Comparison-free polyregular functions

Nguyễn Lê Thành Dũng (a.k.a. Tito) — `nltd@nguyentito.eu`
Laboratoire d'informatique de Paris Nord, Villetaneuse, France
joint work with Pierre Pradic (University of Oxford)
and the fictional author Camille Noûs (`https://www.cogitamus.fr/camilleen.html`)

June 16th, 2021 — Automata Theory seminar, MIMUW, University of Warsaw

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

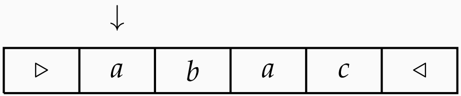
Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \mathtt{reverse}(w)$

$\triangleright$	a	b	a	c	$\triangleleft$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

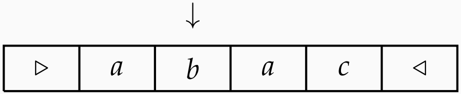


state $= 1$ output $=$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

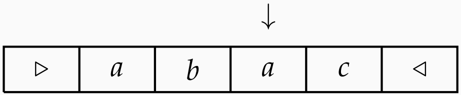


state = 1 output = a

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

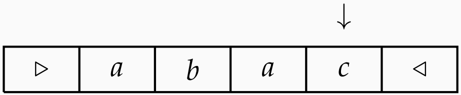


state $= 1$ output $= aa$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

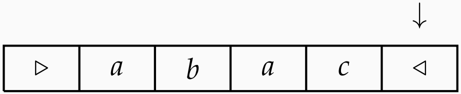


state $= 1$ output $= aaa$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

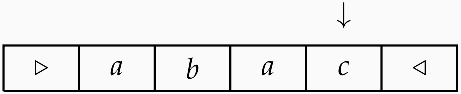


state = 1   output = $aaaa$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

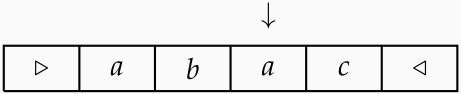


state $= 2$ output $= aaaa$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

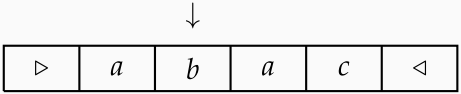


state = 2 output = $aaaac$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

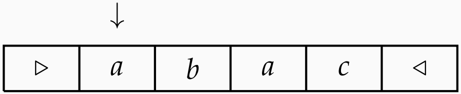


state = 2 output = $aaaaca$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

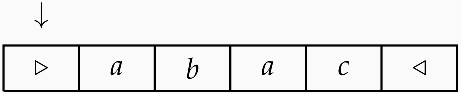


state $= 2$ output $= aaaacab$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$



state = 2 output = *aaaacaba*

Background: two-way transducers $\rightarrow$ regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

$\triangleright$	a	b	a	c	$\triangleleft$

state = $\texttt{end}$ output = $aaaacaba$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \mathtt{reverse}(w)$

$\triangleright$	a	b	a	c	$\triangleleft$

state = `end` output = $aaaacaba$

Can also change direction in middle of input,
e.g. $\mathtt{mapReverse}(abc\#bac\#ca) = cba\#cab\#ac$

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

▷	a	b	a	c	◁

state = `end` output = $aaaacaba$

Can also change direction in middle of input,
e.g. $\texttt{mapReverse}(abc\#bac\#ca) = cba\#cab\#ac$

Properties of regular functions:

- linear growth: $|f(w)| = O(|w|)$
- closed under composition (if $f : \Gamma^* \to \Sigma^*$ and $g : \Sigma^* \to \Pi^*$ are regular then so is $g \circ f$)
- L regular $\implies f^{-1}(L)$ regular

Background: two-way transducers → regular functions

Two-way (deterministic finite) transducer (2DFT):
finite state + bidirectional reading head
+ output produced from left to right

Example: $w \in \{a, b\}^* \mapsto a^{|w|} \cdot \texttt{reverse}(w)$

state = `end` output = $aaaacaba$

Can also change direction in middle of input,
e.g. $\texttt{mapReverse}(abc\#bac\#ca) = cba\#cab\#ac$

Properties of regular functions:

- linear growth: $|f(w)| = O(|w|)$
- closed under composition (if $f : \Gamma^* \to \Sigma^*$ and $g : \Sigma^* \to \Pi^*$ are regular then so is $g \circ f$)
- L regular $\implies f^{-1}(L)$ regular

Some alternative characterizations:

- via Monadic Second-Order logic
- copyless streaming string transducers
- various functional programming or regexp-like (declarative) formalisms
- (self-advertizing) linear λ-calculus (*Implicit automata in typed λ-calculi*)

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

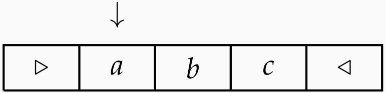
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output =

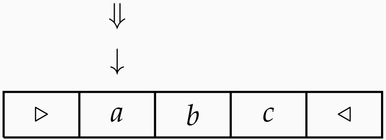
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output =

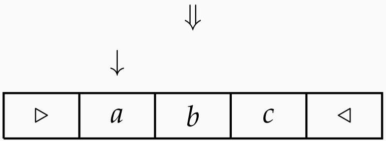
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}$

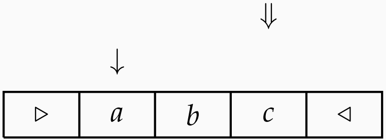
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}b$

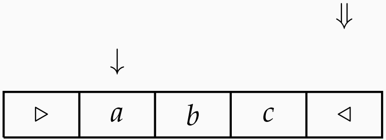
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output = $\underline{a}bc$

