Properties of polyregular functions

Mikołaj Bojańczyk
University of Warsaw

Lê Thành Dũng (Tito) Nguyễn
ENS Lyon

Sandra Kiefer
University of Oxford

Nathan Lhote
Aix-Marseille Université

Cécilia Pradic
Swansea University

GI Theorietag 2023

Kaiserslautern

Regular functions

Regular functions

Regular languages are the ones "expressible" via

- → DFA
- → NFA
- → 2-way FA
- → regular expressions
- → MSO

Regular functions

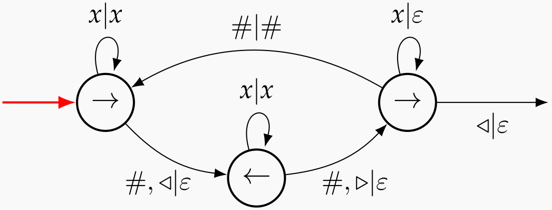
Regular languages are the ones "expressible" via
- → DFA
- → NFA
- → 2-way FA
- → regular expressions
- → MSO

Let's generalise from languages $L \subseteq \Sigma^*$ to functions $f : \Sigma^* \to \Gamma^*$.
To this end, we consider transducers, automata with output.

→ Tito's transducer simulation ←

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \dots \# w_n \cdot \texttt{reverse}(w_n)$



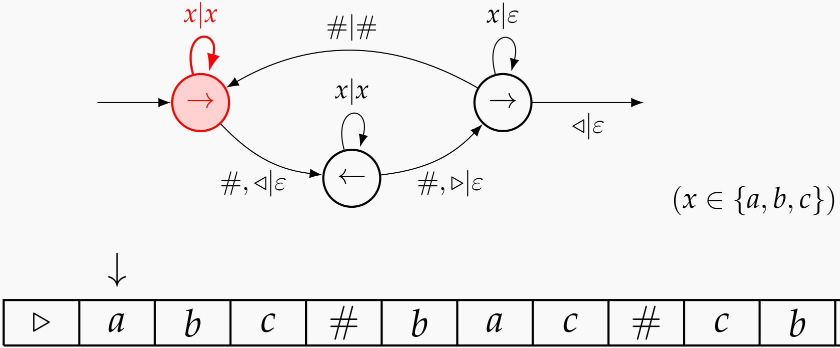
$(x \in \{a, b, c\})$

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

Output:

Two-way transducers (mentioned in [Shepherdson 1958]!)

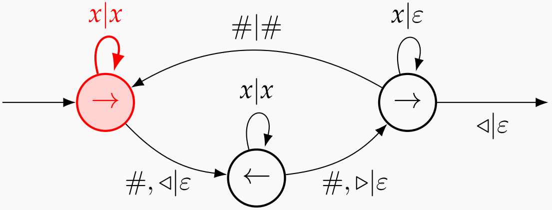
Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



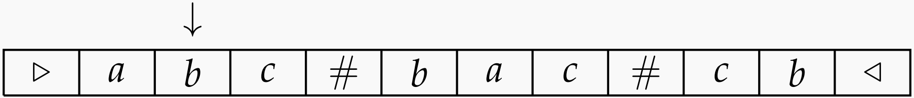
Output:

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



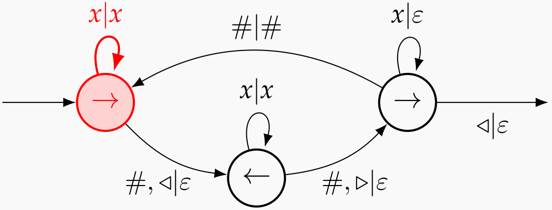
$(x \in \{a, b, c\})$



Output: a

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \ldots \# w_n \cdot \mathtt{reverse}(w_n)$



