A Complexity Approach to Tree Algebras: the Bounded Case

Arthur Jaquard

joint work with Thomas Colcombet

IRIF, CNRS, Université de Paris

Automata Seminar | April 16, 2021

Algebras and classes of languages

Algebras are used to characterize classes of languages

Algebras and classes of languages

Algebras are used to characterize classes of languages

Finite words

Monoids, semigroups

Algebras and classes of languages

Algebras are used to characterize classes of languages

Finite words

Monoids, semigroups

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Algebras and classes of languages

Algebras are used to characterize classes of languages

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Finite words
Monoids, semigroups
Infinite words
Wilke algebras, ω-semigroups, $\circ$-algebras...

Algebras are used to characterize classes of languages

Finite words
Monoids, semigroups

Infinite words
Wilke algebras, ω-semigroups, $\circ$-algebras...

Trees
Deterministic automata, Preclones, Hyperclones, Operads,...

Graphs
HR-algebras, VR-algebras

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Algebras and classes of languages

Algebras are used to characterize classes of languages

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Finite words

Monoids, semigroups

Infinite words

Wilke algebras, ω-semigroups, $\circ$-algebras...

Trees

Deterministic automata, Preclones, Hyperclones, Operads,...

Graphs

HR-algebras, VR-algebras

Infinitely sorted algebras

$(A_n)_{n\in\mathbb{N}}$

$(A_X)_{X \text{ finite}}$

Algebras and classes of languages

Algebras are used to characterize classes of languages

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Finite words

Monoids, semigroups

Infinite words

Wilke algebras, ω-semigroups, $\circ$-algebras...

Trees

Deterministic automata, Preclones, Hyperclones, Operads,...

Graphs

HR-algebras, VR-algebras

Infinitely sorted algebras

$(A_n)_{n\in\mathbb{N}}$

$(A_X)_{X \text{ finite}}$

Problem: Hard to derive characterizations with infinitely sorted algebras

Algebras and classes of languages

Algebras are used to characterize classes of languages

Schützenberger, 1965

A regular language L is star-free if and only if its syntactic monoid is aperiodic.

Finite words

Monoids, semigroups

Infinite words

Wilke algebras, ω-semigroups, $\circ$-algebras...

Trees

Deterministic automata, Preclones, Hyperclones, Operads,...

Graphs

HR-algebras, VR-algebras

Infinitely sorted algebras

$(A_n)_{n\in\mathbb{N}}$

$(A_X)_{X \text{ finite}}$

Problem: Hard to derive characterizations with infinitely sorted algebras

Objective: characterize classes that can be naturally defined using infinitely sorted algebras

Infinitely sorted tree algebras: FT_Σ-algebras

Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

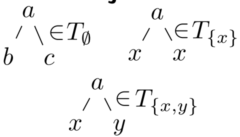
$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Infinitely sorted tree algebras: FT_Σ-algebras

Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Objects

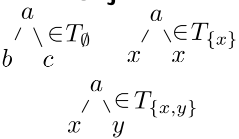


Infinitely sorted tree algebras: FT_Σ-algebras

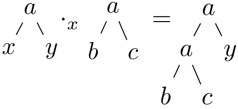
Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Objects



Substitution

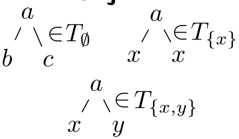


Infinitely sorted tree algebras: FT_Σ-algebras

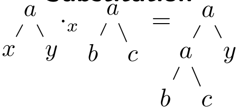
Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Objects

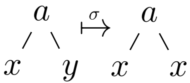


Substitution



Renaming

$$\sigma(x) = \sigma(y) = x$$

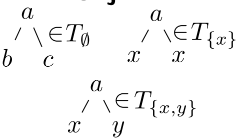


Infinitely sorted tree algebras: FT_Σ-algebras

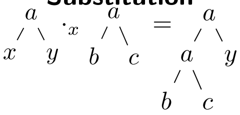
Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Objects

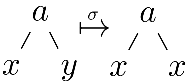


Substitution



Renaming

$$\sigma(x) = \sigma(y) = x$$



Definition (Finite Tree algebras)

A finite FT_Σ-algebra $\mathcal{A}$ consists of an infinite series of finite carrier sets A_X indexed by finite sets of variables X, together with operations:

Constants. $a(x_0, \ldots, x_{n-1})^{\mathcal{A}} \in A_{\{x_0,\ldots,x_{n-1}\}}$ for all $a \in \Sigma_n$ and variables x_i,

Substitution. $\cdot_x^{\mathcal{A}} \colon A_X \times A_Y \to A_{X\setminus\{x\}\cup Y}$ for all finite X, Y and $x \in X$,

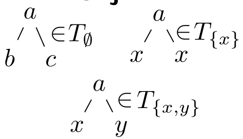
Renaming. $\mathrm{rename}^{\mathcal{A}}[\sigma] \colon A_X \to A_Y$ for all surjective maps $\sigma \colon X \to Y$.

Infinitely sorted tree algebras: FT_Σ-algebras

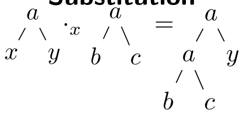
Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$

Objects

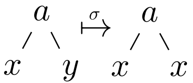


Substitution



Renaming

$$\sigma(x) = \sigma(y) = x$$



Definition (Finite Tree algebras)

A finite FT_Σ-algebra $\mathcal{A}$ consists of an infinite series of finite carrier sets A_X indexed by finite sets of variables X, together with operations:

Constants. $a(x_0, \dots, x_{n-1})^{\mathcal{A}} \in A_{\{x_0,\dots,x_{n-1}\}}$ for all $a \in \Sigma_n$ and variables x_i,

Substitution. $\cdot_x^{\mathcal{A}} \colon A_X \times A_Y \to A_{X \setminus \{x\} \cup Y}$ for all finite X, Y and $x \in X$,

Renaming. $\mathrm{rename}^{\mathcal{A}}[\sigma] \colon A_X \to A_Y$ for all surjective maps $\sigma \colon X \to Y$.

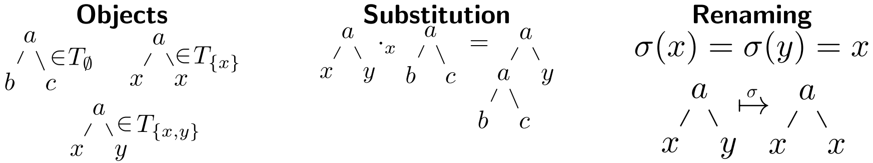
Identities? $\quad a(x,y) \cdot_y b \qquad a(x,z) \cdot_z b$

We also define morphisms, congruences...