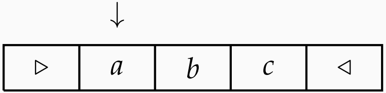
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}bc$

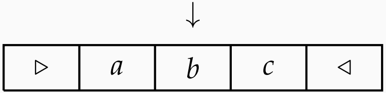
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output = $\underline{a}bc$

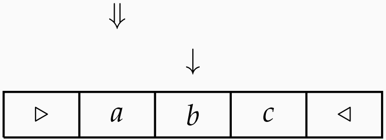
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output $= \underline{a}bc$

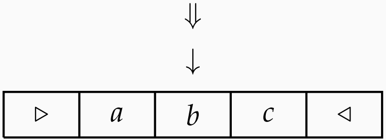
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}bca$

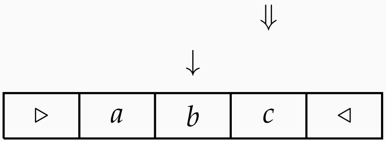
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output $= \underline{a}bca\underline{b}$

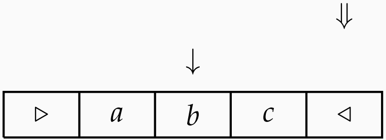
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}bca\underline{b}c$

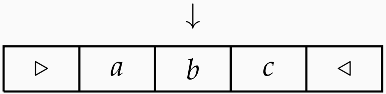
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}bca\underline{b}c$

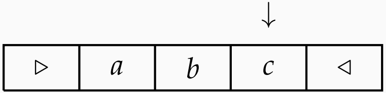
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}bca\underline{b}c$

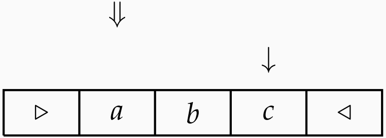
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output = $\underline{a}bca\underline{b}c$

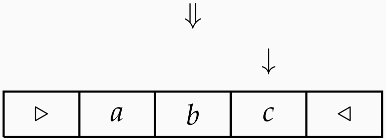
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}bca\underline{b}ca$

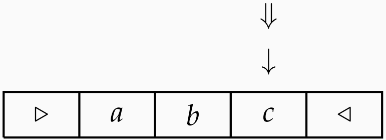
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output = $\underline{a}bca\underline{b}cab$

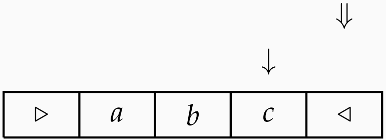
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output $= \underline{a}bca\underline{b}cab\underline{c}$

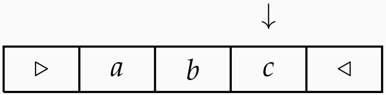
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}bca\underline{b}cab\underline{c}$

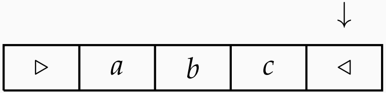
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}bca\underline{b}cab\underline{c}$

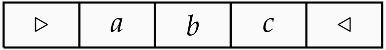
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output = $\underline{a}bca\underline{b}cab\underline{c}$

- Operations on reading head stack: move topmost / push / pop
- Transitions can *compare* head positions

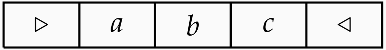
Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: "squaring with underlining" ($k = 2$)



output $= \underline{a}bca\underline{b}cab\underline{c}$

- Operations on reading head stack: move topmost / push / pop
- Transitions can *compare* head positions

Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

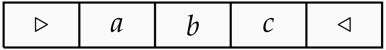
Recent characterizations [Bojańczyk 2018; Bojańczyk, Kiefer & Lhote 2019]:

- multidimensional MSO interpretations
- imperative nested loop programs
- simply typed λ-calculus + list manipulation primitives
- composition closure of [regular functions ∪ "squaring with underlining"]

Background: pebble transducers → polyregular functions

k-pebble transducers ($k \in \mathbb{N}_{\geq 1}$):
finite state + *stack of at most k two-way heads*

Ex: “squaring with underlining” ($k = 2$)



output $= \underline{a}bca\underline{b}cab\underline{c}$

- Operations on reading head stack: move topmost / push / pop
- Transitions can *compare* head positions
  - **Our paper: if they can’t, then what?**

Properties of *polyregular* functions:

- *polynomial* growth: $|f(w)| = O(|w|^k)$
- closed under composition
- L regular $\implies f^{-1}(L)$ regular

Recent characterizations [Bojańczyk 2018; Bojańczyk, Kiefer & Lhote 2019]:

- multidimensional MSO interpretations
- imperative nested loop programs
- simply typed λ-calculus + list manipulation primitives
- composition closure of [regular functions $\cup$ “squaring with underlining”]

Introducing comparison-free polyregular functions

(new definitions/results start here)

Comparison-free k-pebble transducers:

finite state + stack of at most k two-way heads
whose positions cannot be compared

Example

$\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

Introducing comparison-free polyregular functions

(new definitions/results start here)

Comparison-free k-pebble transducers:
finite state + stack of at most k two-way heads
whose positions cannot be compared

Example

$\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

Characterizations of comparison-free polyreg fn:

- composition closure of reg fn $\cup$ `cfsquaring`
  (“squaring with underlining” instead → get all polyreg)
  hard part: cfpolyreg fn are closed under comp
- linear λ-calculus (tweak of the regular case; major motivation for this work)
- *conjecture*: some kind of logical transduction

+ nice properties of polyregular functions

Introducing comparison-free polyregular functions

(new definitions/results start here)

Comparison-free k-pebble transducers:
finite state + stack of at most k two-way heads
whose positions cannot be compared

