

# The separation problem for languages of finite words

Thomas Place, Lorijn van Rooijen, Marc Zeitoun

LaBRI · Univ. Bordeaux · CNRS



May, 2013 · FREC · Île de Ré

*Based on Lorijn's slides*

# Het probleem van de scheiding van talen

- Motivaties voor deze vraag
  - Een eenvoudige klasse van afscheiders maakt algoritmische efficiëntie  
Voorbeeld: toegankelijkheid teller systemen.
  - hergroepering het lidmaatschap probleem.

# Outline

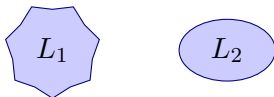
- 1 The separation problem
- 2 Separating regular by piecewise testable languages
- 3 Separation by  $FO^2(<)$ -definable / locally testable languages
- 4 Open problems

# Separation problem

- Let  $\text{Sep}$  be a class of languages closed under complement.
- $L_1, L_2 \in \mathcal{C}$  are **Sep-separable** if  $\exists K \in \text{Sep}$  st  $L_1 \subseteq K$  and  $L_2 \cap K = \emptyset$ .

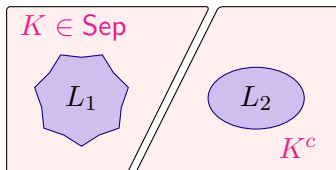
# Separation problem

- Let  $\text{Sep}$  be a class of languages closed under complement.
- $L_1, L_2 \in \mathcal{C}$  are **Sep-separable** if  $\exists K \in \text{Sep}$  st  $L_1 \subseteq K$  and  $L_2 \cap K = \emptyset$ .



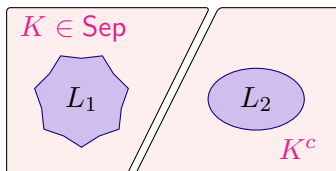
# Separation problem

- Let  $\text{Sep}$  be a class of languages closed under complement.
- $L_1, L_2 \in \mathcal{C}$  are **Sep-separable** if  $\exists K \in \text{Sep}$  st  $L_1 \subseteq K$  and  $L_2 \cap K = \emptyset$ .



# Separation problem

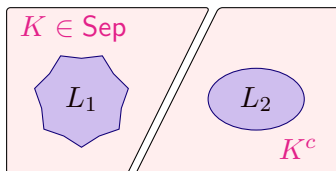
- Let  $\text{Sep}$  be a class of languages closed under complement.
- $L_1, L_2 \in \mathcal{C}$  are **Sep-separable** if  $\exists K \in \text{Sep}$  st  $L_1 \subseteq K$  and  $L_2 \cap K = \emptyset$ .



- No minimal** separator in general.

# Separation problem

- Let  $\text{Sep}$  be a class of languages closed under complement.
- $L_1, L_2 \in \mathcal{C}$  are **Sep-separable** if  $\exists K \in \text{Sep}$  st  $L_1 \subseteq K$  and  $L_2 \cap K = \emptyset$ .



- No minimal** separator in general.
- Co-example:**  $(a^2)^*$  and  $a(a^2)^*$  are disjoint but not  $\text{FO}(<)$ -separable.



# Separation problem: 2 questions

## 1 Separation problem:

“Given  $L_1, L_2 \in \mathcal{C}$ , decide whether they are **Sep**-separable”

- If  $\mathcal{C}$  complement-closed and  $L \in \mathcal{C}$ :

$$L \in \text{Sep} \iff (L, L^c) \text{ Sep-separable}$$

That is, **Sep**-membership reduces to **Sep**-separability.

## Separation problem: 2 questions

### 1 Separation problem:

“Given  $L_1, L_2 \in \mathcal{C}$ , decide whether they are **Sep**-separable”

- If  $\mathcal{C}$  complement-closed and  $L \in \mathcal{C}$ :

$$L \in \text{Sep} \iff (L, L^c) \text{ Sep-separable}$$

That is, **Sep**-membership reduces to **Sep**-separability.