Infinitely sorted tree algebras: FT_Σ-algebras

Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

$$T_X = \{\text{trees in which all the variables of } X \text{ appear on the leaves}\}$$



Objects
$a(b,c) \in T_\emptyset$
$a(x,x) \in T_{\{x\}}$
$a(x,y) \in T_{\{x,y\}}$
Substitution
$a(x,y) \cdot_x a(b,c) = a(a(b,c),y)$
Renaming
$\sigma(x) = \sigma(y) = x$
$a(x,y) \overset{\sigma}{\mapsto} a(x,x)$

Definition (Finite Tree algebras)

A finite FT_Σ-algebra $\mathcal{A}$ consists of an infinite series of finite carrier sets A_X indexed by finite sets of variables X, together with operations:

Constants. $a(x_0, \ldots, x_{n-1})^{\mathcal{A}} \in A_{\{x_0,\ldots,x_{n-1}\}}$ for all $a \in \Sigma_n$ and variables x_i,

Substitution. $\cdot_x^{\mathcal{A}} \colon A_X \times A_Y \to A_{X \setminus \{x\} \cup Y}$ for all finite X, Y and $x \in X$,

Renaming. $\mathrm{rename}^{\mathcal{A}}[\sigma] \colon A_X \to A_Y$ for all surjective maps $\sigma \colon X \to Y$.

Given a finite FT_Σ-algebra $\mathcal{A}$, there is a unique morphism from the free algebra to $\mathcal{A}$. It is called the evaluation morphism of $\mathcal{A}$.

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_Σ-algebra $\mathcal{A}$ if there is a set $P \subseteq A_\emptyset$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

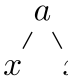
Example $L =$ The language of all trees that only contains a's and b's

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_Σ-algebra $\mathcal{A}$ if there is a set $P \subseteq A_\emptyset$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

Example $L =$ The language of all trees that only contains a's and b's



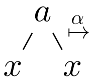
$\overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_Σ-algebra $\mathcal{A}$ if there is a set $P \subseteq A_\emptyset$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

Example $L =$ The language of all trees that only contains a's and b's



$\overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$

$$A_X = 2^\Sigma \text{ for all } X$$

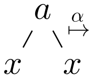
$$A \cdot_x B = A \cup B$$

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_Σ-algebra $\mathcal{A}$ if there is a set $P \subseteq A_\emptyset$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

Example $L =$ The language of all trees that only contains a's and b's



$\overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$

$$A_X = 2^\Sigma \text{ for all } X$$

$$A \cdot_x B = A \cup B$$

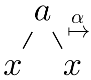
The size of $|A_X|$ is bounded (it does not depend on $|X|$).

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_Σ-algebra $\mathcal{A}$ if there is a set $P \subseteq A_\emptyset$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

Example $L =$ The language of all trees that only contains a's and b's



$\overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$

$$A_X = 2^\Sigma \text{ for all } X$$

$$A \cdot_x B = A \cup B$$

The size of $|A_X|$ is bounded (it does not depend on $|X|$).

Definition (Complexity)

Given a finite FT_Σ-algebra $\mathcal{A}$ with carrier

$$(A_X)_{X \text{ finite}}, \text{ all } A_X \text{ finite}$$

its complexity map is $c_\mathcal{A}(|X|) = |A_X|$. $(|X| = |Y| \text{ implies } |A_X| = |A_Y|)$

Languages and complexity

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite FT_{Σ}-algebra $\mathcal{A}$ if there is a set $P \subseteq A_{\emptyset}$ such that $L = \alpha^{-1}(P)$ in which α is the evaluation morphism of $\mathcal{A}$.

Long term objective

Characterize the languages recognized by algebras of

- Bounded complexity **(This talk)**
- Polynomial complexity
- Exponential complexity
- ...

Given a finite FT_{Σ}-algebra $\mathcal{A}$ with carrier

$$(A_X)_{X \text{ finite}}, \text{ all } A_X \text{ finite}$$

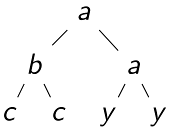
its complexity map is $c_{\mathcal{A}}(|X|) = |A_X|$. $(|X| = |Y| \text{ implies } |A_X| = |A_Y|)$

Example: The language of all trees without b's on the leftmost branch

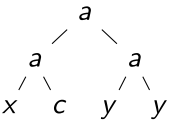
$L =$ trees without b's on the leftmost branch

Example: The language of all trees without *b*'s on the leftmost branch

$L =$ trees without b's on the leftmost branch



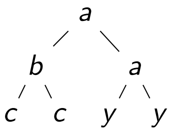
$\overset{\alpha}{\mapsto} \{a, b, c\}$



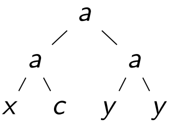
$\overset{\alpha}{\mapsto} (\{a\}, x),$

Example: The language of all trees without b's on the leftmost branch

L = trees without b's on the leftmost branch



$\xmapsto{\alpha} \{a, b, c\}$



$\xmapsto{\alpha} (\{a\}, x),$

$$A_X = 2^{\Sigma} \uplus (2^{\Sigma} \times X)$$

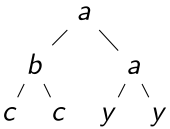
$$\mathrm{lb}(t) = \{a \in \Sigma \mid a \text{ occurs in the leftmost branch of } t\}$$

$$\alpha(t) = \begin{cases} \mathrm{lb}(t) & \text{if there is no variable on the leftmost branch of } t \\ (\mathrm{lb}(t), x) & \text{if } x \text{ is the variable on the leftmost branch of } t \end{cases}$$

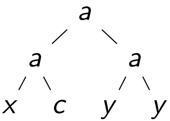
This algebra has linear complexity.

Example: The language of all trees without b's on the leftmost branch

L = trees without b's on the leftmost branch



$\overset{\alpha}{\mapsto} \{a, b, c\}$



$\overset{\alpha}{\mapsto} (\{a\}, x),$

$$A_X = 2^\Sigma \uplus (2^\Sigma \times X)$$

$$\mathrm{lb}(t) = \{a \in \Sigma \mid a \text{ occurs in the leftmost branch of } t\}$$

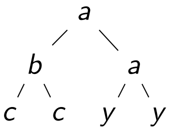
$$\alpha(t) = \begin{cases} \mathrm{lb}(t) & \text{if there is no variable on the leftmost branch of } t \\ (\mathrm{lb}(t), x) & \text{if } x \text{ is the variable on the leftmost branch of } t \end{cases}$$

This algebra has linear complexity.

