Separation and Concatenations (2)

Marc Zeitoun
Joint work with Thomas Place

LaBRI, Bordeaux University

July 16, 2017

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow Pol(\mathcal{C})$-membership decidable.

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Negation is hard, one negation can be circumvented.

Summary of Part 1

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Negation is hard, one negation can be circumvented.

This talk: We focus on $BPol(\mathcal{C})$-**separation**.

Separation for $BPol(\mathcal{C})$ when $\mathcal{C}$ is finite

$BPol(\mathcal{C})$-separation: Three main steps

The algorithm is based on three main steps.

$BPol(\mathcal{C})$-separation: Three main steps

The algorithm is based on three main steps.

- We explain their purpose at a high level.

First step - preliminary remark

Our two closure operations have different properties:

- $Pol(\mathcal{C})$ closed under $\cup$, $\cap$ and **marked concatenation**.

First step - preliminary remark

Our two closure operations have different properties:

- $Pol(\mathcal{C})$ closed under $\cup$, $\cap$ and **marked concatenation**.
- $Bool(\mathcal{C})$ closed under all Boolean operations but

  not **marked concatenation**.

.

First step - preliminary remark

Our two closure operations have different properties:

- $Pol(\mathcal{C})$ closed under $\cup$, $\cap$ and **marked concatenation**.
- $Bool(\mathcal{C})$ closed under all Boolean operations but

  not marked concatenation.

.

Our **techniques rely heavily on concatenation**:

$\Rightarrow$ we like $Pol(\mathcal{C})$ and hate $BPol(\mathcal{C})$.

First step - preliminary remark

Our two closure operations have different properties:

- $Pol(\mathcal{C})$ closed under $\cup$, $\cap$ and **marked concatenation**.
- $Bool(\mathcal{C})$ closed under all Boolean operations but

  not **marked concatenation**.

.

Our **techniques rely heavily on concatenation**:

$\Rightarrow$ we like $Pol(\mathcal{C})$ and hate $BPol(\mathcal{C})$.

Consequence

Even if our goal is $BPol(\mathcal{C})$-separation, we prefer working with $Pol(\mathcal{C})$.

First step - preliminary remark

Meta argument for investigating separation

Learn more on $\mathcal{C}$ to investigate classes built on top of $\mathcal{C}$.

"$\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable."

First step - preliminary remark

Meta argument for investigating separation

Learn more on $\mathcal{C}$ to investigate classes built on top of $\mathcal{C}$.

"$\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable."

Recycle this idea to get rid Boolean closure.

New Goal

Reduce $BPol(\mathcal{C})$-separation to a problem for $Pol(\mathcal{C})$.

$BPol(\mathcal{C})$-separation: Three main steps (1)

Step 1

Step 2

Step 3

Getting rid of Boolean closure

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$$(L_1, L_2, \dots, L_n) \cap K \quad \stackrel{\text{def}}{=} \quad (L_1 \cap K, L_2 \cap K, \dots, L_n \cap K)$$

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$$(L_1, L_2, \ldots, L_n) \cap K \quad \stackrel{\text{def}}{=} \quad (L_1 \cap K, L_2 \cap K, \ldots, L_n \cap K)$$

Tuple $\mathcal{D}$-separability: inductive definition

$(L_1, \ldots, L_n)$ is **$\mathcal{D}$-separable** iff:

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$$(L_1, L_2, \ldots, L_n) \cap K \quad \stackrel{\text{def}}{=} \quad (L_1 \cap K, L_2 \cap K, \ldots, L_n \cap K)$$

Tuple $\mathcal{D}$-separability: inductive definition

$(L_1, \ldots, L_n)$ is **$\mathcal{D}$-separable** iff:

- $n = 1$ and $L_1 = \emptyset$ or,

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$$(L_1, L_2, \ldots, L_n) \cap K \quad \stackrel{\text{def}}{=} \quad (L_1 \cap K, L_2 \cap K, \ldots, L_n \cap K)$$

Tuple $\mathcal{D}$-separability: inductive definition

$(L_1, \ldots, L_n)$ is **$\mathcal{D}$-separable** iff:

- $n = 1$ and $L_1 = \emptyset$ or,
- $n \geq 2$ and there exists $K \in \mathcal{D}$ such that

$$L_1 \subseteq K \quad \text{and} \quad (L_2, \ldots, L_n) \cap K \text{ is } \mathcal{D}\text{-separable}$$

