A Complexity Approach to Tree Algebras: the Exponential Case (ongoing work)

Arthur Jaquard

joint work with Thomas Colcombet

Université Paris Cité, CNRS, IRIF

Highlights 2022 | July 1, 2021

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

Université de Paris

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables. The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables. The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Objects

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

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

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables.
The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Objects

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

$$\begin{array}{c} a \\ / \quad \backslash \\ x \quad\ \ x \end{array} \in T_{\{x,y\}} \qquad \begin{array}{c} a \\ / \quad \backslash \\ x \quad\ \ y \end{array} \in T_{\{x,y\}}$$

Substitution

$$\begin{array}{c} a \\ / \quad \backslash \\ x \quad\ \ y \end{array} \cdot_x \begin{array}{c} a \\ / \quad \backslash \\ b \quad\ \ c \end{array} = \begin{array}{c} a \\ / \quad \backslash \\ a \quad\ \ y \\ / \quad \backslash \qquad \\ b \quad\ \ c \qquad \end{array}$$

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables.
The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Objects

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array} \in T_{\{x,y\}} \qquad \begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \in T_{\{x,y\}}$$

Substitution

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

Renaming

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \xmapsto{\sigma} \begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array}$$

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables. The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Objects

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array} \in T_{\{x,y\}} \qquad \begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \in T_{\{x,y\}}$$

Substitution

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

Renaming

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \overset{\sigma}{\mapsto} \begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array}$$

A tree algebra $\mathcal{A} = (A_X)_{X \text{ finite}}$ is finite if all the sorts A_X are finite.

Infinitely sorted tree algebras

Let Σ be a ranked alphabet and $\mathcal{V}$ be a countably infinite set of variables. The free tree algebra has as carrier sets the $(T_X)_{X \subseteq \mathcal{V} \text{ finite}}$.

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

Objects

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array} \in T_{\{x,y\}} \qquad \begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \in T_{\{x,y\}}$$

Substitution

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

Renaming

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

$$\begin{array}{c} a \\ / \ \backslash \\ x \quad y \end{array} \overset{\sigma}{\longmapsto} \begin{array}{c} a \\ / \ \backslash \\ x \quad x \end{array}$$

A tree algebra $\mathcal{A} = (A_X)_{X \text{ finite}}$ is finite if all the sorts A_X are finite.

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite tree algebra $\mathcal{A}$ if it is the inverse image of a subset of $A_\emptyset$ under a morphism from the free tree algebra to $\mathcal{A}$.

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 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 $\mathcal{A}$.

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 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 $\mathcal{A}$.

Finite tree algebras exactly recognize the regular languages.

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 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 $\mathcal{A}$.

Finite tree algebras exactly recognize the regular languages.

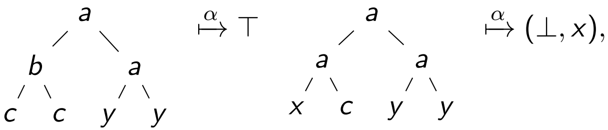
Example $L =$ trees with a b on the leftmost branch

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite 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 $\mathcal{A}$.

Finite tree algebras exactly recognize the regular languages.

Example $L =$ trees with a b on the leftmost branch



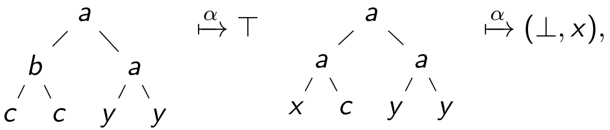
Languages and the size of the algebra

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite 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 $\mathcal{A}$.

Finite tree algebras exactly recognize the regular languages.

Example $L =$ trees with a b on the leftmost branch



$A_X = \{\top, \bot\} \uplus (\{\top, \bot\} \times X)$ $\qquad |A_X| = 2 + 2|X|$ is linear in $|X|$.

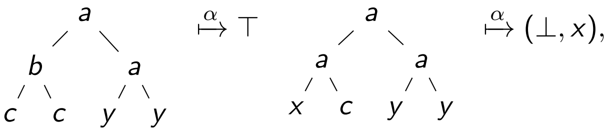
Languages and the size of the algebra

Definition (Language recognized by an algebra)

A language L of finite trees over Σ is recognized by a finite 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 $\mathcal{A}$.

Finite tree algebras exactly recognize the regular languages.

Example $L =$ trees with a b on the leftmost branch



$A_X = \{\top, \perp\} \uplus (\{\top, \perp\} \times X)$ $\qquad$ $|A_X| = 2 + 2|X|$ is linear in $|X|$.

This algebra has linear complexity.

Complexity

Definition (Complexity of an algebra)

The complexity of a finite algebra $\mathcal{A}$ is the asymptotic size of $|A_X|$ as a function of $|X|$.