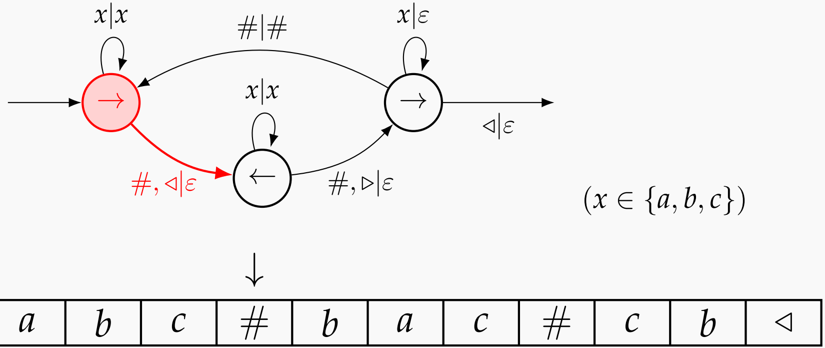
$(x \in \{a, b, c\})$

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

Output: ab

Two-way transducers (mentioned in [Shepherdson 1958]!)

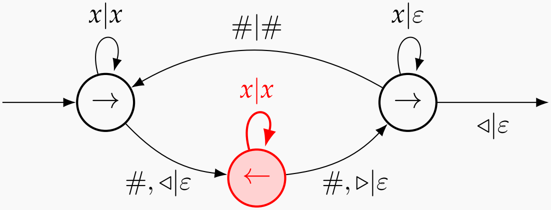
Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



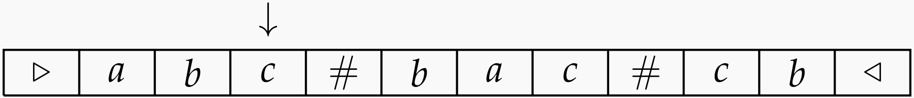
Output: abc

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \dots \# w_n \cdot \mathtt{reverse}(w_n)$



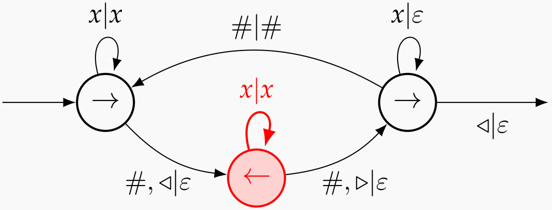
$(x \in \{a, b, c\})$



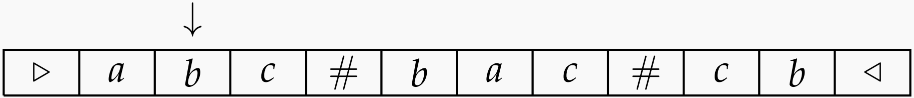
Output: abc

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1\cdot\mathtt{reverse}(w_1)\#\ldots\#w_n\cdot\mathtt{reverse}(w_n)$



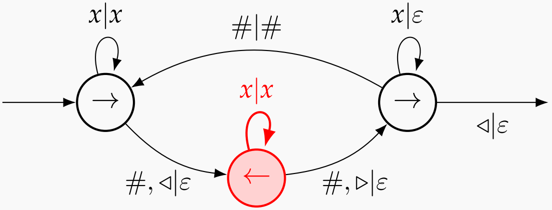
$(x \in \{a,b,c\})$



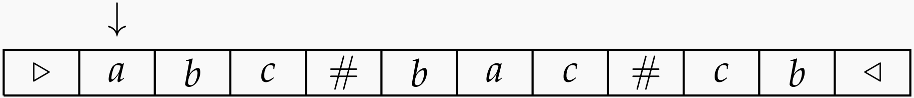
Output: $abcc$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