Example

$\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

Characterizations of comparison-free polyreg fn:

- composition closure of reg fn $\cup$ `cfsquaring`
  (“squaring with underlining” instead $\to$ get all polyreg)
  hard part: cfpolyreg fn are closed under comp
- linear λ-calculus (tweak of the regular case; major motivation for this work)
- *conjecture*: some kind of logical transduction

+ nice properties of polyregular functions

Inductive reformulation of the cf pebble def:
the smallest class such that

- every regular function is cfpolyreg
- if f is regular and g_i is cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i\in I})$ is cfpolyreg

Introducing comparison-free polyregular functions

(new definitions/results start here)

Comparison-free k-pebble transducers:
finite state + stack of at most k two-way heads
whose positions cannot be compared

Example

$$\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$$

Characterizations of comparison-free polyreg fn:

- composition closure of reg fn ∪ `cfsquaring`
  (“squaring with underlining” instead → get all polyreg)
  hard part: cfpolyreg fn are closed under comp
- linear λ-calculus (tweak of the regular case; major motivation for this work)
- *conjecture*: some kind of logical transduction

+ nice properties of polyregular functions

Definition (Composition by substitution)

Let $f : \Gamma^* \to I^*$, $g_i : \Gamma^* \to \Sigma^*$ and $w \in \Gamma^*$.
If $f(w) = i_1 \ldots i_k$ then

$$\mathrm{CbS}(f, (g_i)_{i \in I})(w) = g_{i_1}(w) \ldots g_{i_k}(w)$$

e.g. for `cfsquaring`, $f(abc) = aXbXcX$,
$g_a(w) = \underline{a}$, $g_X(w) = w$

Inductive reformulation of the cf pebble def:
the smallest class such that

- every regular function is cfpolyreg
- if f is regular and g_i is cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i \in I})$ is cfpolyreg

Introducing comparison-free polyregular functions

(new definitions/results start here)

Comparison-free k-pebble transducers:
finite state + stack of at most k two-way heads
whose positions cannot be compared

Example

$\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

Characterizations of comparison-free polyreg fn:

- composition closure of reg fn $\cup$ `cfsquaring`
  (“squaring with underlining” instead → get all polyreg)
  hard part: cfpolyreg fn are closed under comp
- linear λ-calculus (tweak of the regular case; major motivation for this work)
- *conjecture*: some kind of logical transduction

+ nice properties of polyregular functions

Definition (Composition by substitution)

Let $f : \Gamma^* \to I^*$, $g_i : \Gamma^* \to \Sigma^*$ and $w \in \Gamma^*$.
If $f(w) = i_1 \ldots i_k$ then

$$\mathrm{CbS}(f, (g_i)_{i \in I})(w) = g_{i_1}(w) \ldots g_{i_k}(w)$$

e.g. for `cfsquaring`, $f(abc) = aXbXcX$,
$g_a(w) = \underline{a}$, $g_X(w) = w$

Inductive reformulation of the cf pebble def:
the smallest class such that

- every regular function is cfpolyreg
- if f is regular and g_i is cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i \in I})$ is cfpolyreg

Note: both cfpolyreg and polyregular
functions are closed under CbS

Separation results

What about *counter*examples?

Theorem

The function $f : a^n \in \{a\}^* \mapsto a\#aa\# \dots \#a^n$ *is polyregular but not comparison-free.*

Corollary: “squaring with underlining” is not CF.

Theorem

$g : a^{n_1}\# \dots \#a^{n_k} \in \{a, \#\}^* \mapsto a^{n_1 \times n_1}\# \dots \#a^{n_k \times n_k}$ *is polyregular but not comparison-free.*

([Douéneau-Tabot 2021] proves a stronger result)

Separation results

What about *counter*examples?

Theorem

The function $f : a^n \in \{a\}^* \mapsto a\#aa\#\dots\#a^n$ *is polyregular but not comparison-free.*

Corollary: “squaring with underlining” is not CF.

f is also an *HDT0L transduction* ($\iff$ computable by a copyful streaming string transducer / marble transducer [Douéneau-Tabot et al. 2020]). Therefore HDT0L $\not\subset$ cfpolyreg; conversely:

Theorem

$w \in \Gamma^* \mapsto w^{|w|}$ *is comparison-free polyregular, but when* $|\Gamma| \geq 2$*, it is not HDT0L.*

Note: polynomially growing HDT0L $\subset$ polyreg

Theorem

$g : a^{n_1}\#\dots\#a^{n_k} \in \{a,\#\}^* \mapsto a^{n_1 \times n_1}\#\dots\#a^{n_k \times n_k}$ *is polyregular but not comparison-free.*

([Douéneau-Tabot 2021] proves a stronger result)

Separation results

What about *counter*examples?

Theorem

The function $f : a^n \in \{a\}^* \mapsto a\#aa\#\ldots\#a^n$ *is polyregular but not comparison-free.*

Corollary: “squaring with underlining” is not CF.

f is also an *HDT0L transduction* ($\iff$ computable by a copyful streaming string transducer / marble transducer [Douéneau-Tabot et al. 2020]). Therefore HDT0L $\not\subset$ cfpolyreg; conversely:

Theorem

$w \in \Gamma^* \mapsto w^{|w|}$ *is comparison-free polyregular, but when* $|\Gamma| \geq 2$, *it is not HDT0L.*

Note: polynomially growing HDT0L $\subset$ polyreg

Theorem

$g : a^{n_1}\#\ldots\#a^{n_k} \in \{a,\#\}^* \mapsto a^{n_1 \times n_1}\#\ldots\#a^{n_k \times n_k}$ *is polyregular but not comparison-free.*

([Douéneau-Tabot 2021] proves a stronger result)

Definition

For $h : \Gamma^* \to \Sigma^*$, $w_1, \ldots, w_n \in \Gamma^*$ with $\# \notin \Gamma$, $\mathbf{map}(h)(w_1\#\ldots\#w_n) = f(w_1)\#\ldots\#f(w_n)$.