$Bool(\mathcal{D})$-separation $\Rightarrow$ tuple separation for $\mathcal{D}$

- We generalize the notion of separability to **tuples of languages**.

$$(L_1, L_2, \dots, L_n) \cap K \quad \stackrel{\text{def}}{=} \quad (L_1 \cap K, L_2 \cap K, \dots, L_n \cap K)$$

Tuple $\mathcal{D}$-separability: inductive definition

$(L_1, \dots, L_n)$ is **$\mathcal{D}$-separable** iff:

- $n = 1$ and $L_1 = \emptyset$ or,
- $n \geq 2$ and there exists $K \in \mathcal{D}$ such that

$$L_1 \subseteq K \quad \text{and} \quad (L_2, \dots, L_n) \cap K \text{ is } \mathcal{D}\text{-separable}$$

Remarks

- When $n = 2$, we recover the classical notion.
- The longer, the easier to separate.

Boolean closure theorem

$$(L_1, L_2)^k \quad \stackrel{\text{def}}{=} \quad \underbrace{(L_1, L_2, \ldots, L_1, L_2)}_{\text{length } 2k}$$

Boolean closure theorem

$$(L_1, L_2)^k \quad \stackrel{\text{def}}{=} \quad \underbrace{(L_1, L_2, \ldots, L_1, L_2)}_{\text{length } 2k}$$

$Bool(\mathcal{D})$-separation theorem

Given a lattice $\mathcal{D}$ and two languages L_1, L_2, **TFAE**:

1. L_1 is $Bool(\mathcal{D})$-separable from L_2.
2. There exists $k \geq 1$ s.t. $(L_1, L_2)^k$ is $\mathcal{D}$-separable.

Boolean closure theorem

$$(L_1, L_2)^k \quad \stackrel{\text{def}}{=} \quad \underbrace{(L_1, L_2, \ldots, L_1, L_2)}_{\text{length } 2k}$$

$Bool(\mathcal{D})$-separation theorem

Given a lattice $\mathcal{D}$ and two languages L_1, L_2, **TFAE**:

1. L_1 is $Bool(\mathcal{D})$-separable from L_2.
2. There exists $k \geq 1$ s.t. $(L_1, L_2)^k$ is $\mathcal{D}$-separable.

Mission accomplished!

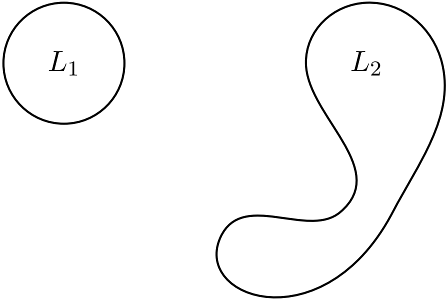
$\mathcal{D} = Pol(\mathcal{C})$: reduction from $BPol(\mathcal{C})$-separation to a problem for $Pol(\mathcal{C})$.

Boolean closure theorem: proof of $2 \Rightarrow 1$

Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.

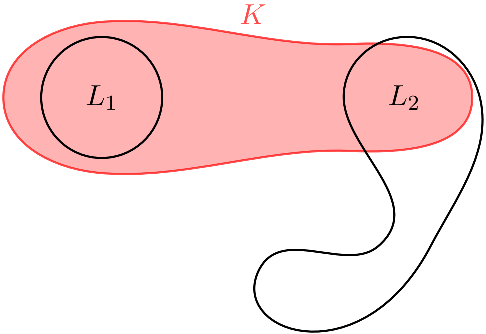
Boolean closure theorem: proof of $2 \Rightarrow 1$

Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.



Boolean closure theorem: proof of $2 \Rightarrow 1$

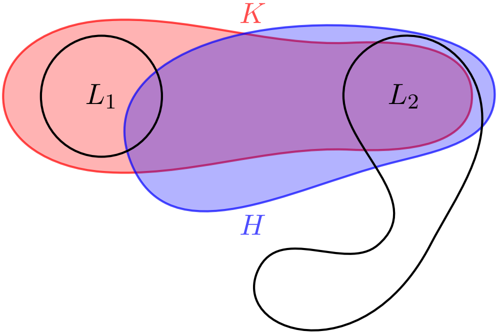
Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.