### 2 Compute a separator.

### 3 How costly is it to decide? to compute a separator?

# Why separation?

# Why separation?

- Separation everywhere!
  - in math (eg, Hahn-Banach, optimization),
  - in CS: approximate query answering, [Czerwiński–Martens–Masopust13]
  - expressiveness & membership: robust property entailing decidability.

# Why separation?

- Separation everywhere!

- in math (eg, Hahn-Banach, optimization),
- in CS: approximate query answering, [Czerwiński–Martens–Masopust13]
- expressiveness & membership: robust property entailing decidability.

- Powerful tool: J. Leroux's proof to decide reachability in VASS.

*"If  $s \not\rightarrow^* t$ , then  $s$  and  $t$  are Presburger separable."*

## Why separation?

- Separation everywhere!
  - in math (eg, Hahn-Banach, optimization),
  - in CS: approximate query answering, [Czerwiński–Martens–Masopust13]
  - expressiveness & membership: robust property entailing decidability.
- Powerful tool: J. Leroux's proof to decide reachability in VASS.  
*"If  $s \not\rightarrow^* t$ , then  $s$  and  $t$  are Presburger separable."*
- Algorithmic effectiveness, eg, interpolation techniques in verification.

# Why separation?

- Separation everywhere!
  - in math (eg, Hahn-Banach, optimization),
  - in CS: approximate query answering, [Czerwiński–Martens–Masopust13]
  - expressiveness & membership: robust property entailing decidability.
- Powerful tool: J. Leroux's proof to decide reachability in VASS.
 

*"If  $s \not\rightarrow^* t$ , then  $s$  and  $t$  are Presburger separable."*
- Algorithmic effectiveness, eg, interpolation techniques in verification.
- It's fun!

## Separation problem vs. profinite approach

- Assume  $\mathcal{C} = \text{Reg}$  and  $\text{Sep}$  is a variety of languages.

**Examples:** groups, aperiodic monoids, J-trivial monoids, locally testable.



## Separation problem vs. profinite approach

- Assume  $\mathcal{C} = \text{Reg}$  and  $\text{Sep}$  is a variety of languages.

**Examples:** groups, aperiodic monoids, J-trivial monoids, locally testable.

- $\hat{F}_A(\text{Sep}) =$  relatively  $\text{Sep}$ -free profinite semigroup over  $A$ .
- Theorem [Almeida96]**  $L_1, L_2$  are  $\text{Sep}$ -separable  $\iff \bar{L}_1 \cap \bar{L}_2 = \emptyset$ .  
where  $\bar{L}$  is the closure of  $L$  in  $\hat{F}_A(\text{Sep})$ .

## Separation problem vs. profinite approach

- Assume  $\mathcal{C} = \text{Reg}$  and  $\text{Sep}$  is a variety of languages.

**Examples:** groups, aperiodic monoids, J-trivial monoids, locally testable.

- $\hat{F}_A(\text{Sep}) =$  relatively  $\text{Sep}$ -free profinite semigroup over  $A$ .
- Theorem [Almeida96]**  $L_1, L_2$  are  $\text{Sep}$ -separable  $\iff \bar{L}_1 \cap \bar{L}_2 = \emptyset$ .  
where  $\bar{L}$  is the closure of  $L$  in  $\hat{F}_A(\text{Sep})$ .
- Deciding  $\bar{L}_1 \cap \bar{L}_2 = \emptyset$  is equivalent to computing the 2-pointlikes for  $\text{Sep}$

## Separation problem vs. profinite approach

- Computing the 2-pointlikes is not easy in general.
- Known to be computable for many classes
  - groups [Ash91, RZ92, Auinger04, Auinger–Steinberg05]
  - aperiodic monoids [Henckell88, Henckell–Rhodes–Steinberg10]
  - J- and R-trivial monoids [Almeida–Z.95, Almeida–Costa–Z.08]
  - locally testable monoids [Costa–Nogueira10]

## Separation problem vs. profinite approach

- Computing the 2-pointlikes is not easy in general.
- Known to be computable for many classes
  - groups [Ash91, RZ92, Auinger04, Auinger–Steinberg05]
  - aperiodic monoids [Henckell88, Henckell–Rhodes–Steinberg10]
  - J- and R-trivial monoids [Almeida–Z.95, Almeida–Costa–Z.08]
  - locally testable monoids [Costa–Nogueira10]
- **Drawbacks** This approach only provides a yes/no answer.  
No separator in the “yes” case.

