A Unifying Framework for Deciding Synchronizability

Laetitia Laversa

10/03/2022

Université Côte d'Azur

Synchronizability problem

Synchronizability problem

For $\mathcal{S}$ a system and $\mathcal{C}$ a class of MSC, do we have

$$L(\mathcal{S}) \subseteq \mathcal{C} \ ?$$

$\mathcal{C}$ can be, for a given k, the set of

- existentially-k-bounded MSCs
- universally-k-bounded MSCs
- k-synchronisable MSCs

The interest of the synchronizability problem

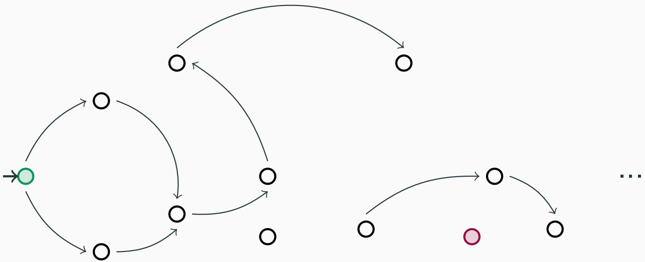
Theorem [Brand and Zafiropoulo, 1983]

Communicating automata systems are Turing-equivalent.

Verification problems are indecidable

so the reachability problem is indecidable.

Reachability problem

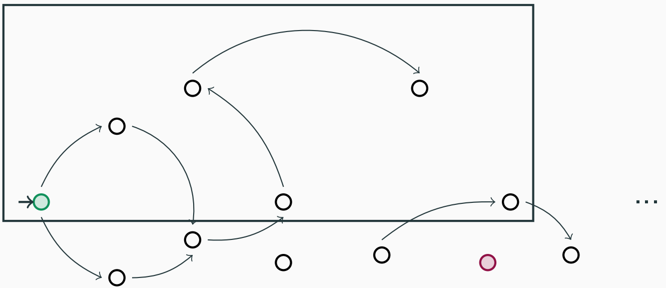


Reachability problem

Is the configuration ○ reachable from the initial configuration ○ ?

Reachability problem

Under-approximation



Under-approximation

Problem applicated to a regular set ⇒

Reachability problem **decidable**

Objective

Membership problem

Does the set of behaviors of the system correspond to the under-approximation ?

Class	pp	mb
$\exists$-k-bounded	Decidable [1]	Decidable [2]
$\forall$-k-bounded	Decidable [1]	Decidable [2]
k-synchronizable	Decidable [3]	Décidable [3]

1. [Genest *et al.*, 2007]
2. [Bollig *et al.*, 2021]
3. [Di Giusto *et al.*, 2020]

Objective

Membership problem

Does the set of behaviors of the system correspond to the under-approximation ?

Class	pp	mb
$\exists$-k-bounded	Decidable [1]	**Decidable** [2]
$\forall$-k-bounded	Decidable [1]	**Decidable** [2]
k-synchronizable	Decidable [3]	Décidable [3]

1. [Genest *et al.*, 2007]
2. [Bollig *et al.*, 2021]
3. [Di Giusto *et al.*, 2020]

Decidability

In [Bollig *et al.*, 2021], the proof of decidability for mailbox $\exists$ / $\forall$-k-bounded systems uses two concepts :

- the monadic second order logic
- the special treewidth of a graph

Monadic Second Order Logic

Generalities

MSO Logic

- Extension of first-order logic : variables can represent sets.
- Restriction of second-order logic : only quantifications on unary predicates.
- Allows to express properties on graphs

A graph class $\mathcal{C}$ is **MSO-definable** if there exists an MSO-formula φ such that $L(\varphi) = \mathcal{C}$.