- $(L_2, (L_1, L_2)^{k-1}) \cap K$ is $\mathcal{D}$-separable

Boolean closure theorem: proof of $2 \Rightarrow 1$

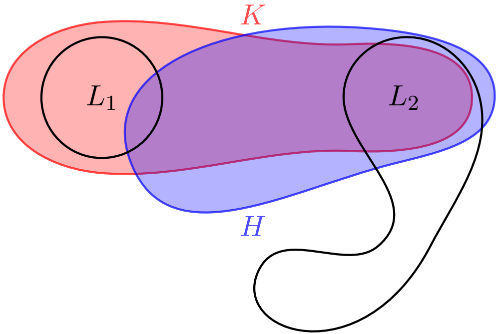
Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.



- $(L_2, (L_1, L_2)^{k-1}) \cap K$ is $\mathcal{D}$-separable
- $(L_1, L_2)^{k-1} \cap K \cap H$ is $\mathcal{D}$-separable

Boolean closure theorem: proof of $2 \Rightarrow 1$

Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.

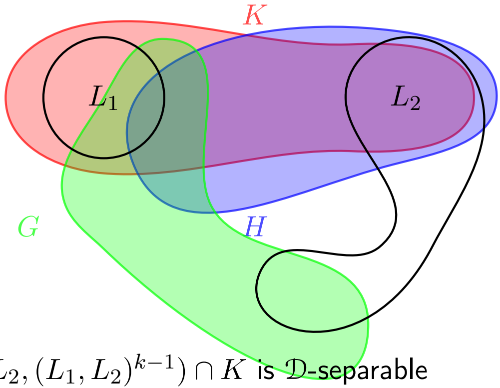


- $(L_2, (L_1, L_2)^{k-1}) \cap K$ is $\mathcal{D}$-separable
- $(L_1, L_2)^{k-1} \cap K \cap H$ is $\mathcal{D}$-separable

Induction $\Rightarrow$ $G \in Bool(\mathcal{D})$ separates $L_1 \cap K \cap H$ from $L_2 \cap K \cap H$

Boolean closure theorem: proof of $2 \Rightarrow 1$

Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow L_1$ $Bool(\mathcal{D})$-separable from L_2.

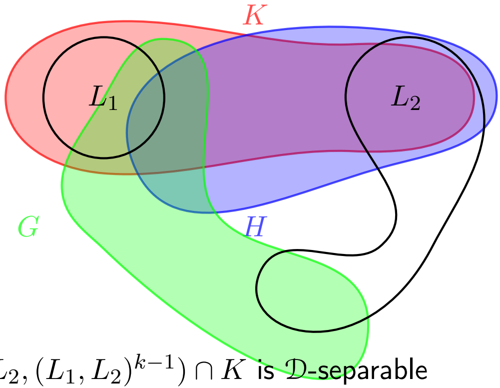


- $(L_2, (L_1, L_2)^{k-1}) \cap K$ is $\mathcal{D}$-separable
- $(L_1, L_2)^{k-1} \cap K \cap H$ is $\mathcal{D}$-separable

Induction $\Rightarrow G \in Bool(\mathcal{D})$ separates $L_1 \cap K \cap H$ from $L_2 \cap K \cap H$

Boolean closure theorem: proof of $2 \Rightarrow 1$

Induction on k: $(L_1, L_2)^k$ $\mathcal{D}$-separable $\Rightarrow$ L_1 $Bool(\mathcal{D})$-separable from L_2.



- $(L_2, (L_1, L_2)^{k-1}) \cap K$ is $\mathcal{D}$-separable
- $(L_1, L_2)^{k-1} \cap K \cap H$ is $\mathcal{D}$-separable

Induction $\Rightarrow G \in Bool(\mathcal{D})$ separates $L_1 \cap K \cap H$ from $L_2 \cap K \cap H$

$(G \cap K) \cup (K \setminus H) \in Bool(\mathcal{D})$ separates L_1 from L_2

$BPol(\mathcal{C})$-separation: Three main steps (2)

Getting rid of Boolean closure

Step 1

Step 2

Step 3

$BPol(\mathcal{C})$-separation: Three main steps (2)

