# The separation problem for languages of finite words

Thomas Place, Lorijn van Rooijen, Marc Zeitoun

LaBRI · Univ. Bordeaux · CNRS

July 2013 – ALFA'13

# Separation problem

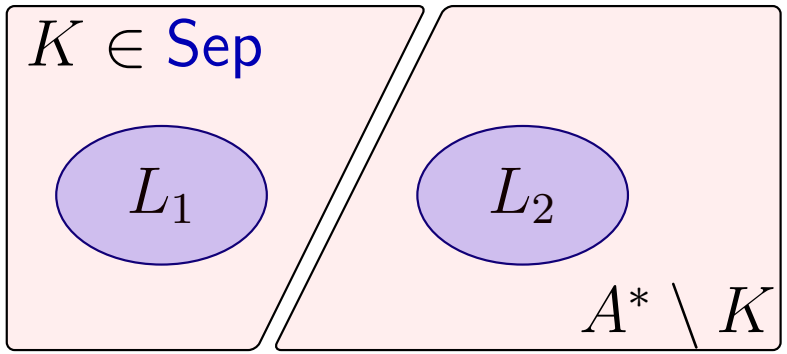
- $L_1, L_2 \in \mathcal{C}$ are Sep-**separable** if $\exists K \in$ Sep, $L_1 \subseteq K$, $L_2 \cap K = \varnothing$.

# Separation problem

- $L_1, L_2 \in \mathcal{C}$ are Sep-**separable** if $\exists K \in \mathsf{Sep}$, $L_1 \subseteq K$, $L_2 \cap K = \varnothing$.

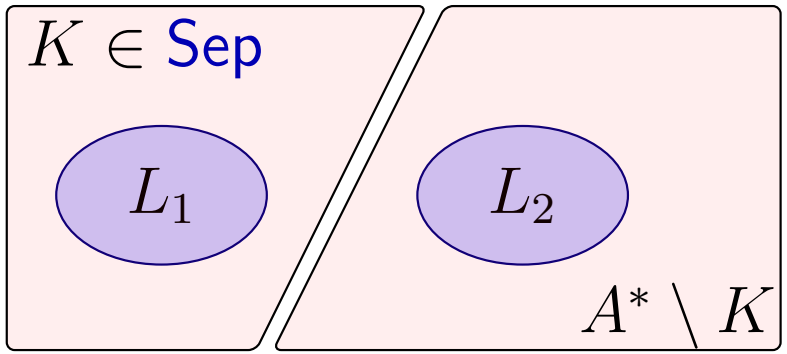
# Separation problem

- $L_1, L_2 \in \mathcal{C}$ are Sep-**separable** if $\exists K \in \mathsf{Sep}$, $L_1 \subseteq K$, $L_2 \cap K = \varnothing$.



# Separation problem

- $L_1, L_2 \in \mathcal{C}$ are Sep-**separable** if $\exists K \in$ Sep, $L_1 \subseteq K$, $L_2 \cap K = \varnothing$.

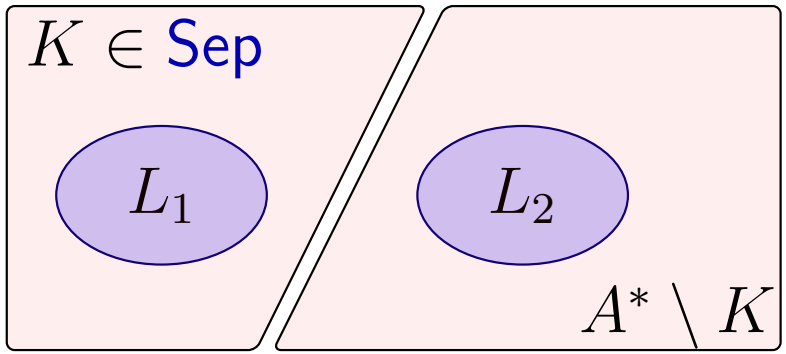



Sep = fixed class of languages *(closed under complement)*.

.

