Semantic reductions in random expression trees

Pablo Rotondo

LIGM, Université Gustave Eiffel

Based on joint work with
Florent Koechlin, Cyril Nicaud, Carine Pivoteau

Journée mathématiques informatique,
Rouen, 12 December, 2025.

Plan of the talk

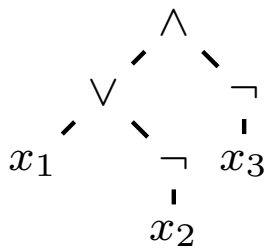
1. Random expression trees: BST and uniform
2. Semantic simplifications
3. Results for uniform random expression trees
4. Result for BST-like trees and elements of the proof
5. Conclusions

Section

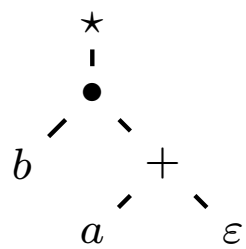
1. Random expression trees: BST and uniform
2. Semantic simplifications
3. Results for uniform random expression trees
4. Result for BST-like trees and elements of the proof
5. Conclusions

Introduction

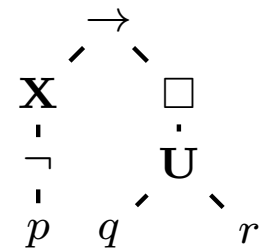
- Expression trees



$(x_1 \vee \neg x_2) \wedge \neg x_3$



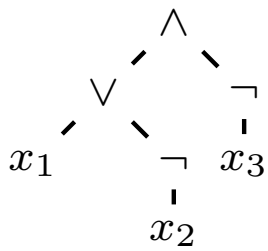
$(b \cdot (a + \varepsilon))^\star$



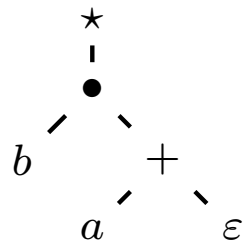
$\mathbf{X}(\neg p) \rightarrow \square(q \mathbf{U} r)$

Introduction

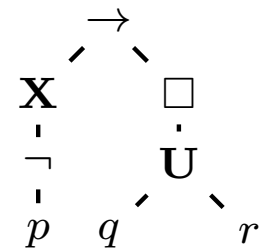
- Expression trees



$(x_1 \vee \neg x_2) \wedge \neg x_3$



$(b \cdot (a + \varepsilon))^\star$

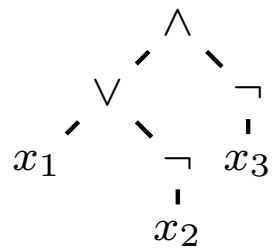


$\mathbf{X}(\neg p) \rightarrow \Box(q \mathbf{U} r)$

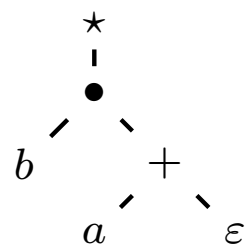
- Automated testing, benchmark testing
  - Correctness and performance of algorithms

Introduction

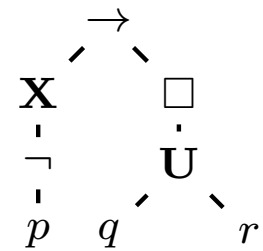
- Expression trees



$(x_1 \vee \neg x_2) \wedge \neg x_3$



$(b \cdot (a + \varepsilon))^\star$



$\mathbf{X}(\neg p) \rightarrow \Box(q \mathbf{U} r)$

- Automated testing, benchmark testing
  - Correctness and performance of algorithms
- Randomly generated input
  - Realistic distribution
  - Simple implementation, possibility of theoretical analysis.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

- If $n > 1$, pick operator[1] following (p_{op}), else pick random leaf (p_a).

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

- If $n > 1$, pick operator[1] following (p_{op}), else pick random leaf (p_a).
- if operator is binary, choose *size* of branches uniformly,

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

- If $n > 1$, pick operator[1] following (p_{op}), else pick random leaf (p_a).
- if operator is binary, choose *size* of branches uniformly,

$$\Pr_n(|T_L| = k) = \tfrac{1}{n-2}\,, \qquad k = 1, 2, \ldots, n-2\,.$$

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

- If $n > 1$, pick operator[1] following (p_{op}), else pick random leaf (p_a).
- if operator is binary, choose *size* of branches uniformly,

$$\Pr_n(|T_L| = k) = \tfrac{1}{n-2}\,, \qquad k = 1, 2, \ldots, n-2\,.$$

- build sub-trees recursively and independently!

[1] If $n = 2$, constrained to arity one.

BST-like trees: a natural construction algorithm

Problem: sample random tree of size n,

- representing expression with unary and binary operators,
- leaves correspond to constants or variables.

Idea: fix probability vectors (p_{op}) and (p_a) for operators and leaves.

- If $n > 1$, pick operator[1] following (p_{op}), else pick random leaf (p_a).
- if operator is binary, choose *size* of branches uniformly,

$$\Pr_n(|T_L| = k) = \tfrac{1}{n-2}\,, \qquad k = 1, 2, \ldots, n-2\,.$$

- build sub-trees recursively and independently!

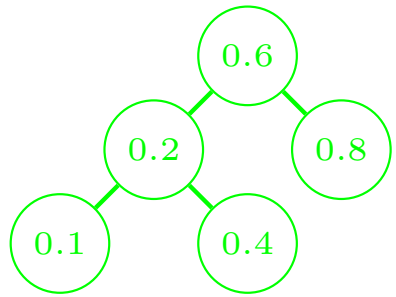
Example.

$$\Pr_n\left(\begin{array}{ccc} & \star & \\ & | & \\ & + & \\ a \swarrow & & \searrow \star \\ & & | \\ & & b \end{array}\right) = p_\star p_+ \tfrac{1}{2} p_a p_b$$

[1] If $n = 2$, constrained to arity one.

Why binary search "like" ?

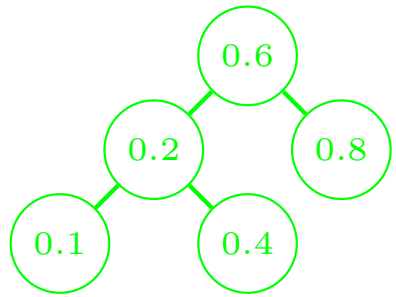
- Build BST from n random numbers $u_i \in [0, 1]$:



- With probability $\frac{1}{n}$, root corresponds to k-th ranked u_i.

Why binary search "like" ?

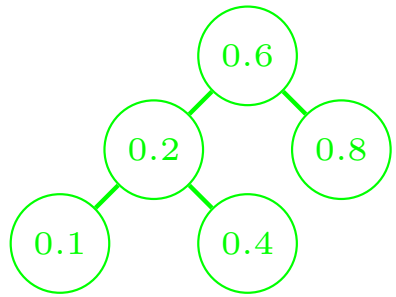
- Build BST from n random numbers $u_i \in [0, 1]$:



- With probability $\frac{1}{n}$, root corresponds to k-th ranked u_i.
- Equivalently, subtrees have sizes $|T_L| = k - 1$ and $|T_R| = n - k$.

Why binary search "like" ?

- Build BST from n random numbers $u_i \in [0, 1]$:



- With probability $\frac{1}{n}$, root corresponds to k-th ranked u_i.
- Equivalently, subtrees have sizes $|T_L| = k - 1$ and $|T_R| = n - k$.

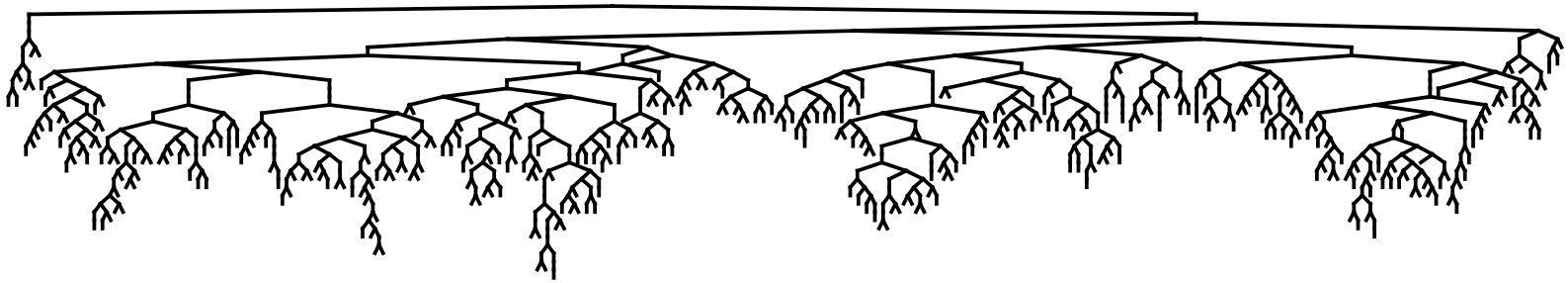
Same construction: force our subtrees to have $|T_L|, |T_R| \geq 1$, as node corresponds to binary operator.

BST-like trees: a natural construction algorithm

Code used in tool `lbtt` (from `TCS`) to draw an LTL formula:

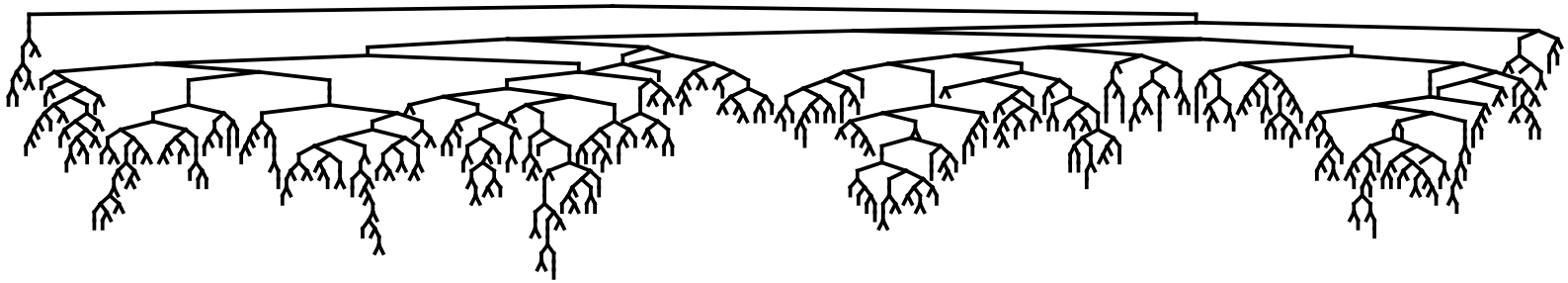
function RandomFormula(n):
if $n = 1$ **then**
  $p :=$ random symbol in $AP \cup \{\top, \bot\}$;
  return p;
else if $n = 2$ **then**
  $op :=$ random unary operator in $\{\neg, \mathbf{X}, \Box, \Diamond\}$;
  $f :=$ RandomFormula(1);
  return $op\ f$;
else
  $op :=$ random operator in $\{\neg, \mathbf{X}, \Box, \Diamond, \wedge, \vee, \rightarrow, \leftrightarrow, \mathbf{U}, \mathbf{R}\}$;
  if op *in* $\{\neg, \mathbf{X}, \Box, \Diamond\}$ **then**
    $f :=$ RandomFormula($n - 1$);
    return $op\ f$;
  else
    $x :=$ uniform integer in $[1, n - 2]$;
    $f_1 :=$ RandomFormula(x);
    $f_2 :=$ RandomFormula($n - x - 1$);
    return $(f_1\ op\ f_2)$;

BST-like trees: distribution over unary-binary trees



BST-like tree distribution is not uniform[a].