Step 1

Getting rid of Boolean closure

(L_1, L_2) $BPol(\mathcal{C})$-separable
iff
$\exists k$ $(L_1, L_2)^k$ $Pol(\mathcal{C})$-separable

Step 2

Step 3

$BPol(\mathcal{C})$-separation: Three main steps (2)

Step 1

Getting rid of Boolean closure

(L_1, L_2) $BPol(\mathcal{C})$-separable
iff
$\exists k$ $(L_1, L_2)^k$ $Pol(\mathcal{C})$-separable

Step 2

Solving tuple $Pol(\mathcal{C})$-separation:
Input: $(L_1, \dots, L_n)$
Output: Is $(L_1, \dots, L_n)$ $Pol(\mathcal{C})$-separable?

Step 3

Tuple $Pol(\mathcal{C})$-separation: Approach (1)

Our input $(L_1, \ldots, L_n)$ is a **tuple of n regular languages**.

We do **not work directly with these languages**.

Tuple $Pol(\mathcal{C})$-separation: Approach (1)

Our input $(L_1, \dots, L_n)$ is a **tuple of n regular languages**.

We do **not work directly with these languages**.

Rule #1 for separation-like problems:

One always looks at several inputs simultaneously.

Tuple $Pol(\mathcal{C})$-separation: Approach (1)

Our input $(L_1, \ldots, L_n)$ is a **tuple of n regular languages**.

We do **not work directly with these languages**.

Rule $\#$1 for separation-like problems:

One always looks at several inputs simultaneously.

We use a **set of inputs** which has a special structure.

Tuple $Pol(\mathcal{C})$-separation: Approach (2)

Input $(L_1, \ldots, L_n)$ is a **tuple of n regular languages**.

$\Rightarrow$ We have **NFAs** for these languages:

Tuple $Pol(\mathcal{C})$-separation: Approach (2)

Input $(L_1, \dots, L_n)$ is a **tuple of n regular languages**.

$\Rightarrow$ We have **NFAs** for these languages:

We work with a **set of languages $\mathbb{L}$** containing all languages

$$L_{p,q} = \{w \mid p \xrightarrow{w} q\} \quad (p, q \text{ two states of these } \textbf{NFAs})$$

Tuple $Pol(\mathcal{C})$-separation: Approach (2)

Input $(L_1, \ldots, L_n)$ is a **tuple of n regular languages**.

$\Rightarrow$ We have **NFAs** for these languages:

We work with a **set of languages $\mathbf{L}$** containing all languages

$$L_{p,q} = \{w \mid p \xrightarrow{w} q\} \quad (p, q \text{ two states of these } \textbf{NFAs})$$

Given $n \geq 1$, we compute the set $\mathcal{I}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

Answer for $(L_1, \ldots, L_n)$ can then be extracted from this information.

Tuple $Pol(\mathcal{C})$-separation: Approach (3)

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

The algorithm uses two hypotheses on $\mathbf{L}$:

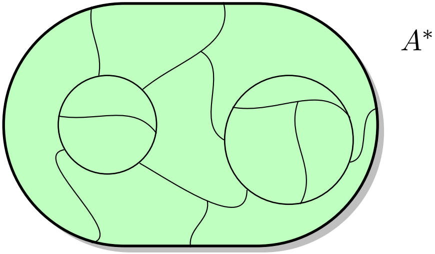
- It has an algebraic structure (given by the **NFAs**).

Tuple $Pol(\mathcal{C})$-separation: Approach (3)

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

The algorithm uses two hypotheses on $\mathbf{L}$:

- It has an algebraic structure (given by the **NFAs**).
- It is **$\mathcal{C}$-compatible**:
  $\mathcal{C}$ is finite $\Rightarrow$ exists **finest partition** of A^* into languages of $\mathcal{C}$:

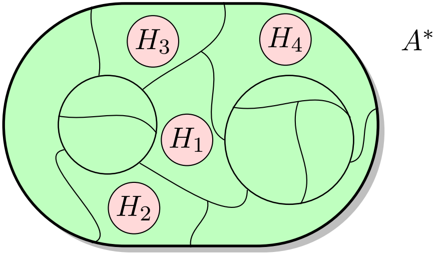


Tuple $Pol(\mathcal{C})$-separation: Approach (3)

