( L_T^B(A) = L_T^B(\varphi) \).

Proof

- Translate \( \varphi \) into normal form

  \[ \exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi \]

  where \( \psi \) is \( r_\varphi \)-local around \( y \).

- Existential quantification \( \Rightarrow \) projection & guessing of truth values for propositions involving only free variables of \( \forall y \psi \) [Gastin-Kuske 2010].

- Construct fixed-topology CA \( A_\theta \) for formula \( \forall y \psi \) over all topology neighborhoods \( \theta \) of radius \( \lceil r_\varphi / 2 \rceil \) [Genest-Kuske-Muscholl 2004].
Unambiguous topology classes

Theorem

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$, $B \geq 1$, and $\mathcal{T}$ be a $(r_{\varphi} + 2)$-unambiguous set of topologies. There is a PCA $A$ such that, for all $T \in \mathcal{T}$, we have $L_T^B(A) = L_T^B(\varphi)$.

Proof

- Translate $\varphi$ into normal form

$$\exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi$$

where $\psi$ is $r_{\varphi}$-local around $y$.

- Existential quantification $\Rightarrow$ projection & guessing of truth values for propositions involving only free variables of $\forall y \psi$ [Gastin-Kuske 2010].

- Construct fixed-topology CA $A_\theta$ for formula $\forall y \psi$ over all topology neighborhoods $\theta$ of radius $\lceil r_{\varphi}/2 \rceil$ [Genest-Kuske-Muscholl 2004].

- Glue fixed-topology CA together to obtain a PCA for $\forall y \psi$. 
Unambiguous topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$
Unambiguous topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lfloor r_\varphi/2 \rfloor = 2$
Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$. 

Unambiguous topology classes

Since $|w| = 4 \leq r_\varphi + 2$, and $T$ is $(r_\varphi + 2)$-unambiguous, $w$ forms a cycle in the topology as well.

Every process has to simulate several automata.
Unambiguous topology classes

Proof (cntd.) suppose \( r_\varphi = 3 \) so that \( \lceil r_\varphi/2 \rceil = 2 \)
Proof (cntd.) suppose \( r_\varphi = 3 \) so that \( \lceil r_\varphi / 2 \rceil = 2 \)

- Process \( p = (2, 3) \) guesses topology neighborhood \( \theta \) and simulates local automaton \( A_\theta[2', 3'] \).
Unambiguous topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$

- Process $p = (2, 3)$ guesses topology neighborhood $\theta$ and simulates local automaton $A_\theta[2', 3']$.
- Process $p$ sends $(\theta, (2', 3'))$ to $(2, 2)$. 

Positive Results
Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$

- Process $p = (2, 3)$ guesses topology neighborhood $\theta$ and simulates local automaton $A_\theta[2', 3']$.
- Process $p$ sends $(\theta, (2', 3'))$ to $(2, 2)$.
- Process $(2, 2)$ receives $(\theta, (1', 2'))$ from $(1, 2)$ and simulates $A_\theta[2', 2']$. 

Topology admits $w = (b, a)(d, c)(a, b)(c, d)$-path from $p$. Since $|w| = 4 \leq r_\varphi + 2$, and $T$ is $(r_\varphi + 2)$-unambiguous, $w$ forms a cycle in the topology as well.
Unambiguos topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$

- Process $p = (2, 3)$ guesses topology neighborhood $\theta$ and simulates local automaton $A_\theta[2', 3']$.
- Process $p$ sends $(\theta, (2', 3'))$ to $(2, 2)$.
- Process $(2, 2)$ receives $(\theta, (1', 2'))$ from $(1, 2)$ and simulates $A_\theta[2', 2']$.
- $\Rightarrow$ Topology admits $w = (b, a)(d, c)(a, b)(c, d)$-path from $p$. 

\[ \text{Positive Results} \]
Unambiguous topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$

