Aperiodicity in Tree Automata

Zoltán ÉSIK Szabolcs IVÁN

Thessaloniki, 22^{th} May, 2007.

What is this talk about

- Aperiodicity is a well-studied property of semigroups.

What is this talk about

- Aperiodicity is a well-studied property of semigroups.
- This property corresponds nicely to a class of regular string languages.

What is this talk about

- Aperiodicity is a well-studied property of semigroups.
- This property corresponds nicely to a class of regular string languages.
- The concept can be extended to tree automata (to tree languages) several ways. One possible extension was introduced by Thomas.

What is this talk about

- Aperiodicity is a well-studied property of semigroups.
- This property corresponds nicely to a class of regular string languages.
- The concept can be extended to tree automata (to tree languages) several ways. One possible extension was introduced by Thomas.
- In this talk we give several other possible definitions of aperiodicity of tree automata, compare them and study them from the algebraic point of view.

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Theorem (Kamp, '68; McNaughton-Papert, '71)

The following are equivalent for any string language L:

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Theorem (Kamp, '68; McNaughton-Papert, '71)

The following are equivalent for any string language L:

- *L is a regular language with an aperiodic minimal automaton;*

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Theorem (Kamp, '68; McNaughton-Papert, '71)

The following are equivalent for any string language L:

- *L is a regular language with an aperiodic minimal automaton;*
- *L is definable in first order logic using the order relation over positions;*

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Theorem (Kamp, '68; McNaughton-Papert, '71)

The following are equivalent for any string language L:

- *L is a regular language with an aperiodic minimal automaton;*
- *L is definable in first order logic using the order relation over positions;*
- *L is definable in the temporal logic LTL.*

Aperiodicity of semigroups

Definition

A semigroup $\mathcal{S} = (S, \cdot)$ is said to be aperiodic iff there exists some $n > 0$ such that $x^n = x^{n+1}$ holds for any $x \in S$.

Definition

A finite automaton is aperiodic iff its translation semigroup is aperiodic.

Aperiodicity nicely corresponds to logical definability in the string case:

Theorem (Kamp, '68; McNaughton-Papert, '71)

The following are equivalent for any string language L:

- *L is a regular language with an aperiodic minimal automaton;*
- *L is definable in first order logic using the order relation over positions;*
- *L is definable in the temporal logic LTL.*

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.
- Σ: finite ranked alphabet of rank type R.

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.
- Σ: finite ranked alphabet of rank type R.
- $X_n = \{x_1, \ldots, x_n\}$: variables (assumed to be disjoint from any Σ).

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.
- Σ: finite ranked alphabet of rank type R.
- $X_n = \{x_1, \ldots, x_n\}$: variables (assumed to be disjoint from any Σ).
- $T_\Sigma(X_n)$: finite terms (trees) over Σ with variables from X_n allowed.

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.
- Σ: finite ranked alphabet of rank type R.
- $X_n = \{x_1, \ldots, x_n\}$: variables (assumed to be disjoint from any Σ).
- $T_\Sigma(X_n)$: finite terms (trees) over Σ with variables from X_n allowed.
- Trees of $T_\Sigma(X_1)$ that have exactly one occurrence of x_1 are called contexts.

Our trees

We will introduce several aperiodicity concepts for finite, ranked, ordered trees with variables:

Definitions

- A rank type R is a finite set of nonnegative integers.
- Σ: finite ranked alphabet of rank type R.
- $X_n = \{x_1, \ldots, x_n\}$: variables (assumed to be disjoint from any Σ).
- $T_\Sigma(X_n)$: finite terms (trees) over Σ with variables from X_n allowed.
- Trees of $T_\Sigma(X_1)$ that have exactly one occurrence of x_1 are called contexts.
- A tree that is not a variable is called proper.

Tree automata

Definitions

Tree automata

Definitions

- Let Σ be an alphabet of rank type R.

Tree automata

Definitions

- Let Σ be an alphabet of rank type R.
- A tree automaton $\mathbb{A} = (A, \Sigma)$ – where A is a finite state set – associates a function $\sigma^{\mathbb{A}} : A^n \to A$ for each function symbol $\sigma \in \Sigma_n$.

Tree automata

Definitions

- Let Σ be an alphabet of rank type R.
- A tree automaton $\mathbb{A} = (A, \Sigma)$ – where A is a finite state set – associates a function $\sigma^{\mathbb{A}} : A^n \to A$ for each function symbol $\sigma \in \Sigma_n$.
- Each term $t \in T_\Sigma(X_n)$ induces a term function $t^{\mathbb{A}} : A^n \to A$.

Tree automata

Definitions

- Let Σ be an alphabet of rank type R.
- A tree automaton $\mathbb{A} = (A, \Sigma)$ – where A is a finite state set – associates a function $\sigma^{\mathbb{A}} : A^n \to A$ for each function symbol $\sigma \in \Sigma_n$.
- Each term $t \in T_\Sigma(X_n)$ induces a term function $t^{\mathbb{A}} : A^n \to A$.
- Term functions that can be induced by some proper term are called proper term functions.