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

The algorithm uses two hypotheses on $\mathbf{L}$:

- It has an algebraic structure (given by the **NFAs**).
- It is **$\mathcal{C}$-compatible**:
  $\mathcal{C}$ is finite $\Rightarrow$ exists **finest partition** of A^* into languages of $\mathcal{C}$:



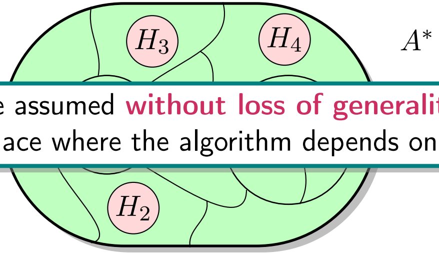
Any $H \in \mathbf{L}$ must be **included in a class of this partition**.

Tuple $Pol(\mathcal{C})$-separation: Approach (3)

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

The algorithm uses two hypotheses on $\mathbf{L}$:

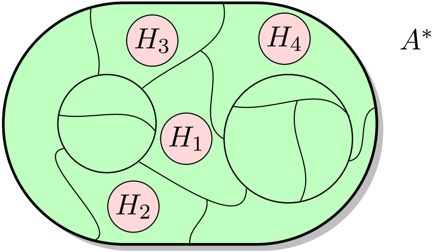
- It has an algebraic structure (given by the **NFAs**).
- It is **$\mathcal{C}$-compatible**:
  $\mathcal{C}$ is finite $\Rightarrow$ exists **finest partition** of A^* into languages of $\mathcal{C}$:



Can be assumed **without loss of generality**
Only place where the algorithm depends on $\mathcal{C}$

Any $H \in \mathbf{L}$ must be **included in a class of this partition**.

Tuple $Pol(\mathcal{C})$-separation: $\mathcal{C}$-compatibility



Any $H \in \mathbf{L}$ must be **included in a class of this partition**.

We can refine the initial input set to fulfill this condition.

Answer for the original input can be recovered from the refinement.

Least fixpoint computation of $\mathcal{T}^n[\mathbf{L}]$

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not** $Pol(\mathcal{C})$**-separable**.

$\mathcal{T}^n[\mathbf{L}]$ is computed by **induction on** n:

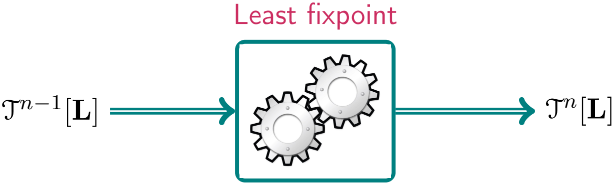
- $\mathcal{T}^1[\mathbf{L}] \subseteq \mathbf{L}$ is the set of **nonempty** languages in $\mathbf{L}$.

Least fixpoint computation of $\mathcal{T}^n[\mathbf{L}]$

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not $Pol(\mathcal{C})$-separable**.

$\mathcal{T}^n[\mathbf{L}]$ is computed by **induction on n**:

- $\mathcal{T}^1[\mathbf{L}] \subseteq \mathbf{L}$ is the set of **nonempty** languages in $\mathbf{L}$.
- For $n \geq 2$, one first computes $\mathcal{T}^{n-1}[\mathbf{L}]$, and



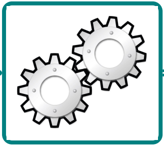
Least fixpoint computation of $\mathcal{T}^n[\mathbf{L}]$

Given $n \geq 1$, we compute the set $\mathcal{T}^n[\mathbf{L}] \subseteq \mathbf{L}^n$ of all tuples $\overline{L} \in \mathbf{L}^n$ which are **not** $Pol(\mathcal{C})$**-separable**.

$\mathcal{T}^n[\mathbf{L}]$ is computed by **induction on** n:

- $\mathcal{T}^1[\mathbf{L}] \subseteq \mathbf{L}$ is the set of **nonempty** languages in $\mathbf{L}$.
- For $n \geq 2$, one first computes $\mathcal{T}^{n-1}[\mathbf{L}]$, and

Least fixpoint

$\mathcal{T}^{n-1}[\mathbf{L}] \Longrightarrow$  $\Longrightarrow \mathcal{T}^n[\mathbf{L}]$

  - Computes $\mathcal{T}^n[\mathbf{L}]$ from a subset of trivial tuples
  - **Adds more with operations until a fixpoint is reached**.
  - Requires having $\mathcal{T}^{n-1}[\mathbf{L}]$ in hand.