BST-like trees: distribution over unary-binary trees



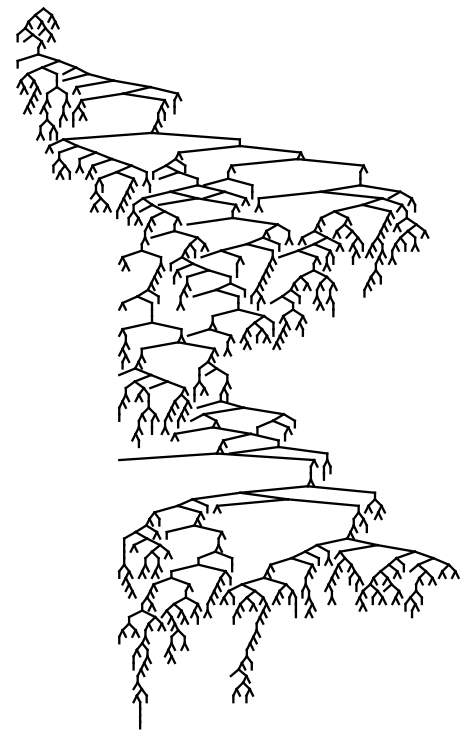
BST-like tree distribution is not uniform[a].

- Binary nodes $\approx$ balanced $\frac{n}{2}$–$\frac{n}{2}$,
  but for uniform trees

$$\mathbb{E}_n[\min(|T_L|, |T_R|)] \sim c_0\sqrt{n}\,.$$

- Expected height of different order

$$\Theta(\log n) \text{ vs } \Theta(\sqrt{n})\,.$$



[a] Tree T chosen uniformly from $\{T : |T| = n\}$.

Uniform and BST-like distributions

The uniform distribution:

- naturally maximizes entropy.
- can be sampled efficiently with a little more effort
  (Recursive method, Boltzmann samplers, Devroye's constrained GW).
- is amenable to theoretical study (Analytic Combinatorics).

Uniform and BST-like distributions

The uniform distribution:

- naturally maximizes entropy.
- can be sampled efficiently with a little more effort
  (Recursive method, Boltzmann samplers, Devroye's constrained GW).
- is amenable to theoretical study (Analytic Combinatorics).

The BST-like distribution:

- must be parametrized (prob. of operators).
- is easy to implement and very efficient.
- is often used in the automated checking of tools.

Uniform and BST-like distributions

The uniform distribution:

- naturally maximizes entropy.
- can be sampled efficiently with a little more effort
  (Recursive method, Boltzmann samplers, Devroye's constrained GW).
- is amenable to theoretical study (Analytic Combinatorics).

The BST-like distribution:

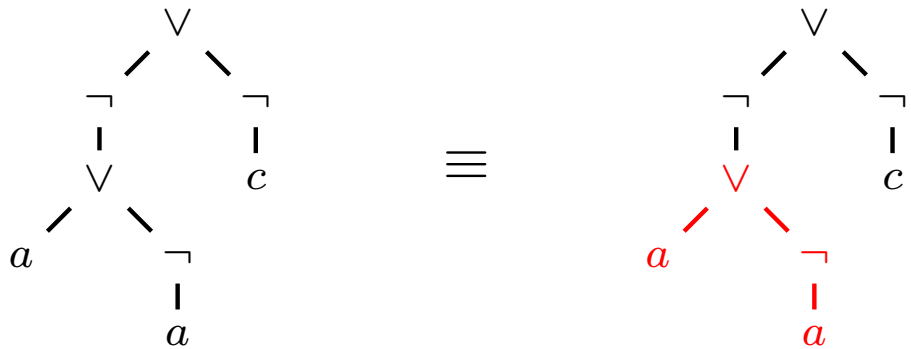
- must be parametrized (prob. of operators).
- is easy to implement and very efficient.
- is often used in the automated checking of tools.

We had previously studied *semantically* **uniform expressions**...