Grammar

Grammar :

$$\varphi := x \rightarrow y \mid x \lhd y \mid \lambda(x) = a \mid x = y \mid$$
$$x \in X \mid \exists x.\varphi \mid \exists X.\varphi \mid \varphi \wedge \varphi \mid \neg\varphi$$

When graphs are MSCs we have :

x, y events, a action, (first-order variables)

X set of events, (second-order variables)

φ formula ,

$\lhd$ relation between two matched actions,

$\rightarrow$ order over events of the same process

Example

A matched action :

$matched(x) = \exists y, x \lhd y$

Mailbox order :

$$x \sqsubset y = \bigvee_{q \in \mathbb{P}, a, b \in Send(\_, q, \_)} \lambda(x) = a \land \lambda(y) = b \land$$
$$((matched(x) \land \neg matched(y)) \lor$$
$$\exists x', \exists y'(x \lhd x' \land y \lhd y' \land x' \rightarrow^{+} y'))$$

Special treewidth

Generalities

Intuition

Specify how far a graph is from being a tree.

- Trees have a special treewidth of 1.
- A graph with a treewidth of k can be described by a k-STT (Special Tree Term).
- If a graphs set has a bounded special treewidth, the satisfiability of an MSO-formula on this set is decidable.

k-STT

Syntax

$$\tau := i \mid Add_{i,j}\ \tau \mid Add_i^a\ \tau \mid Forget_i\ \tau \mid \tau \oplus \tau$$

- color
- add edge between nodes of color i and j
- add label to nodes of color i (message or process)
- forget color i
- union of two term if there is no common color

If an STT uses $k+1$ colors, it is a **k-STT**.

Example of k-STT

Create a new node :

$$Add_i^p \ Add_i^a \ i = (i, a, p)$$

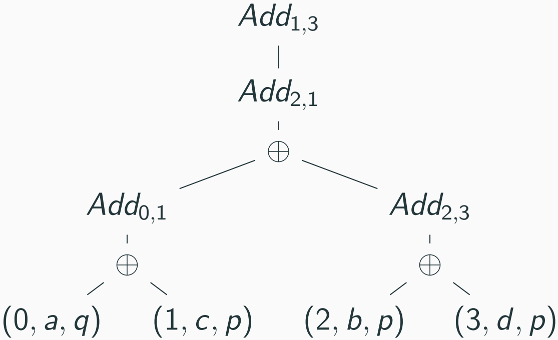
An arbitrary graph :

$$\tau = Add_{1,3} Add_{2,1}(Add_{0,1}((0, a, q) \oplus (1, c, p)) \oplus Add_{2,3}((2, b, p) \oplus (3, d, p)))$$

Example of k-STT

An arbitrary graph :

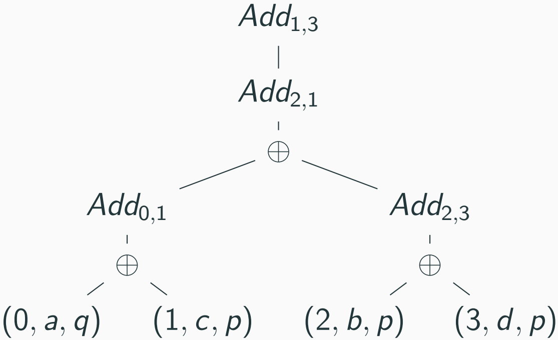
$$\tau = Add_{1,3}Add_{2,1}(Add_{0,1}((0, a, q) \oplus (1, c, p)) \oplus Add_{2,3}((2, b, p) \oplus (3, d, p)))$$

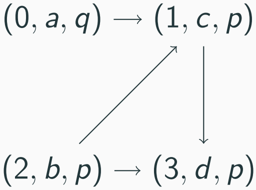


Example of k-STT

An arbitrary graph :

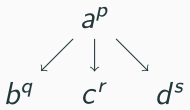
$$\tau = Add_{1,3}Add_{2,1}(Add_{0,1}((0,a,q) \oplus (1,c,p)) \oplus Add_{2,3}((2,b,p) \oplus (3,d,p)))$$