$BPol(\mathcal{C})$-separation: Three main steps (3)

Step 1

Getting rid of Boolean closure

(L_1, L_2) $BPol(\mathcal{C})$-separable
iff
$\exists k$ $(L_1, L_2)^k$ $Pol(\mathcal{C})$-separable

Step 2

Solving tuple $Pol(\mathcal{C})$-separation:
Input: $(L_1, \ldots, L_n)$ (regular)
Output: Is $(L_1, \ldots, L_n)$ $Pol(\mathcal{C})$-separable?

Step 3

$BPol(\mathcal{C})$-separation: Three main steps (3)

Getting rid of Boolean closure

Step 1

(L_1, L_2) $BPol(\mathcal{C})$-separable
iff
$\exists k\ (L_1, L_2)^k$ $Pol(\mathcal{C})$-separable

Step 2

Solving tuple $Pol(\mathcal{C})$-separation:
Input: $(L_1, \dots, L_n)$ (regular)
Output: Is $(L_1, \dots, L_n)$ $Pol(\mathcal{C})$-separable?

Step 3

Reuse Step 2 as a subprocedure in our $BPol(\mathcal{C})$ algorithm

$BPol(\mathcal{C})$-separation - General approach

We continue to work with a set of languages $\mathbf{L}$ with appropriate properties.

We compute the set $\mathcal{A}[\mathbf{L}] \subseteq \mathbf{L}^2$ of pairs $(K, L) \in \mathbf{L}^2$ which are **not** $BPol(\mathcal{C})$**-separable**.

$BPol(\mathcal{C})$-separation - General approach

We continue to work with a set of languages $\mathbf{L}$ with appropriate properties.

We compute the set $\mathcal{A}[\mathbf{L}] \subseteq \mathbf{L}^2$ of pairs $(K, L) \in \mathbf{L}^2$ which are **not** $BPol(\mathcal{C})$**-separable**.

By the previous steps

Given two languages L_1, L_2, **TFAE**:

1. L_1 $BPol(\mathcal{C})$-separable from L_2.
2. There exists $k \geq 1$ s.t. $(L_1, L_2)^k$ is $Pol(\mathcal{C})$-separable.

$BPol(\mathcal{C})$-separation - General approach

We continue to work with a set of languages $\mathbf{L}$ with appropriate properties.

We compute the set $\mathcal{A}[\mathbf{L}] \subseteq \mathbf{L}^2$ of pairs $(K, L) \in \mathbf{L}^2$ which are **not** $BPol(\mathcal{C})$**-separable**.

By the previous steps

Given two languages L_1, L_2, **TFAE**:

1. L_1 is **not** $BPol(\mathcal{C})$-separable from L_2.
2. For all $k \geq 1$, $(L_1, L_2)^k$ is **not** $Pol(\mathcal{C})$-separable.

$BPol(\mathcal{C})$-separation - General approach

We continue to work with a set of languages $\mathbf{L}$ with appropriate properties.

We compute the set $\mathcal{A}[\mathbf{L}] \subseteq \mathbf{L}^2$ of pairs $(K, L) \in \mathbf{L}^2$ which are **not** $BPol(\mathcal{C})$**-separable**.

By the previous steps

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k$ **not** $Pol(\mathcal{C})$-separable, i.e. $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

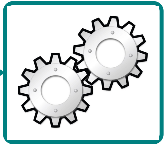
Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

Least fixpoint



$BPol(\mathcal{C})$-separation - Greatest fixpoint

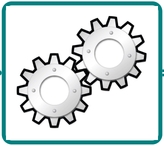
Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

Least fixpoint



$\mathbf{R} \subseteq \mathbf{L}^3$

$(\mathbf{R} = \mathcal{T}^3[\mathbf{L}])$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

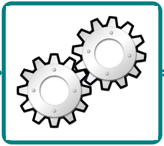
Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

Least fixpoint



$\mathbf{R} \subseteq \mathbf{L}^3$

$(\mathbf{R} = \mathcal{T}^3[\mathbf{L}])$

$$\boxed{\mathbf{T}_1 = \{(K, L) \mid (K, L, K) \in \mathbf{R}\} \subseteq \mathbf{L}^2}$$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