# Separation problem

- $L_1, L_2 \in \mathcal{C}$ are Sep-**separable** if $\exists K \in$ Sep, $L_1 \subseteq K$, $L_2 \cap K = \varnothing$.




Sep = fixed class of languages *(closed under complement)*.

.

- **No minimal** separator for in general.
- **Example**: $(a^2)^*$ and $a(a^2)^*$ are disjoint but not FO$(<)$-separable.

# Separation problem: some questions

1. **Decision problem**

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

# Separation problem: some questions

1. **Decision problem**

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

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

$$L \in \text{Sep} \iff (L,\ A^* \setminus L) \text{ Sep-separable}$$

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

# Separation problem: some questions

1. **Decision problem**

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

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

   $$L \in \text{Sep} \iff (L,\ A^*\backslash L) \text{ Sep-separable}$$

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

2. **Computation** of a separator.

3. **Complexity**: How costly is it to decide? to compute a separator?

# Outline

# Why separation?

# Why separation?

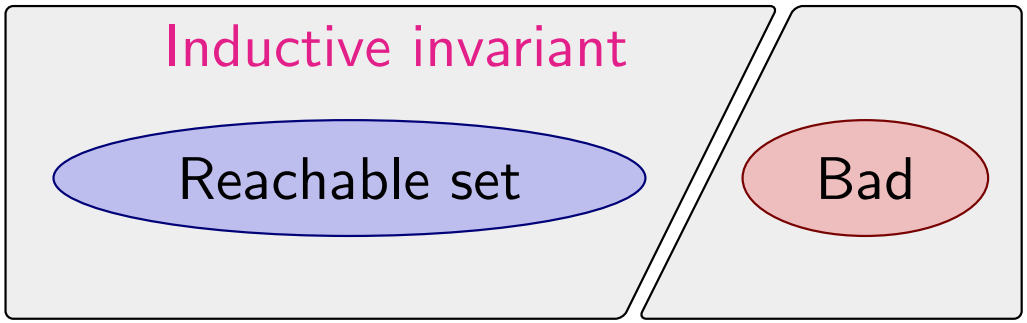
- Captures the discriminating power of logics.
  More accurate than expressive power.

# Why separation?

- Captures the discriminating power of logics.
  More accurate than expressive power.
- Separation everywhere!

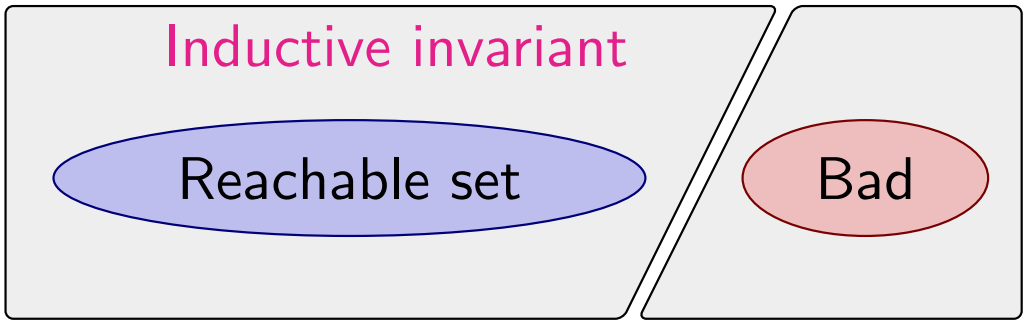
# Why separation?

- Captures the discriminating power of logics.
  More accurate than expressive power.
- Separation everywhere!
  - In math *(topology, Hahn-Banach, optimization)*,
  - Automatic verification: interpolation.



# Why separation?

- Captures the discriminating power of logics.
  More accurate than expressive power.
- Separation everywhere!
  - In math *(topology, Hahn-Banach, optimization)*,
  - Automatic verification: interpolation.




  - Reachability in VASS (J. Leroux's proof).
  - Algorithmic effectiveness, eg, interpolation techniques.