Section

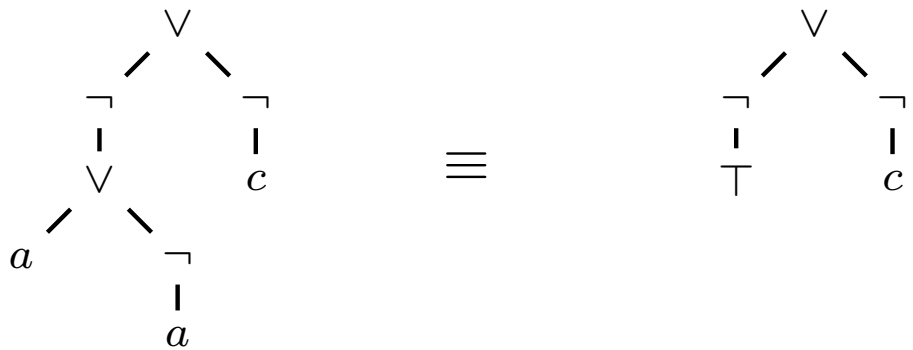
Semantic simplification

Given tree may be redundant



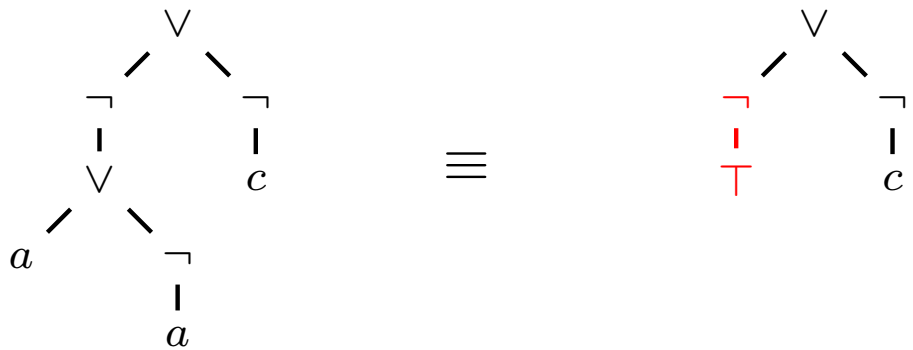
Semantic simplification

Given tree may be redundant



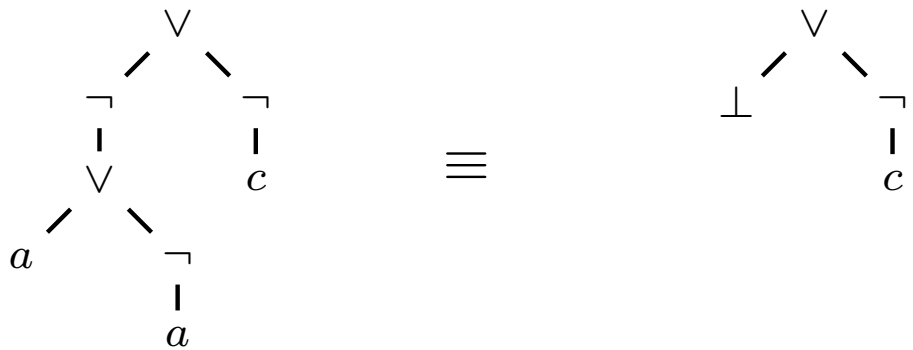
Semantic simplification

Given tree may be redundant



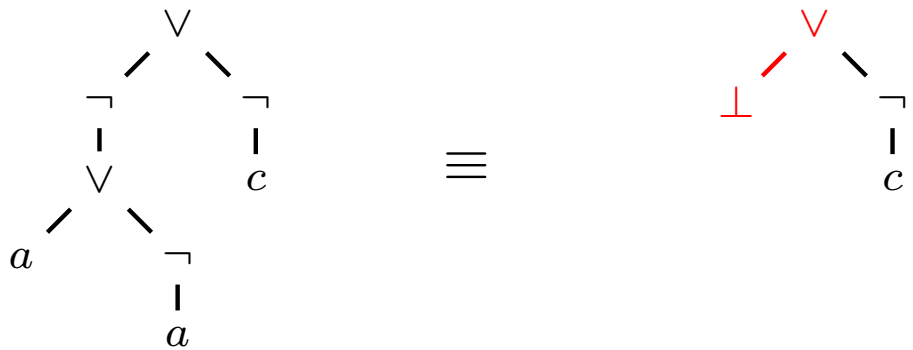
Semantic simplification

Given tree may be redundant



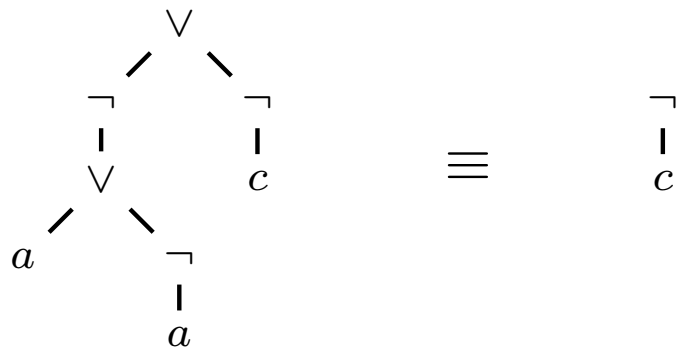
Semantic simplification

Given tree may be redundant



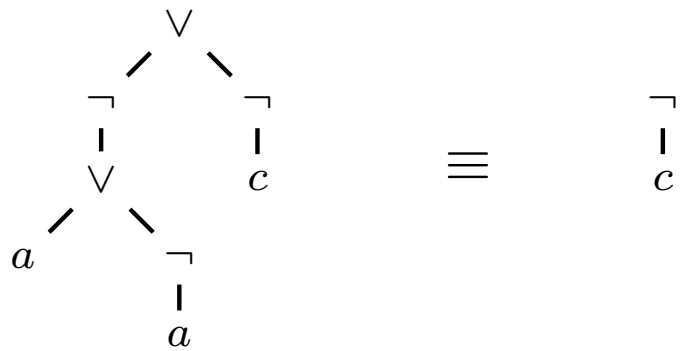
Semantic simplification

Given tree may be redundant

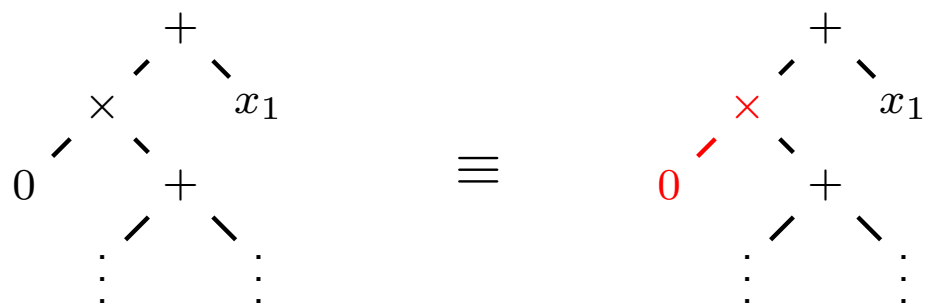


Semantic simplification

Given tree may be redundant

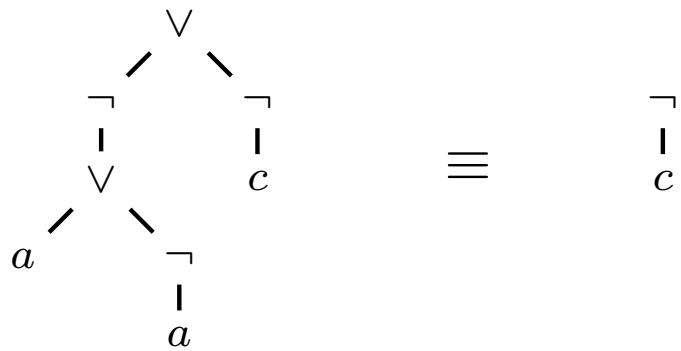


Or even *more:*

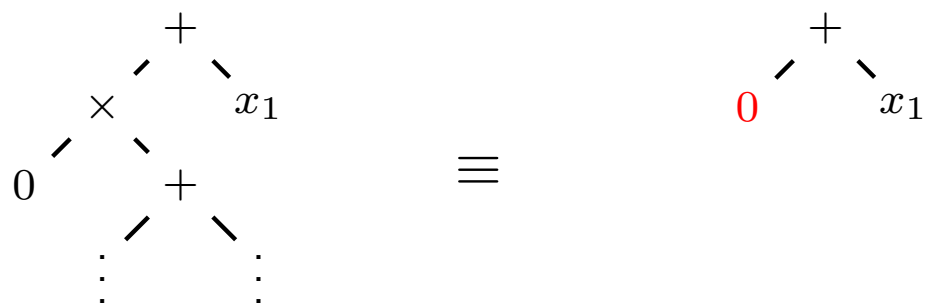


Semantic simplification

Given tree may be redundant

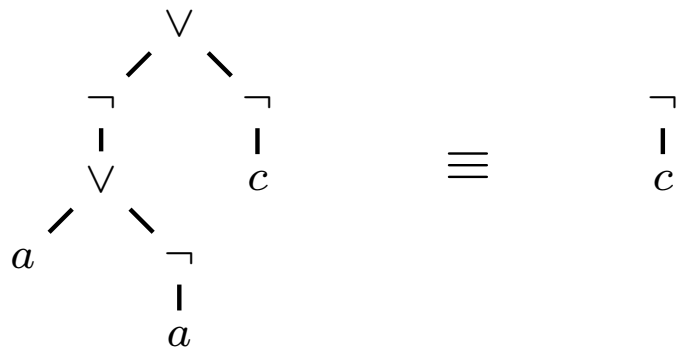


Or even *more:*

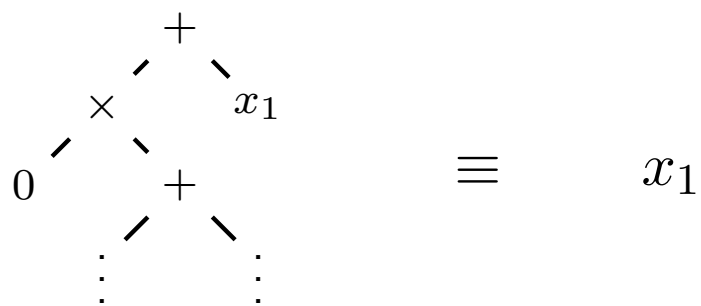


Semantic simplification

Given tree may be redundant

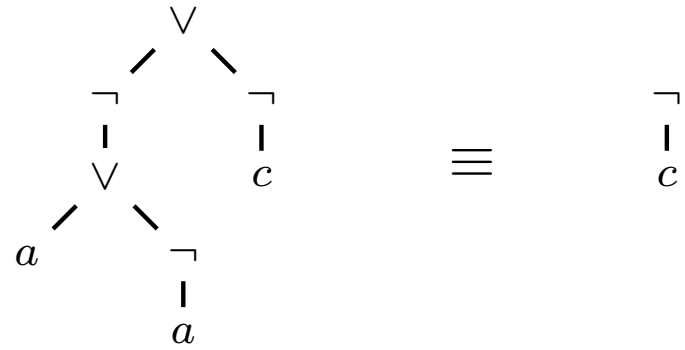


Or even *more:*

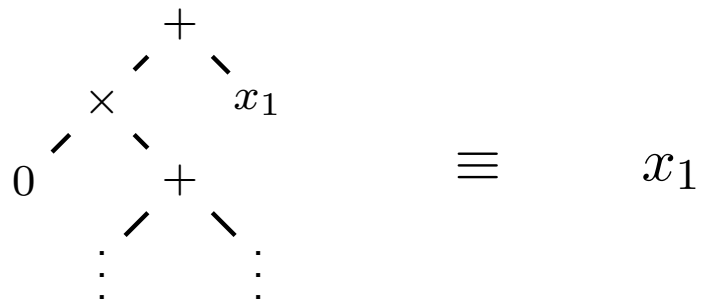


Semantic simplification

Given tree may be redundant



Or even *more:*



Question

Do semantic reductions affect the size of the random expressions?

Section

Semantic simplification on uniform trees

Universal result for uniform tree model:

Theorem (Koechlin,Nicaud,R,'20)

Expected size of reduction of uniform tree bounded, as size $\to \infty$.

Semantic simplification on uniform trees

Universal result for uniform tree model:

Theorem (Koechlin,Nicaud,R,'20)

Expected size of reduction of uniform tree bounded, as size $\to \infty$.

- Idea based on absorbing pattern $\mathcal{P}$, e.g., `false` $\wedge (\ldots) \equiv$ `false`,

Semantic simplification on uniform trees

Universal result for uniform tree model:

Theorem (Koechlin,Nicaud,R,'20)

Expected size of reduction of uniform tree bounded, as size $\to \infty$.

- Idea based on absorbing pattern $\mathcal{P}$, e.g., $\texttt{false} \wedge (\ldots) \equiv \texttt{false}$,

- Works also for systems and different arities,

Semantic simplification on uniform trees

Universal result for uniform tree model:

Theorem (Koechlin,Nicaud,R,'20)

Expected size of reduction of uniform tree bounded, as size $\to \infty$.

- Idea based on absorbing pattern $\mathcal{P}$, e.g., $\texttt{false} \wedge (\ldots) \equiv \texttt{false}$,

$$\overset{\circledast}{\underset{\mathcal{P}}{/}\backslash\underset{T}{}} \rightsquigarrow \mathcal{P} \qquad \overset{\circledast}{\underset{T}{/}\backslash\underset{\mathcal{P}}{}} \rightsquigarrow \mathcal{P}$$

- Works also for systems and different arities,

$$\texttt{Example} : \begin{cases} \mathcal{L}_R = \overset{\neg}{|}_{\mathcal{S}} + \mathcal{S}, \\ \mathcal{S} = a + b + \overset{\vee}{\underset{\mathcal{L}_R}{/}\backslash\underset{\mathcal{L}_R}{}} + \overset{\wedge}{\underset{\mathcal{L}_R}{/}\backslash\underset{\mathcal{L}_R}{}}. \end{cases}$$

Absorbing patters: simplifying the trees

Definition (Simplification, absorbing pattern)

To define an absorbing pattern we fix:

- an "operator" $\circledast \in \mathcal{A}_{ops}$ with arity $(\geq)2$,
- an expression tree $\mathcal{P}$ in the family.

Absorbing patters: simplifying the trees

Definition (Simplification, absorbing pattern)

To define an absorbing pattern we fix:

- an "operator" $\circledast \in \mathcal{A}_{ops}$ with arity $(\geq)2$,
- an expression tree $\mathcal{P}$ in the family.

Simplify by applying bottom-up the rule:

$$\begin{array}{ccc} & \circledast & \\ / & & \backslash \\ C_1 & & C_2 \end{array} \rightsquigarrow \mathcal{P}\text{, whenever } C_i = \mathcal{P} \text{ for some } i \in \{1, 2\}.$$

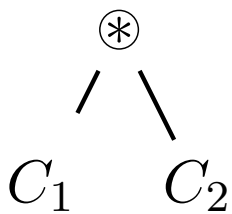
Absorbing patters: simplifying the trees

Definition (Simplification, absorbing pattern)

To define an absorbing pattern we fix:

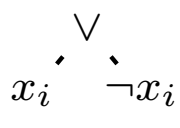
- an "operator" $\circledast \in \mathcal{A}_{ops}$ with arity $(\geq)2$,
- an expression tree $\mathcal{P}$ in the family.

Simplify by applying bottom-up the rule:

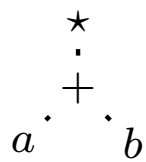


$\rightsquigarrow \mathcal{P}$, whenever $C_i = \mathcal{P}$ for some $i \in \{1, 2\}$.

- Wide variety of examples:



operator $\vee$



operator $+$

$x \mapsto 0$

operator $\times$

- Weak hypothesis: in practice often several coexisting patterns !

Regular Expressions

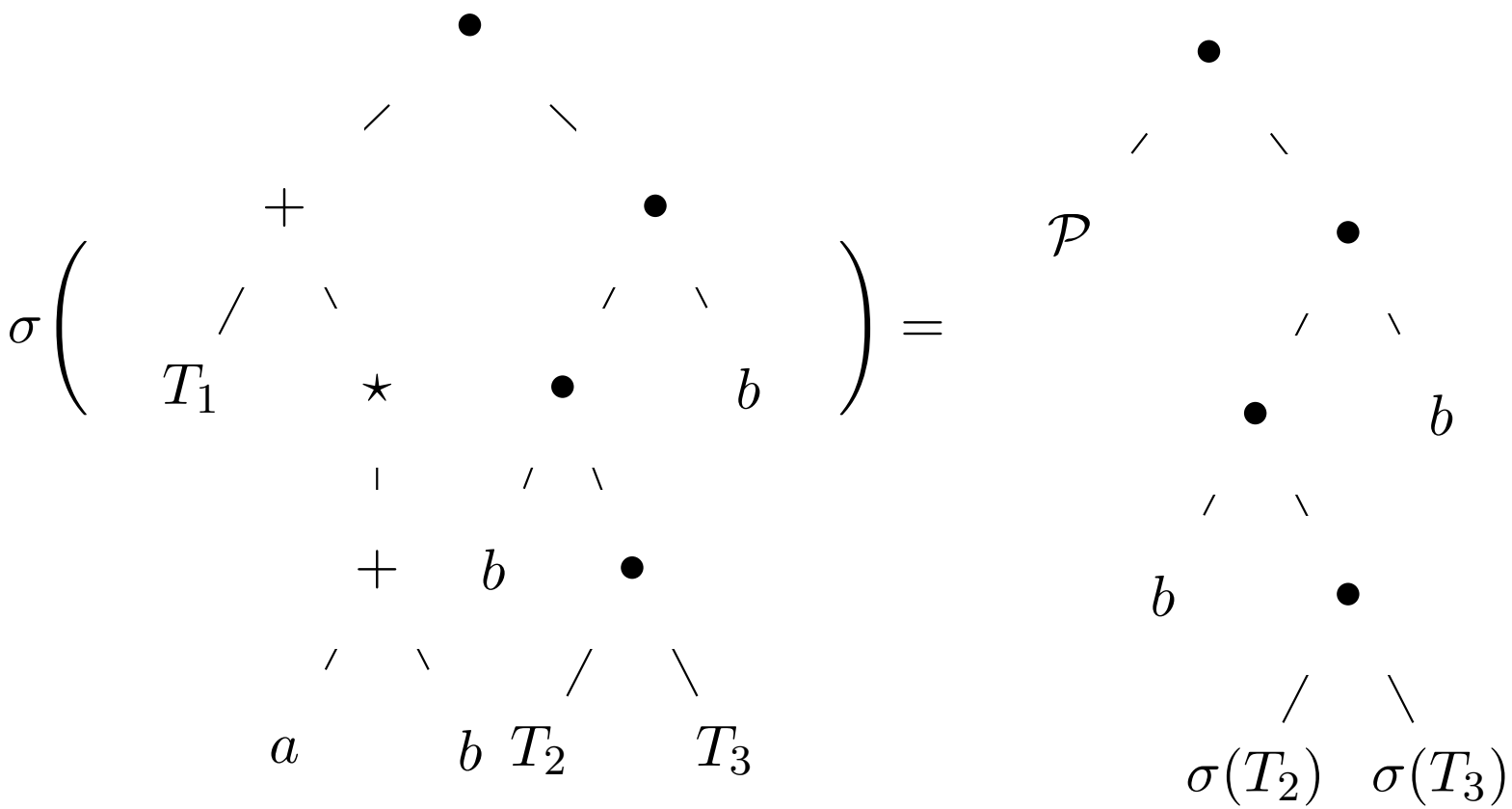
Denote by $\sigma(T) = \sigma(T, \mathcal{P}, \circledast)$ the simplified T.