## Separation problem vs. profinite approach

- Computing the 2-pointlikes is not easy in general.
- Known to be computable for many classes
  - groups [Ash91, RZ92, Auinger04, Auinger–Steinberg05]
  - aperiodic monoids [Henckell88, Henckell–Rhodes–Steinberg10]
  - J- and R-trivial monoids [Almeida–Z.95, Almeida–Costa–Z.08]
  - locally testable monoids [Costa–Nogueira10]
- **Drawbacks** This approach only provides a yes/no answer.  
No separator in the “yes” case.
- Papers read/understood mainly by their referees/authors.

# Outline

- 1 The separation problem
- 2 Separating regular by piecewise testable languages
- 3 Separation by  $FO^2(<)$ -definable / locally testable languages
- 4 Open problems

# Piecewise testable languages

- Want to separate regular by piecewise testable languages.
- $u$  is a subword of  $v$  if

$$u = a_1 \cdots a_n \quad v = v_0 a_1 v_1 \cdots v_{n-1} a_n v_n, \quad a_i \in A, \quad v_i \in A^*.$$

# Piecewise testable languages

- Want to separate **regular** by **piecewise testable** languages.
- $u$  is a **subword** of  $v$  if

$$u = a_1 \cdots a_n \quad v = v_0 a_1 v_1 \cdots v_{n-1} a_n v_n, \quad a_i \in A, \quad v_i \in A^*.$$

- $u, v$  are  **$\sim_n$ -equivalent** if they have the same subwords up to length  $n$ .
- A **piecewise-testable language** is a union of  $\sim_n$ -classes.



## Piecewise testable languages

A regular language  $L$  over  $A$  is **piecewise testable (PT)** if

$L$  is a union of  $\sim_n$ -classes, for some  $n$

# Piecewise testable languages

A regular language  $L$  over  $A$  is **piecewise testable (PT)** if

$L$  is a union of  $\sim_n$ -classes, for some  $n$

$\iff L$  is a finite Boolean combination of languages of the form  $A^*a_1A^*a_2\dots A^*a_nA^*$ , where every  $a_i \in A$

$\iff L$  is determined by a  $B\Sigma_1(<)$  formula.

## Piecewise testable languages

A regular language  $L$  over  $A$  is **piecewise testable (PT)** if

$L$  is a union of  $\sim_n$ -classes, for some  $n$

$\iff L$  is a finite Boolean combination of languages of the form  $A^*a_1A^*a_2\dots A^*a_nA^*$ , where every  $a_i \in A$

$\iff L$  is determined by a  $B\Sigma_1(<)$  formula.

$\iff$  its syntactic monoid is  $\mathcal{J}$ -trivial (Simon's Theorem)

# Piecewise testable languages

A regular language  $L$  over  $A$  is **piecewise testable (PT)** if

$L$  is a union of  $\sim_n$ -classes, for some  $n$

$\iff L$  is a finite Boolean combination of languages of the form  $A^*a_1A^*a_2\dots A^*a_nA^*$ , where every  $a_i \in A$

$\iff L$  is determined by a  $B\Sigma_1(<)$  formula.

$\iff$  its syntactic monoid is  $\mathcal{J}$ -trivial (Simon's Theorem)

$\iff$  its minimal automaton avoids some forbidden patterns (can be checked in PTIME, Stern '82).

# Piecewise testable languages

A regular language  $L$  over  $A$  is **piecewise testable (PT)** if

$L$  is a union of  $\sim_n$ -classes, for some  $n$

$\iff L$  is a finite Boolean combination of languages of the form  $A^*a_1A^*a_2\dots A^*a_nA^*$ , where every  $a_i \in A$

$\iff L$  is determined by a  $B\Sigma_1(<)$  formula.