# Why separation?

- Captures the discriminating power of logics.
  More accurate than expressive power.
- Separation everywhere!
  - In math *(topology, Hahn-Banach, optimization)*,
  - Automatic verification: interpolation.
  - Approximate query answering, [Czerwiński–Martens–Masopust13]
  - XML schemas. [idem]

# Why separation?

- Captures the discriminating power of logics.
  More accurate than expressive power.
- Separation everywhere!
  - In math *(topology, Hahn-Banach, optimization)*,
  - Automatic verification: interpolation.
  - Approximate query answering, [Czerwiński–Martens–Masopust13]
  - XML schemas. [idem]
  - Semigroup theory: specific reductions membership $\leqslant$ separation.
    Separation for some classes entails membership for larger classes.

# Separation problem: profinite approach

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

  **Ex**: FO-, $\mathrm{FO}^2$-definable, piecewise-testable, locally testable.

# Separation problem: profinite approach

- Assume $\mathcal{C} = \mathsf{Reg}$ and Sep is a variety of languages.
  **Ex**: FO-, $\mathrm{FO}^2$-definable, piecewise-testable, locally testable.
- $\hat{\mathsf{F}}_A(\mathsf{Sep}) =$ relatively Sep-free profinite semigroup over $A$.
- Theorem [Almeida96] $L_1, L_2$ Sep-separable $\iff \overline{L}_1 \cap \overline{L}_2 = \varnothing$.
  $\overline{L} =$ closure of $L$ in $\hat{\mathsf{F}}_A(Sep)$.

# Separation problem: profinite approach

- Assume $\mathcal{C} = \mathsf{Reg}$ and Sep is a variety of languages.
  **Ex**: FO-, $\mathrm{FO}^2$-definable, piecewise-testable, locally testable.
- $\hat{\mathsf{F}}_A(\mathsf{Sep}) =$ relatively Sep-free profinite semigroup over $A$.
- Theorem [Almeida96] $L_1, L_2$ Sep-separable $\iff \overline{L}_1 \cap \overline{L}_2 = \varnothing$.
  $\overline{L} =$ closure of $L$ in $\hat{\mathsf{F}}_A(Sep)$.
- An approach for separation: "computing" closures.

# Separation problem vs. profinite approach

- Testing $\overline{L}_1 \cap \overline{L}_2 = \varnothing$ not easy in general.
- Already solved for separator languages recognized by
  - 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 [Steinberg01, Costa–Nogueira10]

# Separation problem vs. profinite approach

- Testing $\overline{L}_1 \cap \overline{L}_2 = \varnothing$ not easy in general.
- Already solved for separator languages from several classes.
- Drawbacks Only provides a yes/no answer.
  No separator in the "yes" case.
  Requires involved tools (profinite semigroup theory).

# Outline

# Generic proof outline

- Separation is RE when Sep is RE and $\subseteq, \cap$ against $\mathcal{C}$ decidable.

# Generic proof outline

- Separation is RE when Sep is RE and $\subseteq, \cap$ against $\mathcal{C}$ decidable.
- Need: RE witness for non-separability.

# Generic proof outline

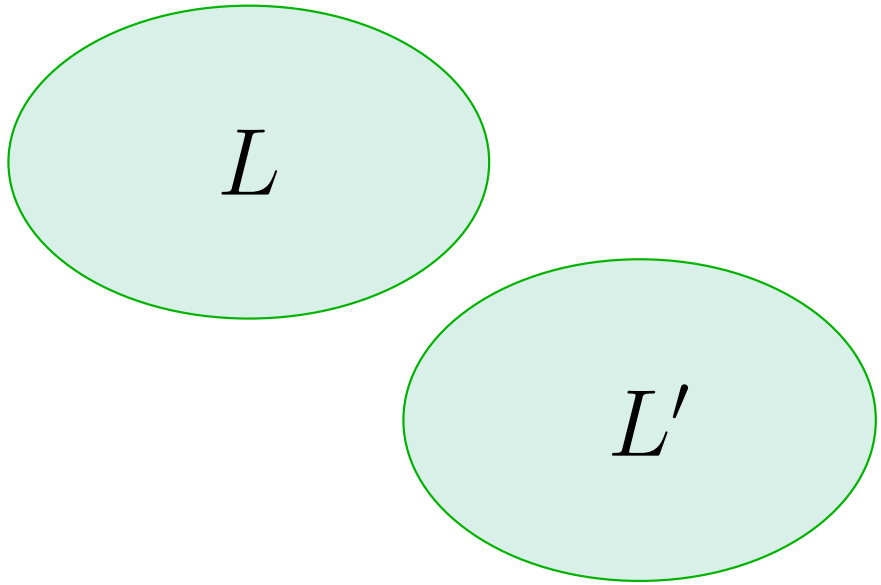
- Separation is RE when Sep is RE and $\subseteq, \cap$ against $\mathcal{C}$ decidable.
- Need: RE witness for non-separability.
- 2 approaches for getting such a witness.

# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Examples of layered classes:
     - FO-definable — formulas of quantifier depth $\leqslant n$.
     - Piecewise-testable — pieces of length $\leqslant n$.
     - Locally testable — windows of length $\leqslant n$.

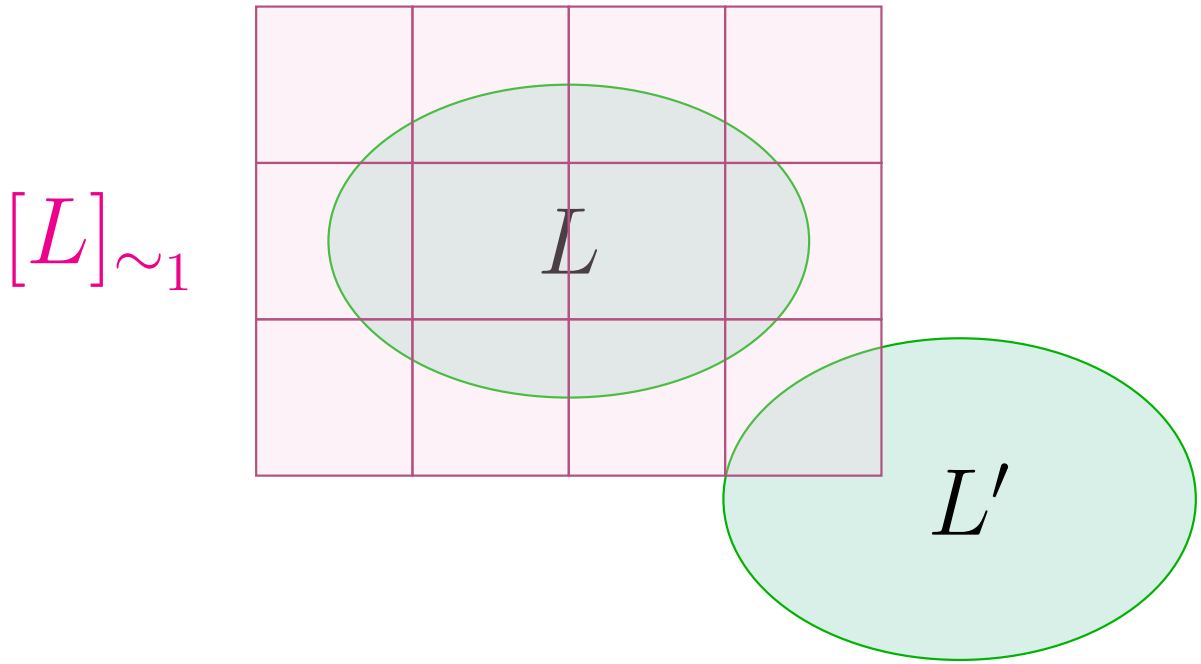
# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Examples of layered classes:
     - FO-definable — formulas of quantifier depth $\leqslant n$.
     - Piecewise-testable — pieces of length $\leqslant n$.
     - Locally testable — windows of length $\leqslant n$.