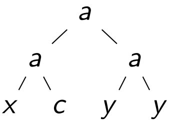
Better algebra: $A_X = X \uplus \{\bot, \top\}$

Example: The language of all trees without b's on the leftmost branch

L = trees without b's on the leftmost branch



$\overset{\alpha}{\longmapsto} \{a, b, c\}$



$\overset{\alpha}{\longmapsto} (\{a\}, x),$

$$A_X = 2^\Sigma \uplus (2^\Sigma \times X)$$

$$\mathrm{lb}(t) = \{a \in \Sigma \mid a \text{ occurs in the leftmost branch of } t\}$$

$$\alpha(t) = \begin{cases} \mathrm{lb}(t) & \text{if there is no variable on the leftmost branch of } t \\ (\mathrm{lb}(t), x) & \text{if } x \text{ is the variable on the leftmost branch of } t \end{cases}$$

This algebra has linear complexity.

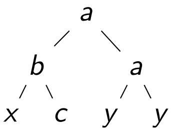
Better algebra: $A_X = X \uplus \{\bot, \top\}$(it is the syntactic algebra of L)

Example: The language of all trees with at least a b on every branch

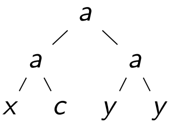
$L =$ trees with at least a b on every branch

Example: The language of all trees with at least a *b* on every branch

$L =$ trees with at least a b on every branch



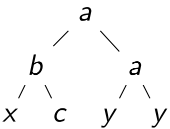
$\overset{\alpha}{\mapsto} \{y\}$



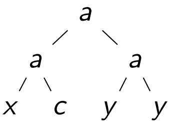
$\overset{\alpha}{\mapsto} \bot$

Example: The language of all trees with at least a *b* on every branch

$L =$ trees with at least a b on every branch



$\overset{\alpha}{\mapsto} \{y\}$



$\overset{\alpha}{\mapsto} \bot$

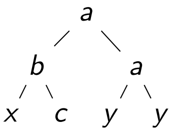
$$A_X = 2^X \uplus \{\bot\}$$

$$\mathrm{vw}_b(t) = \{x \in X \mid x \text{ occurs on a branch that has no } b\text{'s}\}$$

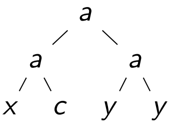
$$\alpha(t) = \begin{cases} \bot & \text{if there is branch without a } b \text{ that ends with a constant} \\ \mathrm{vw}_b(t) & \text{otherwise} \end{cases}$$

Example: The language of all trees with at least a *b* on every branch

$L =$ trees with at least a b on every branch



$\overset{\alpha}{\longmapsto} \{y\}$



$\overset{\alpha}{\longmapsto} \bot$

$$A_X = 2^X \uplus \{\bot\}$$

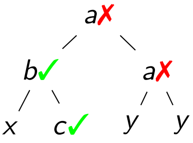
$$\mathrm{vw}_b(t) = \{x \in X \mid x \text{ occurs on a branch that has no } b\text{'s}\}$$

$$\alpha(t) = \begin{cases} \bot & \text{if there is branch without a } b \text{ that ends with a constant} \\ \mathrm{vw}_b(t) & \text{otherwise} \end{cases}$$

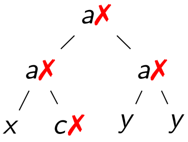
This algebra has exponential complexity and is syntactic for L

Example: The language of all trees with at least a b on every branch

$L =$ trees with at least a b on every branch



$\overset{\alpha}{\longmapsto} \{y\}$



$\overset{\alpha}{\longmapsto} \bot$

$$A_X = 2^X \uplus \{\bot\}$$

$$\mathrm{vw}_b(t) = \{x \in X \mid x \text{ occurs on a branch that has no } b\text{'s}\}$$

$$\alpha(t) = \begin{cases} \bot & \text{if there is branch without a } b \text{ that ends with a constant} \\ \mathrm{vw}_b(t) & \text{otherwise} \end{cases}$$

This algebra has exponential complexity and is syntactic for L

Example: The language of all trees with at least a b on every branch

$L =$ trees with at least a b on every branch

Languages recognized by top-down deterministic automata

All languages recognized by top-down deterministic automata are recognized by FT_Σ-algebras of exponential complexity.

$$A_X = 2^X \uplus \{\bot\}$$

$$\mathrm{vw}_b(t) = \{x \in X \mid x \text{ occurs on a branch that has no } b\text{'s}\}$$

$$\alpha(t) = \begin{cases} \bot & \text{if there is branch without a } b \text{ that ends with a constant} \\ \mathrm{vw}_b(t) & \text{otherwise} \end{cases}$$

This algebra has exponential complexity and is syntactic for L

Example: The language of all trees with at least a *b* on every branch

L = trees with at least a b on every branch

Languages recognized by top-down deterministic automata

All languages recognized by top-down deterministic automata are recognized by FT_Σ-algebras of exponential complexity.

Regular languages

A top-down nondeterministic automaton can be transformed into a FT_Σ-algebras of doubly-exponential complexity that recognizes the same language.

$$\alpha(t) = \begin{cases} \bot & \text{if there is branch without a } b \text{ that ends with a constant} \\ \mathrm{vw}_b(t) & \text{otherwise} \end{cases}$$

This algebra has exponential complexity and is syntactic for L

Main result

Characterization theorem

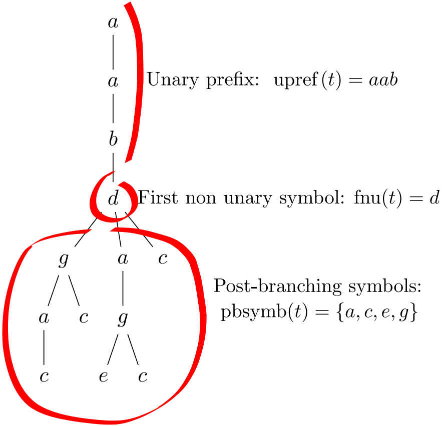
A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.