$\iff$  its syntactic monoid is  $\mathcal{J}$ -trivial (Simon's Theorem)

$\iff$  its minimal automaton avoids some forbidden patterns  
(can be checked in PTIME, Stern '82).

**Co-example.**  $(ab)^*$  is not PT.

## Separability: co-examples

- $(ab)^+$  and  $(ba)^+$  are **not** PT-separable.

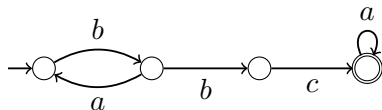
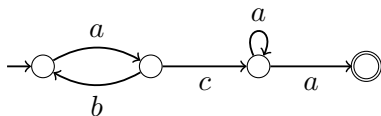
Indeed,  $\forall n \in \mathbb{N}, (ab)^n \sim_n (ba)^n$ .

## Separability: co-examples

- $(ab)^+$  and  $(ba)^+$  are **not** PT-separable.

Indeed,  $\forall n \in \mathbb{N}, (ab)^n \sim_n (ba)^n$ .

- The languages recognized by the following DFAs are **not** PT-separable.



## Main result for PT-separability

**Theorem** Given NFAs  $\mathcal{A}_1, \mathcal{A}_2$ , one can determine in  $\text{PTIME}(|Q_1|, |Q_2|, |A|)$ , whether  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are PT-separable.

### Notes

- Obvious semi-algorithm for testing separability.
- Need a witness for **non**-separability.
- 2 independent proofs
  - Czerwiński–Martens–Masopust.
  - Place-van Rooijen–Z.



# $(\vec{u}, \vec{B})$ -paths

A **factorization pattern** is a pair  $(\vec{u}, \vec{B})$ , where

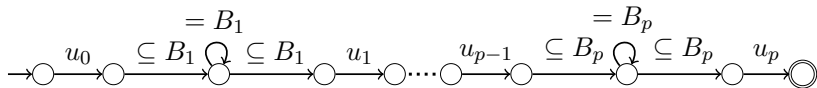
$$\begin{array}{ll} \vec{u} = (u_0, \dots, u_p) & \text{with } u_0, \dots, u_p \in A^*, \\ \vec{B} = (B_1, \dots, B_p) & \text{with } B_1, \dots, B_p \text{ nonempty subalphabets of } A. \end{array}$$

$(\vec{u}, \vec{B})$ -paths

A **factorization pattern** is a pair  $(\vec{u}, \vec{B})$ , where

$$\begin{aligned} \vec{u} &= (u_0, \dots, u_p) & \text{with } & u_0, \dots, u_p \in A^*, \\ \vec{B} &= (B_1, \dots, B_p) & \text{with } & B_1, \dots, B_p \text{ nonempty subalphabets of } A. \end{aligned}$$

A  $(\vec{u}, \vec{B})$ -**path** in  $\mathcal{A}$  is a successful path, of the form:



# Proof of the theorem

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

# Proof of the theorem

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

**Proposition 2** One can determine in  $\text{PTIME}(|Q_1|, |Q_2|, |A|)$  whether

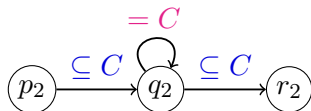
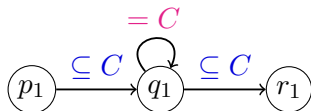
$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

# Detecting forbidden patterns

**Proposition 2** One can determine in  $\text{PTIME}(|Q_1|, |Q_2|, |A|)$  whether  $\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- By adding meta-transitions in  $\mathcal{A}_i$ , finding a  $(\vec{u}, \vec{B})$ -witness reduces to:

**Given** states  $p_i, q_i, r_i$  of  $\mathcal{A}_i$ , is there  $B \subseteq A$  st. the paths



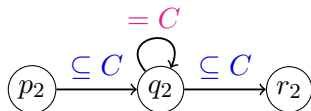
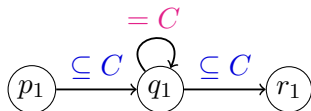
occur in  $\mathcal{A}_1$  resp.  $\mathcal{A}_2$ ?