Aperiodicity in Tree Automata (Thomas)

For any tree automaton $\mathbb{A} = (A, \Sigma)$ we can associate a semigroup $C(\mathbb{A})$ as follows:

Aperiodicity in Tree Automata (Thomas)

For any tree automaton $\mathbb{A} = (A, \Sigma)$ we can associate a semigroup $C(\mathbb{A})$ as follows:

- $C(\mathbb{A})$ consists of the term functions of $\mathbb{A}$ that can be induced by proper contexts;

Aperiodicity in Tree Automata (Thomas)

For any tree automaton $\mathbb{A} = (A, \Sigma)$ we can associate a semigroup $C(\mathbb{A})$ as follows:

- $C(\mathbb{A})$ consists of the term functions of $\mathbb{A}$ that can be induced by proper contexts;
- product is function composition.

Aperiodicity in Tree Automata (Thomas)

For any tree automaton $\mathbb{A} = (A, \Sigma)$ we can associate a semigroup $C(\mathbb{A})$ as follows:

- $C(\mathbb{A})$ consists of the term functions of $\mathbb{A}$ that can be induced by proper contexts;
- product is function composition.

Definition (Thomas, '82)

A tree automaton $\mathbb{A}$ is said to be aperiodic iff $C(\mathbb{A})$ is an aperiodic semigroup.

Aperiodicity in Tree Automata (Thomas)

For any tree automaton $\mathbb{A} = (A, \Sigma)$ we can associate a semigroup $C(\mathbb{A})$ as follows:

- $C(\mathbb{A})$ consists of the term functions of $\mathbb{A}$ that can be induced by proper contexts;
- product is function composition.

Definition (Thomas, '82)

A tree automaton $\mathbb{A}$ is said to be aperiodic iff $C(\mathbb{A})$ is an aperiodic semigroup.

We will denote the class of all aperiodic finite tree automata **CAper**.

More restrictive variants of aperiodicity

It seems that the branching structure of trees cannot be captured using only contexts. It may worth studying more restrictive variants of aperiodicity, via a broader class of allowed terms.

More restrictive variants of aperiodicity

It seems that the branching structure of trees cannot be captured using only contexts. It may worth studying more restrictive variants of aperiodicity, via a broader class of allowed terms.

However, when $t \notin T_\Sigma(X_1)$, the induced term function is not a transformation.

Tupling terms

Definitions

Tupling terms

Definitions

- If $\mathbb{A} = (A, \Sigma)$ is a tree automaton, $n > 0$ is an integer and $t_1, \ldots, t_n \in T_\Sigma(X_n)$ are terms with (at most) n variables, then the tuple $\underline{t} = (t_1, \ldots, t_n)$ induces a term function $\underline{t}^{\mathbb{A}} : A^n \to A^n$ as follows: for any n-tuple of states $\underline{a} = (a_1, \ldots, a_n)$ let

$$\underline{t}^{\mathbb{A}}(\underline{a}) = (t_1^{\mathbb{A}}(\underline{a}), \ldots, t_n^{\mathbb{A}}(\underline{a})).$$

Tupling terms

Definitions

- If $\mathbb{A} = (A, \Sigma)$ is a tree automaton, $n > 0$ is an integer and $t_1, \ldots, t_n \in T_\Sigma(X_n)$ are terms with (at most) n variables, then the tuple $\underline{t} = (t_1, \ldots, t_n)$ induces a term function $\underline{t}^{\mathbb{A}} : A^n \to A^n$ as follows: for any n-tuple of states $\underline{a} = (a_1, \ldots, a_n)$ let

$$\underline{t}^{\mathbb{A}}(\underline{a}) = (t_1^{\mathbb{A}}(\underline{a}), \ldots, t_n^{\mathbb{A}}(\underline{a})).$$

- A tuple of terms is proper if each member of the tuple is a proper term.

Tupling terms

Definitions

- If $\mathbb{A} = (A, \Sigma)$ is a tree automaton, $n > 0$ is an integer and $t_1, \ldots, t_n \in T_\Sigma(X_n)$ are terms with (at most) n variables, then the tuple $\underline{t} = (t_1, \ldots, t_n)$ induces a term function $\underline{t}^{\mathbb{A}} : A^n \to A^n$ as follows: for any n-tuple of states $\underline{a} = (a_1, \ldots, a_n)$ let

$$\underline{t}^{\mathbb{A}}(\underline{a}) = (t_1^{\mathbb{A}}(\underline{a}), \ldots, t_n^{\mathbb{A}}(\underline{a})).$$

- A tuple of terms is proper if each member of the tuple is a proper term.
- Term functions that can be induced by some proper tuple of terms are also called proper.

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;
- product is function composition.

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;
- product is function composition.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called n-aperiodic iff $S_n(\mathbb{A})$ is an aperiodic semigroup.

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;
- product is function composition.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called n-aperiodic iff $S_n(\mathbb{A})$ is an aperiodic semigroup.
The class of all n-aperiodic finite tree automata is denoted $\mathbf{SAper}_n$.