Regular Expressions

Denote by $\sigma(T) = \sigma(T, \mathcal{P}, \circledast)$ the simplified T.

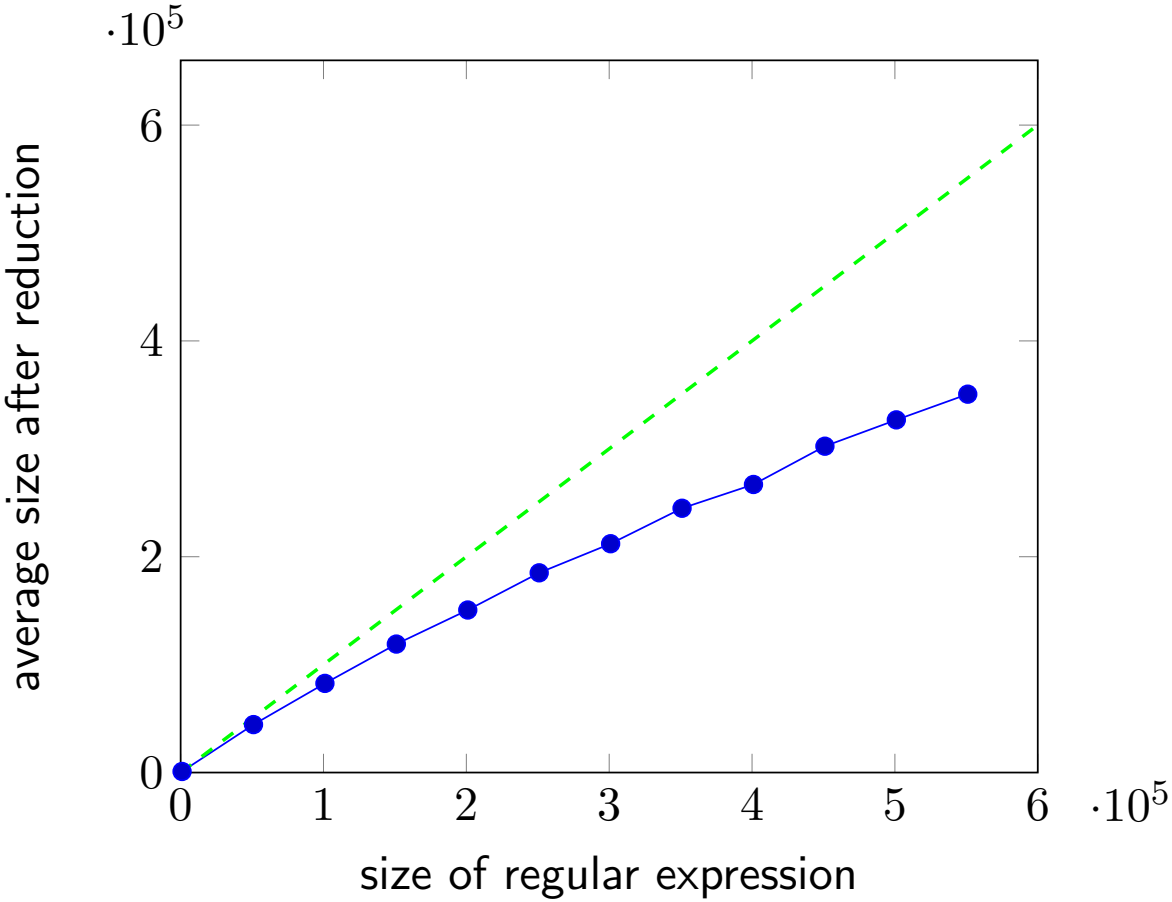
Example: regular expressions $(+, \bullet, \star)$ on two letters a, b:

$\mathcal{P} = (a + b)^\star$ absorbing for union $\circledast = +$



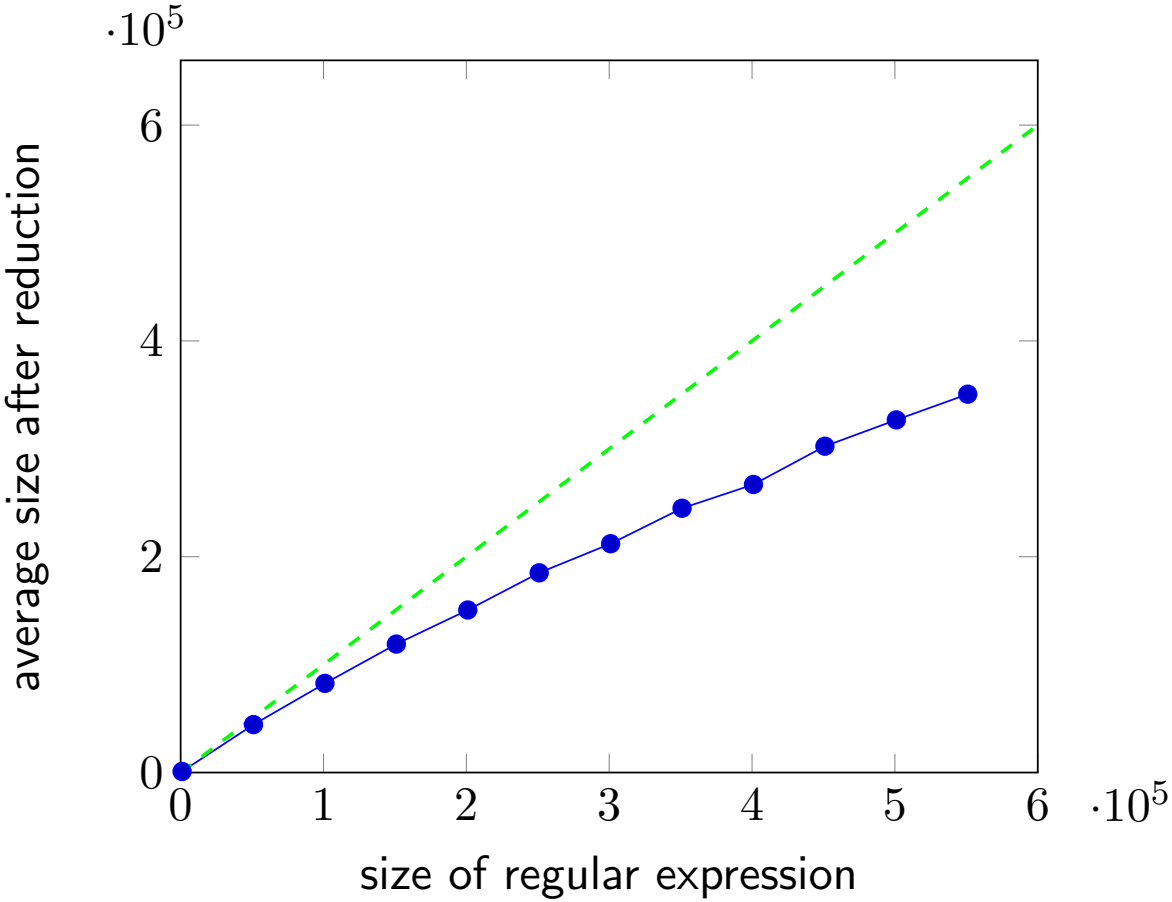
Regular Expressions II

Hidden constant $O(1)$: for the set of all regular expressions tree on two letters, the limit size after reduction is $3\ 624\ 217$.



Regular Expressions II

Hidden constant $O(1)$: for the set of all regular expressions tree on two letters, the limit size after reduction is $3\ 624\ 217$.



Question. Are uniform regular expressions useful nonetheless?

Regular Expressions III

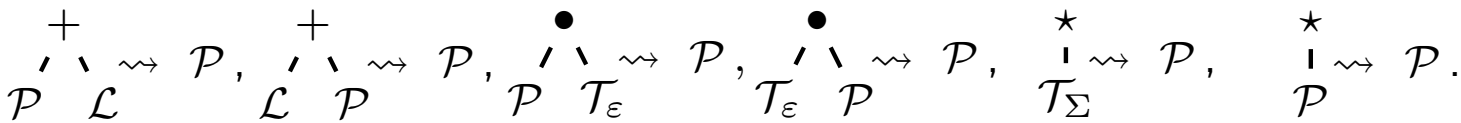
Extend idea: substitute universal subtrees[2] by tree $\mathcal{P}$ corresponding to $\Sigma^\star$.

[2] The associated language is $\Sigma^\star$, all words.

Regular Expressions III

Extend idea: substitute universal subtrees[2] by tree $\mathcal{P}$ corresponding to $\Sigma^\star$.

- We define bottom-up propagation rules

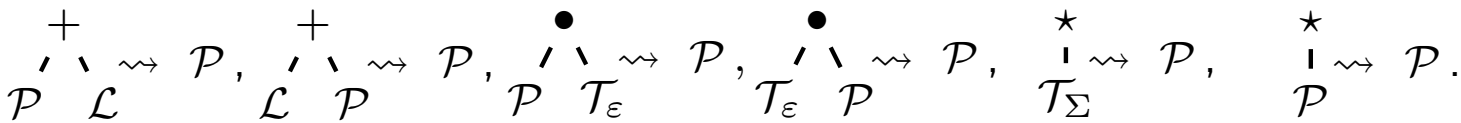


[2] The associated language is $\Sigma^\star$, all words.

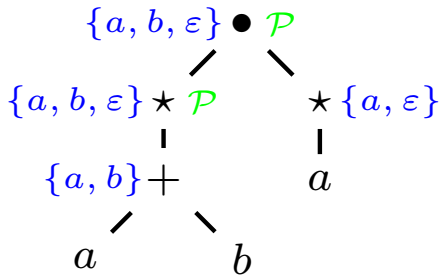
Regular Expressions III

Extend idea: substitute universal subtrees[2] by tree $\mathcal{P}$ corresponding to $\Sigma^\star$.

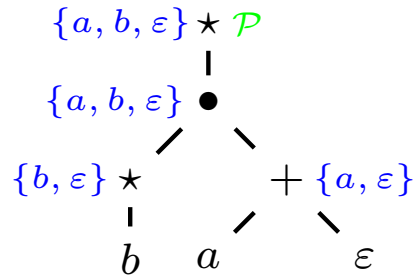
- We define bottom-up propagation rules



- Examples for $\Sigma = \{a, b\}$,



(I) : $(a + b)^\star \cdot a^\star$



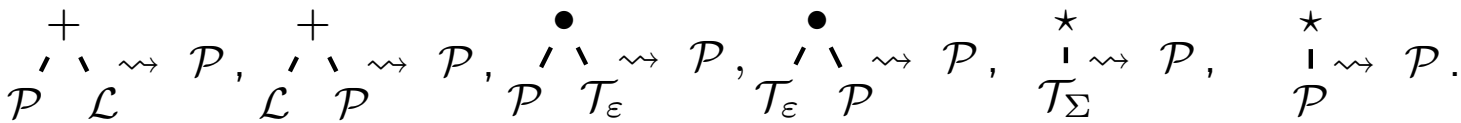
(II) : $(b^\star \cdot (a + \varepsilon))^\star$

[2] The associated language is $\Sigma^\star$, all words.

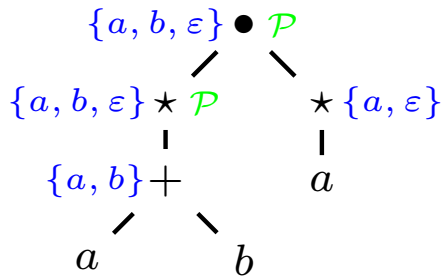
Regular Expressions III

Extend idea: substitute universal subtrees[2] by tree $\mathcal{P}$ corresponding to $\Sigma^\star$.