# Detecting forbidden patterns

**Proposition 2** One can determine in  $\text{PTIME}(|Q_1|, |Q_2|, |A|)$  whether  $\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- By adding meta-transitions in  $\mathcal{A}_i$ , finding a  $(\vec{u}, \vec{B})$ -witness reduces to:

**Given** states  $p_i, q_i, r_i$  of  $\mathcal{A}_i$ , is there  $B \subseteq A$  st. the paths



occur in  $\mathcal{A}_1$  resp.  $\mathcal{A}_2$ ?

- This is in  $\text{PTIME}$ : iteratively use Tarjan's algorithm.

## Detecting forbidden patterns

- If  $C$  exists,  $C \subseteq C_1 \stackrel{\text{def}}{=} \text{alph\_scc}(q_1, \mathcal{A}_1) \cap \text{alph\_scc}(q_2, \mathcal{A}_2)$  (LINEAR)

- Restrict the automata to alphabet  $C_1$ , and repeat the process:

$$C_{i+1} \stackrel{\text{def}}{=} \text{alph\_scc}(q_1, \mathcal{A}_1 \upharpoonright_{C_i}) \cap \text{alph\_scc}(q_2, \mathcal{A}_2 \upharpoonright_{C_i}).$$

- After  $n \leq |A|$  iterations,  $C_n = C_{n+1}$ .
  - If  $C_n = \emptyset$ , the answer is no.
  - If  $C_n \neq \emptyset$ , it is the maximal possible  $C$  with  $(= C)$ -loops around  $q_1, q_2$ .

- Then, determine the remaining paths. (LINEAR)

- Overall LINEAR algorithm.

## Separability reduces to finding $(\vec{u}, \vec{B})$ -paths

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- **Easy.** If there are such  $(\vec{u}, \vec{B})$ -paths, then

$$\forall n, \exists u_n \in L(\mathcal{A}_1), w_n \in L(\mathcal{A}_2) \quad \text{s.t.} \quad u_n \sim_n w_n.$$

Hence,  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not separable**.



## Separability reduces to finding $(\vec{u}, \vec{B})$ -paths

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- $C^{\otimes} =$  set of words with alphabet exactly  $C$ .

$$\mathcal{L}(\vec{u}, \vec{B}, n) = u_0(B_1^{\otimes})^n u_1 \cdots u_{p-1}(B_p^{\otimes})^n u_p.$$

# Separability reduces to finding $(\vec{u}, \vec{B})$ -paths

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- $C^{\otimes} =$  set of words with alphabet exactly  $C$ .

$$\mathcal{L}(\vec{u}, \vec{B}, n) = u_0(B_1^{\otimes})^n u_1 \cdots u_{p-1}(B_p^{\otimes})^n u_p.$$

- $(w_n)_n$  is  $(\vec{u}, \vec{B})$ -adequate if

$$\forall n \geq 0, w_n \in \mathcal{L}(\vec{u}, \vec{B}, n).$$

- $(w_n)_n$  is adequate if it is  $(\vec{u}, \vec{B})$ -adequate for some  $(\vec{u}, \vec{B})$ .

## Separability reduces to finding $(\vec{u}, \vec{B})$ -paths

**Proposition 1**  $L(\mathcal{A}_1)$  and  $L(\mathcal{A}_2)$  are **not PT-separable**

$\exists (\vec{u}, \vec{B})$  such that both  $\mathcal{A}_1$  and  $\mathcal{A}_2$  have a  $(\vec{u}, \vec{B})$ -path.

- **Lemma 1** Every sequence  $(w_n)_n$  of words has an adequate subsequence.

**Immediate** from Simon's Factorization Forest Theorem.

- **Lemma 2** If  $(v_n)_n$  and  $(w_n)_n$  are such that

- $(v_n)_n$  is  $(\vec{u}, \vec{B})$ -adequate
- $(w_n)_n$  is  $(\vec{t}, \vec{C})$ -adequate
- $v_n \sim_n w_n$  for every  $n \geq 0$ .
- (+ simple technical condition)

Then,  $\vec{u} = \vec{t}$  and  $\vec{B} = \vec{C}$ .