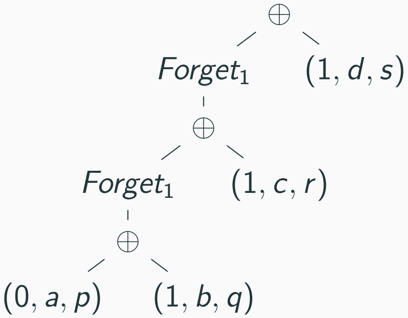




Example

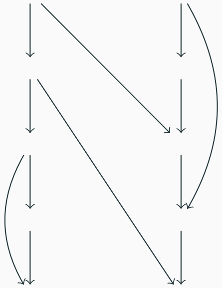
For a tree :





Decomposition game for special treewidth

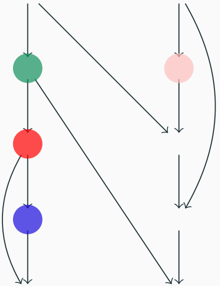
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.

Decomposition game for special treewidth

For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.

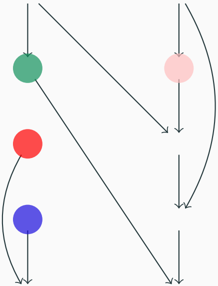


1) Eve has $k + 1$ colors and put it on nodes.

2) If the source and the end of an edge are colored, the edge is removed.

Decomposition game for special treewidth

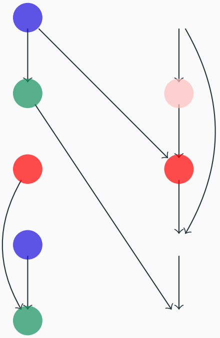
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.
2) If the source and the end of an edge are colored, the edge is removed.
3) Adam choose the next graph.
4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

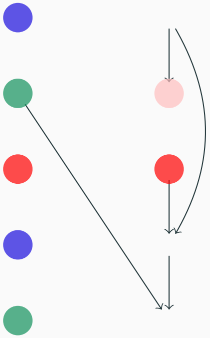
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.
2) If the source and the end of an edge are colored, the edge is removed.
3) Adam choose the next graph.
4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

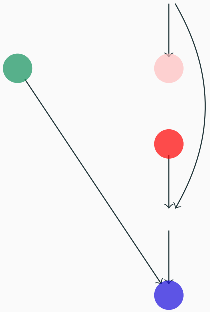
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.
2) If the source and the end of an edge are colored, the edge is removed.
3) Adam choose the next graph.
4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

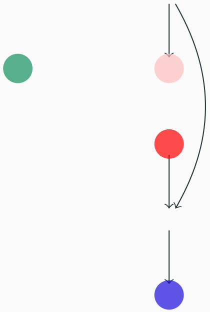
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.
2) If the source and the end of an edge are colored, the edge is removed.
3) Adam choose the next graph.
4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k+1$ colors and put it on nodes.

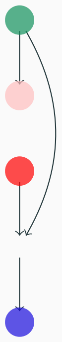
2) If the source and the end of an edge are colored, the edge is removed.

3) Adam choose the next graph.

4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.

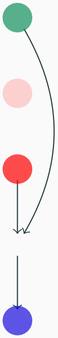
2) If the source and the end of an edge are colored, the edge is removed.

3) Adam choose the next graph.

4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

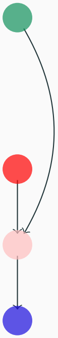
For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.
2) If the source and the end of an edge are colored, the edge is removed.
3) Adam choose the next graph.
4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.



1) Eve has $k + 1$ colors and put it on nodes.

2) If the source and the end of an edge are colored, the edge is removed.

3) Adam choose the next graph.

4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decomposition game for special treewidth

For a given graph and a given k, we check that the graph has a special treewidth of k. Here, $k = 3$.

1) Eve has $k + 1$ colors and put it on nodes.

2) If the source and the end of an edge are colored, the edge is removed.

3) Adam choose the next graph.

4) Eve can put the absent colors on nodes, etc. Until there is no more possible moves.