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;
- product is function composition.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called n-aperiodic iff $S_n(\mathbb{A})$ is an aperiodic semigroup.
The class of all n-aperiodic finite tree automata is denoted $\mathbf{SAper}_n$.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called strongly aperiodic iff it is n-aperiodic for all $n > 0$. The corresponding class is denoted **SAper**.

The $S_n(\mathbb{A})$ semigroups

Definition

Let $\mathbb{A} = (A, \Sigma)$ be a tree automaton and $n > 0$ be an integer. The semigroup $S_n(\mathbb{A})$ is defined as follows:

- the elements of $S_n(\mathbb{A})$ are the proper term functions from A^n to A^n;
- product is function composition.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called n-aperiodic iff $S_n(\mathbb{A})$ is an aperiodic semigroup.
The class of all n-aperiodic finite tree automata is denoted $\mathbf{SAper}_n$.

Definition

A tree automaton $\mathbb{A} = (A, \Sigma)$ is called strongly aperiodic iff it is n-aperiodic for all $n > 0$. The corresponding class is denoted **SAper**.

Studied aspects

We have studied the following:

- the hierarchy of the above defined aperiodicity concepts;

Studied aspects

We have studied the following:

- the hierarchy of the above defined aperiodicity concepts;
- algebraic properties of the classes;

Studied aspects

We have studied the following:

- the hierarchy of the above defined aperiodicity concepts;
- algebraic properties of the classes;
- complexity of the membership problem at each level of the hierarchy;

Studied aspects

We have studied the following:

- the hierarchy of the above defined aperiodicity concepts;
- algebraic properties of the classes;
- complexity of the membership problem at each level of the hierarchy;
- connection to logics;

Studied aspects

We have studied the following:

- the hierarchy of the above defined aperiodicity concepts;
- algebraic properties of the classes;
- complexity of the membership problem at each level of the hierarchy;
- connection to logics;
- a variant of n-aperiodicity has also been introduced and described.

The hierarchy of the classes

Proposition

The hierarchy of the classes

Proposition

- Strong aperiodicity implies n-aperiodicity for any $n > 0$.

The hierarchy of the classes

Proposition

- Strong aperiodicity implies n-aperiodicity for any $n > 0$.
- n-aperiodicity implies k-aperiodicity for any $n \geq k > 0$.

The hierarchy of the classes

Proposition

- Strong aperiodicity implies n-aperiodicity for any $n > 0$.
- n-aperiodicity implies k-aperiodicity for any $n \geq k > 0$.
- 1-aperiodicity implies aperiodicity.

The hierarchy of the classes

Proposition

- Strong aperiodicity implies n-aperiodicity for any $n > 0$.
- n-aperiodicity implies k-aperiodicity for any $n \geq k > 0$.
- 1-aperiodicity implies aperiodicity.

Hence, we can talk about the aperiodicity hierarchy: we have that

$$\mathbf{SAper} \subseteq \cdots \subseteq \mathbf{SAper_2} \subseteq \mathbf{SAper_1} \subseteq \mathbf{CAper}.$$

The hierarchy of the classes

Proposition

- Strong aperiodicity implies n-aperiodicity for any $n > 0$.
- n-aperiodicity implies k-aperiodicity for any $n \geq k > 0$.
- 1-aperiodicity implies aperiodicity.

Hence, we can talk about the aperiodicity hierarchy: we have that

$$\mathbf{SAper} \subsetneq \ldots \subsetneq \mathbf{SAper_2} \subsetneq \mathbf{SAper_1} \subsetneq \mathbf{CAper}.$$

Theorem

For any nonclassical rank type R it holds that the above hierarchy is proper.

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

- *renamings;*

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

- *renamings;*
- *taking subautomata and homomorphic images;*

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

- *renamings;*
- *taking subautomata and homomorphic images;*
- *taking generalized cascade products.*

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

- *renamings;*
- *taking subautomata and homomorphic images;*
- *taking generalized cascade products.*

Renaming

The tree automaton $\mathbb{B} = (B, \Delta)$ is a renaming of $\mathbb{A} = (A, \Sigma)$ if $A = B$ and for each $\delta \in \Delta_n$ there exists a $\sigma \in \Sigma_n$ such that $\delta^{\mathbb{B}} = \sigma^{\mathbb{A}}$ holds.

Algebraic properties

Theorem

Each member of the aperiodicity hierarchy is a generalized cascade variety: closed under

- *renamings;*
- *taking subautomata and homomorphic images;*
- *taking generalized cascade products.*

Renaming

The tree automaton $\mathbb{B} = (B, \Delta)$ is a renaming of $\mathbb{A} = (A, \Sigma)$ if $A = B$ and for each $\delta \in \Delta_n$ there exists a $\sigma \in \Sigma_n$ such that $\delta^{\mathbb{B}} = \sigma^{\mathbb{A}}$ holds.

Note that a renaming is not neccessarily reversible: it cannot introduce new function interpretations but can erase some of them.

