A Complexity Approach to Tree Algebras

Arthur Jaquard

joint work with Thomas Colcombet

Université de Paris, CNRS, IRIF, F-75006, Paris, France

June 28, 2021

INSTITUT
DE RECHERCHE
EN INFORMATIQUE
FONDAMENTALE

Université de Paris

Algebras are used to characterize classes of languages

Algebras are used to characterize classes of languages

Finite words

Monoids, semigroups

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

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

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

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 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 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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} \in T_\emptyset \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \in T_{\{x\}}$$

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \in T_{\{x,y\}}$$

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} \in T_\emptyset \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \in T_{\{x\}}$$

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \in T_{\{x,y\}}$$

Substitution

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \cdot_x \begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} = \begin{array}{ccccc} & & & a & \\ & & / & & \backslash \\ & & a & & y \\ & / & & \backslash & \\ & b & & c & \end{array}$$

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} \in T_\emptyset \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \in T_{\{x\}}$$

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \in T_{\{x,y\}}$$

Substitution

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \cdot_x \begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} = \begin{array}{ccccc} & & a & & \\ & / & & \backslash & \\ & a & & & y \\ / & & \backslash & & \\ b & & c & & \end{array}$$

Renaming

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \overset{\sigma}{\longmapsto} \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array}$$

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} \in T_\emptyset \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \in T_{\{x\}}$$

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \in T_{\{x,y\}}$$

Substitution

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \cdot_x \begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} = \begin{array}{ccccc} & & & a & \\ & & / & & \backslash \\ & & a & & y \\ & / & & \backslash & \\ & b & & c & \end{array}$$

Renaming

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \overset{\sigma}{\mapsto} \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array}$$

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 $x \in X$,

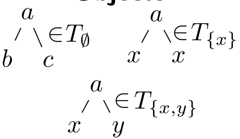
Renaming. $\mathrm{rename}^{\mathcal{A}}[\sigma] \colon A_X \to A_Y$ for all surjective maps $\sigma \colon X \to Y$.

Infinitely sorted tree algebras: FT_Σ-algebras

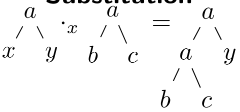
Let Σ be a ranked alphabet. The free FT_Σ-algebra has as carrier $(T_X)_{X \text{ finite}}$ where the X's are finite sets of variables.

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

Objects

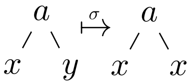


Substitution



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. $a(x_0, \ldots, x_{n-1})^{\mathcal{A}} \in A_{\{x_0,\ldots,x_{n-1}\}}$ 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 $\sigma \colon X \to Y$.

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

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} \in T_\emptyset \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \in T_{\{x\}} \qquad \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \in T_{\{x,y\}}$$

Substitution

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \cdot_x \begin{array}{ccc} & a & \\ / & & \backslash \\ b & & c \end{array} = \begin{array}{ccccc} & & & a & \\ & & / & & \backslash \\ & & a & & y \\ & / & & \backslash & \\ b & & & & c \end{array}$$

Renaming

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

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & y \end{array} \overset{\sigma}{\mapsto} \begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array}$$

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 the size of the algebra

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 contain a's and b's

Languages and the size of the algebra

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 contain a's and b's

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$$

Languages and the size of the algebra

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 contain a's and b's

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$$

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

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

Languages and the size of the algebra

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 contain a's and b's

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$$

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

$$|A_X| = 2^{|\Sigma|}).$$

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

Languages and the size of the algebra

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 contain a's and b's

$$\begin{array}{ccc} & a & \\ / & & \backslash \\ x & & x \end{array} \overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}} \qquad A_X = 2^\Sigma \text{ for all } X \qquad |A_X| = 2^{|\Sigma|}).$$

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

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

Languages and the size of the algebra

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 contain a's and b's

$$\begin{array}{c} a \\ / \quad \backslash \\ x \quad\quad x \end{array} \overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}} \qquad A_X = 2^\Sigma \text{ for all } X \qquad |A_X| = 2^{|\Sigma|}).$$

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

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

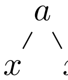
$$\begin{array}{c} a \\ / \qquad \backslash \\ b \qquad\qquad a \\ / \; \backslash \qquad / \; \backslash \\ c \quad c \qquad y \quad y \end{array} \overset{\alpha}{\mapsto} \{a, b, c\} \qquad \begin{array}{c} a \\ / \qquad \backslash \\ a \qquad\qquad a \\ / \; \backslash \qquad / \; \backslash \\ x \quad c \qquad y \quad y \end{array} \overset{\alpha}{\mapsto} (\{a\}, x),$$

Languages and the size of the algebra

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 contain a's and b's

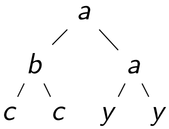

$\overset{\alpha}{\mapsto} \{a\} \in A_{\{x\}}$

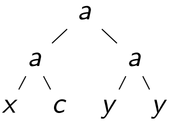
$$A_X = 2^\Sigma \text{ for all } X$$

$$|A_X| = 2^{|\Sigma|}).$$

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

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


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


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

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

$|A_X| = 2^{|\Sigma|} + 2^{|\Sigma|}|X|$ is linear in $|X|$.