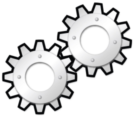
In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

$\cup|$

$\mathbf{T}_1$

Least fixpoint



$\mathbf{R} \subseteq \mathbf{L}^3$

$(\mathbf{R} = \mathcal{T}^3[\mathbf{L}])$

$\boxed{\mathbf{T}_1 = \{(K, L) \mid (K, L, K) \in \mathbf{R}\} \subseteq \mathbf{L}^2}$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

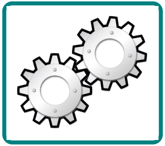
Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$$
$$\cup\!|$$
$$\mathbf{T}_1$$

Least fixpoint



$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

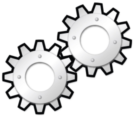
$\supseteq$

$\mathbf{T}_1$

$\supseteq$

$\mathbf{T}_2$

Least fixpoint



$\mathbf{R} \subseteq \mathbf{L}^3$

$\boxed{\mathbf{T}_2 = \{(K, L) \mid (K, L, K) \in \mathbf{R}\} \subseteq \mathbf{L}^2}$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

$\cup|$

$\mathbf{T}_1$

$\cup|$

$\mathbf{T}_2$

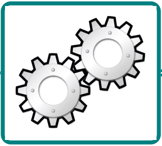
$\vdots$

$\cup|$

$\vdots$

$\mathbf{T}_n$

Least fixpoint



$\mathbf{R} \subseteq \mathbf{L}^3$

$\mathbf{T}_2 = \{(K, L) \mid (K, L, K) \in \mathbf{R}\} \subseteq \mathbf{L}^2$

Eventually, a **fixpoint is reached**, $\mathbf{T}_n = \mathbf{T}_{n+1} = \mathcal{A}[\mathbf{L}]$

$BPol(\mathcal{C})$-separation - Greatest fixpoint

Given two languages $K, L \in \mathbf{L}$, **TFAE**:

1. $(K, L) \in \mathcal{A}[\mathbf{L}]$.
2. For all $k \geq 1$, $(K, L)^k \in \mathcal{T}^{2k}[\mathbf{L}]$.

In particular, $\mathcal{A}[\mathbf{L}] \subseteq \mathcal{T}^2[\mathbf{L}]$. **Greatest fixpoint**:
From $\mathcal{T}^2[\mathbf{L}]$, remove elements with an operation until fixpoint.

Least fixpoint

$\mathbf{T}_0 = \mathcal{T}^2[\mathbf{L}]$

$\cup|$

$\mathbf{T}_1$

The $BPol(\mathcal{C})$-algorithm is a **greatest fixpoint** whose **operation is a least fixpoint**

$\mathbf{R} \subseteq \mathbf{L}^3$

$\cup|$

$\mathbf{T}_2$

$\mathbf{T}_2 = \{(K, L) \mid (K, L, K) \in \mathbf{R}\} \subseteq \mathbf{L}^2$

$\vdots$

$\cup|$

$\vdots$

$\mathbf{T}_n$

Eventually, a **fixpoint is reached**, $\mathbf{T}_n = \mathbf{T}_{n+1} = \mathcal{A}[\mathbf{L}]$

Conclusion

Conclusion (1)

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Conclusion (1)

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Bonus corresponding to enrichment with successor

For any $\mathcal{C}$, $\mathcal{C}$-separation decidable $\Rightarrow$ $\mathcal{C}^+$-separation decidable.

Conclusion (1)

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Bonus corresponding to enrichment with successor

For any $\mathcal{C}$, $\mathcal{C}$-separation decidable $\Rightarrow$ $\mathcal{C}^+$-separation decidable.

Some words about **complexity**:

1. Complexity depends on $|\mathcal{C}|$ (tied to the implicit alphabet).
2. $Pol(\text{AT})$ and $BPol(\text{AT})$ are **PSpace**(-complete).