- Process $p = (2, 3)$ guesses topology neighborhood $\theta$ and simulates local automaton $A_\theta[2', 3']$.
- Process $p$ sends $(\theta, (2', 3'))$ to $(2, 2)$.
- Process $(2, 2)$ receives $(\theta, (1', 2'))$ from $(1, 2)$ and simulates $A_\theta[2', 2']$.
- $\Rightarrow$ Topology admits $w = (b, a)(d, c)(a, b)(c, d)$-path from $p$.
- Since $|w| = 4 \leq r_\varphi + 2$, and $\mathcal{T}$ is $(r_\varphi + 2)$-unambiguous, $w$ forms a cycle in the topology as well.
Unambiguous topology classes

Proof (cntd.) suppose $r_\varphi = 3$ so that $\lceil r_\varphi / 2 \rceil = 2$

- Process $p = (2, 3)$ guesses topology neighborhood $\theta$ and simulates local automaton $A_\theta[2', 3']$.
- Process $p$ sends $(\theta, (2', 3'))$ to $(2, 2)$.
- Process $(2, 2)$ receives $(\theta, (1', 2'))$ from $(1, 2)$ and simulates $A_\theta[2', 2']$.
- $\Rightarrow$ Topology admits $w = (b, a)(d, c)(a, b)(c, d)$-path from $p$.
- Since $|w| = 4 \leq r_\varphi + 2$, and $\mathcal{T}$ is $(r_\varphi + 2)$-unambiguous, $w$ forms a cycle in the topology as well.
- Every process has to simulate several automata.

Positive Results
Unambiguous topology classes

Almost the same proof works for a weaker logic without channel bound:

**Theorem**

Let \( \varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}, \sim] \), and \( \mathcal{T} \) be a \((r_\varphi + 2)\)-unambiguous set of topologies. There is a PCA \( A \) such that, for all \( T \in \mathcal{T} \), \( L_T(A) = L_T(\varphi) \).
Unambiguous topology classes

Almost the same proof works for a weaker logic without channel bound:

**Theorem**

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}, \sim]$, and $\mathcal{I}$ be a $(r_{\varphi} + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{I}$, $L_\mathcal{T}(\mathcal{A}) = L_\mathcal{T}(\varphi)$.

**Proof**

- Translate $\varphi$ into normal form

  $$\exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi$$

  where $\psi$ is $r_{\varphi}$-local around $y$.

- Existential quantification $\Rightarrow$ projection & guessing of truth values for propositions involving only free variables of $\forall y \psi$ [Gastin-Kuske 2010].

- Construct **fixed-topology CA** $\mathcal{A}_\theta$ for formula $\forall y \psi$ over all topology neighborhoods $\theta$ of radius $\lceil r_{\varphi}/2 \rceil$ [Genest-Kuske-Muscholl 2004].

- Glue fixed-topology CA together to obtain a PCA for $\forall y \psi$. 

Positive Results
Unambiguous topology classes

Almost the same proof works for a weaker logic without channel bound:

**Theorem**

Let $\varphi \in \text{EMSO}[\ll_{\text{proc}}, \ll_{\text{msg}}, \sim]$, and $\mathcal{T}$ be a $(r_\varphi + 2)$-unambiguous set of topologies. There is a PCA $\mathcal{A}$ such that, for all $\mathcal{T} \in \mathcal{T}$, $L_\mathcal{T}(\mathcal{A}) = L_\mathcal{T}(\varphi)$.

**Proof**

- Translate $\varphi$ into normal form
  \[
  \exists X_1 \ldots \exists X_m \exists x_1 \ldots \exists x_n \forall y \psi
  \]
  where $\psi$ is $r_\varphi$-local around $y$.
- Existential quantification $\Rightarrow$ projection & guessing of truth values for propositions involving only free variables of $\forall y \psi$ [Gastin-Kuske 2010].
- Construct fixed-topology CA $\mathcal{A}_\theta$ for formula $\forall y \psi$ over all topology neighborhoods $\theta$ of radius $\lceil r_\varphi / 2 \rceil$ [B.-Leucker 2004].
- Glue fixed-topology CA together to obtain a PCA for $\forall y \psi$. 