Complexity

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

Complexity

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 $c_\mathcal{A}(|X|) = |A_X|$. $\quad (|X| = |Y| \text{ implies } |A_X| = |A_Y|)$

$$A_X = 2^\Sigma$$

$$|A_X| = 2^{|\Sigma|}$$

Bounded complexity

Complexity

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

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

$$|A_X| = 2^{|\Sigma|} \qquad\qquad |A_X| = 2^{|\Sigma|} + 2^{|\Sigma|}|X|$$

Bounded complexity — Linear complexity

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

$$a(b(x, c),\ a(y, y)) \overset{\alpha}{\mapsto} (\{y\}, \top)$$

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

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

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

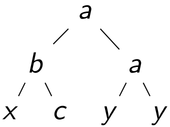
$$a(b(x, c), a(y, y)) \overset{\alpha}{\mapsto} (\{y\}, \top) \qquad a(a(x, c), a(y, y)) \overset{\alpha}{\mapsto} (\{x, y\}, \bot)$$

$$A_X = 2^X \times \{\top, \bot\}$$

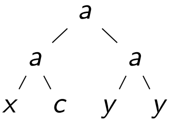
$$|A_X| = 2^{|X|} \times 2$$

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

L = trees with at least a b on every branch



$\xrightarrow{\alpha} (\{y\}, \top)$



$\xrightarrow{\alpha} (\{x, y\}, \bot)$

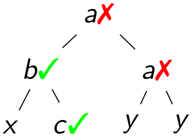
$$A_X = 2^X \times \{\top, \bot\}$$

$$|A_X| = 2^{|X|} \times 2$$

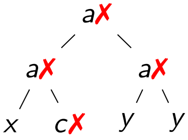
This algebra has exponential complexity.

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

L = trees with at least a b on every branch



$\xmapsto{\alpha} (\{y\}, \top)$



$\xmapsto{\alpha} (\{x, y\}, \bot)$

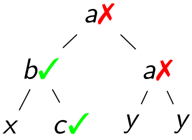
$$A_X = 2^X \times \{\top, \bot\}$$

$$|A_X| = 2^{|X|} \times 2$$

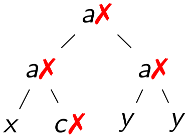
This algebra has exponential complexity.

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

L = trees with at least a b on every branch



$\overset{\alpha}{\mapsto} (\{y\}, \top)$



$\overset{\alpha}{\mapsto} (\{x, y\}, \bot)$

$$A_X = 2^X \times \{\top, \bot\}$$

$$|A_X| = 2^{|X|} \times 2$$

This algebra has exponential complexity.

Languages recognized by top-down deterministic automata

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

What complexity means

Bounded complexity

The algebra does not remember anything about the variables.

$A_X \rightsquigarrow$ the variables that appear in the tree are exactly X.

What complexity means

Bounded complexity
The algebra does not remember anything about the variables.
$A_X \rightsquigarrow$ the variables that appear in the tree are exactly X.
Polynomial complexity
$A_X = X^k \rightsquigarrow k$ variables (e.g. k branches)

What complexity means

Bounded complexity
The algebra does not remember anything about the variables.
$A_X \rightsquigarrow$ the variables that appear in the tree are exactly X.
Polynomial complexity
$A_X = X^k \rightsquigarrow k$ variables (e.g. k branches)
Exponential complexity
$A_X = k^X \rightsquigarrow$ a function from X to k (e.g. a set of variables when $k = 2$, or modulo counting if $k = \mathbb{Z}/q\mathbb{Z}$)

What complexity means

Bounded complexity
The algebra does not remember anything about the variables.
$A_X \rightsquigarrow$ the variables that appear in the tree are exactly X.
Polynomial complexity
$A_X = X^k \rightsquigarrow k$ variables (e.g. k branches)
Exponential complexity
$A_X = k^X \rightsquigarrow$ a function from X to k (e.g. a set of variables when $k = 2$, or modulo counting if $k = \mathbb{Z}/q\mathbb{Z}$)
Doubly exponential complexity

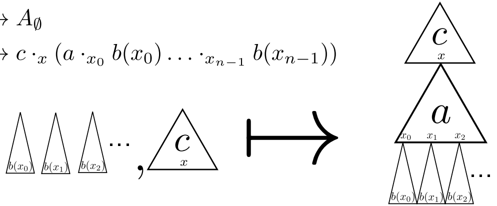
Regular languages

A top-down nondeterministic automaton can be transformed into a FT_Σ-algebras of doubly-exponential complexity that recognizes the same language.
Conversely, any language recognized by a finite FT_Σ-algebra is regular.