a
a
b
d
g
a
c
a
c
g
c
e
c
Unary prefix: upref $(t) = aab$
First non unary symbol: fnu$(t) = d$
Post-branching symbols:
pbsymb$(t) = \{a, c, e, g\}$

Bounded branching: $\exists k$ all trees in K have at most k branches

Main result

Characterization theorem

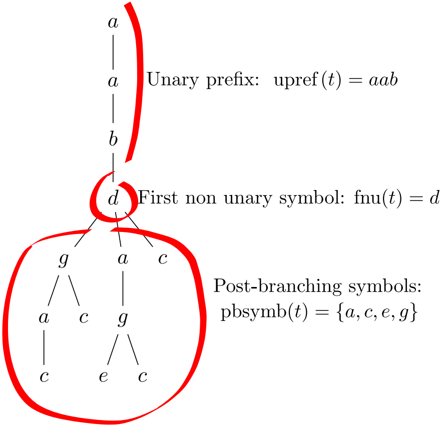
A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

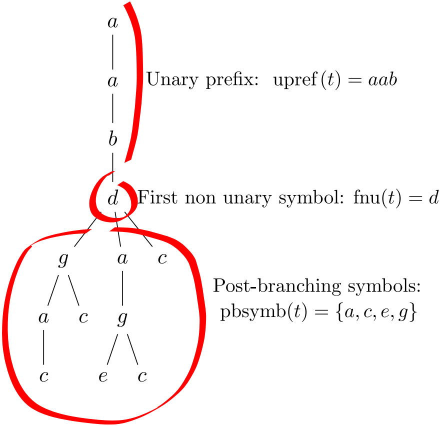

a
a
b
Unary prefix: $\mathrm{upref}(t) = aab$
d
First non unary symbol: $\mathrm{fnu}(t) = d$
g
a
c
a
c
g
c
e
c
Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Easy direction: any Boolean combination of a.-d. is recognized by an FT_Σ-algebra of bounded complexity.

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}\,(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

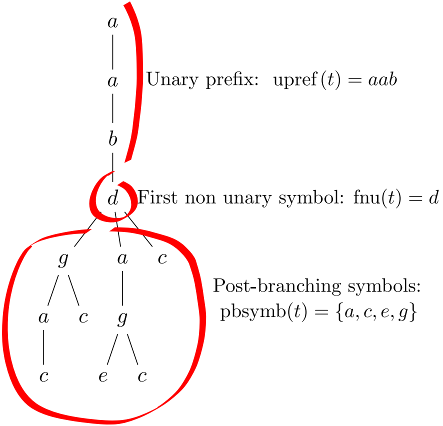
Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols:
$\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

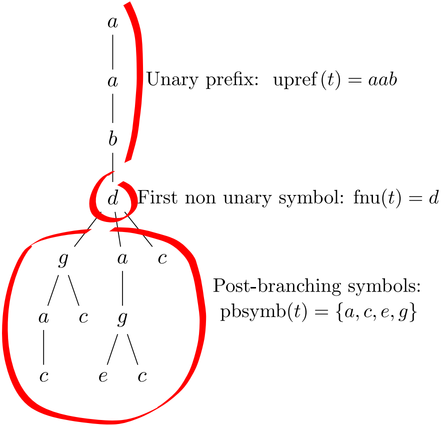
$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

$$\alpha_1(t) = \varphi(\mathrm{upref}(t))$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

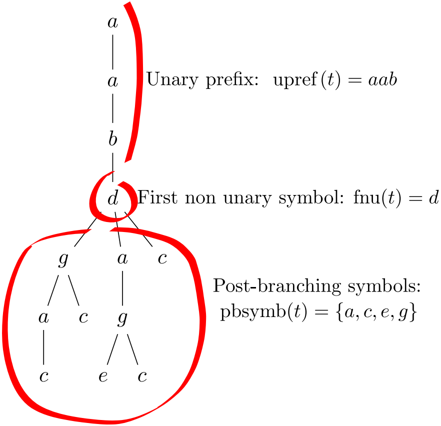
$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

$$\alpha_1(t) = \varphi(\mathrm{upref}(t))$$
$$\alpha_2(t) = \{\text{letters of } \mathrm{upref}(t)\}$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}\,(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

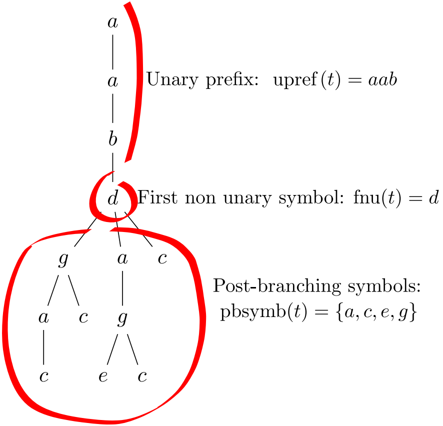
$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

$$\alpha_1(t) = \varphi(\mathrm{upref}(t))$$
$$\alpha_2(t) = \{\text{letters of } \mathrm{upref}(t)\}$$
$$\alpha_3(t) = \mathrm{fnu}(t)$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

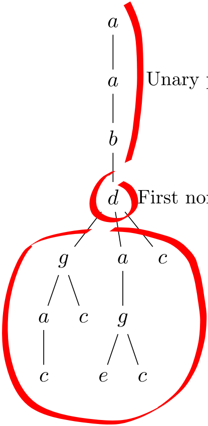
$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

$$\alpha_1(t) = \varphi(\mathrm{upref}(t))$$
$$\alpha_2(t) = \{\text{letters of } \mathrm{upref}(t)\}$$
$$\alpha_3(t) = \mathrm{fnu}(t)$$
$$\alpha_4(t) = \mathrm{pbsymb}(t)$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.



Unary prefix: $\mathrm{upref}\,(t) = aab$

First non unary symbol: $\mathrm{fnu}(t) = d$

Post-branching symbols: $\mathrm{pbsymb}(t) = \{a, c, e, g\}$

Let $\varphi\colon \Sigma_1^* \to M$ recognize L.

$$A_X = M \times 2^{\Sigma_1} \times \Sigma_{\neq 1} \times 2^{\Sigma}$$

$$\alpha_1(t) = \varphi(\mathrm{upref}(t))$$
$$\alpha_2(t) = \{\text{letters of } \mathrm{upref}(t)\}$$
$$\alpha_3(t) = \mathrm{fnu}(t)$$
$$\alpha_4(t) = \mathrm{pbsymb}(t)$$

$$A_{\{x\}} = M \times 2^{\Sigma_1} \times (\Sigma_{\neq 1} \cup \{x\}) \times 2^{\Sigma}$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.