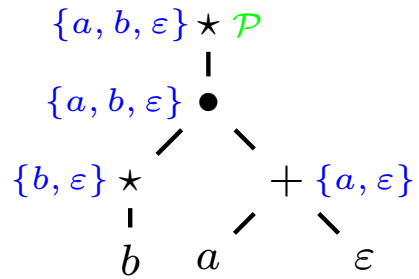
- We define bottom-up propagation rules



- Examples for $\Sigma = \{a, b\}$,



(I) : $(a + b)^\star \cdot a^\star$



(II) : $(b^\star \cdot (a + \varepsilon))^\star$

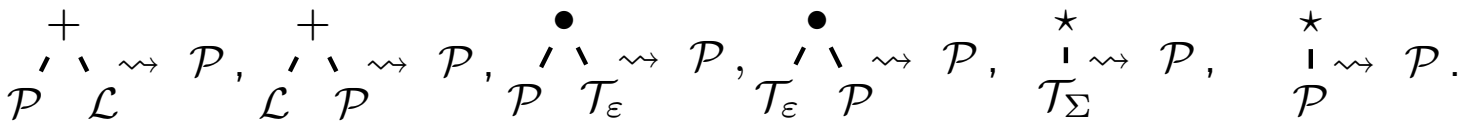
- Detection only partial: universality problem is PSPACE-complete.

[2] The associated language is $\Sigma^\star$, all words.

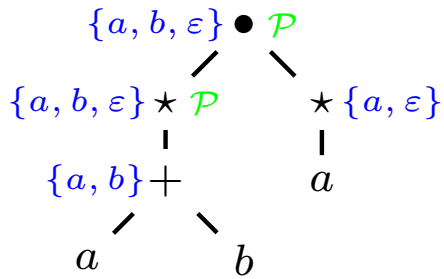
Regular Expressions III

Extend idea: substitute universal subtrees[2] by tree $\mathcal{P}$ corresponding to $\Sigma^\star$.

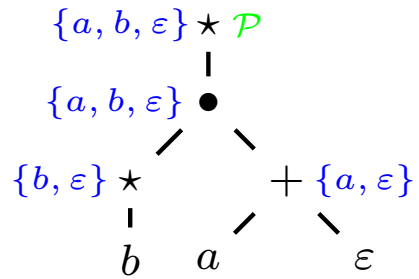
- We define bottom-up propagation rules



- Examples for $\Sigma = \{a, b\}$,



(I) : $(a + b)^\star \cdot a^\star$



(II) : $(b^\star \cdot (a + \varepsilon))^\star$

- Detection only partial: universality problem is PSPACE-complete.
- Yet average size of reduced **unifrom** expression ≈ 77.8 .

[2] The associated language is $\Sigma^\star$, all words.

Systems of Regexp [Koechlin, Pivoteau, R. '25]

We compute the probabilities of

- reducing to $\mathcal{P}$: denoted $\mathbb{P}_S^{\langle \mathcal{U} \rangle}$
- recognizing all $\overline{\Sigma} = \{\varepsilon\} \cup \Sigma$: denoted $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$.

Systems of Regexp [Koechlin, Pivoteau, R. '25]

We compute the probabilities of

- reducing to $\mathcal{P}$: denoted $\mathbb{P}_S^{\langle \mathcal{U} \rangle}$
- recognizing all $\overline{\Sigma} = \{\varepsilon\} \cup \Sigma$: denoted $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$. Note $\mathbb{P}_S^{\langle \mathcal{U} \rangle} \leq \mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$

Consider the grammar preventing the redundancies from tower of $\star$ and

1. the associativity of $+$,
2. both the associativity of $+$ and $\bullet$.

$$(1)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S^{\bullet} \\ S_p = a + b + \varepsilon + S\ S^{\bullet} + S_s^{\star} \end{cases} \qquad (2)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_p = a + b + \varepsilon + S_s^{\star} + S\ S_c^{\bullet} \\ S_c = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} \end{cases}$$

(1) $\mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.16\ldots$, $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.27\ldots$; (2) $\mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.38\ldots$, $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.51\ldots$.

Systems of Regexp [Koechlin, Pivoteau, R. '25]

We compute the probabilities of

- reducing to $\mathcal{P}$: denoted $\mathbb{P}_S^{\langle \mathcal{U} \rangle}$
- recognizing all $\overline{\Sigma} = \{\varepsilon\} \cup \Sigma$: denoted $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$. Note $\mathbb{P}_S^{\langle \mathcal{U} \rangle} \leq \mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$

Consider the grammar preventing the redundancies from tower of $\star$ and

1. the associativity of $+$,
2. both the associativity of $+$ and $\bullet$.

$$(1)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S^{\bullet} \\ S_p = a + b + \varepsilon + S\ S^{\bullet} + S_s^{\star} \end{cases} \qquad (2)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_p = a + b + \varepsilon + S_s^{\star} + S\ S_c^{\bullet} \\ S_c = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} \end{cases}$$

$$(1)\quad \mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.16\ldots,\ \mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.27\ldots\ ; \qquad (2)\quad \mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.38\ldots,\ \mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.51\ldots\ .$$

System (2) produces more universal expressions than (1) ...

Systems of Regexp [Koechlin, Pivoteau, R. '25]

We compute the probabilities of

- reducing to $\mathcal{P}$: denoted $\mathbb{P}_S^{\langle \mathcal{U} \rangle}$
- recognizing all $\overline{\Sigma} = \{\varepsilon\} \cup \Sigma$: denoted $\mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$. Note $\mathbb{P}_S^{\langle \mathcal{U} \rangle} \leq \mathbb{P}_S^{\langle \overline{\Sigma} \rangle}$

Consider the grammar preventing the redundancies from tower of $\star$ and

1. the associativity of $+$,
2. both the associativity of $+$ and $\bullet$.

$$(1)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S^{\bullet} \\ S_p = a + b + \varepsilon + S\ S^{\bullet} + S_s^{\star} \end{cases} \qquad (2)\begin{cases} S = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_s = a + b + \varepsilon + S\ S_p^{+} + S\ S_c^{\bullet} \\ S_p = a + b + \varepsilon + S_s^{\star} + S\ S_c^{\bullet} \\ S_c = a + b + \varepsilon + S_s^{\star} + S\ S_p^{+} \end{cases}$$

$$(1)\quad \mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.16\ldots,\ \mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.27\ldots\ ; \qquad (2)\quad \mathbb{P}_S^{\langle \mathcal{U} \rangle} = 0.38\ldots,\ \mathbb{P}_S^{\langle \overline{\Sigma} \rangle} = 0.51\ldots\ .$$

System (2) produces more universal expressions than (1) ...

$\Rightarrow$ we provide automatic tool to test *overabundance* of universal expressions.

Section

Zoom into the case of BST-like trees

If we draw a random BST-like tree expression of size n:

- do we have the same flaw as uniform trees?

[3] Left to right
$(p_\star, p_\bullet, p_+) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$,$(p_\star, p_\bullet, p_+) = (\frac{5}{29}, \frac{5}{29}, \frac{19}{29})$,$(p_\star, p_\bullet, p_+) = (\frac{1}{10}, \frac{1}{10}, \frac{8}{10})$

Zoom into the case of BST-like trees

If we draw a random BST-like tree expression of size n:

- do we have the same flaw as uniform trees?

$$\Longrightarrow \text{we characterize 5 regimes}$$

[3] Left to right
$(p_\star, p_\bullet, p_+) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$,$(p_\star, p_\bullet, p_+) = (\frac{5}{29}, \frac{5}{29}, \frac{19}{29})$,$(p_\star, p_\bullet, p_+) = (\frac{1}{10}, \frac{1}{10}, \frac{8}{10})$

Zoom into the case of BST-like trees

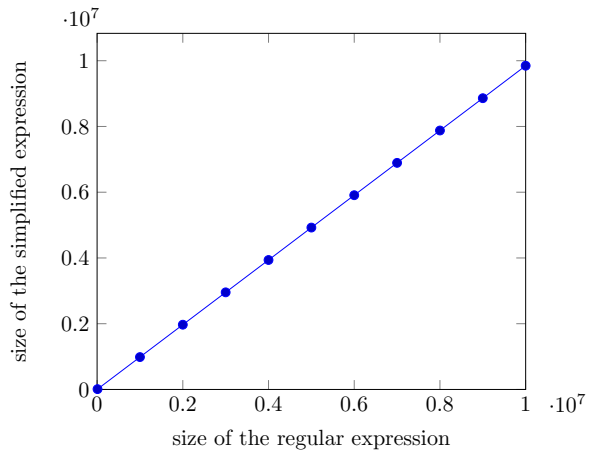
If we draw a random BST-like tree expression of size n:

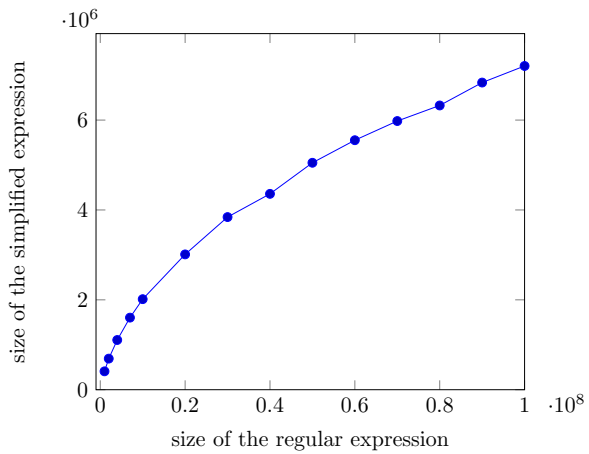
- do we have the same flaw as uniform trees?

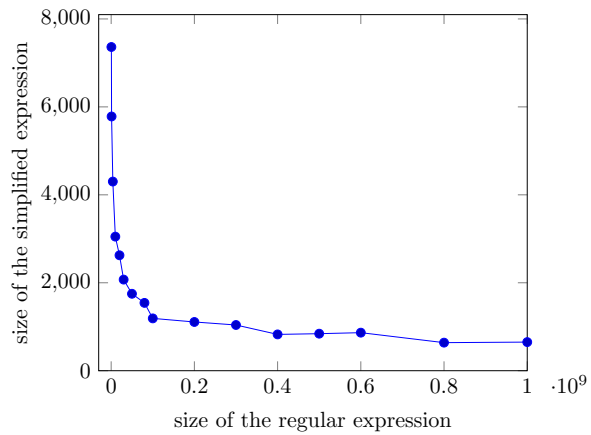
$\Longrightarrow$ we characterize 5 regimes

Experimental expected size (10 000 samples)[3] on regular expressions $(+, \bullet, \star)$ on two letters a, b:

$\mathcal{P} = (a + b)^\star$ absorbing for union $\circledast = +$







[3] Left to right
$(p_\star, p_\bullet, p_+) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}), (p_\star, p_\bullet, p_+) = (\frac{5}{29}, \frac{5}{29}, \frac{19}{29}), (p_\star, p_\bullet, p_+) = (\frac{1}{10}, \frac{1}{10}, \frac{8}{10})$

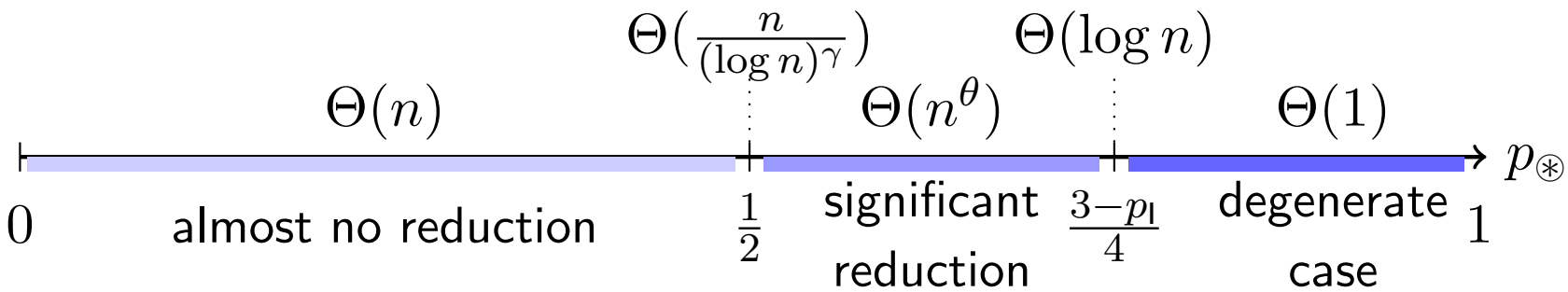
Theorem. *Consider a family of expression trees defined from unary and binary operators with an absorbing pattern $\mathcal{P}$ for an operator $\circledast$ of arity 2.*

Take the simplification consisting in inductively changing a $\circledast$-node by $\mathcal{P}$ whenever one of its children simplifies to $\mathcal{P}$.