Complexity

Definition (Complexity of an algebra)

The complexity of a finite algebra $\mathcal{A}$ is the asymptotic size of $|A_X|$ as a function of $|X|$.

All regular languages are recognized by algebras of doubly-exponential complexity.

Complexity

Definition (Complexity of an algebra)

The complexity of a finite algebra $\mathcal{A}$ is the asymptotic size of $|A_X|$ as a function of $|X|$.

All regular languages are recognized by algebras of doubly-exponential complexity.

Describe the languages recognized by algebras of bounded / polynomial / exponential complexity.

Complexity

Definition (Complexity of an algebra)

The complexity of a finite algebra $\mathcal{A}$ is the asymptotic size of $|A_X|$ as a function of $|X|$.

All regular languages are recognized by algebras of doubly-exponential complexity.

Describe the languages recognized by algebras of bounded / polynomial / exponential complexity.

Bounded complexity	[Colcombet, J, 2021]
Polynomial complexity	To appear at MFCS 2022
Exponential complexity	This talk (ongoing)
$\vdots$	$\vdots$
Doubly-exponential complexity	All regular languages

Complexity

Definition (Complexity of an algebra)

The complexity of a finite algebra $\mathcal{A}$ is the asymptotic size of $|A_X|$ as a function of $|X|$.

All regular languages are recognized by algebras of doubly-exponential complexity.

Describe the languages recognized by algebras of bounded / polynomial / exponential complexity.

Bounded complexity	[Colcombet, J, 2021]
Polynomial complexity	To appear at MFCS 2022
Exponential complexity	This talk (ongoing)
$\vdots$	$\vdots$
Doubly-exponential complexity	All regular languages

The objective is to identify new classes of languages and to gain a better understanding of tree algebras.

Tree algebras of exponential complexity

Exponential complexity:

$$|A_X| = \mathsf{Poly}(2^{|X|}) \qquad\qquad |A_X| = 2^{\mathsf{Poly}(|X|)}$$

Tree algebras of exponential complexity

Exponential complexity:

$$|A_X| = \mathrm{Poly}(2^{|X|}) \qquad\qquad |A_X| = 2^{\mathrm{Poly}(|X|)}$$

$\mathrm{Poly}(2^{|X|})$ is the smallest algebra complexity class closed under Boolean operations that contains $2^{|X|}$.

Tree algebras of exponential complexity

Exponential complexity:

$$|A_X| = \mathrm{Poly}(2^{|X|}) \qquad\qquad |A_X| = 2^{\mathrm{Poly}(|X|)}$$

$\mathrm{Poly}(2^{|X|})$ is the smallest algebra complexity class closed under Boolean operations that contains $2^{|X|}$.

Equivalence theorem

For a regular language of finite trees, the following are equivalent:

a. Being recognized by a finite tree algebra of exponential complexity.

Tree algebras of exponential complexity

Exponential complexity:

$$|A_X| = \mathrm{Poly}(2^{|X|}) \qquad\qquad |A_X| = 2^{\mathrm{Poly}(|X|)}$$

$\mathrm{Poly}(2^{|X|})$ is the smallest algebra complexity class closed under Boolean operations that contains $2^{|X|}$.

Equivalence theorem

For a regular language of finite trees, the following are equivalent:

a. Being recognized by a finite tree algebra of exponential complexity.
b. Being recognized by a color tree automaton.

Tree algebras of exponential complexity

Exponential complexity:

$$|A_X| = \mathrm{Poly}(2^{|X|}) \qquad\qquad |A_X| = 2^{\mathrm{Poly}(|X|)}$$

$\mathrm{Poly}(2^{|X|})$ is the smallest algebra complexity class closed under Boolean operations that contains $2^{|X|}$.

Equivalence theorem

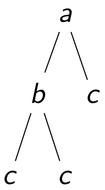
For a regular language of finite trees, the following are equivalent:

a. Being recognized by a finite tree algebra of exponential complexity.
b. Being recognized by a color tree automaton.

What is a color tree automaton?

Color tree automaton

How to build the following tree ?



Color tree automaton

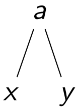
How to build the following tree ?

x

$[x]$

Color tree automaton

How to build the following tree ?

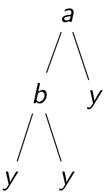


$$[x]$$

$$[\cdot_x a(x, y)]$$

Color tree automaton

How to build the following tree ?



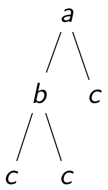
$[x]$

$[\cdot_x a(x, y)]$

$[\cdot_x b(y, y)]$

Color tree automaton

How to build the following tree ?



$[x]$

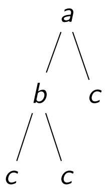
$[\cdot_x a(x, y)]$

$[\cdot_x b(y, y)]$

$[\cdot_y c]$

Color tree automaton

How to build the following tree ?