Lemma

A regular language K of bounded branching is recognized by an algebra of bounded complexity.

Let $\mathcal{A}$ recognize K. Let k be such that trees with more than k branches never belong to K.

$$\begin{array}{cccccc}
\mathcal{A} & A_\emptyset & A_{\{x_0\}} & \dots & A_{\{x_0,\dots,x_k\}} & \dots \\
\mathcal{A}' & A_\emptyset \times \{1, \dots, k-1, \bot\} & A_{\{x_0\}} \times \{1, \dots, k-1, \bot\} & \dots & \{\bot\} & \dots
\end{array}$$

Easy direction of the characterization theorem

Lemma

The languages $\mathrm{UPref}(L)$, $\mathrm{FNU}(b)$ and $\mathrm{PBSymb}(B)$ are recognized by algebras of bounded complexity for all $b \in \Sigma_{\neq 1}$, $B \subseteq \Sigma$ and $L \subseteq \Sigma_1^*$ that is regular.

Lemma

A regular language K of bounded branching is recognized by an algebra of bounded complexity.

Let $\mathcal{A}$ recognize K. Let k be such that trees with more than k branches never belong to K.

$$\begin{array}{cccccc} \mathcal{A} & A_\emptyset & A_{\{x_0\}} & \dots & A_{\{x_0,\dots,x_k\}} & .. \\ \mathcal{A}' & A_\emptyset \times \{1,...,k-1,\bot\} & A_{\{x_0\}} \times \{1,...,k-1,\bot\} & \dots & \{\bot\} & .. \end{array}$$

Lemma

A Boolean combination of FT_Σ-algebras of bounded complexity has bounded complexity.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.
b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.
c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.
d. A regular language K of bounded branching.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations for large X.

   The converse is also true.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations for large X.
   The converse is also true.
2. For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then $\mathcal{A}$ does not distinguish between s and t.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations for large X.
   The converse is also true.
2. For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then $\mathcal{A}$ does not distinguish between s and t.
3. A language recognized by an algebra of bounded complexity is a Boolean combination of a.-d.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations for large X.

   The converse is also true.

2. For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then $\mathcal{A}$ does not distinguish between s and t.

3. A language recognized by an algebra of bounded complexity is a Boolean combination of a.-d.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned} \varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X) \\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma] \end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned} \varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X) \\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma] \end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma, folklore: $\mathrm{Ker}(\varphi_X)$ may only be $\mathbf{Sym}(X)$, $\mathbf{Alt}(X)$ or $\{\mathrm{id}_X\}$ whenever $|X| \geq 5$.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned}\varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X) \\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma]\end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma, folklore: $\mathrm{Ker}(\varphi_X)$ may only be $\mathbf{Sym}(X)$, $\mathbf{Alt}(X)$ or $\{\mathrm{id}_X\}$ whenever $|X| \geq 5$.
Lemma 1a: In a syntactic algebra, $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned} \varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X) \\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma] \end{aligned}$$

Lemma 1 (Invariance under permutations)

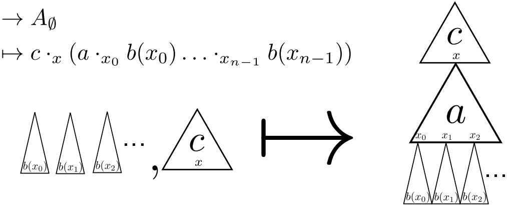
A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma, folklore: $\mathrm{Ker}(\varphi_X)$ may only be $\mathbf{Sym}(X)$, $\mathbf{Alt}(X)$ or $\{\mathrm{id}_X\}$ whenever $|X| \geq 5$.
Lemma 1a: In a syntactic algebra, $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.
Lemma 1b: In a syntactic algebra of bounded complexity, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.

Syntactic FT_Σ-algebras

Let $\mathcal{A}$ be some FT_Σ-algebra and let $X = \{x_0, ..., x_{n-1}\}$ be a finite set of variables. Define for all $a \in A_X$

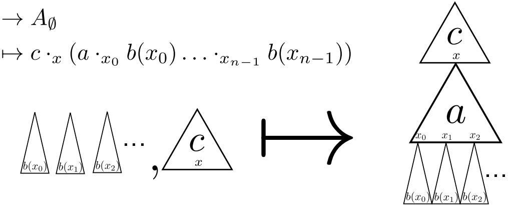
$$\begin{aligned}\langle a\rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\rightarrow A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1}))\end{aligned}$$



Syntactic FT_Σ-algebras

Let $\mathcal{A}$ be some FT_Σ-algebra and let $X = \{x_0, ..., x_{n-1}\}$ be a finite set of variables. Define for all $a \in A_X$

$$\begin{aligned} \langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1})) \end{aligned}$$



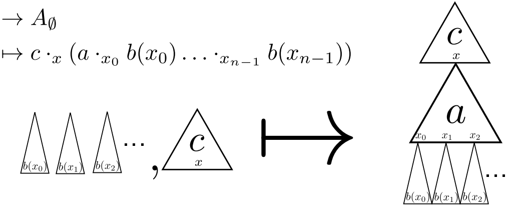
Lemma

If $\mathcal{A}$ is a *syntactic algebra* then $a = b$ iff $\langle a \rangle = \langle b \rangle$, for all a, b in $\mathcal{A}$.

Syntactic FT_Σ-algebras

Let $\mathcal{A}$ be some FT_Σ-algebra and let $X = \{x_0, ..., x_{n-1}\}$ be a finite set of variables. Define for all $a \in A_X$

$$\begin{aligned}\langle a\rangle\colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset\\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \dots \cdot_{x_{n-1}} b(x_{n-1}))\end{aligned}$$



Lemma

If $\mathcal{A}$ is a *syntactic algebra* then $a = b$ iff $\langle a\rangle = \langle b\rangle$, for all a, b in $\mathcal{A}$.