Theorem. *Consider a family of expression trees defined from unary and binary operators with an absorbing pattern $\mathcal{P}$ for an operator $\circledast$ of arity 2.*

Take the simplification consisting in inductively changing a $\circledast$-node by $\mathcal{P}$ whenever one of its children simplifies to $\mathcal{P}$.

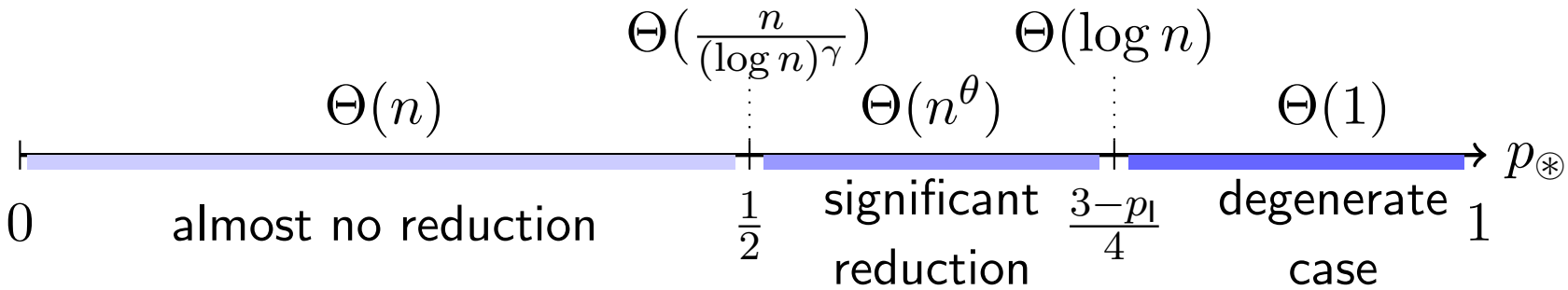
Then the expected size of the simplification of a random BST-like tree has an asymptotic behaviour given by the following cases, depending on the probability $p_\circledast$ of the absorbing operator:



Theorem. *Consider a family of expression trees defined from unary and binary operators with an absorbing pattern $\mathcal{P}$ for an operator $\circledast$ of arity 2.*

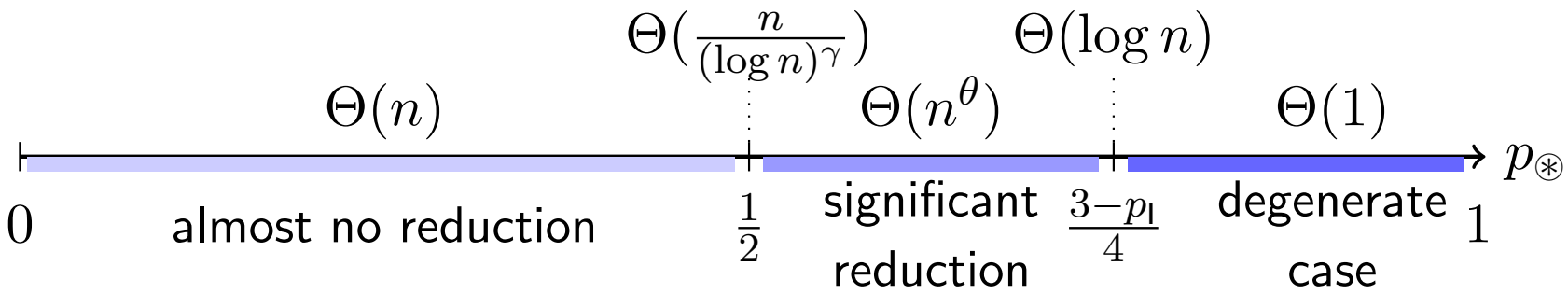
Take the simplification consisting in inductively changing a $\circledast$-node by $\mathcal{P}$ whenever one of its children simplifies to $\mathcal{P}$.

Then the expected size of the simplification of a random BST-like tree has an asymptotic behaviour given by the following cases, depending on the probability $p_\circledast$ of the absorbing operator:

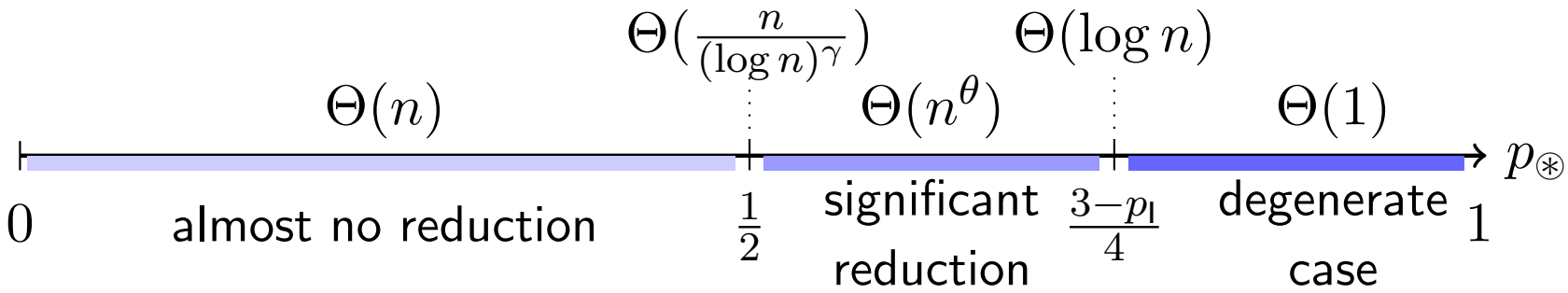


- Probability $p_\circledast$ of $\circledast$, and p_l of picking unary operator.
- Two critical points $p_\circledast = 1/2$ and $p_\circledast = (3 - p_\mathsf{l})/4$
- Regimes from no reduction $\Theta(n)$ to complete reduction $\Theta(1)$

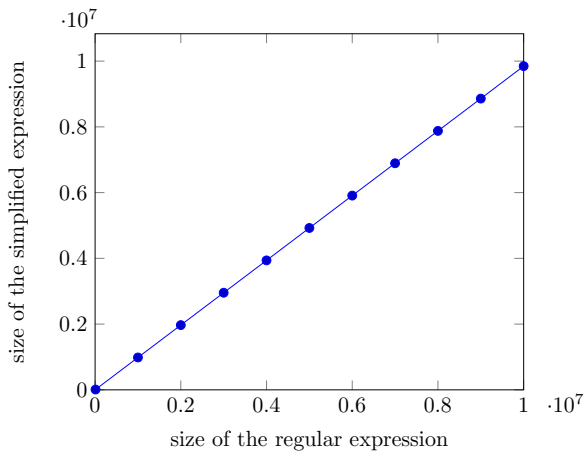
The main regimes experimentally

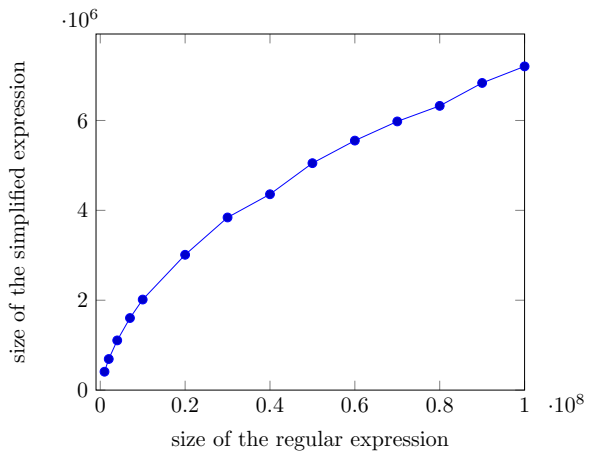


The main regimes experimentally



Experimental plots (10 000 samples)for regular expressions on two letters a, b: $\mathcal{P} = (a + b)^\star$ absorbing for union $\circledast = +$





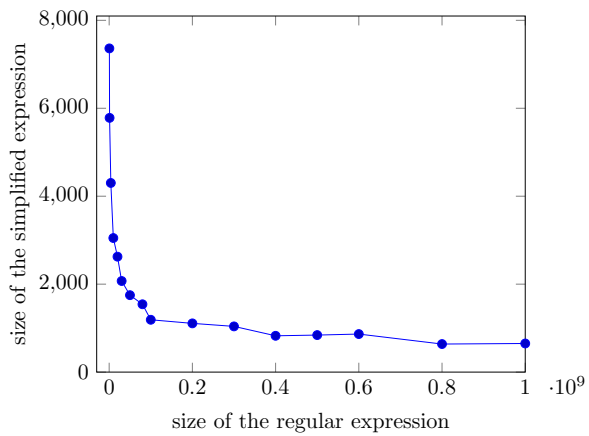


Figure: Left to right: linear ($p_+ = p_\star = p_\cdot = \frac{1}{3}$), sublinear ($p_+ = \frac{19}{29}$, $p_\star = p_\cdot = \frac{5}{29}$) and constant ($p_+ = \frac{8}{10}$, $p_\star = p_\cdot = \frac{1}{10}$).

Scheme of the proof: steps from Analytic combinatorics

We employ Analytic Combinatorics to study the expectation,

- Ordinary generating function

$$E(z) := \sum_{n=0}^{\infty} e_n z^n \,,$$

encodes sequence $e_n := \mathbb{E}_n[\#\sigma(T)]$.

Scheme of the proof: steps from Analytic combinatorics

We employ Analytic Combinatorics to study the expectation,

- Ordinary generating function

$$E(z) := \sum_{n=0}^{\infty} e_n z^n \,,$$

encodes sequence $e_n := \mathbb{E}_n[\#\sigma(T)]$.

- **Symbolic Step.** We find a formal equation describing $E(z)$.

Here this will be an *ordinary differential equation*

$$E'(z) = B(z) + C(z) \cdot E(z) \,.$$

Scheme of the proof: steps from Analytic combinatorics

We employ Analytic Combinatorics to study the expectation,

- Ordinary generating function

$$E(z) := \sum_{n=0}^{\infty} e_n z^n \,,$$

encodes sequence $e_n := \mathbb{E}_n[\#\sigma(T)]$.

- **Symbolic Step.** We find a formal equation describing $E(z)$.