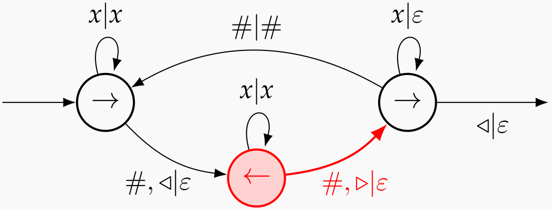
$(x \in \{a, b, c\})$



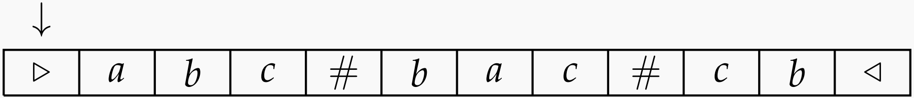
Output: *abccb*

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



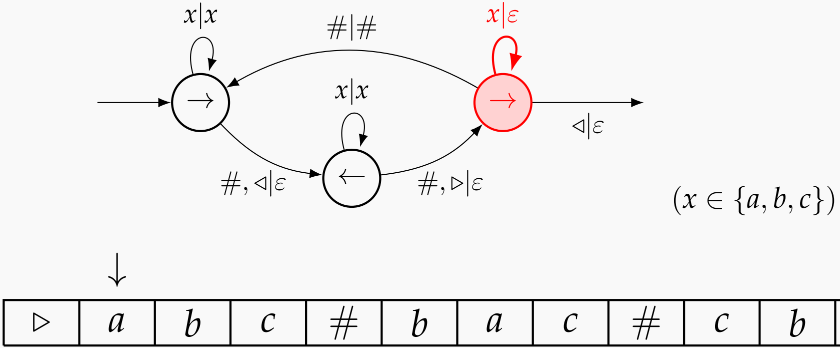
$(x \in \{a, b, c\})$



Output: *abccba*

Two-way transducers (mentioned in [Shepherdson 1958]!)

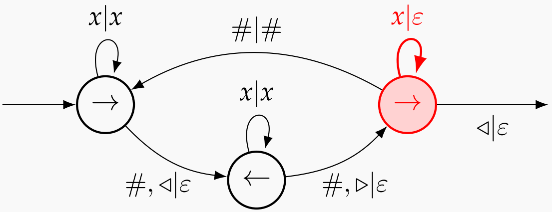
Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



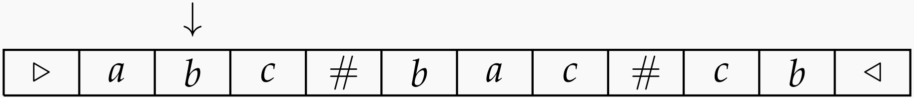
Output: *abccba*

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \dots \# w_n \cdot \mathtt{reverse}(w_n)$



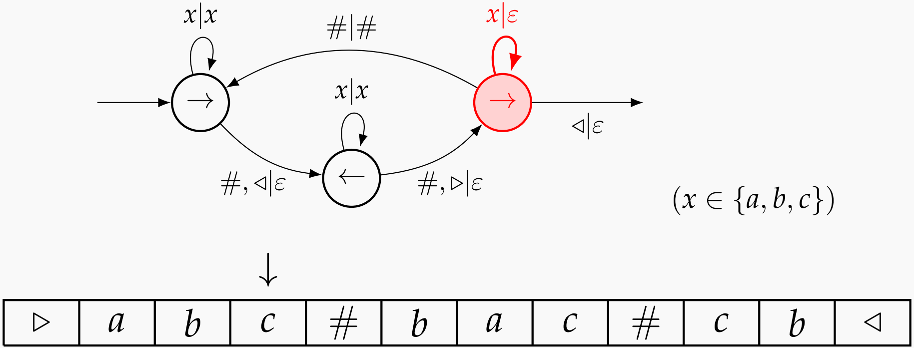
$(x \in \{a, b, c\})$



Output: *abccba*

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \ldots \# w_n \cdot \mathtt{reverse}(w_n)$

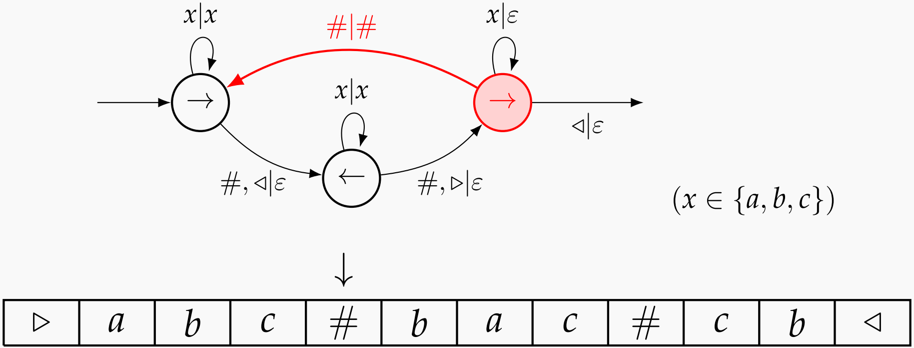


$(x \in \{a, b, c\})$

Output: *abccba*

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \ldots \# w_n \cdot \mathtt{reverse}(w_n)$