Decidability proof of synchronizability for existentially bounded systems

Synchronizability problem

Synchronizability problem

For $\mathcal{S}$ a system and $\mathcal{C}$ a class of MSC, do we have

$$L(\mathcal{S}) \subseteq \mathcal{C} \text{ ?}$$

This problem is decidable if :

1. $\mathcal{C}$ is MSO-definable
2. $\mathcal{C}$ is has a k bounded treewidth

Example with $\exists$-B-mailbox bounded MSCs

We choose $\mathcal{C}$ the class of $\exists$-B-mailbox bounded MSCs.

1. Is $\mathcal{C}$ MSO-definable ?
    - We are looking for a formula verifying that there exists a linearization $\exists$-B bounded and runnable with a mailbox communication.
2. Does $\mathcal{C}$ have a k bounded treewidth ?
    - Can we find a bound such that all $\exists$-B-mailbox bounded MSCs have a special treewidth of at most k.

1. Is $\mathcal{C}$ MSO-definable ?

A MSC is $\exists$-B-mailbox bounded if $(\rightarrow \cup \lhd \cup \sqsubset \cup \xrightarrow{rev}_k)^*$ is acyclic.

- we have seen that $\sqsubset$ is MSO-definable
- $\xrightarrow{rev}_k$ makes the link between a reception and the sending of the k-th reception which follows it.
- $r \xrightarrow{rev}_k s = \exists r_1, r_2, \cdots, r_n, r \rightarrow r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_n \wedge s \lhd r_n$

So $\mathcal{C}$ is MSO-definable.

2. Does $\mathcal{C}$ have a k bounded treewidth ?

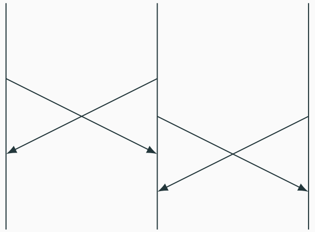
We can prove that for all $\exists$-B-mailbox bounded MSCs in a system, we can bound the special treewidth by

$$k = B \mid \mathbb{P} \mid^2 + \mid \mathbb{P} \mid$$

Eve's strategy for $\mathcal{C}$

A $\exists$-2-mailbox bounded MSC

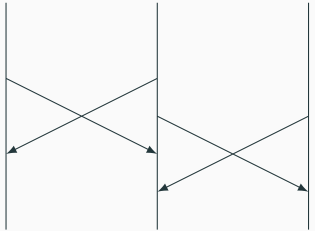
Let's show that it has a special treewidth of 3
$(3 < 2 \times 3^2 + 3 = 21)$.

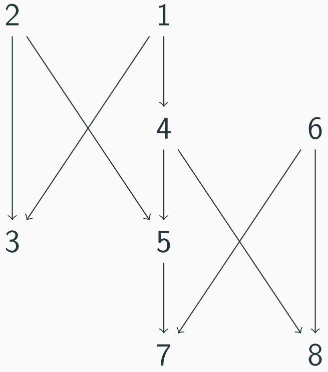


Eve's strategy for $\mathcal{C}$

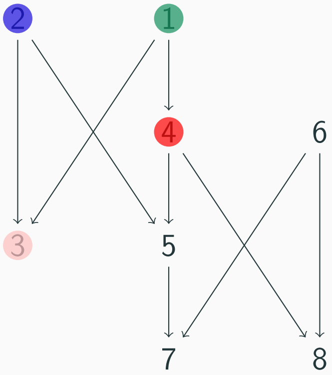
A ∃-2-mailbox bounded MSC

Let's show that it has a special treewidth of 3
$(3 < 2 \times 3^2 + 3 = 21)$.