## Computing a separator

**Theorem** Let  $L_1 = L(\mathcal{A}_1)$  and  $L_2 = L(\mathcal{A}_2)$ . Let  $p = \max(|Q_1|, |Q_2|) + 1$  and  $\kappa = p|A|2^{2^{|A|}|A|^{(p|A|+1)}}$ . **TFAE**:

- $L_1$  and  $L_2$  are PT-separable.
- $L_1$  and  $L_2$  are  $PT[\kappa]$ -separable.
- The language  $[L_1]_{\sim\kappa}$  separates  $L_1$  from  $L_2$ .
- $(\mathcal{A}_1, \mathcal{A}_2)$  have no witness of non PT-separability.

## Computing a separator

**Theorem** Let  $L_1 = L(\mathcal{A}_1)$  and  $L_2 = L(\mathcal{A}_2)$ . Let  $p = \max(|Q_1|, |Q_2|) + 1$  and  $\kappa = p|A|2^{2^{|A|}|A|^{(p|A|+1)}}$ . **TFAE**:

- $L_1$  and  $L_2$  are PT-separable.
- $L_1$  and  $L_2$  are  $PT[\kappa]$ -separable.
- The language  $[L_1]_{\sim\kappa}$  separates  $L_1$  from  $L_2$ .
- $(\mathcal{A}_1, \mathcal{A}_2)$  have no witness of non PT-separability.

$\kappa$  comes from Erdős-Szekeres upper bound of Ramsey numbers.  
(expon. improvement observed by W. Czerwiński).

# Computing a separator

**Theorem** Let  $L_1 = L(\mathcal{A}_1)$  and  $L_2 = L(\mathcal{A}_2)$ . Let  $p = \max(|Q_1|, |Q_2|) + 1$  and  $\kappa = p|A|2^{2^{|A|}|A|^{(p|A|+1)}}$ . **TFAE**:

- $L_1$  and  $L_2$  are PT-separable.
- $L_1$  and  $L_2$  are  $PT[\kappa]$ -separable.
- The language  $[L_1]_{\sim \kappa}$  separates  $L_1$  from  $L_2$ .
- $(\mathcal{A}_1, \mathcal{A}_2)$  have no witness of non PT-separability.

$\kappa$  comes from Erdős-Szekeres upper bound of Ramsey numbers.  
(expon. improvement observed by W. Czerwiński).

**Ramsey+Pumping** to show: if  $\exists u_1 \in L_1, u_2 \in L_2$  and  $u_1 \sim_{\kappa} u_2$ , then

$$\forall n \quad \exists u_{1,n} \in L_1 \quad \exists u_{2,n} \in L_2 \quad u_{1,n} \sim_n u_{2,n}.$$

## Separation by $\text{FO}^2(<)$ -definable languages

Same power:  $\text{FO}^2(<)$ , Unambiguous Languages (UL),  $\Delta_2$ ,  $\Sigma_2 \cap \Pi_2$ , DA, UTL  
[Schützenberger, Schwentick–Therien–Vollmer, Pin–Weil, Therien–Wilke].

## Separation by $\text{FO}^2(<)$ -definable languages

Same power:  $\text{FO}^2(<)$ , Unambiguous Languages (UL),  $\Delta_2$ ,  $\Sigma_2 \cap \Pi_2$ , DA, UTL  
[Schützenberger, Schwentick–Therien–Vollmer, Pin–Weil, Therien–Wilke].

$u \cong_k v$  if  $u, v$  belong to the same unambiguous products of size at most  $k$ .



## Separation by $\text{FO}^2(<)$ -definable languages

Same power:  $\text{FO}^2(<)$ , Unambiguous Languages (UL),  $\Delta_2$ ,  $\Sigma_2 \cap \Pi_2$ , DA, UTL  
 [Schützenberger, Schwentick–Therien–Vollmer, Pin–Weil, Therien–Wilke].

$u \cong_k v$  if  $u, v$  belong to the same unambiguous products of size at most  $k$ .