Generalized cascade product

Definition

Suppose $\mathbb{A} = (A, \Sigma)$ and $\mathbb{B} = (B, \Delta)$ are tree automata and $\alpha = \{\alpha_n : n \in R\}$ is a family of functions such that each α_n maps each element of $A^n \times \Sigma_n$ to some proper n-ary term over Δ.

Generalized cascade product

Definition

Suppose $\mathbb{A} = (A, \Sigma)$ and $\mathbb{B} = (B, \Delta)$ are tree automata and $\alpha = \{\alpha_n : n \in R\}$ is a family of functions such that each α_n maps each element of $A^n \times \Sigma_n$ to some proper n-ary term over Δ.
Then the generalized cascade product $\mathbb{C} = \mathbb{A} \times_\alpha \mathbb{B}$ is the Σ-tree automaton with carrier set $A \times B$ and where the function symbols are interpreted as follows:

Generalized cascade product

Definition

Suppose $\mathbb{A} = (A, \Sigma)$ and $\mathbb{B} = (B, \Delta)$ are tree automata and $\alpha = \{\alpha_n : n \in R\}$ is a family of functions such that each α_n maps each element of $A^n \times \Sigma_n$ to some proper n-ary term over Δ.
Then the generalized cascade product $\mathbb{C} = \mathbb{A} \times_\alpha \mathbb{B}$ is the Σ-tree automaton with carrier set $A \times B$ and where the function symbols are interpreted as follows:

$$\sigma^{\mathbb{C}}\big((a_1, b_1), \ldots, (a_n, b_n)\big) = \big(\sigma^{\mathbb{A}}(a_1, \ldots, a_n), t^{\mathbb{B}}(b_1, \ldots, b_n)\big),$$

where $t = \alpha_n(a_1, \ldots, a_n, \sigma)$.

Complexity of membership

Theorem (Cho-Huynh, '91)

Deciding aperiodicity of string automata is a PSPACE*-complete problem.*

Complexity of membership

Theorem (Cho-Huynh, '91)

Deciding aperiodicity of string automata is a PSPACE*-complete problem.*

A logspace reduction is possible from the string case to any nonclassical rank type R and integer $n > 0$.

Complexity of membership

Theorem (Cho-Huynh, '91)

Deciding aperiodicity of string automata is a PSPACE*-complete problem.*

A logspace reduction is possible from the string case to any nonclassical rank type R and integer $n > 0$.

Theorem

For any nonclassical rank type R and integer $n > 0$ it holds that the membership problem of the class $\mathbf{SAper}_n$ *of finite tree automata is* PSPACE*-hard and contained in* EXPTIME*.*

Complexity of membership

Theorem (Cho-Huynh, '91)

Deciding aperiodicity of string automata is a PSPACE*-complete problem.*

A logspace reduction is possible from the string case to any nonclassical rank type R and integer $n > 0$.

Theorem

For any nonclassical rank type R and integer $n > 0$ it holds that the membership problem of the class $\mathbf{SAper}_n$ *of finite tree automata is* PSPACE*-hard and contained in* EXPTIME*.*

Theorem

For any rank type R it holds that the membership problem of the class **SAper** *of finite tree automata is contained in* P*.*

Complexity of membership

Theorem (Cho-Huynh, '91)

Deciding aperiodicity of string automata is a PSPACE*-complete problem.*

A logspace reduction is possible from the string case to any nonclassical rank type R and integer $n > 0$.

Theorem

For any nonclassical rank type R and integer $n > 0$ it holds that the membership problem of the class $\mathbf{SAper}_n$ *of finite tree automata is* PSPACE*-hard and contained in* EXPTIME*.*

Theorem

For any rank type R it holds that the membership problem of the class **SAper** *of finite tree automata is contained in* P*.*

Correspondence to logic

Theorem

The class **CTL** *is a strict subclass of* $\mathbf{SAper}_1$.

Correspondence to logic

Theorem

The class **CTL** *is a strict subclass of* $\mathbf{SAper}_1$.

Theorem

There exists a tree language definable in $\mathrm{FO}(<)$ *that is not 1-aperiodic.*

Correspondence to logic

Theorem

The class **CTL** *is a strict subclass of* $\mathbf{SAper}_1$.

Theorem

There exists a tree language definable in $\mathrm{FO}(<)$ *that is not 1-aperiodic.*

Theorem

There exists a 1-aperiodic regular tree language that is not definable in $\mathrm{FO}(<, S_i)$.

The polynomial variant

We could also talk about proper polynomial functions instead of proper term functions (i.e. having all the constants even if there is no constant symbol at all).

The polynomial variant

We could also talk about proper polynomial functions instead of proper term functions (i.e. having all the constants even if there is no constant symbol at all).
The corresponding classes are denoted $\mathbf{SAper}_n^{(p)}$ and $\mathbf{SAper}^{(p)}$.

The polynomial variant

We could also talk about proper polynomial functions instead of proper term functions (i.e. having all the constants even if there is no constant symbol at all).
The corresponding classes are denoted $\mathbf{SAper}_n^{(p)}$ and $\mathbf{SAper}^{(p)}$.