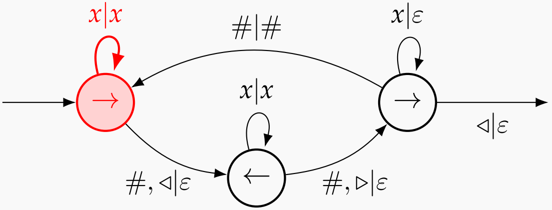


$(x \in \{a, b, c\})$

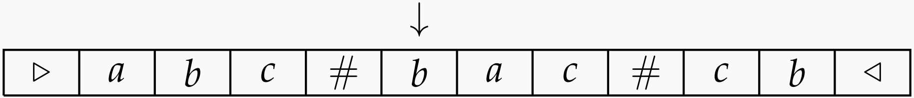
Output: *abccba*

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



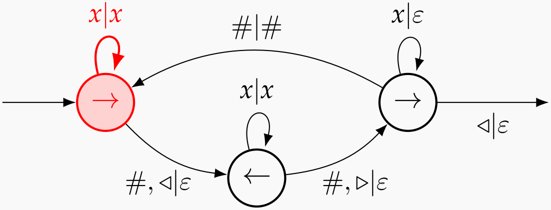
$(x \in \{a, b, c\})$



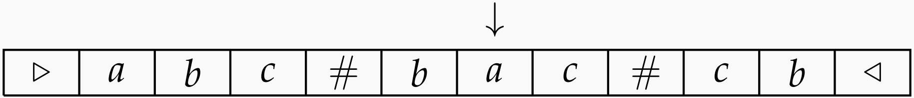
Output: $abccba\#$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \dots \# w_n \cdot \texttt{reverse}(w_n)$



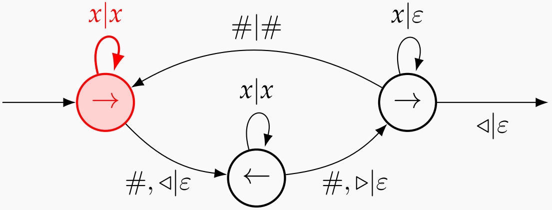
$(x \in \{a, b, c\})$



Output: $abccba\#b$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1) \# \ldots \# w_n \cdot \mathtt{reverse}(w_n)$



$(x \in \{a, b, c\})$

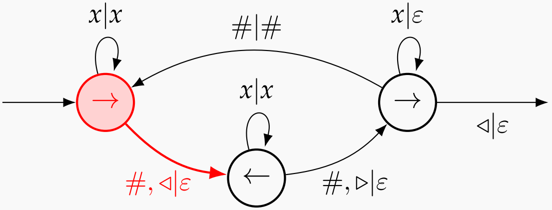
$\downarrow$

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

Output: $abccba\#ba$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \dots \# w_n \cdot \texttt{reverse}(w_n)$



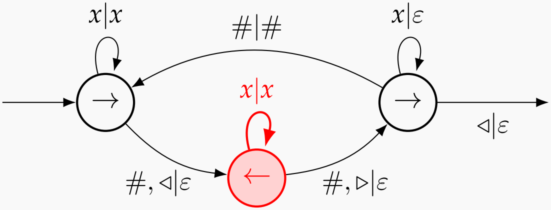
$(x \in \{a, b, c\})$

$\rhd$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\lhd$

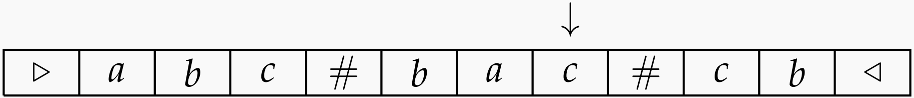
Output: $abccba\#bac$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