Corollary: A *syntactic algebra* is of complexity at most doubly-exponential:

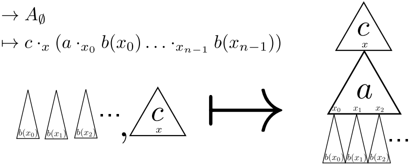
$$|A_X| \leq |A_\emptyset|^{|A_{\{x\}}||A_\emptyset|^{|X|}}$$

Lemmas 1a and 1b

Lemma 1a

In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

$$\begin{aligned} \langle a\rangle : (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \dots \cdot_{x_{n-1}} b(x_{n-1})) \end{aligned}$$



Lemmas 1a and 1b

Lemma 1a

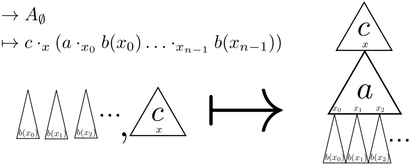
In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

$M = \max(5, |A_\emptyset| + 1)$

Suppose for the sake of contradiction that $|X| \geq M$ and $\mathrm{Ker}(\varphi_X) = \mathbf{Alt}(X)$

$$\mathrm{Im}(\varphi_X) = \{\mathrm{id}_{A_X}, \tau\}$$

$$\begin{aligned} \langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1})) \end{aligned}$$



Lemmas 1a and 1b

Lemma 1a

In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

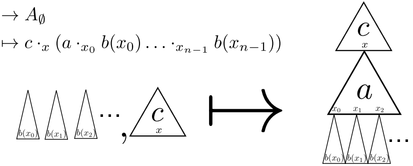
$M = \max(5, |A_\emptyset| + 1)$

Suppose for the sake of contradiction that $|X| \geq M$ and $\mathrm{Ker}(\varphi_X) = \mathbf{Alt}(X)$

$\mathrm{Im}(\varphi_X) = \{\mathrm{id}_{A_X}, \tau\}$

Prove $\mathrm{rename}^{\mathcal{A}}[t] = \mathrm{id}_{A_X}$ by showing $\langle \mathrm{rename}^{\mathcal{A}}[t](a)\rangle = \langle a \rangle$ for all $a \in A_X$.

$$\begin{aligned} \langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1})) \end{aligned}$$



Lemmas 1a and 1b

Lemma 1a

In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

$M = \max(5, |A_\emptyset| + 1)$

Suppose for the sake of contradiction that $|X| \geq M$ and $\mathrm{Ker}(\varphi_X) = \mathbf{Alt}(X)$

$\mathrm{Im}(\varphi_X) = \{\mathrm{id}_{A_X}, \tau\}$

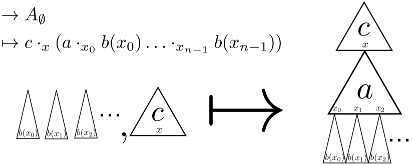
Prove $\mathrm{rename}^{\mathcal{A}}[t] = \mathrm{id}_{A_X}$ by showing $\langle \mathrm{rename}^{\mathcal{A}}[t](a) \rangle = \langle a \rangle$ for all $a \in A_X$.
Fix $a \in A_X$

$$c \in A_{\{x\}}, b \in (A_\emptyset)^X$$

$$x \neq y \text{ with } b(x) = b(y)$$

$$\begin{aligned}\langle \mathrm{rename}^{\mathcal{A}}[t](a) \rangle (b, c) &= \langle \tau(a) \rangle (b, c) \\ &= \langle \mathrm{rename}^{\mathcal{A}}[(x\ y)](a) \rangle (b, c) \\ &= \langle a \rangle (b, c)\end{aligned}$$

$$\begin{aligned}\langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1}))\end{aligned}$$



Lemmas 1a and 1b

Lemma 1a

In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \textbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Lemma 1b

In a syntactic algebra of bounded complexity, $\mathrm{Ker}(\varphi_X) = \textbf{Sym}(X)$ whenever X is large enough.

Suppose $|A_X| \leq k$ for all X and $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$

Lemmas 1a and 1b

Lemma 1a

In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Lemma 1b

In a syntactic algebra of bounded complexity, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ whenever X is large enough.

Suppose $|A_X| \leq k$ for all X and $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$

$$|X|! = |\mathrm{Im}(\varphi_X)| \leq |\mathbf{Sym}(A_X)| = |A_X|! \leq k!$$

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned}\varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X)\\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma]\end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma 1a: In a syntactic algebra, $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned}\varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X)\\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma]\end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma 1a: In a syntactic algebra, $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.
Lemma, admitted: In a syntactic algebra, either $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ for large X, or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.

Invariance under permutations

Consider for all X the group morphism induced by renaming

$$\begin{aligned} \varphi_X \colon \mathbf{Sym}(X) &\to \mathbf{Sym}(A_X) \\ \sigma &\mapsto \mathrm{rename}^{\mathcal{A}}[\sigma] \end{aligned}$$

Lemma 1 (Invariance under permutations)

A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Lemma 1a: In a syntactic algebra, $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.
Lemma, admitted: In a syntactic algebra, either $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ for large X, or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.
Lemma 1c: A syntactic algebra in which $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for every sufficiently large X is of bounded complexity.

Lemma 1c

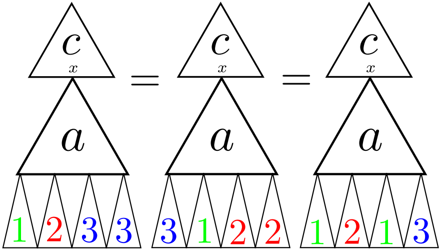
Lemma

Suppose that $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ whenever $|X| \in \{n, n-1\}$. Then for all $a \in A_X$ with $|X| = n$, and all $b, b' \in (A_\emptyset)^X$, $c \in A_{\{x\}}$

$$\langle a \rangle(b, c) = \langle a \rangle(b', c)$$

whenever $\mathrm{Im}(b) = \mathrm{Im}(b')$.

$$\langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} \to A_\emptyset$$



Lemma 1c

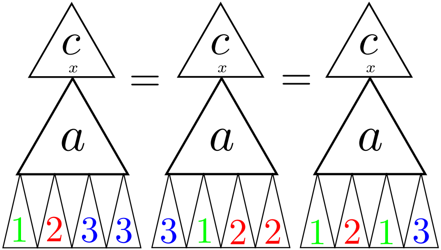
Lemma

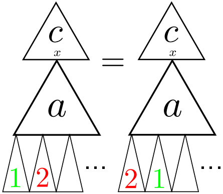
Suppose that $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ whenever $|X| \in \{n, n-1\}$. Then for all $a \in A_X$ with $|X| = n$, and all $b, b' \in (A_\emptyset)^X$, $c \in A_{\{x\}}$