$g = \mathbf{map}(w \mapsto w^{|w|})$ therefore comparison-free polyregular functions are *not* closed under **map**, unlike regular and polyreg functions

→ obstruction to characterizing cfpolyreg fn by list-processing functional programs (à la [Bojańczyk, Daviaud & Krishna 2018])

Separation results

What about *counter*examples? (Next: proofs)

Theorem

The function $f : a^n \in \{a\}^* \mapsto a\#aa\#\dots\#a^n$ *is polyregular but not comparison-free.*

Corollary: “squaring with underlining” is not CF.

f is also an *HDT0L transduction* ($\iff$ computable by a copyful streaming string transducer / marble transducer [Douéneau-Tabot et al. 2020]). Therefore HDT0L $\not\subset$ cfpolyreg; conversely:

Theorem

$w \in \Gamma^* \mapsto w^{|w|}$ *is comparison-free polyregular, but when* $|\Gamma| \geq 2$*, it is not HDT0L.*

Note: polynomially growing HDT0L $\subset$ polyreg

Theorem

$g : a^{n_1}\#\dots\#a^{n_k} \in \{a,\#\}^* \mapsto a^{n_1 \times n_1}\#\dots\#a^{n_k \times n_k}$ *is polyregular but not comparison-free.*

([Douéneau-Tabot 2021] proves a stronger result)

Definition

For $h : \Gamma^* \to \Sigma^*$, $w_1, \dots, w_n \in \Gamma^*$ with $\# \notin \Gamma$, $\mathbf{map}(h)(w_1\#\dots\#w_n) = f(w_1)\#\dots\#f(w_n)$.

$g = \mathbf{map}(w \mapsto w^{|w|})$ therefore comparison-free polyregular functions are *not* closed under **map**, unlike regular and polyreg functions

→ obstruction to characterizing cfpolyreg fn
by list-processing functional programs
(à la [Bojańczyk, Daviaud & Krishna 2018])

Separation proof for “map unary square” + pebble minimization

Theorem

$g(a^{n_1}\# \dots \#a^{n_k}) = a^{n_1 \times n_1}\# \dots \#a^{n_k \times n_k}$
is not *comparison-free polyregular.*

Proof by contradiction: assume g is cfpolyreg.
First, $|g(w)| = O(|w|^2)$ *therefore* g is computed by some **2**-cf-pebble transducer.

Separation proof for “map unary square” + pebble minimization

Theorem (Pebble minimization – major result of our paper)

If f is cfpolyreg and $|f(w)| = O(|w|^k)$ then some comparison-free k-pebble transducer computes f.

Very technical proof adapted from the analogous result for pebble transducers [Lhote 2020].

Theorem

$g(a^{n_1}\# \dots \# a^{n_k}) = a^{n_1 \times n_1}\# \dots \# a^{n_k \times n_k}$
is not *comparison-free polyregular.*

Proof by contradiction: assume g is cfpolyreg.
First, $|g(w)| = O(|w|^2)$ *therefore* g is computed by some **2**-cf-pebble transducer.

Separation proof for “map unary square” + pebble minimization

Theorem (Pebble minimization – major result of our paper)

If f is cfpolyreg and $|f(w)| = O(|w|^k)$ then some comparison-free k-pebble transducer computes f.

Very technical proof adapted from the analogous result for pebble transducers [Lhote 2020].

Theorem

$g(a^{n_1}\# \dots \# a^{n_k}) = a^{n_1 \times n_1}\# \dots \# a^{n_k \times n_k}$
is not *comparison-free polyregular.*

Proof by contradiction: assume g is cfpolyreg.
First, $|g(w)| = O(|w|^2)$ *therefore* g is computed by some **2**-cf-pebble transducer. Equivalently, for some *finite* I and *regular* functions f and h_i,

$$g = \mathrm{CbS}(f, (h_i)_{i \in I}) \quad \text{(composition by substitution)}$$

$$\text{i.e.} \quad f(w) = i_1 \dots i_m \implies g(w) = h_{i_1}(w) \dots h_{i_m}(w)$$

Separation proof for “map unary square” + pebble minimization

Theorem (Pebble minimization – major result of our paper)

If f is cfpolyreg and $|f(w)| = O(|w|^k)$ then some comparison-free k-pebble transducer computes f.

Very technical proof adapted from the analogous result for pebble transducers [Lhote 2020].

Theorem

$g(a^{n_1}\# \dots \# a^{n_k}) = a^{n_1 \times n_1}\# \dots \# a^{n_k \times n_k}$
is not *comparison-free polyregular.*

Proof by contradiction: assume g is cfpolyreg.
First, $|g(w)| = O(|w|^2)$ *therefore* g is computed by some **2**-cf-pebble transducer. Equivalently, for some *finite* I and *regular* functions f and h_i,

$$g = \mathrm{CbS}(f, (h_i)_{i \in I}) \quad \text{(composition by substitution)}$$

$$\text{i.e.} \quad f(w) = i_1 \dots i_m \implies g(w) = h_{i_1}(w) \dots h_{i_m}(w)$$

(Oversimplified) idea: consider $g(w)$ where

$$w = \underbrace{aa\dots a\#\dots\#aa\dots a}_{|I| \text{ times } \# \text{ i.e. } |I|+1 \text{ blocks of } a\text{s}}$$

Separation proof for "map unary square" + pebble minimization

Theorem (Pebble minimization – major result of our paper)

If f is cfpolyreg and $|f(w)| = O(|w|^k)$ then some comparison-free k-pebble transducer computes f.