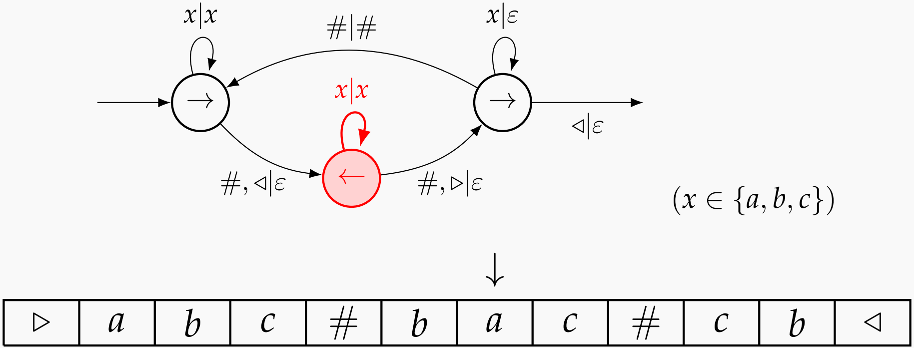
$(x \in \{a, b, c\})$



Output: $abccba\#bac$

Two-way transducers (mentioned in [Shepherdson 1958]!)

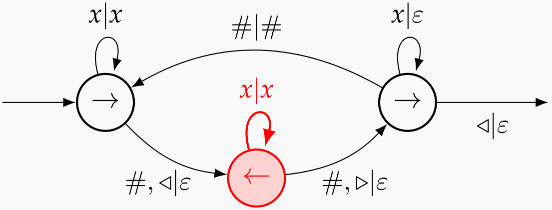
Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \ldots \# w_n \cdot \mathtt{reverse}(w_n)$



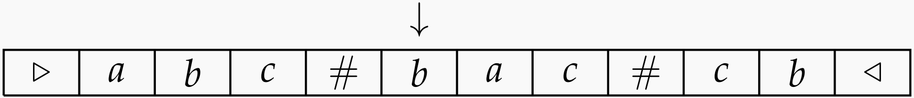
Output: $abccba\#bacc$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \dots \# w_n \cdot \texttt{reverse}(w_n)$



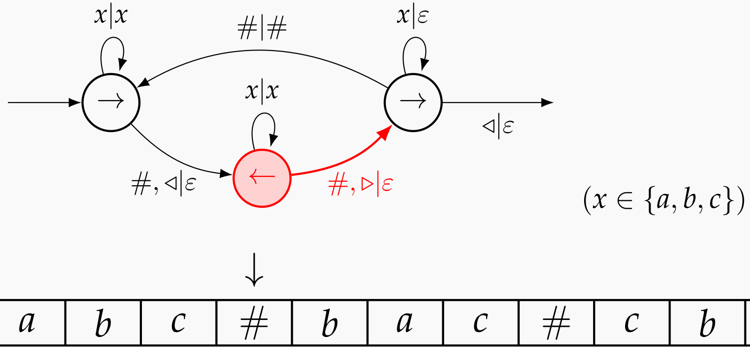
$(x \in \{a, b, c\})$



Output: $abccba\#bacca$

Two-way transducers (mentioned in [Shepherdson 1958]!)

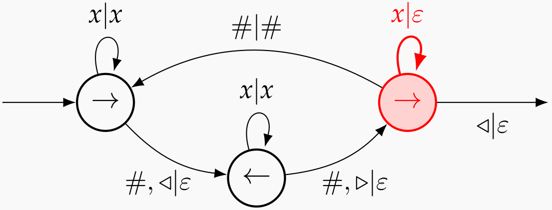
Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



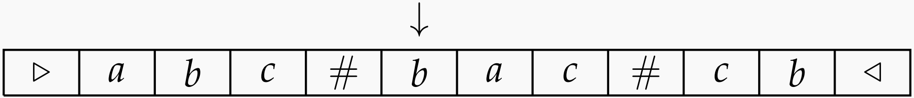
Output: $abccba\#baccab$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



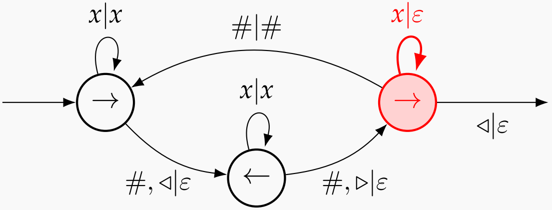
$(x \in \{a, b, c\})$



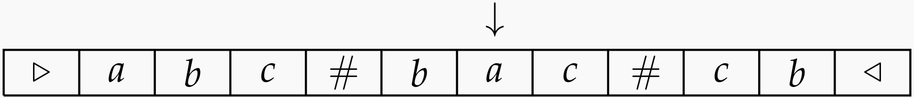
Output: $abccba\#baccab$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