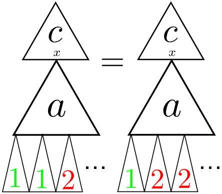
$$\langle a\rangle(b, c) = \langle a\rangle(b', c)$$

whenever $\mathrm{Im}(b) = \mathrm{Im}(b')$.

$$\langle a\rangle : (A_\emptyset)^X \times A_{\{x\}} \to A_\emptyset$$







Lemma 1c

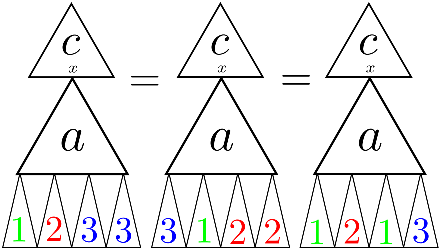
Lemma

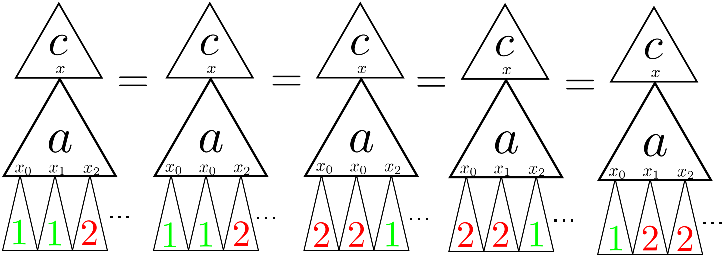
Suppose that $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ whenever $|X| \in \{n, n-1\}$. Then for all $a \in A_X$ with $|X| = n$, and all $b, b' \in (A_\emptyset)^X$, $c \in A_{\{x\}}$

$$\langle a \rangle(b, c) = \langle a \rangle(b', c)$$

whenever $\mathrm{Im}(b) = \mathrm{Im}(b')$.

$$\langle a \rangle : (A_\emptyset)^X \times A_{\{x\}} \to A_\emptyset$$





Lemma 1c

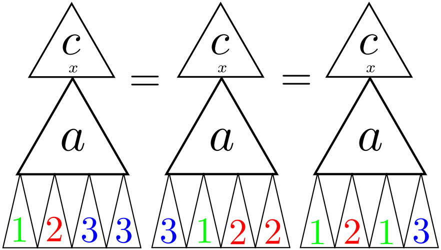
Lemma

Suppose that $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ whenever $|X| \in \{n, n-1\}$. Then for all $a \in A_X$ with $|X| = n$, and all $b, b' \in (A_\emptyset)^X$, $c \in A_{\{x\}}$

$$\langle a \rangle(b, c) = \langle a \rangle(b', c)$$

whenever $\mathrm{Im}(b) = \mathrm{Im}(b')$.

$$\langle a \rangle \colon (A_\emptyset)^X \times A_{\{x\}} \to A_\emptyset$$



Lemma 1c

A finite syntactic algebra such that $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for all sufficiently large set of variables X has bounded complexity.

For all a, $\langle a \rangle$ must be chosen in a set of at most $|A_\emptyset|^{|A_{\{x\}}|2^{|A_\emptyset|}}$ functions.
Lemma: for all a, b, $a = b$ if and only if $\langle a \rangle = \langle b \rangle$.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations for large X.

   The converse is also true.

2. For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then $\mathcal{A}$ does not distinguish between s and t.

3. A language recognized by an algebra of bounded complexity is a Boolean combination of a.-d.

Lemma 2 (Trees with many branches) 1/2

Fix a syntactic FT_Σ-algebra $\mathcal{A}$ of bounded complexity, with evaluation morphism α. Write $s \simeq_{\mathcal{A}} t$ if $\alpha(s) = \alpha(t)$.

Permutation lemma

If a tree $t(x, y)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2) \simeq_{\mathcal{A}} t(t_2, t_1)$$

Duplication lemma

If a tree $t(x, y, z)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2, t_2) \simeq_{\mathcal{A}} t(t_1, t_1, t_2)$$

Lemma 2 (Trees with many branches) 1/2

Fix a syntactic FT_Σ-algebra $\mathcal{A}$ of bounded complexity, with evaluation morphism α. Write $s \simeq_{\mathcal{A}} t$ if $\alpha(s) = \alpha(t)$.

Permutation lemma

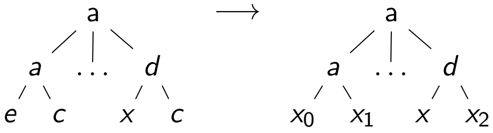
If a tree $t(x, y)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2) \simeq_{\mathcal{A}} t(t_2, t_1)$$

Duplication lemma

If a tree $t(x, y, z)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2, t_2) \simeq_{\mathcal{A}} t(t_1, t_1, t_2)$$



Lemma 2 (Trees with many branches) 1/2

Fix a syntactic FT_Σ-algebra $\mathcal{A}$ of bounded complexity, with evaluation morphism α. Write $s \simeq_\mathcal{A} t$ if $\alpha(s) = \alpha(t)$.

Permutation lemma: after repetitive and careful applications

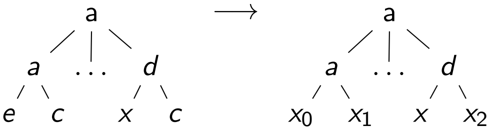
If a tree $t(t_1, t_2)$ has sufficiently many branches then

$$t(t_1, t_2) \simeq_\mathcal{A} t(t_2, t_1)$$

Duplication lemma

If a tree $t(x, y, z)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2, t_2) \simeq_\mathcal{A} t(t_1, t_1, t_2)$$



Lemma 2 (Trees with many branches) 1/2

Fix a syntactic FT_Σ-algebra $\mathcal{A}$ of bounded complexity, with evaluation morphism α. Write $s \simeq_{\mathcal{A}} t$ if $\alpha(s) = \alpha(t)$.