Very technical proof adapted from the analogous result for pebble transducers [Lhote 2020].

Theorem

$g(a^{n_1}\# \dots \# a^{n_k}) = a^{n_1 \times n_1}\# \dots \# a^{n_k \times n_k}$
is not *comparison-free polyregular.*

Proof by contradiction: assume g is cfpolyreg.
First, $|g(w)| = O(|w|^2)$ *therefore* g is computed by some **2**-cf-pebble transducer. Equivalently, for some *finite* I and *regular* functions f and h_i,

$$g = \mathrm{CbS}(f, (h_i)_{i \in I}) \quad \text{(composition by substitution)}$$

$$\text{i.e.} \quad f(w) = i_1 \dots i_m \implies g(w) = h_{i_1}(w) \dots h_{i_m}(w)$$

(Oversimplified) idea: consider $g(w)$ where

$$w = \underbrace{aa\dots a\#\dots\#aa\dots a}_{|I| \text{ times } \# \text{ i.e. } |I|+1 \text{ blocks of } as}$$

"pump" p-th factor $aa\dots a$ $(1 \le p \le |I|+1)$ in w
→ p-th factor in $g(w)$ grows as $\Theta(n^2)$
→ $\exists$ corresponding $j[p] \in I$ such that $|f(w)|_{j[p]}$ and $|h_{j[p]}(w)|$ grow as $\Theta(n)$

Pigeonhole principle: $j[p] = j[q]$ for some $p \neq q$
→ "pumping" p-th factor of w makes q-th factor of $g(w)$ grow, contradiction

Separation proofs continued + unary inputs

Theorem

$f(a^n) = a\#aa\# \dots \#a^n$ is not cfpolyreg.

Observation: $f(a^n)$ has the n maximal a-factors

$$a \quad aa \quad \dots \quad a^n$$

Lemma

For any cfpolyreg $g : \{a\}^* \to \Sigma^*$, there are $O(1)$ possible lengths for maximal a-factors in $g(a^n)$.

Separation proofs continued + unary inputs

Theorem

$f(a^n) = a\#aa\# \ldots \#a^n$ *is* not *cfpolyreg*.

Observation: $f(a^n)$ has the n maximal a-factors

$$a \quad aa \quad \ldots \quad a^n$$

Lemma

For any cfpolyreg $g : \{a\}^* \to \Sigma^*$, there are $O(1)$ possible lengths *for maximal a-factors in* $g(a^n)$.

In fact, $\exists \mathcal{S} \subseteq \mathbb{Q}[X]$ *finite* such that $\{P(n) \mid P \in \mathcal{S}\}$ contains $\{$lengths of maximal a-factors of $g(a^n)\}$, by structural induction on *poly-pumping sequences*.

Separation proofs continued + unary inputs

Theorem

$f(a^n) = a\#aa\# \ldots \#a^n$ is not cfpolyreg.

Observation: $f(a^n)$ has the n maximal a-factors

$$a \quad aa \quad \ldots \quad a^n$$

Lemma

For any cfpolyreg $g : \{a\}^* \to \Sigma^*$, there are $O(1)$ possible lengths for maximal a-factors in $g(a^n)$.

In fact, $\exists \mathcal{S} \subseteq \mathbb{Q}[X]$ *finite* such that $\{P(n) \mid P \in \mathcal{S}\}$ contains $\{$lengths of maximal a-factors of $g(a^n)\}$, by structural induction on *poly-pumping sequences*.

Definition (poly-pumping sequence of words)

Smallest subclass of $(\Sigma^*)^{\mathbb{N}}$

- containing the constant sequences $\alpha_n = w$
- closed under concatenation $\alpha_n = \beta_n \cdot \gamma_n$
- closed under "iteration" $\alpha_n = (\beta_n)^n$

Separation proofs continued + unary inputs

Theorem

$f(a^n) = a\#aa\# \dots \#a^n$ *is* not *cfpolyreg.*

Observation: $f(a^n)$ has the n maximal a-factors

$$a \quad aa \quad \dots \quad a^n$$

Lemma

For any cfpolyreg $g : \{a\}^* \to \Sigma^*$, there are $O(1)$ possible lengths *for maximal a-factors in* $g(a^n)$.

In fact, $\exists \mathcal{S} \subseteq \mathbb{Q}[X]$ *finite* such that $\{P(n) \mid P \in \mathcal{S}\}$ contains {lengths of maximal a-factors of $g(a^n)$}, by structural induction on *poly-pumping sequences*.

Definition (poly-pumping sequence of words)

Smallest subclass of $(\Sigma^*)^{\mathbb{N}}$

- containing the constant sequences $\alpha_n = w$
- closed under concatenation $\alpha_n = \beta_n \cdot \gamma_n$
- closed under "iteration" $\alpha_n = (\beta_n)^n$

Theorem (cfpolyreg with unary input)

$f : \{a\}^* \to \Sigma^*$ *is comparison-free polyregular if and only if* $\exists p \in \mathbb{N}$ *such that* $(f(a^{(n+1)p+m}))_{n\in\mathbb{N}}$ *is poly-pumping for every* $m < p$.

→ "ultimately periodic combinations" (u.p.c.)

Separation proofs continued + unary inputs

Theorem

$f(a^n) = a\#aa\#\dots\#a^n$ *is* not *cfpolyreg.*

Observation: $f(a^n)$ has the n maximal a-factors

$$a \quad aa \quad \dots \quad a^n$$

Lemma

For any cfpolyreg $g : \{a\}^* \to \Sigma^*$, there are $O(1)$ possible lengths *for maximal a-factors in* $g(a^n)$.