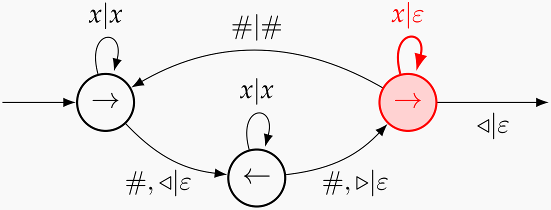
$(x \in \{a,b,c\})$



Output: $abccba\#baccab$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



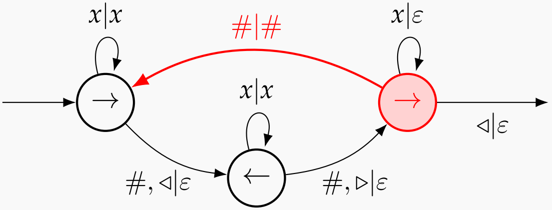
$(x \in \{a, b, c\})$

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

Output: $abccba\#baccab$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \dots \# w_n \cdot \texttt{reverse}(w_n)$



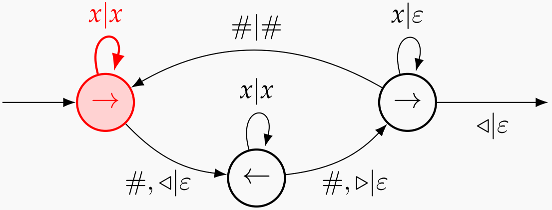
$(x \in \{a, b, c\})$

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

Output: $abccba\#baccab$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



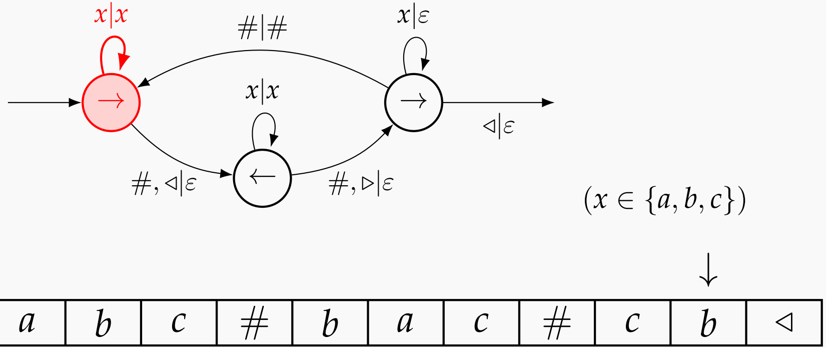
$(x \in \{a, b, c\})$

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

Output: $abccba\#baccab\#$

Two-way transducers (mentioned in [Shepherdson 1958]!)

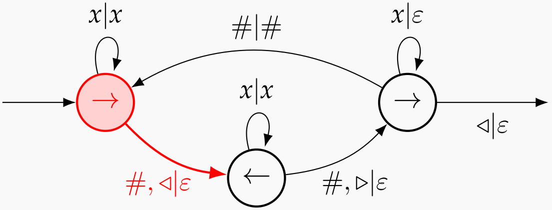
Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



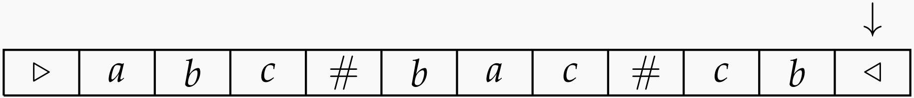
Output: $abccba\#baccab\#c$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\#\ldots\#w_n \cdot \mathtt{reverse}(w_n)$