**Theorem** Let  $M_1$  and  $M_2$  be monoids recognizing  $L_1, L_2 \subseteq A^*$ .

Let  $\kappa = (2|M_1||M_2| + 1)(|A| + 1)^2$ . **TFAE**:

- $L_1$  and  $L_2$  are UL-separable.
- $L_1$  and  $L_2$  are  $\text{UL}[\kappa]$ -separable.
- The language  $[L_1]_{\cong_\kappa}$  separates  $L_1$  from  $L_2$ .

## Separation by $\text{FO}^2(<)$ -definable languages

Same power:  $\text{FO}^2(<)$ , Unambiguous Languages (UL),  $\Delta_2$ ,  $\Sigma_2 \cap \Pi_2$ , DA, UTL  
 [Schützenberger, Schwentick–Therien–Vollmer, Pin–Weil, Therien–Wilke].

$u \cong_k v$  if  $u, v$  belong to the same unambiguous products of size at most  $k$ .

**Theorem** Let  $M_1$  and  $M_2$  be monoids recognizing  $L_1, L_2 \subseteq A^*$ .

Let  $\kappa = (2|M_1||M_2| + 1)(|A| + 1)^2$ . **TFAE**:

- $L_1$  and  $L_2$  are UL-separable.
- $L_1$  and  $L_2$  are  $\text{UL}[\kappa]$ -separable.
- The language  $[L_1]_{\cong_\kappa}$  separates  $L_1$  from  $L_2$ .

### Note

- New result (pointlike sets not known to be computable for DA).
- No algorithm in terms of forbidden patterns.

## Separation by LT languages

**Locally testable language:** finite boolean combination of  $uA^*$ ,  $A^*uA^*$ ,  $A^*u$ .

$u \equiv_k v$  if  $u$  and  $v$  have the same factors of length  $\leq k$  (+ prefix, suffix).

**Theorem** Let  $M_1$  and  $M_2$  be monoids recognizing  $L_1, L_2 \subseteq A^*$ .

Let  $\kappa = 4(|M_1| + |M_2| + 1)$ . **TFAE:**

- $L_1$  and  $L_2$  are LT-separable.
- $L_1$  and  $L_2$  are  $\text{LT}[\kappa]$ -separable.
- The language  $[L_1]_{\equiv_\kappa}$  separates  $L_1$  from  $L_2$ .

## Separation by LT languages

**Locally testable language:** finite boolean combination of  $uA^*$ ,  $A^*uA^*$ ,  $A^*u$ .

**Theorem Forbidden pattern characterization.**

- $L_1, L_2$  separable iff no two sublanguages of the form

$$u_1v_1^*u_2v_2^*\cdots v_{n-1}^*u_n$$

inducing same infinite prefix, infinite suffix, bi-infinite infixes  $u_{i-1}^{-\infty}v_iu_{i-1}^{+\infty}$ .

- Yields NP-hardness.

## Open problems

- $FO^2(+1)$ , modular predicates,
- FO-separability, nice proof by Thomas from Thomas :).

# Open problems

- $\text{FO}^2(+1)$ , modular predicates,
- FO-separability, nice proof by Thomas from Thomas :).
- Separating non-regular languages.  
Reg-separation of CF languages [Szymanski–Williams76, Hunt III82]
- Infinite words, trees.

# Open problems

- $FO^2(+1)$ , modular predicates,
- FO-separability, nice proof by Thomas from Thomas :).
- Separating non-regular languages.  
Reg-separation of CF languages [Szymanski–Williams76, Hunt III82]
- Infinite words, trees.
- Separation by positive varieties, lattices of languages, etc.
- Complexity issues (time, size of separators).
- Efficient computation of separators.

# Open problems

- $\text{FO}^2(+1)$ , modular predicates,
- FO-separability, nice proof by Thomas from Thomas :).
- Separating non-regular languages.  
Reg-separation of CF languages [Szymanski–Williams76, Hunt III82]
- Infinite words, trees.
- Separation by positive varieties, lattices of languages, etc.
- Complexity issues (time, size of separators).
- Efficient computation of separators.
- Algebraic object from  $L_1, L_2$  on which one could test separability.