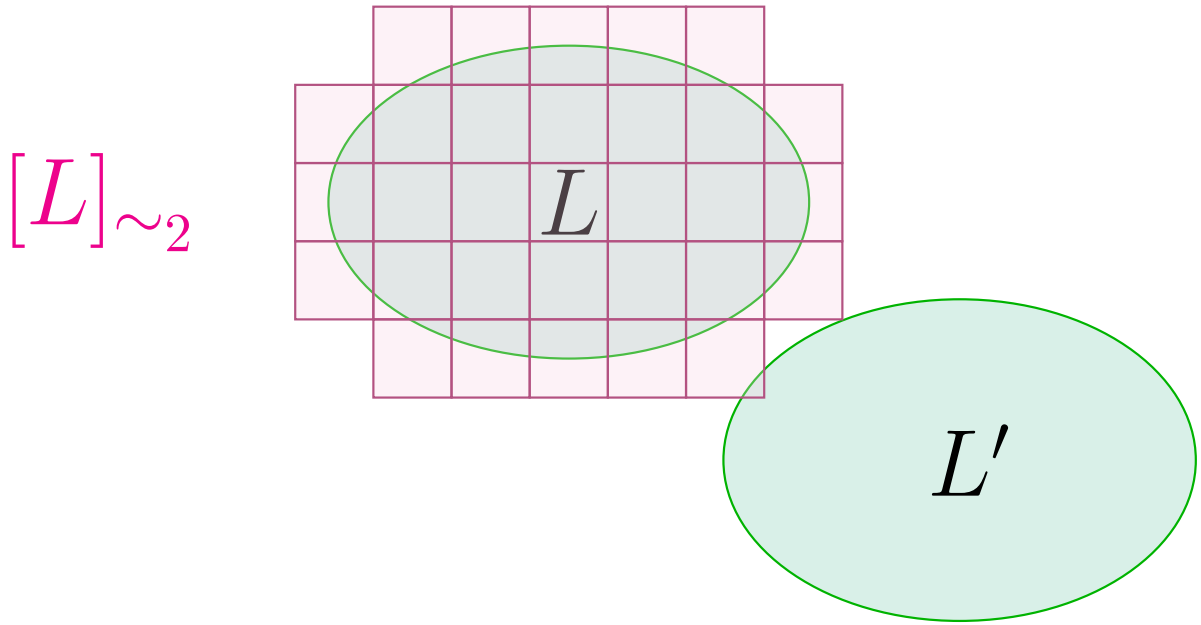
# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Examples of layered classes:
     - FO-definable — formulas of quantifier depth $\leqslant n$.
     - Piecewise-testable — pieces of length $\leqslant n$.
     - Locally testable — windows of length $\leqslant n$.



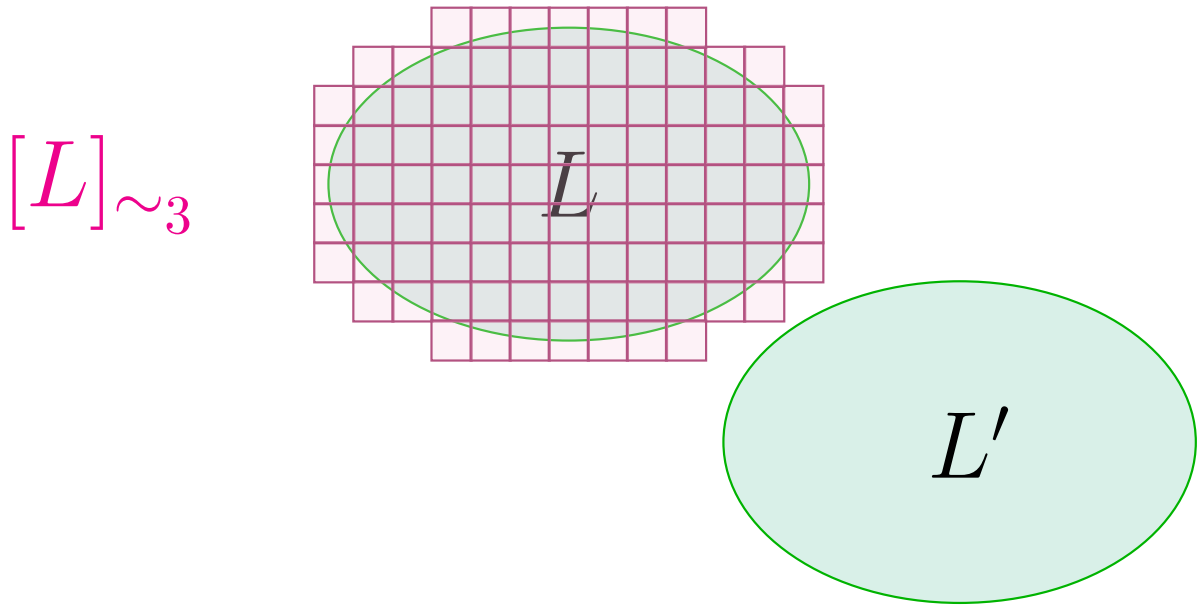
# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Examples of layered classes:
     - FO-definable — formulas of quantifier depth $\leqslant n$.
     - Piecewise-testable — pieces of length $\leqslant n$.
     - Locally testable — windows of length $\leqslant n$.



# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Examples of layered classes:
     - FO-definable — formulas of quantifier depth $\leqslant n$.
     - Piecewise-testable — pieces of length $\leqslant n$.
     - Locally testable — windows of length $\leqslant n$.



# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Non-separability by languages up to index $n$ is RE.
   - Reduction to a computable sufficient index $N$.

$\rightsquigarrow$ Brute force algorithm.

# Witnesses of non-separability: 2 approaches

1. For layered classes of separators: index up to which to test.
   - Non-separability by languages up to index $n$ is RE.
   - Reduction to a computable sufficient index $N$.

   $\rightsquigarrow$ Brute force algorithm.

2. Witnesses as common patterns in automata recognizing $L_1, L_2$.

   $\rightsquigarrow$ Practical algorithms.

# Bounding index for layered classes: ingredients

- $L_1, L_2$ are not separable iff.

$$(\mathcal{P}) \qquad \forall n \qquad \exists w_1 \in L_1,\ \exists w_2 \in L_2 \qquad w_1 \sim_n w_2.$$

# Bounding index for layered classes: ingredients

- $L_1, L_2$ are not separable iff.

$$(\mathcal{P}) \qquad \forall n \qquad \exists w_1 \in L_1,\ \exists w_2 \in L_2 \qquad w_1 \sim_n w_2.$$

- Entails suitable decompositions of $w_1, w_2$.
- For computable index $N \geqslant f(L_1, L_2)$, pump decomposition factors
  - without changing syntactical values wrt. $L_1, L_2$.
  - entailing $(\mathcal{P})$.

# Bounding index for layered classes: ingredients

- $L_1, L_2$ are not separable iff.

$$(\mathcal{P}) \qquad \forall n \qquad \exists w_1 \in L_1,\ \exists w_2 \in L_2 \qquad w_1 \sim_n w_2.$$

- Entails suitable decompositions of $w_1, w_2$.
- For computable index $N \geqslant f(L_1, L_2)$, pump decomposition factors
  - without changing syntactical values wrt. $L_1, L_2$.
  - entailing $(\mathcal{P})$.
- Byproduct: decompositions yield common patterns.

# Meta theorem

Theorem One can compute $N = f(L_1, L_2)$ st. TFAE:

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

# Meta theorem

Theorem One can compute $N = f(L_1, L_2)$ st. TFAE:

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

- $N$ comes from pumping arguments

  (pigeonhole principle, Ramsey's theorem, Simon's FFT).

- Witness $=$ pattern with same shape (depends on Sep).

# Outline

# Piecewise testable languages

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

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

# Piecewise testable languages

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

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

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

# Piecewise testable languages

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

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

- $u, v$ $\sim_n$-**equivalent** if they have the same pieces up to length $n$.
- A piecewise-testable language is a union of $\sim_n$-classes.
- Are $(ab)^+$ and $(ba)^+$ PT-separable?