Our results carry over to this polynomial variant, with one difference:

The polynomial variant

We could also talk about proper polynomial functions instead of proper term functions (i.e. having all the constants even if there is no constant symbol at all).
The corresponding classes are denoted $\mathbf{SAper}_n^{(p)}$ and $\mathbf{SAper}^{(p)}$.

Our results carry over to this polynomial variant, with one difference:

Theorem

Taking polynomial functions instead of term functions, the corresponding hierarchy of classes of finite tree automata collapses to level 2:

The polynomial variant

We could also talk about proper polynomial functions instead of proper term functions (i.e. having all the constants even if there is no constant symbol at all).
The corresponding classes are denoted $\mathbf{SAper}_n^{(p)}$ and $\mathbf{SAper}^{(p)}$.

Our results carry over to this polynomial variant, with one difference:

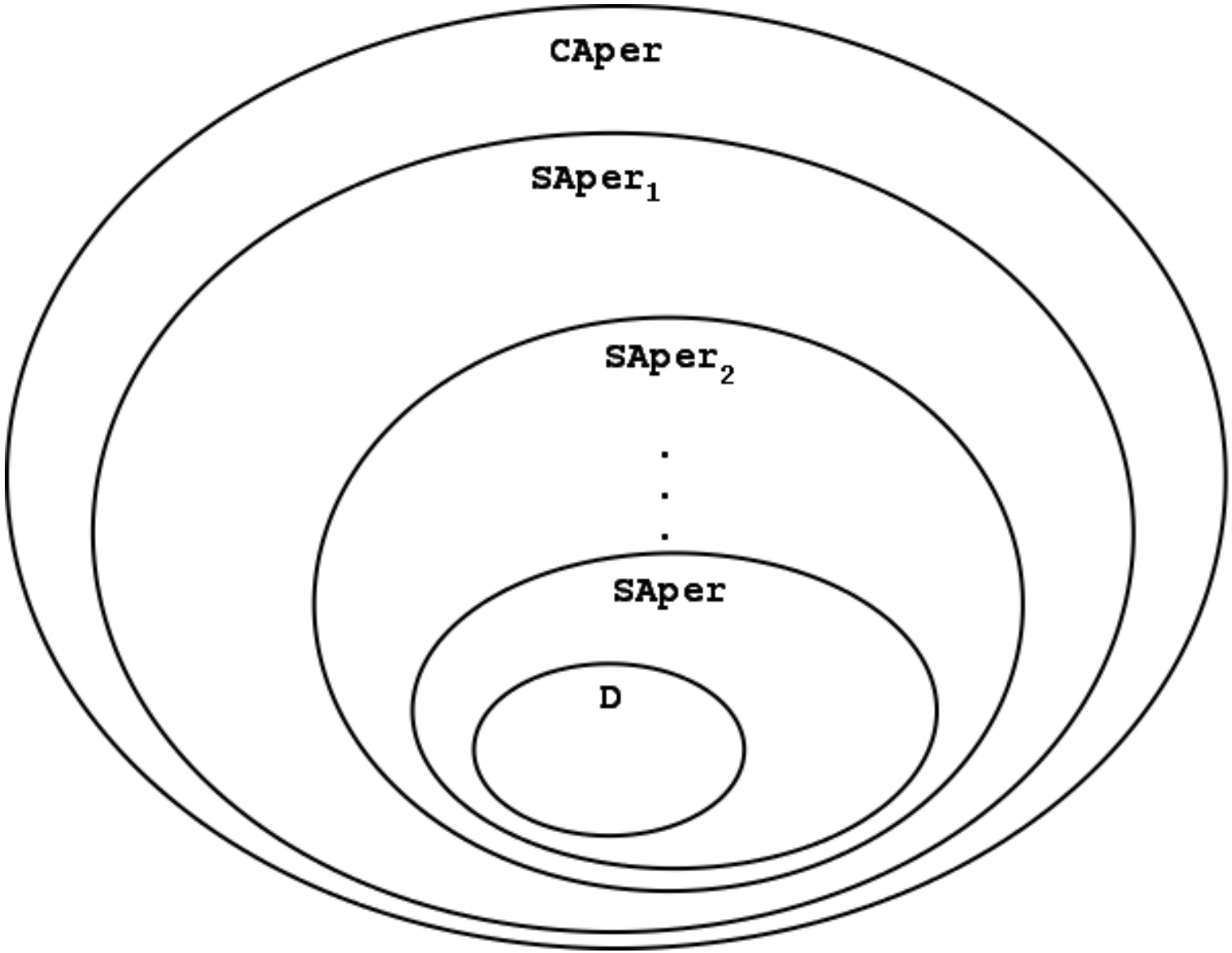
Theorem

Taking polynomial functions instead of term functions, the corresponding hierarchy of classes of finite tree automata collapses to level 2:

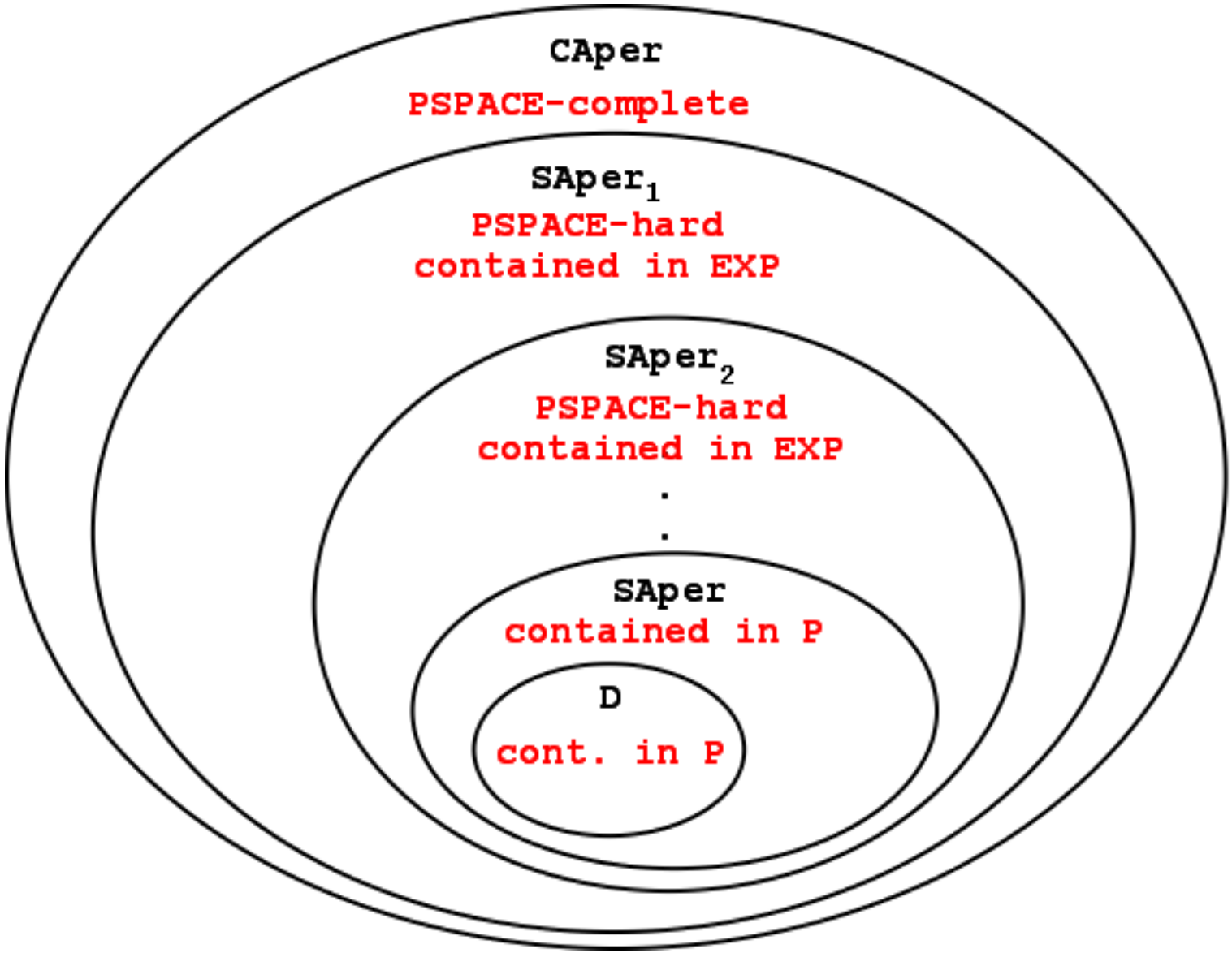
$$\mathbf{D} = \mathbf{SAper}^{(p)} = \mathbf{SAper}_2^{(p)} \subsetneq \mathbf{SAper}_1^{(p)} \subsetneq \mathbf{SAper}_1.$$