In fact, $\exists \mathcal{S} \subseteq \mathbb{Q}[X]$ *finite* such that $\{P(n) \mid P \in \mathcal{S}\}$ contains $\{$lengths of maximal a-factors of $g(a^n)\}$, by structural induction on *poly-pumping sequences*.

Definition (poly-pumping sequence of words)

Smallest subclass of $(\Sigma^*)^{\mathbb{N}}$

- containing the constant sequences $\alpha_n = w$
- closed under concatenation $\alpha_n = \beta_n \cdot \gamma_n$
- closed under "iteration" $\alpha_n = (\beta_n)^n$

Theorem (cfpolyreg with unary input)

$f : \{a\}^* \to \Sigma^*$ *is comparison-free polyregular if and only if* $\exists p \in \mathbb{N}$ *such that* $(f(a^{(n+1)p+m}))_{n\in\mathbb{N}}$ *is poly-pumping for every* $m < p$.

→ "ultimately periodic combinations" (u.p.c.)

- Regular word sequences are u.p.c. of pumping sequences $(u_0(v_1)^n \dots (v_l)^n u_l)_{n\in\mathbb{N}}$ [Choffrut 2017]

  Proof idea: find an idempotent in a suitable transition monoid of your favorite machine model for reg fn
- Proof for general cfpolyreg sequences: induction on the CbS-based definition

First-order comparison-free polyregular functions (not in the paper)

Poly-pumping sequences: generated from constants by concat and "iteration" $\alpha_n = (\beta_n)^n$.

Theorem

"ultimately periodic combinations" of poly-pumping sequences = all comparison-free polyreg sequences

What about the *aperiodic* ($\Leftrightarrow$ *first-order*) case?

First-order comparison-free polyregular functions (not in the paper)

Poly-pumping sequences: generated from constants by concat and “iteration” $\alpha_n = (\beta_n)^n$.

Theorem

“ultimately periodic combinations” of poly-pumping sequences = all comparison-free polyreg sequences

What about the *aperiodic* ($\Leftrightarrow$ *first-order*) case?

FO-regular functions are characterized by

- logic: replace MSO by FO
- 2DFT with aperiodic monoid of behaviors
- regexp-like e.g. [Dartois, Gastin & Krishna 2021]
- more self-advertizing:
  - non-commutative λ-calculus – should extend to FO-cfpolyreg characterization
  - 2DFT with *planar* behaviors (upcoming)

First-order comparison-free polyregular functions (not in the paper)

Poly-pumping sequences: generated from constants by concat and "iteration" $\alpha_n = (\beta_n)^n$.

Theorem

"ultimately periodic combinations" of poly-pumping sequences = all comparison-free polyreg sequences

What about the *aperiodic* ($\Leftrightarrow$ *first-order*) case?

FO-regular functions are characterized by

- logic: replace MSO by FO
- 2DFT with aperiodic monoid of behaviors
- regexp-like e.g. [Dartois, Gastin & Krishna 2021]
- more self-advertizing:
  - non-commutative λ-calculus – should extend to FO-cfpolyreg characterization
  - 2DFT with *planar* behaviors (upcoming)

Definition (First-order cfpolyreg functions)

FO-cfpolyreg = smallest class such that

- every FO-regular function is FO-cfpolyreg
- if f is FO-regular and g_i is FO-cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i \in I})$ is FO-cfpolyreg

First-order comparison-free polyregular functions (not in the paper)

Poly-pumping sequences: generated from constants by concat and “iteration” $\alpha_n = (\beta_n)^n$.

Theorem

“ultimately periodic combinations” of poly-pumping sequences = all comparison-free polyreg sequences

What about the *aperiodic* ($\Leftrightarrow$ *first-order*) case?

Also, FO-cfpolyreg $\subsetneq$ FO-polyregular (the previous counterexamples are all FO-polyreg); and composition, pebble minimization, etc. should work *mutatis mutandis*

FO-regular functions are characterized by

- logic: replace MSO by FO
- 2DFT with aperiodic monoid of behaviors
- regexp-like e.g. [Dartois, Gastin & Krishna 2021]
- more self-advertizing:
  - non-commutative λ-calculus – should extend to FO-cfpolyreg characterization
  - 2DFT with *planar* behaviors (upcoming)

Definition (First-order cfpolyreg functions)

FO-cfpolyreg = smallest class such that

- every FO-regular function is FO-cfpolyreg
- if f is FO-regular and g_i is FO-cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i \in I})$ is FO-cfpolyreg

First-order comparison-free polyregular functions (not in the paper)

Poly-pumping sequences: generated from constants by concat and “iteration” $\alpha_n = (\beta_n)^n$.

Theorem

“ultimately periodic combinations” of poly-pumping sequences = all comparison-free polyreg sequences

What about the *aperiodic* ($\Leftrightarrow$ *first-order*) case?

Theorem

$f : \{a\}^* \to \Sigma^*$ *is FO-cfpolyreg iff* $\exists p \in \mathbb{N}$ *such that* $n \mapsto f(a^{n+p})$ *is poly-pumping.*

Also, FO-cfpolyreg $\subsetneq$ FO-polyregular (the previous counterexamples are all FO-polyreg);

and composition, pebble minimization, etc. should work *mutatis mutandis*

FO-regular functions are characterized by

- logic: replace MSO by FO
- 2DFT with aperiodic monoid of behaviors
- regexp-like e.g. [Dartois, Gastin & Krishna 2021]
- more self-advertizing:
  - non-commutative λ-calculus – should extend to FO-cfpolyreg characterization
  - 2DFT with *planar* behaviors (upcoming)

Definition (First-order cfpolyreg functions)

FO-cfpolyreg = smallest class such that

- every FO-regular function is FO-cfpolyreg
- if f is FO-regular and g_i is FO-cfpolyreg $\forall i \in I$ then $\mathrm{CbS}(f, (g_i)_{i\in I})$ is FO-cfpolyreg

Integer sequences (not in the paper)

Previously: unary *inputs*. [Douéneau-Tabot 2021]:
unary *outputs*. Let's consider both: let $f : \mathbb{N} \to \mathbb{N}$.