Conclusion (1)

Everything we know is captured by only **four generic results**:

1. $\mathcal{C}$ finite $\Rightarrow$ $Pol(\mathcal{C})$-separation decidable.
2. $\mathcal{C}$ finite $\Rightarrow$ $BPol(\mathcal{C})$-separation decidable.
3. $\mathcal{C}$ finite $\Rightarrow$ $Pol(BPol(\mathcal{C}))$-separation decidable.
4. For any $\mathcal{C}$,
   $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Bonus corresponding to enrichment with successor

For any $\mathcal{C}$, $\mathcal{C}$-separation decidable $\Rightarrow$ $\mathcal{C}^+$-separation decidable.

Some words about **complexity**:

1. Complexity depends on $|\mathcal{C}|$ (tied to the implicit alphabet).
2. $Pol(\text{AT})$ and $BPol(\text{AT})$ are **PSpace**(-complete).
3. If the alphabet is fixed, or $|\mathcal{C}|$ is constant,

   $Pol(\mathcal{C})$-separation and $BPol(\mathcal{C})$-separation are in **PTime**

Conclusion (2)

Future work

We have the result,

- For any $\mathcal{C}$,
  $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

Conclusion (2)

Future work

We have the result,

- For any $\mathcal{C}$,
  $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

A nontrivial corollary of the $BPol(\mathcal{C})$-algorithm is as follows:

- For any $\mathcal{C}$,
  $\mathcal{C}$-"something" decidable $\Rightarrow$ $BPol(\mathcal{C})$-membership decidable.

Conclusion (2)

Future work

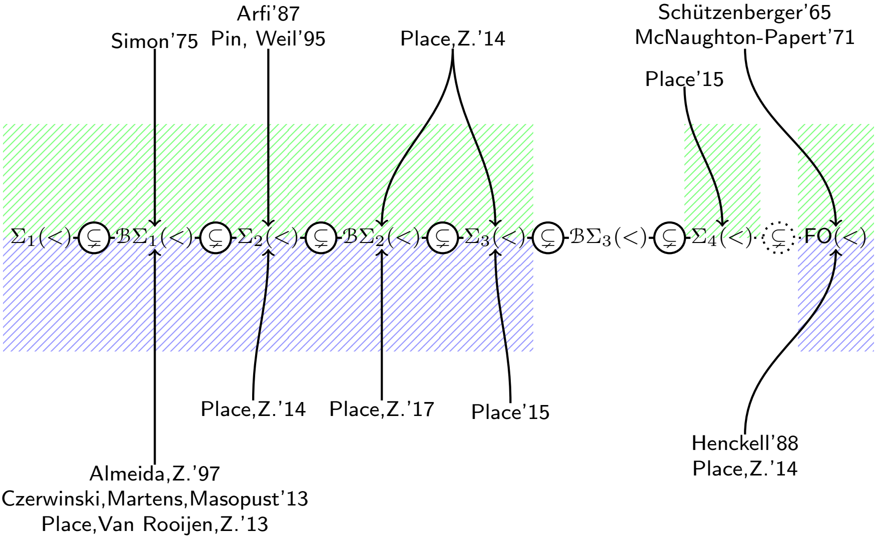
We have the result,

- For any $\mathcal{C}$,
  $\mathcal{C}$-separation decidable $\Rightarrow$ $Pol(\mathcal{C})$-membership decidable.

A nontrivial corollary of the $BPol(\mathcal{C})$-algorithm is as follows:

- For any $\mathcal{C}$,
  $\mathcal{C}$-"something" decidable $\Rightarrow$ $BPol(\mathcal{C})$-membership decidable.

$\mathcal{B}\Sigma_2(<)$-"something" seems to be decidable which would yield a membership algorithm for $\mathcal{B}\Sigma_3(<)$.


Arfi'87
Pin, Weil'95
Simon'75
Place,Z.'14
Schützenberger'65
McNaughton-Papert'71
Place'15
$\Sigma_1(<) \subsetneq \mathcal{B}\Sigma_1(<) \subsetneq \Sigma_2(<) \subsetneq \mathcal{B}\Sigma_2(<) \subsetneq \Sigma_3(<) \subsetneq \mathcal{B}\Sigma_3(<) \subsetneq \Sigma_4(<) \subsetneq$ FO$(<)$
Place,Z.'14
Place,Z.'17
Place'15
Henckell'88
Place,Z.'14
Almeida,Z.'97
Czerwinski,Martens,Masopust'13
Place,Van Rooijen,Z.'13

Thank You