Here this will be an *ordinary differential equation*

$$E'(z) = B(z) + C(z) \cdot E(z) \,.$$

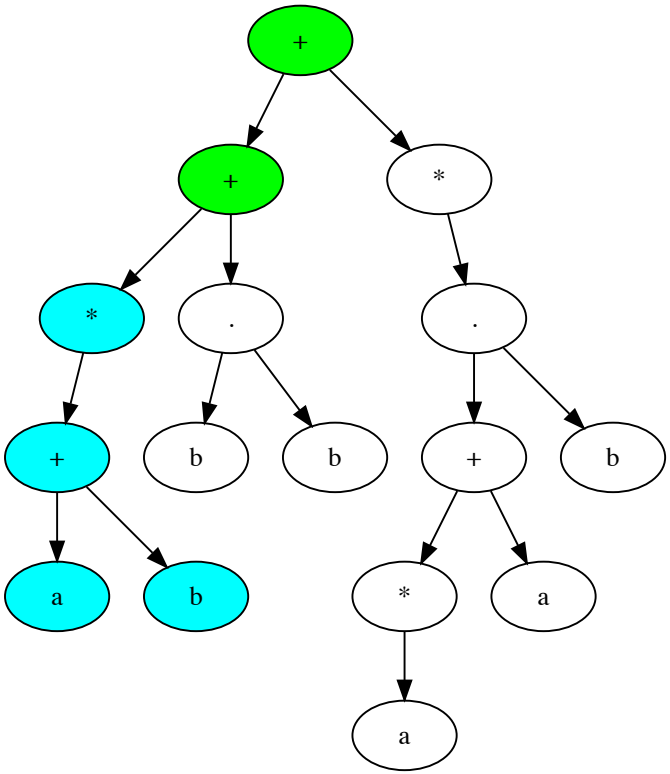
- **Analytic Step.** We look at $E(z)$ over the complex $z \in \mathbb{C}$.

A *Transfer Theorem* links the behaviour of $E(z)$ at its dominant singularity to asymptotics of $e_n \Rightarrow$ Study singularities

$$E(z) \sim_{z\to 1} \lambda(1-z)^{-\alpha} \Longrightarrow e_n \sim \lambda n^{\alpha-1}/\Gamma(\alpha)$$

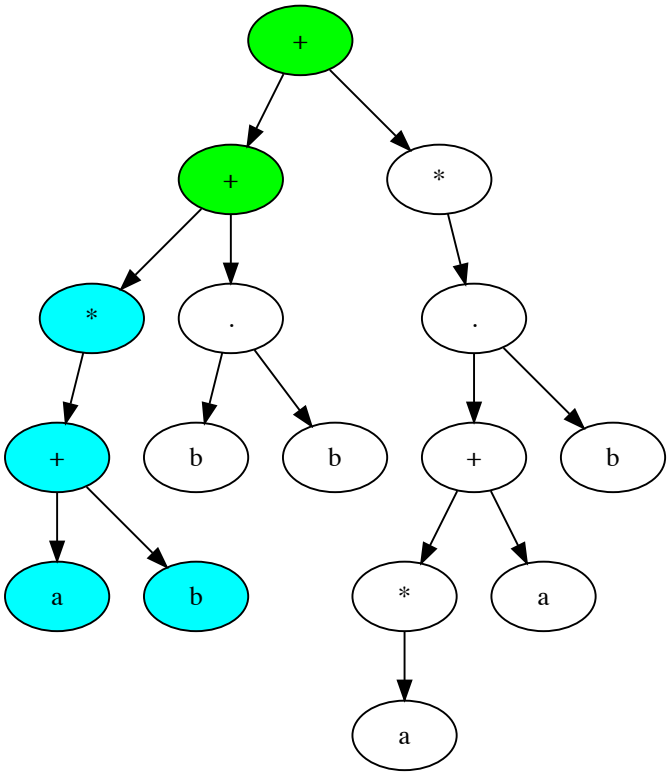
Symbolic step: fully reducible trees

An expression tree T is *fully reducible* when $\sigma(T) = \mathcal{P}$.



Symbolic step: fully reducible trees

An expression tree T is *fully reducible* when $\sigma(T) = \mathcal{P}$.

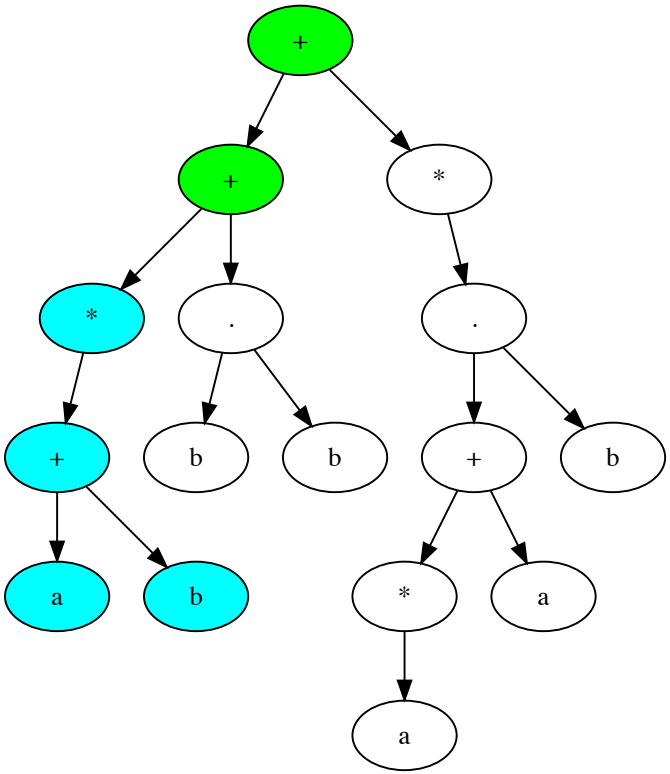


Fully reducible expressions

- dictate the reduction process:
  leaves of the reduced expression.

Symbolic step: fully reducible trees

An expression tree T is *fully reducible* when $\sigma(T) = \mathcal{P}$.



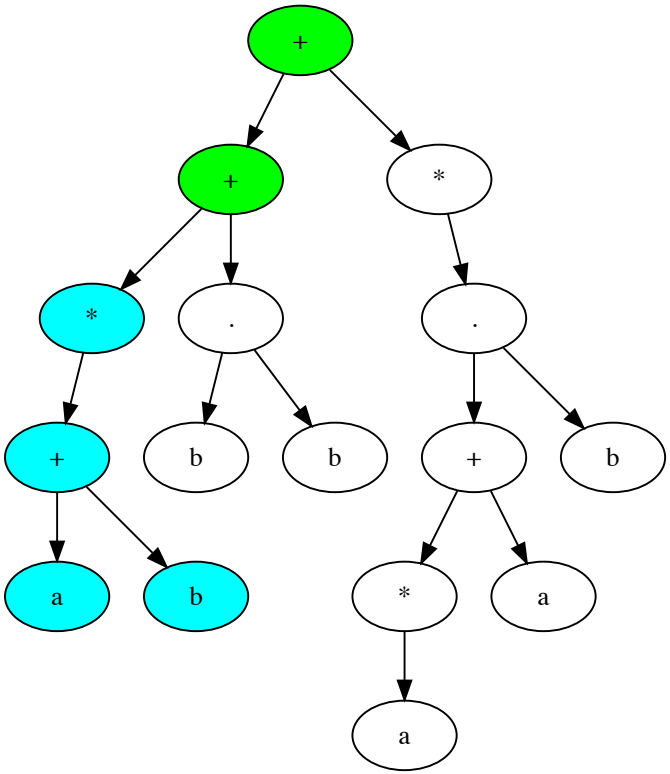
Fully reducible expressions

- dictate the reduction process:
  leaves of the reduced expression.
- can also be specified recursively, e.g.,

$$\mathcal{P} = \overset{\star}{\underset{a}{\cdot}}\overset{\cdot}{+}\overset{\cdot}{\underset{b}{\cdot}} \; ; \quad \mathcal{R} = \mathcal{P} + \overset{+}{\underset{\mathcal{R}}{/}}\backslash\overset{}{\underset{\mathcal{L}}{}} + \overset{+}{\underset{\mathcal{L}}{/}}\backslash\overset{}{\underset{\mathcal{R}}{}} .$$

Symbolic step: fully reducible trees

An expression tree T is *fully reducible* when $\sigma(T) = \mathcal{P}$.



Fully reducible expressions

- dictate the reduction process:
  leaves of the reduced expression.
- can also be specified recursively, e.g.,

$$\mathcal{P} = \overset{\star}{\underset{a}{\cdot}}\!\!\overset{\cdot}{+}\!\!\underset{b}{\cdot} \; ; \quad \mathcal{R} = \mathcal{P} + \overset{+}{\underset{\mathcal{R}}{/}}\!\!\underset{\mathcal{L}}{\backslash} + \overset{+}{\underset{\mathcal{L}}{/}}\!\!\underset{\mathcal{R}}{\backslash} \, .$$

We consider a fundamental sequence

$$\gamma_n := \Pr_n \{\sigma(T) = \mathcal{P}\} \, , \qquad A(z) := \sum \gamma_n z^n \, ,$$

of probabilities of *full* reduction and their generating function.

Symbolic step: recurrence = differential equation

Recurrence for e_n in terms of γ_n yields first order differential equation

$$E'(z) = F(z, A(z)) + \tfrac{1}{1-p_{\mathsf{I}} z}\left(\tfrac{2}{z} - p_{\mathsf{I}} + 2\,(1-p_{\mathsf{I}})\,\frac{z}{1-z} - 2p_{\circledast} A(z)\right)\cdot E(z)\,,$$

in terms of $A(z) = \sum_n \gamma_n z^n$.

Symbolic step: recurrence = differential equation

Recurrence for e_n in terms of γ_n yields first order differential equation

$$E'(z) = F(z, A(z)) + \frac{1}{1-p_{\mathsf{I}} z}\left(\frac{2}{z} - p_{\mathsf{I}} + 2\,(1-p_{\mathsf{I}})\,\frac{z}{1-z} - 2p_{\circledast} A(z)\right)\cdot E(z)\,,$$

in terms of $A(z) = \sum_n \gamma_n z^n$.

First order differential equations can be solved explicitly:

Proposition

Let f, g analytic on a star-shaped Ω and consider

$$U'(z) = f(z) + g(z)U(z)\,.$$

The equation has a unique solution analytic on Ω, with $U(0) = u_0$,

$$U(z) = \exp\left(\int_0^z g(\zeta)d\zeta\right)\left(u_0 + \int_0^z f(\zeta)\exp\left(-\int_0^\zeta g(w)dw\right)d\zeta\right).$$

Symbolic step: recurrence = differential equation

Recurrence for e_n in terms of γ_n yields first order differential equation

$$E'(z) = F(z, A(z)) + \tfrac{1}{1-p_{\mathsf{I}} z}\left(\tfrac{2}{z} - p_{\mathsf{I}} + 2\,(1-p_{\mathsf{I}})\,\frac{z}{1-z} - 2p_{\circledast} A(z)\right)\cdot E(z)\,,$$

in terms of $A(z) = \sum_n \gamma_n z^n$.

First order differential equations can be solved explicitly:

Proposition

Let f, g analytic on a star-shaped Ω and consider

$$U'(z) = f(z) + g(z)U(z)\,.$$

The equation has a unique solution analytic on Ω, with $U(0) = u_0$,

$$U(z) = \exp\left(\int_0^z g(\zeta)d\zeta\right)\left(u_0 + \int_0^z f(\zeta)\exp\left(-\int_0^\zeta g(w)dw\right)d\zeta\right).$$

Our coefficients depend on z and the unknown generating function $A(z)$.

Analytic step: the singularity $z = 1$

The generating functions $A(z)$ and $E(z)$

- are analytic for $|z| < 1$, as the series converge absolutely.

Analytic step: the singularity $z = 1$

The generating functions $A(z)$ and $E(z)$

- are analytic for $|z| < 1$, as the series converge absolutely.
- have $z = 1$ as a dominant singularity: $\rho = 1$ radius of convergence.