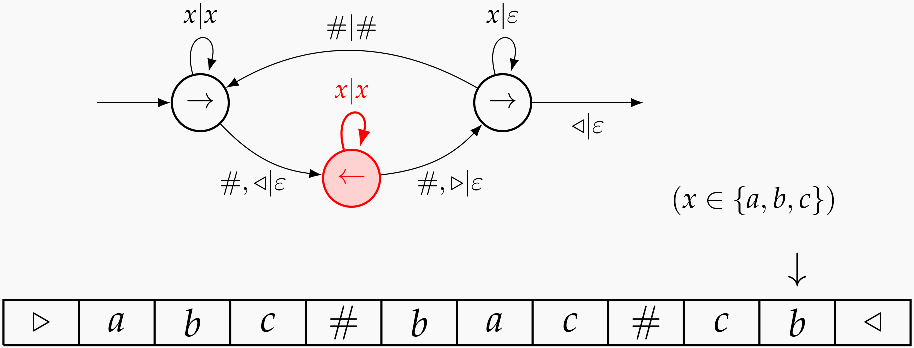
$(x \in \{a, b, c\})$



Output: $abccba\#baccab\#cb$

Two-way transducers (mentioned in [Shepherdson 1958]!)

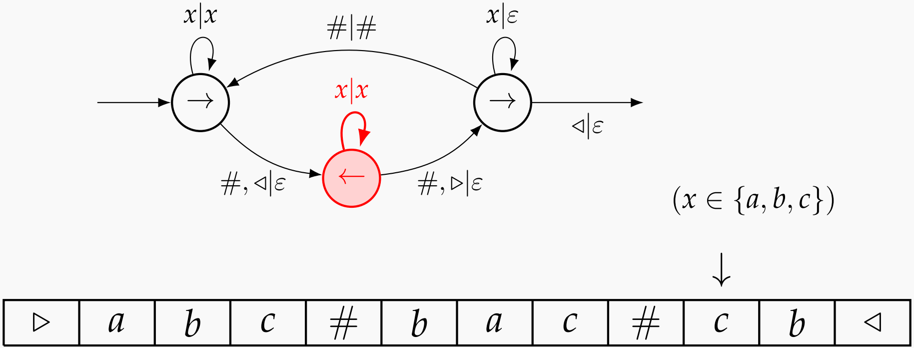
Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



Output: $abccba\#baccab\#cb$

Two-way transducers (mentioned in [Shepherdson 1958]!)

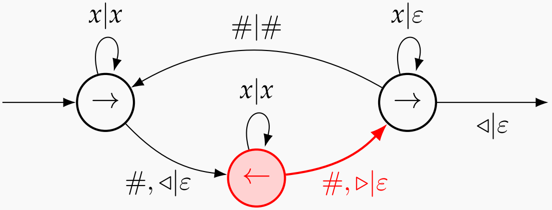
Example: $w_1\# \ldots \# w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\# \ldots \# w_n \cdot \texttt{reverse}(w_n)$



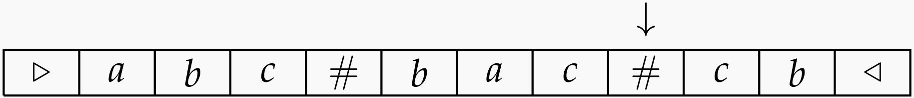
Output: $abccba\#baccab\#cbb$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1\cdot\texttt{reverse}(w_1)\#\ldots\#w_n\cdot\texttt{reverse}(w_n)$



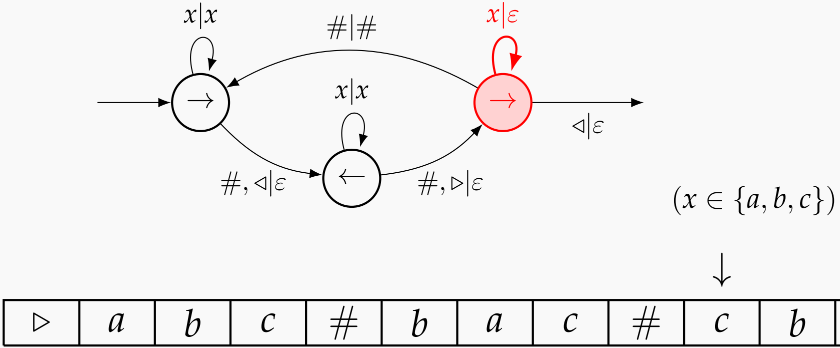
$(x \in \{a,b,c\})$



Output: $abccba\#baccab\#cbbc$

Two-way transducers (mentioned in [Shepherdson 1958]!)