# Piecewise testable languages

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

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

- $u, v$ $\sim_n$-**equivalent** if they have the same pieces up to length $n$.
- A piecewise-testable language is a union of $\sim_n$-classes.
- Are $(ab)^+$ and $(ba)^+$ PT-separable?

No $\qquad \forall n \in \mathbb{N},\ (ab)^n \sim_n (ba)^n.$

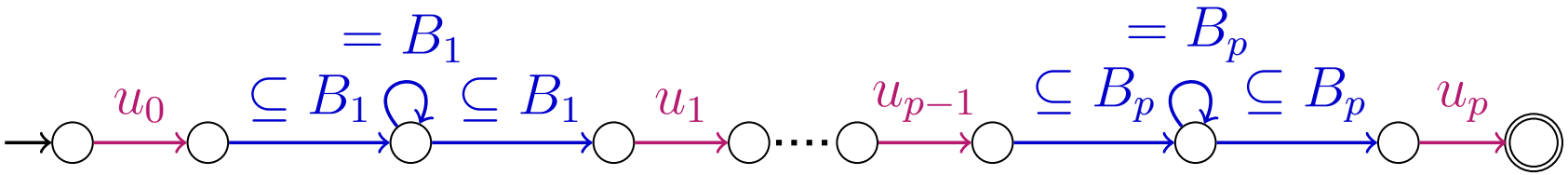
# PT-separability

Theorem From NFAs $\mathcal{A}_1, \mathcal{A}_2$, one determines in PTIME$(|Q_1|, |Q_2|, |A|)$ whether $L(\mathcal{A}_1)$ and $L(\mathcal{A}_2)$ are PT-separable.

- Recall: semi-algorithm for testing separability.
- Need a witness for non-separability.
- 2 independent, different proofs with different motivations
  - Czerwiński–Martens–Masopust [ICALP'13, talk on Wednesday].
  - Place-van Rooijen–Zeitoun [MFCS'13].

# Patterns for non-PT-separability: $(\vec{u}, \vec{B})$-paths

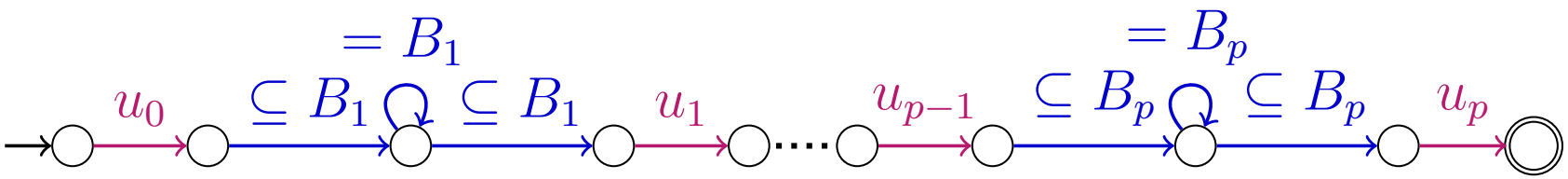
A $(\vec{u}, \vec{B})$-**path** in $\mathcal{A}$ is a successful path, of the form:




$u_i =$ words, $\quad B_i =$ subalphabets.

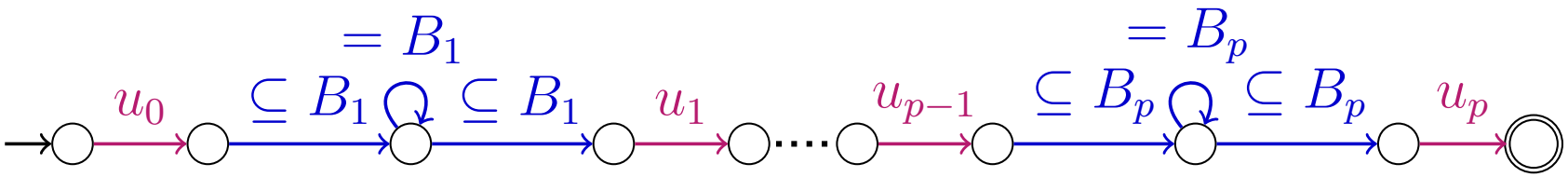
# PT-separability in PTIME: ingredients

Proposition 1 $L(\mathcal{A}_1)$ and $L(\mathcal{A}_2)$ are **not PT-separable** iff
$\exists\ (\vec{u}, \vec{B})$ such that both $\mathcal{A}_1$ and $\mathcal{A}_2$ have a $(\vec{u}, \vec{B})$-path.