Syntactic 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 $a \in A_X$

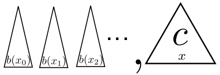
$$\begin{aligned}\langle a\rangle \colon (A_\emptyset)^X \times A_{\{x\}} &\to A_\emptyset \\ (b, c) &\mapsto c \cdot_x (a \cdot_{x_0} b(x_0) \ldots \cdot_{x_{n-1}} b(x_{n-1}))\end{aligned}$$

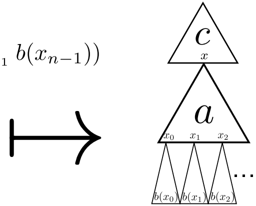


Syntactic algebras

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 $\mathcal{A}$.

Characterizing bounded complexity

What are the languages recognized by FT_{Σ}-algebras of bounded complexity?

Characterizing bounded complexity

What are the languages recognized by FT_Σ-algebras of bounded complexity?

Consider for all X the group morphism induced by renaming

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

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Invariance under permutations

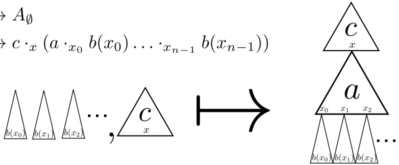
A finite syntactic FT_Σ-algebra is of bounded complexity if and only if for all sufficiently large finite set of variables X, $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$.

Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

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



Proofs

Kernel of φ_X

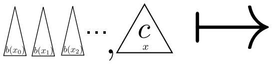
In a syntactic algebra $\mathcal{A}$, there is an integer M such that for all X of cardinal at least M, either $\mathrm{Ker}(\varphi_X) = \textbf{Sym}(X)$ or $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

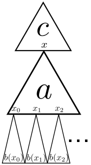
$M = \max(5, |A_\emptyset| + 1)$

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

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

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





Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

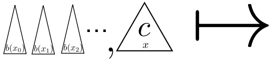
$M = \max(5, |A_\emptyset| + 1)$

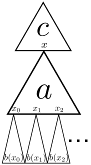
Suppose for the sake of contradiction that $|X| \geq M$ and $\mathrm{Ker}(\varphi_X) = \mathbf{Alt}(X)$

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

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 $a \in A_X$.

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





Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \mathbf{Alt}(X)$

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

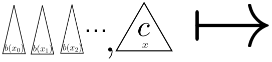
Prove $\mathrm{rename}^{\mathcal{A}}[t] = \mathrm{id}_{A_X}$ by showing $\langle \mathrm{rename}^{\mathcal{A}}[t](a)\rangle = \langle a \rangle$ for all $a \in A_X$.
Fix $a \in A_X$

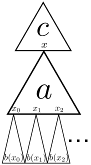
$$c \in A_{\{x\}}, b \in (A_\emptyset)^X$$

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

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

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





Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Invariance under permutations (easy direction)

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$

Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Invariance under permutations (easy direction)

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$

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

Proofs

Kernel of φ_X

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$.

Invariance under permutations (easy direction)

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 $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$

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

id and **Sym** do not alternate

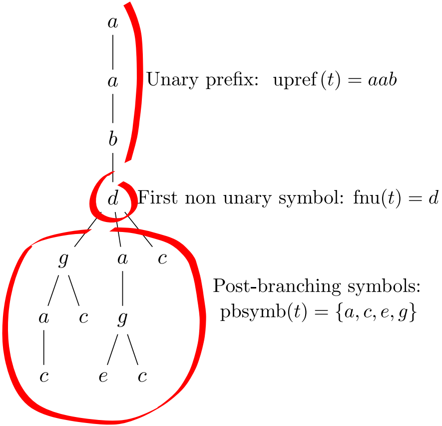
In a syntactic algebra, either $\mathrm{Ker}(\varphi_X) = \{\mathrm{id}_X\}$ for large X, or $\mathrm{Ker}(\varphi_X) = \mathbf{Sym}(X)$ for large X.

Characterisation of 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.



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

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 ?

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 ?

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.

Conclusion

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

Bounded complexity ✓

Polynomial complexity ?

Exponential complexity ?

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...

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.