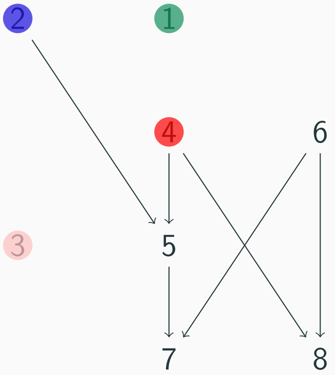




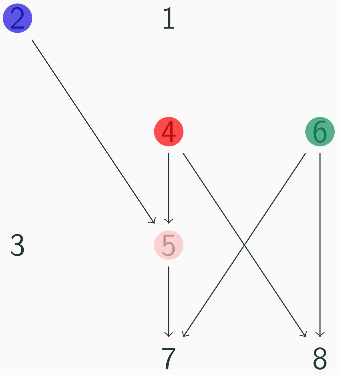
Eve's strategy for $\mathcal{C}$



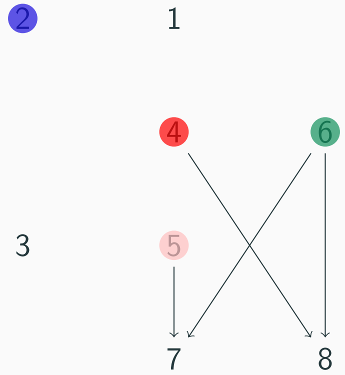
Eve's strategy for $\mathcal{C}$



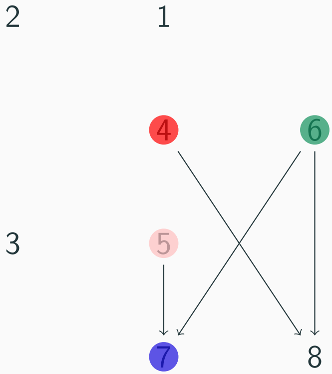
Eve's strategy for $\mathcal{C}$



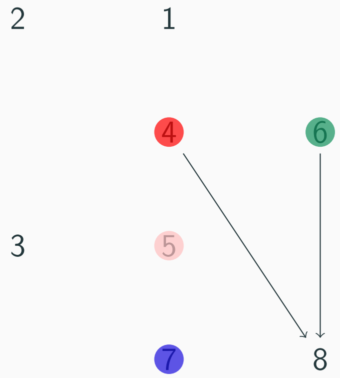
Eve's strategy for $\mathcal{C}$



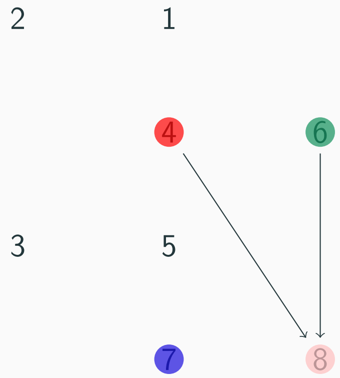
Eve's strategy for $\mathcal{C}$



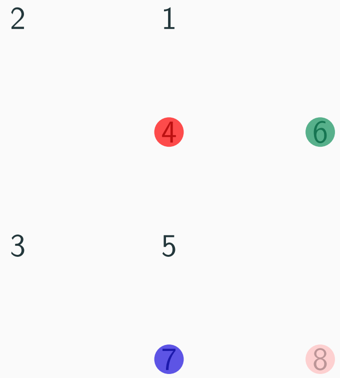
Eve's strategy for $\mathcal{C}$



Eve's strategy for $\mathcal{C}$



Eve's strategy for $\mathcal{C}$



Example with $\exists$-B-mailbox bounded MSCs

We chose $\mathcal{C}$ the class of $\exists$-B-mailbox bounded MSCs.

1. Is $\mathcal{C}$ MSO-definable ? $\rightarrow$ Yes
2. Does $\mathcal{C}$ have a k bounded treewidth ? $\rightarrow$ Yes

$\Rightarrow$ The synchronizability problem is decidable for this class.

Conclusion

- Decidability proof for synchronizability problem.
- This method works with different classes, as long as they are MSO-definable and with a bounded special treewidth.
- It has a drawback : if the k is not passed as a parameter, it does not help to find an adequate k
  - "Is there a k such that our system is $\exists$-k-mailbox-bounded ?" is an open problem.