Analytic step: the singularity $z = 1$

The generating functions $A(z)$ and $E(z)$

- are analytic for $|z| < 1$, as the series converge absolutely.
- have $z = 1$ as a dominant singularity: $\rho = 1$ radius of convergence.

Solution of ODE yields, as $z \to 1$

$$E(z) \sim \frac{c}{(1-z)^2} \left(2 + \int_0^z F(w, A(w)) I(w) dw \right) (I(z))^{-1} , ,$$

where $I(z) := \exp\left(2p_{\circledast} \int_0^z \frac{A(w)}{1-p_{\mathsf{l}} w} dw \right)$.

Analytic step: the singularity $z = 1$

The generating functions $A(z)$ and $E(z)$

- are analytic for $|z| < 1$, as the series converge absolutely.
- have $z = 1$ as a dominant singularity: $\rho = 1$ radius of convergence.

Solution of ODE yields, as $z \to 1$

$$E(z) \sim \frac{c}{(1-z)^2} \left(2 + \int_0^z F(w, A(w)) I(w) dw\right) (I(z))^{-1}, ,$$

where $I(z) := \exp\left(2p_{\circledast} \int_0^z \frac{A(w)}{1-p_{\mathsf{I}} w} dw\right)$.

To apply the Transfer Theorem and complete the proof:

- we require precise asymptotics for $A(z)$ at $z = 1$,
- we show that $A(z)$ and $E(z)$ are analytic over $\Omega = \mathbb{C} \setminus [1, \infty)$.

Fully reducible trees: probabilities

We study the generating function $A(z) = \sum \gamma_n z^n$

Proposition: recurrence for γ_n

The probabilities $\gamma_n = \Pr_n \{\sigma(T) = \mathcal{P}\}$ satisfy, for $n \geq |\mathcal{P}|$,

$$\gamma_{n+1} = \frac{p_{\circledast}}{n-1} \sum_{k=1}^{n-1} (\gamma_k + \gamma_{n-k} - \gamma_k \gamma_{n-k}) \,.$$

Fully reducible trees: probabilities

We study the generating function $A(z) = \sum \gamma_n z^n$

Proposition: recurrence for γ_n

The probabilities $\gamma_n = \Pr_n \{\sigma(T) = \mathcal{P}\}$ satisfy, for $n \geq |\mathcal{P}|$,

$$\gamma_{n+1} = \frac{p_\circledast}{n-1} \sum_{k=1}^{n-1} (\gamma_k + \gamma_{n-k} - \gamma_k \gamma_{n-k}) \,.$$

Recurrence translates into Riccati differential equation

$$A'(z) = (s-2)\gamma_s z^{s-1} + \left(\frac{2}{z} + 2p_\circledast \frac{z}{1-z} \right) A(z) - p_\circledast \cdot (A(z))^2 \,,$$

where $s = |\mathcal{P}|$ is the size of the absorbing pattern.

Behaviour of solutions of Riccati ODE

Considering $v(z)$ such that $p_{\circledast} A(z) = v'(z)/v(z)$,

Riccati equation becomes linear

$$v''(z) = p_{\circledast} \cdot (s-2)\gamma_s z^{s-1} v(z) + \left(\frac{2}{z} + 2p_{\circledast} \frac{z}{1-z} \right) v'(z) \,.$$

Behaviour of solutions of Riccati ODE

Considering $v(z)$ such that $p_{\circledast} A(z) = v'(z)/v(z)$,

Riccati equation becomes linear

$$v''(z) = p_{\circledast} \cdot (s-2)\gamma_s z^{s-1} v(z) + \left(\frac{2}{z} + 2p_{\circledast} \frac{z}{1-z} \right) v'(z) \,.$$

For linear ODEs:

- domain of analyticity well-understood.
- Frobenius method characterizes behaviour at singularities.

Behaviour of solutions of Riccati ODE

Considering $v(z)$ such that $p_{\circledast} A(z) = v'(z)/v(z)$,
Riccati equation becomes linear

$$v''(z) = p_{\circledast} \cdot (s-2)\gamma_s z^{s-1} v(z) + \left(\frac{2}{z} + 2p_{\circledast} \frac{z}{1-z} \right) v'(z) \,.$$

For linear ODEs:

- domain of analyticity well-understood.
- Frobenius method characterizes behaviour at singularities.

Proposition

The generating function $A(z)$ satisfies, $z \to 1$

- For $p_{\circledast} > \frac{1}{2}$, $A(z) = \frac{\gamma_\infty}{1-z} + O((1-z)^{2p_{\circledast}-2})$,
- For $p_{\circledast} = \frac{1}{2}$, $A(z) = \frac{2}{1-z} \left(\log\left(\frac{1}{1-z}\right) \right)^{-1} \left(1 + O\left(\log\left(\frac{1}{1-z}\right)^{-1} \right) \right)$
- For $p_{\circledast} < \frac{1}{2}$, $A(z) \sim \frac{D}{(1-z)^{2p_{\circledast}}}$,

where $\gamma_\infty := (2p_{\circledast} - 1)/p_{\circledast}$ and $D > 0$ is a constant.

Probability of full reduction

Theorem

The probability γ_n of being fully reducible tends to the constant $\gamma_\infty := (2p_\circledast - 1)/p_\circledast$ for $p_\circledast > \frac{1}{2}$ and to zero if $p_\circledast \leq \frac{1}{2}$.

Moreover, for the cases in which it tends to zero,

- *for $p_\circledast = \frac{1}{2}$ we have $\gamma_n \sim \frac{2}{\log n}$,*
- *for $p_\circledast < \frac{1}{2}$, $\gamma_n \sim D \cdot n^{2p_\circledast - 1}$.*

Probability of full reduction

Theorem

The probability γ_n of being fully reducible tends to the constant $\gamma_\infty := (2p_\circledast - 1)/p_\circledast$ for $p_\circledast > \frac{1}{2}$ and to zero if $p_\circledast \leq \frac{1}{2}$.

Moreover, for the cases in which it tends to zero,

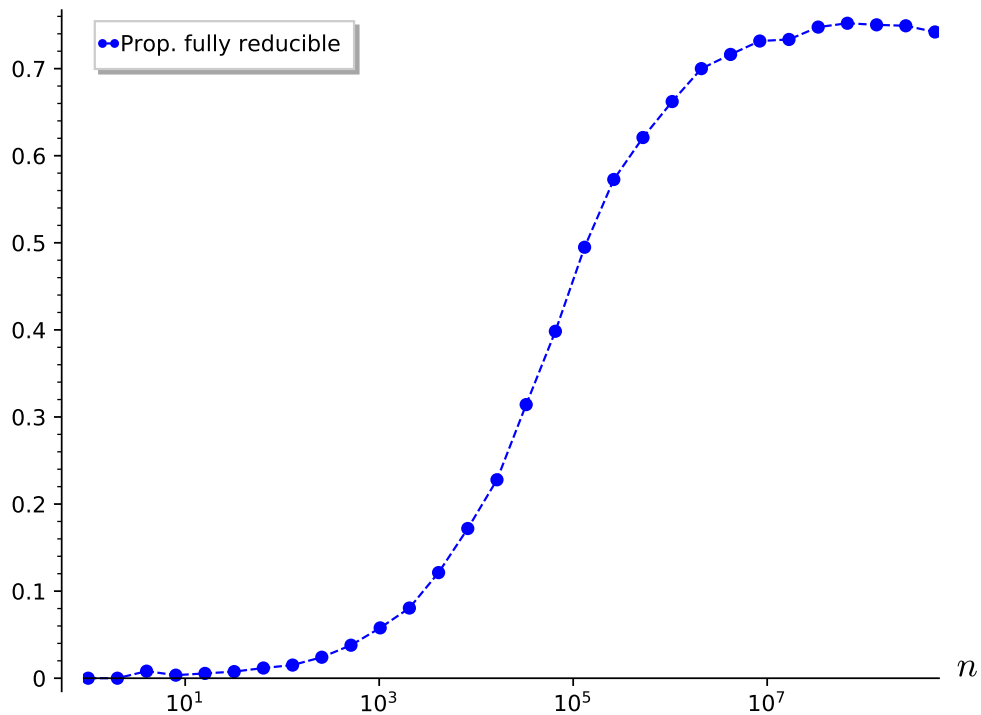
- *for $p_\circledast = \frac{1}{2}$ we have $\gamma_n \sim \frac{2}{\log n}$,*
- *for $p_\circledast < \frac{1}{2}$, $\gamma_n \sim D \cdot n^{2p_\circledast - 1}$.*

Experimental plot:
regular expressions on two letters with

$$(p_+, p_\bullet, p_\star) = (\tfrac{8}{10}, \tfrac{1}{10}, \tfrac{1}{10})\,.$$

Then

$$\lim_{n\to\infty} \gamma_n = 3/4\,.$$



Section

Conclusion and further work

- A simple linear algorithm, reduces uniform regular expressions to small constant size.
- Uniform random regular expression trees tend to describe very limited languages.
- BST-like expression trees present a richer range of behaviours than the uniform ones.

Conclusion and further work

- A simple linear algorithm, reduces uniform regular expressions to small constant size.
- Uniform random regular expression trees tend to describe very limited languages.
- BST-like expression trees present a richer range of behaviours than the uniform ones.

Questions and further work

1. Absorbing operator $\circledast$ of arity $a \geq 3$ for BST-like ?

Conclusion and further work

- A simple linear algorithm, reduces uniform regular expressions to small constant size.
- Uniform random regular expression trees tend to describe very limited languages.
- BST-like expression trees present a richer range of behaviours than the uniform ones.

Questions and further work

1. Absorbing operator $\circledast$ of arity $a \geq 3$ for BST-like ?
   $\Rightarrow$ expect similar results, threshold $\frac{1}{a}$ instead of $\frac{1}{2}$.

Conclusion and further work

- A simple linear algorithm, reduces uniform regular expressions to small constant size.
- Uniform random regular expression trees tend to describe very limited languages.
- BST-like expression trees present a richer range of behaviours than the uniform ones.

Questions and further work

1. Absorbing operator $\circledast$ of arity $a \geq 3$ for BST-like ?
   $\Rightarrow$ expect similar results, threshold $\frac{1}{a}$ instead of $\frac{1}{2}$.
2. Absorbing pattern model is general
   $\Rightarrow$ consider interactions between several operators?

Conclusion and further work

- A simple linear algorithm, reduces uniform regular expressions to small constant size.
- Uniform random regular expression trees tend to describe very limited languages.
- BST-like expression trees present a richer range of behaviours than the uniform ones.

Questions and further work

1. Absorbing operator $\circledast$ of arity $a \geq 3$ for BST-like ?
   $\Rightarrow$ expect similar results, threshold $\frac{1}{a}$ instead of $\frac{1}{2}$.
2. Absorbing pattern model is general
   $\Rightarrow$ consider interactions between several operators?
3. Take a concrete case: LTL formulas.

References

Florent Koechlin, Cyril Nicaud, and Pablo Rotondo.
On the Degeneracy of Random Expressions Specified by Systems of Combinatorial Equations.
Proceedings of DLT 2020, LNCS vol. 12086, pp 164–177.
https://doi.org/10.1007/978-3-030-48516-0_13

Florent Koechlin and Pablo Rotondo.
Absorbing patterns in BST-like expression-trees.
Proceedings of STACS 2021, LIPICs, vol.187, 48:1–48:15
https://doi.org/10.4230/LIPIcs.STACS.2021.48

Florent Koechlin, Carine Pivoteau and Pablo Rotondo.
Heuristic universality detection over regular expressions specified by systems.
Proceedings of DLT 2025, August 2025, pp. 278 – 293.
https://doi.org/10.1007/978-3-032-01475-7_19

Thank you!