$[x]$

$[\cdot_x a(x, y)]$

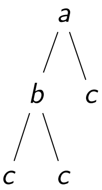
$[\cdot_x b(y, y)]$

$[\cdot_y c]$

Such a word is called a tree coding.

Color tree automaton

How to build the following tree ?



$[x]$

$[\cdot_x a(x, y)]$

$[\cdot_x b(y, y)]$

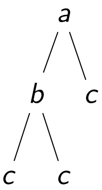
$[\cdot_y c]$

Such a word is called a tree coding.
A color tree automaton works on a tree coding. It is given by

- Q finite set of states
- K final set of colors.

Color tree automaton

How to build the following tree ?



$[x]$

$[\cdot_x a(x, y)]$

$[\cdot_x b(y, y)]$

$[\cdot_y c]$

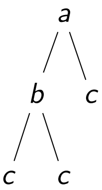
Such a word is called a tree coding.
A color tree automaton works on a tree coding. It is given by

- Q finite set of states
- K final set of colors.

partial tree $t \in T_X \quad \rightsquigarrow \quad$ state and colouring $\in Q \times K^X$

Color tree automaton

How to build the following tree ?



$[x]$

$[\cdot_x a(x, y)]$

$[\cdot_x b(y, y)]$

$[\cdot_y c]$

Such a word is called a tree coding.
A color tree automaton works on a tree coding. It is given by

- Q finite set of states
- K final set of colors.

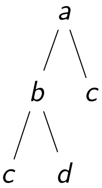
partial tree $t \in T_X$ $\rightsquigarrow$ state and colouring $\in Q \times K^X$

An automaton accepts a tree if it accepts all of its codings.

Example

L = trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad\qquad K = \{\text{red}, \mathit{green}\}$$



$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Example

L = trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad K = \{\text{red}, \mathit{green}\}$$

x

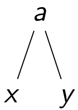
$$\bot, \{x\}, \{\}$$

$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Example

L = trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad K = \{\text{red}, \textit{green}\}$$



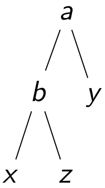
$$\bot, \{x, y\}, \{\}$$

$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Example

$L =$ trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$Q = \{\top, \bot\}$ $K = \{\text{red}, \textit{green}\}$



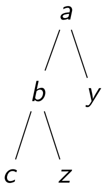
$\bot, \{y\}, \{x, z\}$

$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$

Example

L = trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad K = \{\text{red}, \mathit{green}\}$$



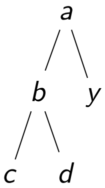
$$\bot, \{y\}, \{z\}$$

$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Example

L = trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad K = \{\text{red}, \textit{green}\}$$



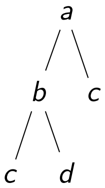
$$\top, \{y\}, \{\}$$

$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Example

$L =$ trees in which there is a d below a b, $\Sigma = \{(a,2),(b,2),(c,0),(d,0)\}$

$$Q = \{\top, \bot\} \qquad K = \{\text{red}, \textit{green}\}$$

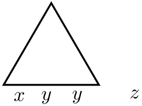


$$\top, \{\}, \{\}$$

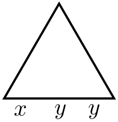
$$[x][\cdot_x a(x,y)][\cdot_x b(x,z)][\cdot_x c][\cdot_z d][\cdot_y c]$$

Expressive power of the different types of algebras

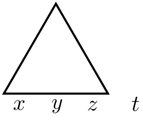
Unrestrained tree algebras $T_{\{x,y,z\}}$



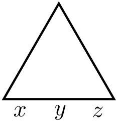
Superlinear tree algebras $T_{\{x,y\}}$



Sublinear tree algebras $T_{\{x,y,z,t\}}$

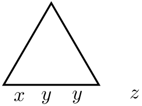


Linear tree algebras $T_{\{x,y,z\}}$

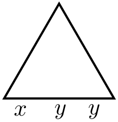


Expressive power of the different types of algebras

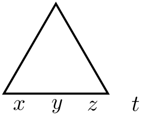
Unrestrained tree algebras $T_{\{x,y,z\}}$



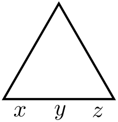
Superlinear tree algebras $T_{\{x,y\}}$



Sublinear tree algebras $T_{\{x,y,z,t\}}$



Linear tree algebras $T_{\{x,y,z\}}$



Expressive power

If we allow exponential complexity, the four variants of tree algebras have the same expressive power.

This is not the case for bounded complexity and polynomial complexity.

Extensions

- Find an algebraic definition of color tree automata.
- Relation to polynomial orbit-complexity.
- The Poly($2^{|X|}$) algebra complexity class.
- Relation to logic.

Extensions

- Find an algebraic definition of color tree automata.
- Relation to polynomial orbit-complexity.
- The Poly($2^{|X|}$) algebra complexity class.
- Relation to logic.

Questions?