Positive Results
An orthogonal approach

**Theorem**

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$. There is a PCA $\mathcal{A}$ that is equivalent to $\varphi$ on all pipelines, trees, and grids.
An orthogonal approach

Theorem

Let $\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$. There is a PCA $A$ that is equivalent to $\varphi$ on all pipelines, trees, and grids.

Proof

$\varphi \in \text{EMSO}[\prec_{\text{proc}}, \prec_{\text{msg}}]$.

$A$ is equivalent to $\varphi$ on all pipelines, trees, and grids.

$\varphi = \prec_{\text{proc}}, \prec_{\text{msg}}$

Exploit sphere automaton from [B.-Leucker] to compute $\{\prec_{\text{proc}}, \prec_{\text{msg}}\}$-neighborhoods.

Positive Results
An orthogonal approach

Theorem

Let $\varphi \in \text{EMSO}[\sqsubseteq_{\text{proc}}, \sqsubseteq_{\text{msg}}]$. There is a PCA $\mathcal{A}$ that is equivalent to $\varphi$ on all pipelines, trees, and grids.

Proof

Exploit sphere automaton from [B.-Leucker] to compute $\{\sqsubseteq_{\text{proc}}, \sqsubseteq_{\text{msg}}\}$-neighborhoods.
Summary of results

Negative results

- There is an $\text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}]$-formula that is not realizable for the class of ring forests.
- There is an FO-formula that is not realizable for the class of trees.

Positive results

Under a channel bound, every $\text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}]$-formula is realizable for the classes of pipelines, trees, grids, and rings.

Every $\text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}, \sim]$-formula is realizable for the classes of pipelines, trees, grids, and rings.

Open problems

- Is every $\text{FO}[\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}]$-formula realizable without channel bound?
- Is every $\text{FO}[\triangleleft_{\text{msg}}]$-formula realizable (for interesting classes of topologies)?
Summary of results

Negative results

- There is an FO[$\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}$]-formula that is not realizable for the class of ring forests.
- There is an FO-formula that is not realizable for the class of trees.

Positive results

- Under a channel bound, every FO[$\triangleleft^*_{\text{proc}}, \triangleleft_{\text{msg}}$]-formula is realizable for the classes of pipelines, trees, grids, and rings.
- Every FO[$\triangleleft_{\text{proc}}, \triangleleft_{\text{msg}}, \sim$]-formula is realizable for the classes of pipelines, trees, grids, and rings.
Summary of results

Negative results

- There is an FO[≺_{proc}, ≺_{msg}]-formula that is not realizable for the class of ring forests.
- There is an FO-formula that is not realizable for the class of trees.

Positive results

- Under a channel bound, every FO[≺^{*}_{proc}, ≺_{msg}]-formula is realizable for the classes of pipelines, trees, grids, and rings.
- Every FO[≺_{proc}, ≺_{msg}, ~]-formula is realizable for the classes of pipelines, trees, grids, and rings.

Open problems

- Is every FO[≺^{*}_{proc}, ≺_{msg}]-formula realizable without channel bound?
- Is every FO[≺^{*}]-formula realizable (for interesting classes of topologies)?
Related work

- Parameterized synthesis [Jacobs-Bloem 2012]
- Distributed algorithms [Grumbach-Wu 2010], [Chalopin-Das-Kosowski 2010]
- Automata from normal forms [Schwentick-Barthelmann 1999], [Gastin-Kuske 2010]
Conclusion

**Contribution**

- A notion of communicating automaton that is independent of a concrete topology
- Büchi-Elgot-Trakhtenbrot theorems for PCA
Conclusion

Contribution

- A notion of communicating automaton that is independent of a concrete topology
- Büchi-Elgot-Trakhtenbrot theorems for PCA

Future work

- Topologies of unbounded degree (unranked trees, star architectures)
- Parameterized verification
Thank You!