Permutation lemma: after repetitive and careful applications

If a tree $t(t_1, t_2)$ has sufficiently many branches then

$$t(t_1, t_2) \simeq_{\mathcal{A}} t(t_2, t_1)$$

Duplication lemma

If a tree $t(x, y, z)$ has sufficiently many branches then, for all trees t_1, t_2,

$$t(t_1, t_2, t_2) \simeq_{\mathcal{A}} t(t_1, t_1, t_2)$$

Creation lemma

If a tree t has sufficiently many branches then, for all trees $s(x, y)$ and all c, d symbols that appear in t (c constant),

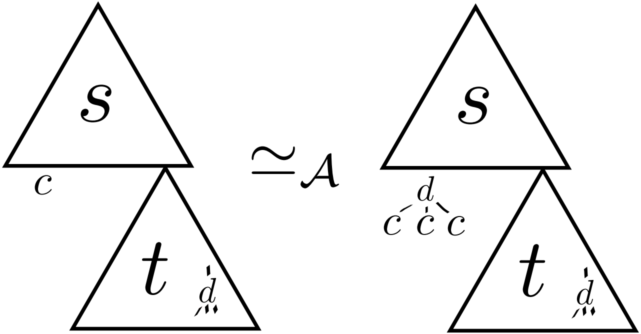
$$s(t, c) \simeq_{\mathcal{A}} s(t, d(c, ..., c))$$

Lemma 2 (Trees with many branches) 2/2

Creation lemma

If a tree t has sufficiently many branches then, for all trees $s(x, y)$ and all symbols c, d that appear in t (c constant),

$$s(t, c) \simeq_{\mathcal{A}} s(t, d(c, ..., c))$$



Lemma 2 (Trees with many branches) 2/2

Creation lemma

If a tree t has sufficiently many branches then, for all trees $s(x, y)$ and all symbols c, d that appear in t (c constant),

$$s(t, c) \simeq_{\mathcal{A}} s(t, d(c, ..., c))$$

Lemma 2

For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then

$$s \simeq_{\mathcal{A}} t$$

Lemma 2 (Trees with many branches) 2/2

Creation lemma

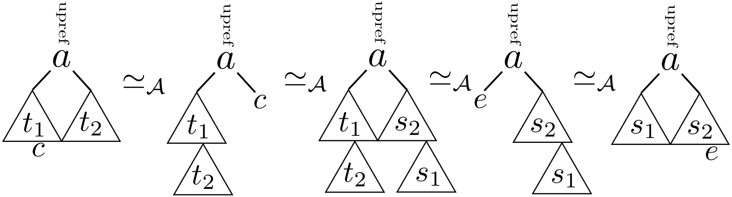
If a tree t has sufficiently many branches then, for all trees $s(x, y)$ and all symbols c, d that appear in t (c constant),

$$s(t, c) \simeq_{\mathcal{A}} s(t, d(c, ..., c))$$

Lemma 2

For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then

$$s \simeq_{\mathcal{A}} t$$



Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

1. In syntactic algebras of bounded complexity, the elements of A_X are invariant under permutations whenever X is large enough,

1'. The converse is also true,

2. For all finite trees s and t with sufficiently many branches, if $\mathrm{upref}(s) = \mathrm{upref}(t)$, $\mathrm{fnu}(s) = \mathrm{fnu}(t)$ and $\mathrm{pbsymb}(s) = \mathrm{pbsymb}(t)$ then $\mathcal{A}$ does not distinguish between s and t,

3. Express the language as a Boolean combination of a.-d.

Structure of the proof of the hard direction

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Conclusion

Complexity map: $c_{\mathcal{A}}(|X|) = |A_X|$

Bounded complexity ✓

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Conclusion

Complexity map: $c_{\mathcal{A}}(|X|) = |A_X|$

Bounded complexity ✓

Polynomial complexity ?

Exponential complexity ?

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Conclusion

Complexity map: $c_{\mathcal{A}}(|X|) = |A_X|$

Bounded complexity ✓

Polynomial complexity ?

Exponential complexity ?

A similar characterization of languages of infinite regular trees as Boolean combinations of a.-d. and other languages

Characterization theorem

A language of finite trees is recognized by an FT_Σ-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Conclusion

Complexity map: $c_{\mathcal{A}}(|X|) = |A_X|$

Bounded complexity ✓

Polynomial complexity ?

Exponential complexity ?

A similar characterization of languages of infinite regular trees as Boolean combinations of a.-d. and other languages

Orbit complexity: renaming yields an action of $\mathbf{Sym}(X)$ over A_X.

$$c^{\circ}_{\mathcal{A}}(|X|) = |A_X/\mathbf{Sym}(X)|$$

Characterization theorem

A language of finite trees is recognized by an FT_{Σ}-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Conclusion

Complexity map: $c_{\mathcal{A}}(|X|) = |A_X|$

Bounded complexity ✓

Polynomial complexity ?

Exponential complexity ?

A similar characterization of languages of infinite regular trees as Boolean combinations of a.-d. and other languages

Orbit complexity: renaming yields an action of $\mathbf{Sym}(X)$ over A_X.

$$c^{\circ}_{\mathcal{A}}(|X|) = |A_X / \mathbf{Sym}(X)|$$

Ongoing: polynomial complexity, bounded orbit complexity...

Characterization theorem

A language of finite trees is recognized by an FT_{Σ}-algebra of bounded complexity if and only if it is a Boolean combination of languages of the following kinds:

a. The language of finite trees with unary prefix in a given regular language of words $L \subseteq \Sigma_1^*$.

b. The language of finite trees with first non unary symbol b for a fixed non unary symbol b.

c. The language of finite trees with post-branching symbols B, for $B \subseteq \Sigma$.

d. A regular language K of bounded branching.

Characterization theorem for languages of regular trees

A regular language of trees is recognized by an algebra of bounded complexity if and only if it is a Boolean combination of languages of the kinds a.-d. and:

e. The language of finite trees.
f. The language of regular trees with a finite number of branches.
g. The language of regular trees that have a subtree u that is both infinite and only has symbols of arity 1, such that $u \in L$, where $L \subseteq \Sigma_1^\omega$ is regular and prefix-invariant.
h. The language of regular trees that have a subtree t that is B-dense, for some $B \subseteq \Sigma$.
   Where B-dense means that all the symbols of t belong to B, and that every symbol $b \in B$ occurs in every subtree of t.

Characterization theorem for languages of regular trees

A regular language of trees is recognized by an algebra of bounded complexity if and only if it is a Boolean combination of languages of the kinds a.-d. and:

e. The language of finite trees.
f. The language of regular trees with a finite number of branches.
g. The language of regular trees that have a subtree u that is both infinite and only has symbols of arity 1, such that $u \in L$, where $L \subseteq \Sigma_1^\omega$ is regular and prefix-invariant.
h. The language of regular trees that have a subtree t that is B-dense, for some $B \subseteq \Sigma$.
   Where B-dense means that all the symbols of t belong to B, and that every symbol $b \in B$ occurs in every subtree of t.