# PT-separability in PTIME: ingredients

**Proposition 1** $L(\mathcal{A}_1)$ and $L(\mathcal{A}_2)$ are **not PT-separable** iff
$\exists\, (\vec{u}, \vec{B})$ such that both $\mathcal{A}_1$ and $\mathcal{A}_2$ have a $(\vec{u}, \vec{B})$-path.




**Proposition 2** One can determine in 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.

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

Same power: $\mathrm{FO}^2(<)$, Unambiguous Languages, $\Delta_2$, $\Sigma_2 \cap \Pi_2$, DA, UTL

[Schützenberger,Schwentick–Therien–Vollmer,Pin–Weil,Therien–Wilke]

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

Same power: $\mathrm{FO}^2(<)$, Unambiguous Languages, $\Delta_2$, $\Sigma_2 \cap \Pi_2$, DA, UTL

[Schützenberger,Schwentick–Therien–Vollmer,Pin–Weil,Therien–Wilke]

Layers:

$u \cong_k v$ if

$u, v$ belong to the same unambiguous products of size $\leqslant k$.

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

Same power: $\mathrm{FO}^2(<)$, Unambiguous Languages, $\Delta_2$, $\Sigma_2 \cap \Pi_2$, DA, UTL

[Schützenberger,Schwentick–Therien–Vollmer,Pin–Weil,Therien–Wilke]

Layers:

$u \cong_k v$ if

$u, v$ belong to the same unambiguous products of size $\leqslant k$.

- Meta-theorem holds. *(Forbidden patterns require precomputation.)*
- NEXPSPACE from NFAs.
- Decidability result is new: DA has decidable 2-pointlike sets.

# Separation by LT and LTT languages

- Locally testable language LT:

  finite boolean combination of $uA^*$, $A^*uA^*$, $A^*u$.

- Locally threshold testable language LTT:

  in addition, can count up to some threshold.

- **Ex.** $(ab)^+$ is LT
  $\#^*a\#^*b\#^*$ is not LT.

- Layers: $u \equiv_n v$ if $u$ and $v$ have the same $n$-windows.

# Separation by LT and LTT languages

- Meta-theorem holds.
- Common pattern:


  with same triples $(\alpha_i, u_i, \alpha_{i+1})$.

- From NFAs, between NP and NEXPTIME for LT.
  Compare to: Membership is PTIME from DFA [Pin05]
- Can be adapted for LTT.

# Open problems

- Other logics: FO, $\mathrm{FO}^2(+1)$, modular predicates,...
- Other separators: positive varieties, lattices of languages, etc.
- Other separated: Separating non-regular languages.

  Reg-separation of CF languages [Szymanski–Williams76]
- Other structures (infinite words, trees).

# Open problems

- Other logics: FO, $\mathrm{FO}^2(+1)$, modular predicates,...
- Other separators: positive varieties, lattices of languages, etc.
- Other separated: Separating non-regular languages.

  Reg-separation of CF languages [Szymanski–Williams76]
- Other structures (infinite words, trees).

- Complexity issues (time, size of separators).
- Efficient computation of separators.

# Open problems

- Other logics: FO, $\mathrm{FO}^2(+1)$, modular predicates,...
- Other separators: positive varieties, lattices of languages, etc.
- Other separated: Separating non-regular languages.

  Reg-separation of CF languages [Szymanski–Williams76]
- Other structures (infinite words, trees).

- Complexity issues (time, size of separators).
- Efficient computation of separators.

- Algebraic interpretation?
- Links between separation and decidability?

# Questions?