Integer sequences (not in the paper)

Previously: unary *inputs*. [Douéneau-Tabot 2021]: unary *outputs*. Let's consider both: let $f : \mathbb{N} \to \mathbb{N}$.

Theorem

f is polyregular $\iff$ any of those:

1. *f is HDT0L ($\Leftrightarrow$ linear recurrence) & $f(n) = n^{O(1)}$*
2. *f is computable by a "marble bimachine"*
3. *f is an ultimately periodic combination of $\mathbb{N}[X]$*

   (*unary poly-pumping sequences = polynomials in* $\mathbb{N}[X]$)
4. *f is comparison-free polyregular*

→ single canonical automata-theoretic class of polynomial growth integer sequences?

polyreg ⇔ (1) ⇔ (2) holds for arbitrary input and unary ouptut [DT21]; the rest is elementary

Integer sequences (not in the paper)

Previously: unary *inputs*. [Douéneau-Tabot 2021]: unary *outputs*. Let's consider both: let $f : \mathbb{N} \to \mathbb{N}$.

Theorem

f is polyregular $\iff$ any of those:

1. *f is HDT0L ($\Leftrightarrow$ linear recurrence)* & $f(n) = n^{O(1)}$
2. *f is computable by a "marble bimachine"*
3. *f is an ultimately periodic combination of $\mathbb{N}[X]$*

   (*unary poly-pumping sequences = polynomials in* $\mathbb{N}[X]$)
4. *f is comparison-free polyregular*

→ single canonical automata-theoretic class of polynomial growth integer sequences?

polyreg ⇔ (1) ⇔ (2) holds for arbitrary input and unary ouptut [DT21]; the rest is elementary

Theorem

f is FO-cfpolyreg $\Leftrightarrow \exists p \in \mathbb{N} : f(X + p) \in \mathbb{N}[X] \Leftrightarrow$ $\exists P \in \mathbb{Z}[X] : \forall n$ except finitely many, $f(n) = P(n)$

Thus, the following is not FO-cfpolyreg for $k \geq 2$:

$$b_k(n) = \binom{n}{k} \qquad \text{e.g. } b_2(n) = \frac{n(n-1)}{2}$$

Integer sequences (not in the paper)

Previously: unary *inputs*. [Douéneau-Tabot 2021]: unary *outputs*. Let's consider both: let $f : \mathbb{N} \to \mathbb{N}$.

Theorem

f is polyregular $\iff$ any of those:

1. *f is HDT0L ($\Leftrightarrow$ linear recurrence) & $f(n) = n^{O(1)}$*
2. *f is computable by a "marble bimachine"*
3. *f is an ultimately periodic combination of $\mathbb{N}[X]$*
   (*unary poly-pumping sequences = polynomials in* $\mathbb{N}[X]$)
4. *f is comparison-free polyregular*

→ single canonical automata-theoretic class of polynomial growth integer sequences?

polyreg ⇔ (1) ⇔ (2) holds for arbitrary input and unary ouptut [DT21]; the rest is elementary

Theorem

f is FO-cfpolyreg $\Leftrightarrow \exists p \in \mathbb{N} : f(X+p) \in \mathbb{N}[X] \Leftrightarrow$ $\exists P \in \mathbb{Z}[X] : \forall n$ except finitely many, $f(n) = P(n)$

Thus, the following is not FO-cfpolyreg for $k \geq 2$:

$$b_k(n) = \binom{n}{k} \qquad \text{e.g. } b_2(n) = \frac{n(n-1)}{2}$$

Yet all b_k are both cfpolyreg (divide by $k!$) and FO-polyreg (think of $a^n \mapsto a\#aa\#\ldots\#a^n$)
→ **FO-cfpolyreg ≠ (FO-polyreg ∩ cfpolyreg)**

Integer sequences (not in the paper)

Previously: unary *inputs*. [Douéneau-Tabot 2021]: unary *outputs*. Let's consider both: let $f : \mathbb{N} \to \mathbb{N}$.

Theorem

f is polyregular $\iff$ any of those:

1. *f is HDT0L ($\Leftrightarrow$ linear recurrence) & $f(n) = n^{O(1)}$*
2. *f is computable by a "marble bimachine"*
3. *f is an ultimately periodic combination of $\mathbb{N}[X]$*
   (*unary poly-pumping sequences = polynomials in* $\mathbb{N}[X]$)
4. *f is comparison-free polyregular*

→ single canonical automata-theoretic class of polynomial growth integer sequences?

polyreg ⇔ (1) ⇔ (2) holds for arbitrary input and unary ouptut [DT21]; the rest is elementary

Theorem

f is FO-cfpolyreg $\Leftrightarrow \exists p \in \mathbb{N} : f(X+p) \in \mathbb{N}[X] \Leftrightarrow$ $\exists P \in \mathbb{Z}[X] : \forall n$ except finitely many, $f(n) = P(n)$

Thus, the following is not FO-cfpolyreg for $k \geq 2$:

$$b_k(n) = \binom{n}{k} \qquad \text{e.g. } b_2(n) = \frac{n(n-1)}{2}$$

Yet all b_k are both cfpolyreg (divide by $k!$) and FO-polyreg (think of $a^n \mapsto a\#aa\#\ldots\#a^n$)
→ **FO-cfpolyreg $\neq$ (FO-polyreg $\cap$ cfpolyreg)**

$$\forall P \in \mathbb{Q}[X],\ P(X+a) = \sum_{i=0}^{\deg P} \Delta^k(P)(a) \binom{X}{k}$$

→ all polynomials P s.t. $P(\mathbb{N}) \subseteq \mathbb{N}$ are FO-polyreg (*conjecture*: some kind of converse holds)

Conclusion

A new(?) class of string-to-string functions: *comparison-free polyregular functions*.