(Here $\mathbf{D}$ *denotes the class of all definite automata.)*

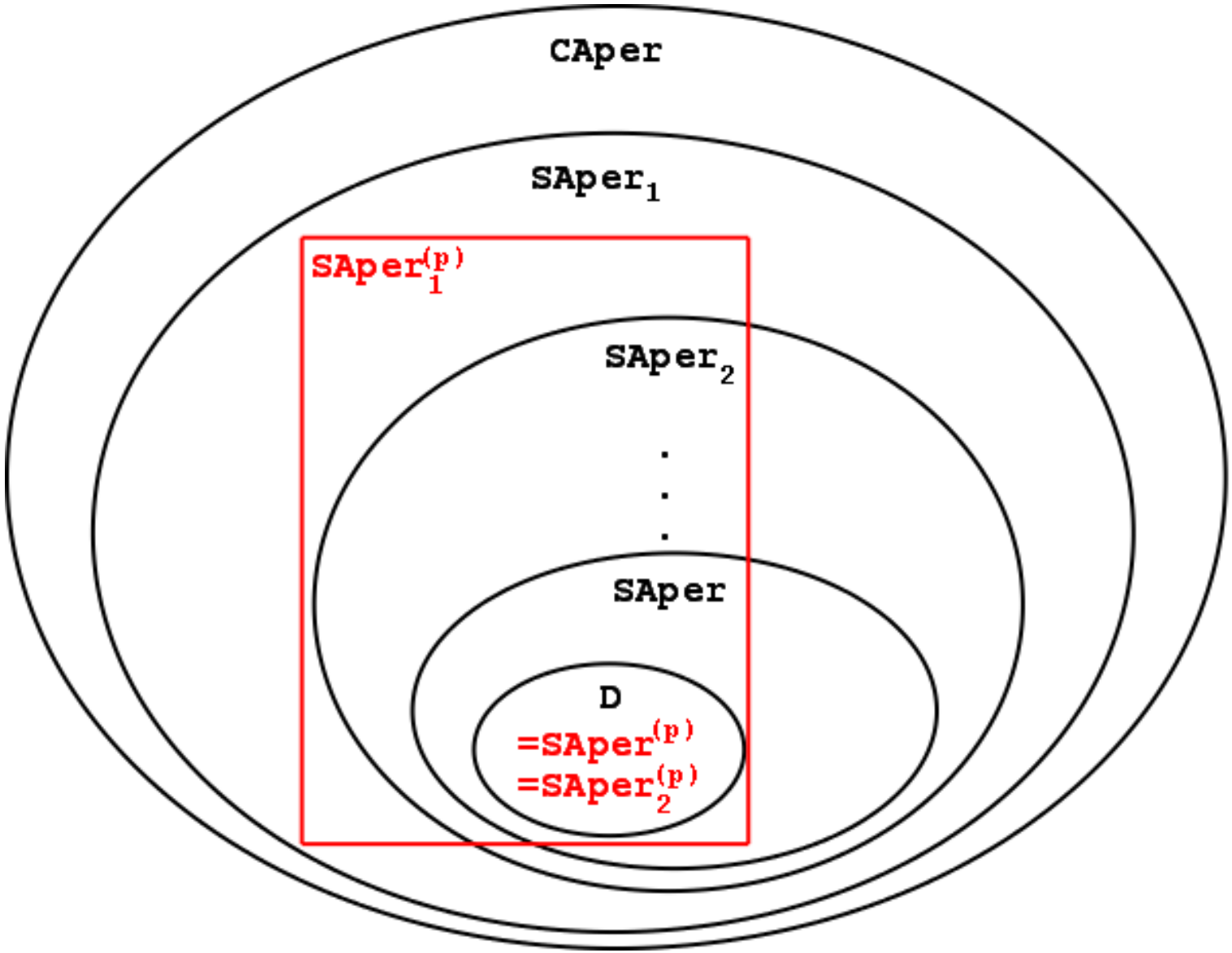
The hierarchy



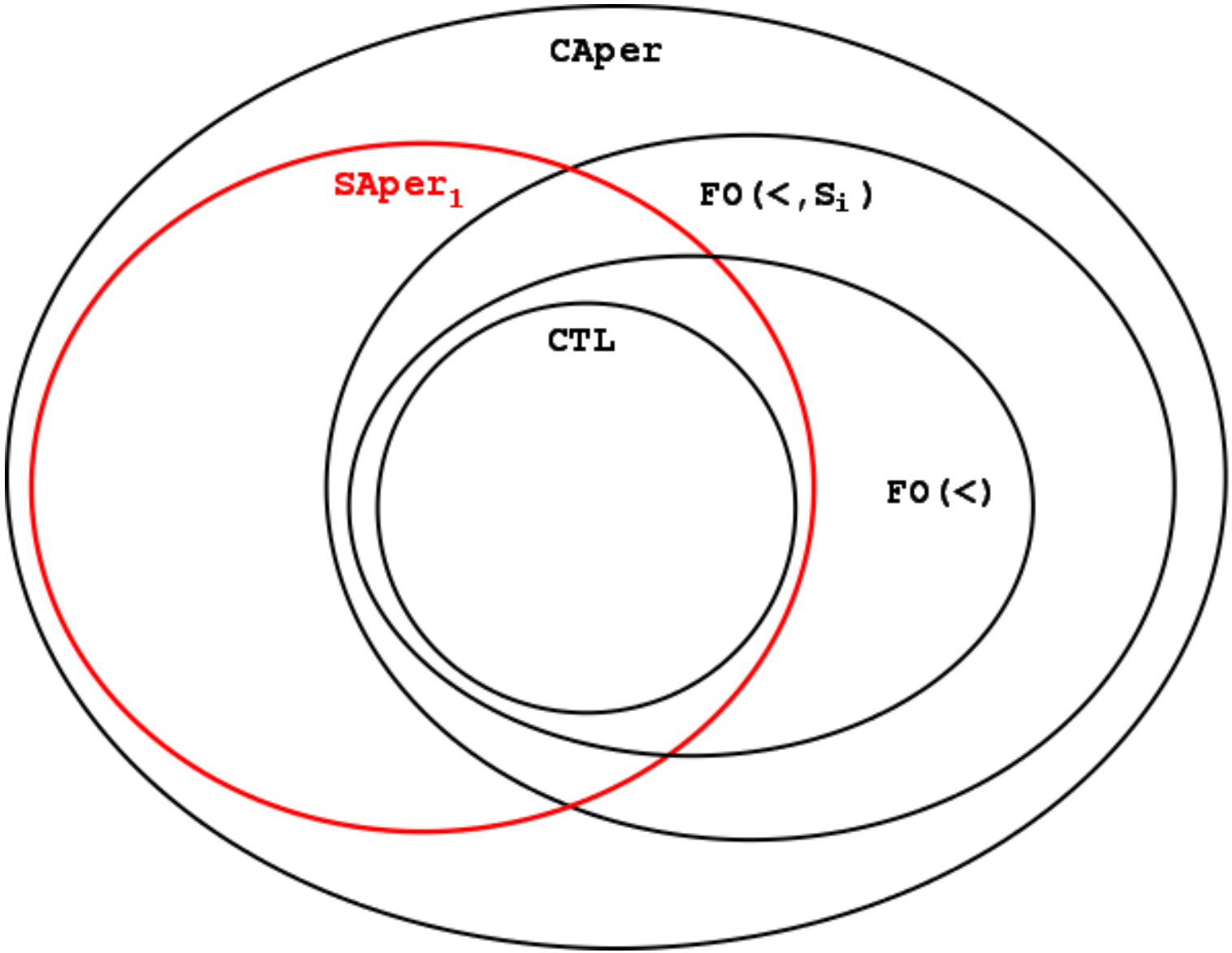
Complexities



Polynomial variant



Logics



Open problems

- Give an algebraic characterization (or a decidability result) of the logics CTL, $\mathrm{FO}(<)$ and $\mathrm{FO}(<, S_i)$.

Open problems

- Give an algebraic characterization (or a decidability result) of the logics CTL, $\mathrm{FO}(<)$ and $\mathrm{FO}(<, S_i)$.
  - for $\mathrm{FO}(S_i)$ it has been done by Benedikt and Segoufin.

Open problems

- Give an algebraic characterization (or a decidability result) of the logics CTL, $\mathrm{FO}(<)$ and $\mathrm{FO}(<, S_i)$.
  - for $\mathrm{FO}(S_i)$ it has been done by Benedikt and Segoufin.
- Describe the exact complexity of the membership problem of the classes $\mathbf{SAper}_n$.

Thank you.

Thank You.

References

- J. Almeida. On pseudovarieties, varieties of languages, filters of congruences, pseudoidentitiel and related topics. *Algebra Universalis,* 27:333–350, 1990.
- L. Bernátsky. regular expression star-freeness is PSPACE-complete. *Acta Cybernetica*, 13:1–21, 1997.
- S. Cho and Dung T. Huynh. Finite automaton aperiodicity is PSPACE-complete. *theoretical Computer Science*, 88:99–116, 1991.
- S. Eilenberg. *Automata, Languages and Machines*, vol. A and B, Academic Press, 1974 and 1976.