Equivalent definitions

- by comparison-free pebble transducers
- inductively (composition by substitution)
- **as the composition closure of regular functions +** $\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

Conclusion

A new(?) class of string-to-string functions: *comparison-free polyregular functions*.

Equivalent definitions

- by comparison-free pebble transducers
- inductively (composition by substitution)
- **as the composition closure of regular functions +** $\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

- L regular language $\implies f^{-1}(L)$ also regular
- polynomial growth: $|f(w)| = O(|w|^k)$
  - **pebble minimization theorem:** $k =$ **number of heads necessary to compute** f
- strictly included in polyregular functions
  - $a^n \mapsto a\#aa\#\dots\#a^n$ / "map unary square"
  - for $\{a\}^* \to \{a\}^*$ cfpolyreg = polyreg
- well-behaved first-order counterpart

Conclusion

A new(?) class of string-to-string functions: *comparison-free polyregular functions*.

Equivalent definitions

- by comparison-free pebble transducers
- inductively (composition by substitution)
- **as the composition closure of regular functions +** `cfsquaring`$(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

- L regular language $\implies f^{-1}(L)$ also regular
- polynomial growth: $|f(w)| = O(|w|^k)$
  - **pebble minimization theorem:** $k =$ **number of heads necessary to compute** f
- strictly included in polyregular functions
  - $a^n \mapsto a\#aa\#\ldots\#a^n$ / “map unary square”
  - for $\{a\}^* \to \{a\}^*$ cfpolyreg = polyreg
- well-behaved first-order counterpart

Open questions

- Membership problems: for partial results, see [Douéneau-Tabot 2021] (again!)
- Equivalence problem: seems hard
- Logical characterization

Conclusion

A new(?) class of string-to-string functions: *comparison-free polyregular functions*.

Equivalent definitions

- by comparison-free pebble transducers
- inductively (composition by substitution)
- **as the composition closure of regular functions +** $\texttt{cfsquaring}(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

- L regular language $\Longrightarrow f^{-1}(L)$ also regular
- polynomial growth: $|f(w)| = O(|w|^k)$
  - **pebble minimization theorem:** $k =$ **number of heads necessary to compute** f
- strictly included in polyregular functions
  - $a^n \mapsto a\#aa\#\ldots\#a^n$ / "map unary square"
  - for $\{a\}^* \to \{a\}^*$ cfpolyreg = polyreg
- well-behaved first-order counterpart

Open questions

- Membership problems: for partial results, see [Douéneau-Tabot 2021] (again!)
- Equivalence problem: seems hard
- Logical characterization: below

Let $\mathfrak{M} : \{\text{words}\} \to \{\text{finite models}\}$ be as usual. For $\mathfrak{U} = (U, R, \ldots)$, let $\mathfrak{U}^k = (U^k, R_1, \ldots, R_k, \ldots)$ where $R_i(x_1, \ldots, x_m) :\Leftrightarrow R(\pi_i(x_1), \ldots, \pi_i(x_m))$.

Conjecture (does not work naively with MSO!)

f is *first-order* comparison-free polyregular

$\iff$ there exist $k \in \mathbb{N}$ & a FO transduction φ such that $\forall w,\ \mathfrak{M}(f(w)) \simeq \varphi\left(\mathfrak{M}(w)^k\right)$.

Conclusion

(Thanks for your attention! Questions?)

A new(?) class of string-to-string functions: *comparison-free polyregular functions*.

Equivalent definitions

- by comparison-free pebble transducers
- inductively (composition by substitution)
- **as the composition closure of regular functions +** `cfsquaring`$(abc) = \underline{a}abc\underline{b}abc\underline{c}abc$

- L regular language $\implies f^{-1}(L)$ also regular
- polynomial growth: $|f(w)| = O(|w|^k)$
  - **pebble minimization theorem:** $k =$ **number of heads necessary to compute** f
- strictly included in polyregular functions
  - $a^n \mapsto a\#aa\#\ldots\#a^n$ / "map unary square"
  - for $\{a\}^* \to \{a\}^*$ cfpolyreg = polyreg
- well-behaved first-order counterpart

Open questions

- Membership problems: for partial results, see [Douéneau-Tabot 2021] (again!)
- Equivalence problem: seems hard
- Logical characterization: below

Let $\mathfrak{M} : \{\text{words}\} \to \{\text{finite models}\}$ be as usual. For $\mathfrak{U} = (U, R, \ldots)$, let $\mathfrak{U}^k = (U^k, R_1, \ldots, R_k, \ldots)$ where $R_i(x_1, \ldots, x_m) :\Leftrightarrow R(\pi_i(x_1), \ldots, \pi_i(x_m))$.

Conjecture (does not work naively with MSO!)

f is *first-order* comparison-free polyregular

$\iff$ there exist $k \in \mathbb{N}$ & a FO transduction φ such that $\forall w,\ \mathfrak{M}(f(w)) \simeq \varphi\left(\mathfrak{M}(w)^k\right)$.