- Z. Ésik. Definite tree automata and their cascade compositions. *Publ. Math.*, 48:243–262, 1996.
- Z. Ésik. An algebraic characterization of temporal logics on finite trees. Parts I, II, III. In *1st International Conference on Algebraic Informatics, 2005*, pages 53–77, 79–99, 101-110. Aristotle Univ. Thessaloniki, Thessaloniki, 2005.

References

- Z. Ésik. Characterizing CTL-like logics on finite trees. *Theoretical computer Science* 356:136–152, 2006.
- F. Gécseg and M. Steinby. *Tree Automata*. Akadémiai Kiadó, 1984.
- G. Grätzer. *Universal Algebra.* 2nd ed., Springer-Verlag, 1979.
- U. Heuter. Definite tree languages. *Bulletin of the EATCS,* 35:137–142. 1988.
- U. Heuter. First-order properties of trees, star-free expressions, and aperiodicity. *RAIRO Inform. Théor. Appl.*, 25:125–145, 1991.
- R. McNaughton and S. Papert. *Counter-free automata*. M.I.T. Press, Cambridge, Mass.-London, 1971.
- A. Potthoff. Modulo-counting quantifiers over finite trees. *Theoretical Computer Science*, 126:97–112, 1994.

References

- M. Steinby. A theory of tree language varieties. In *Tree Automata and Languages*, pages 57–81, North-Holland, Amsterdam, 1992.
- M. Steinby. general varieties of tree languages. *Theoretical Computer Science*, 205:1–43, 1998.
- W. Thomas. Logical aspects in the study of tree languages. In *Ninth colloquium on trees in algebra and programming (Bordeaux, 1984)*, 31–49, Cambridge Univ. Press, Cambridge, 1984.