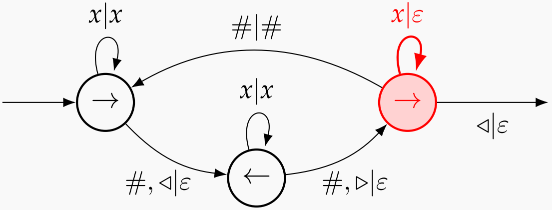
Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



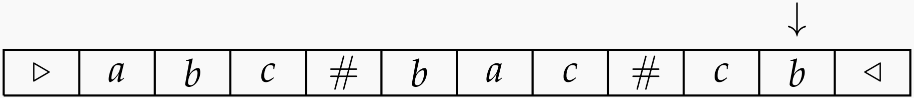
Output: $abccba\#baccab\#cbbc$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\#\ldots\#w_n \longmapsto w_1 \cdot \texttt{reverse}(w_1)\#\ldots\#w_n \cdot \texttt{reverse}(w_n)$



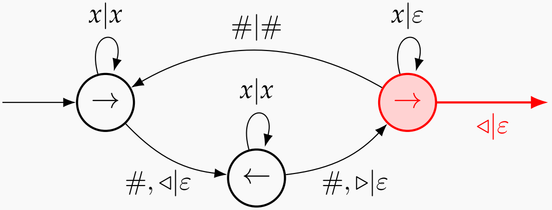
$(x \in \{a, b, c\})$



Output: $abccba\#baccab\#cbbc$

Two-way transducers (mentioned in [Shepherdson 1958]!)

Example: $w_1\# \dots \# w_n \longmapsto w_1 \cdot \mathtt{reverse}(w_1)\# \dots \# w_n \cdot \mathtt{reverse}(w_n)$



$(x \in \{a, b, c\})$

$\downarrow$

$\rhd$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\lhd$

Output: $abccba\#baccab\#cbbc$

Regular functions

Regular languages are the ones "expressible" via
→ DFA
→ NFA
→ 2-way FA
→ regular expressions
→ MSO

Let's generalise from languages $L \subseteq \Sigma^*$ to functions $f : \Sigma^* \to \Gamma^*$.
To this end, we consider transducers, automata with output.

→ Tito's transducer simulation ←

Regular functions

Regular languages are the ones "expressible" via
- → DFA
- → NFA
- → 2-way FA
- → regular expressions
- → MSO

Let's generalise from languages $L \subseteq \Sigma^*$ to functions $f : \Sigma^* \to \Gamma^*$.
To this end, we consider transducers, automata with output.

→ Tito's transducer simulation ←

Regular functions = functions computed by deterministic 2-way transducers

Regular functions

Regular languages are the ones "expressible" via
- → DFA
- → NFA
- → 2-way FA
- → regular expressions
- → MSO

Let's generalise from languages $L \subseteq \Sigma^*$ to functions $f : \Sigma^* \to \Gamma^*$.
To this end, we consider transducers, automata with output.

→ Tito's transducer simulation ←

Regular functions = functions computed by deterministic 2-way transducers

Those form a well-understood class with nice properties:
- closed under composition
- preimages of regular languages are regular

Regular functions

Regular languages are the ones "expressible" via
- → DFA
- → NFA
- → 2-way FA
- → regular expressions
- → MSO

Let's generalise from languages $L \subseteq \Sigma^*$ to functions $f : \Sigma^* \to \Gamma^*$.
To this end, we consider transducers, automata with output.

→ Tito's transducer simulation ←

Regular functions = functions computed by deterministic 2-way transducers

Those form a well-understood class with nice properties:
- closed under composition
- preimages of regular languages are regular
- robust, many equivalent definitions, e.g. MSO transductions

From "regular" to "polyregular"

Regular functions = functions computed by deterministic 2-way transducers

For regular functions, the output length is always at most linear in the input length: $f(|w|) = \mathcal{O}(|w|)$ $\Rightarrow$ linear growth rate

From "regular" to "polyregular"

Regular functions = functions computed by deterministic 2-way transducers

For regular functions, the output length is always at most linear in the input length: $f(|w|) = \mathcal{O}(|w|)$ $\Rightarrow$ linear growth rate

How can we modify the model to go <u>beyond</u> linear growth?

From "regular" to "polyregular"

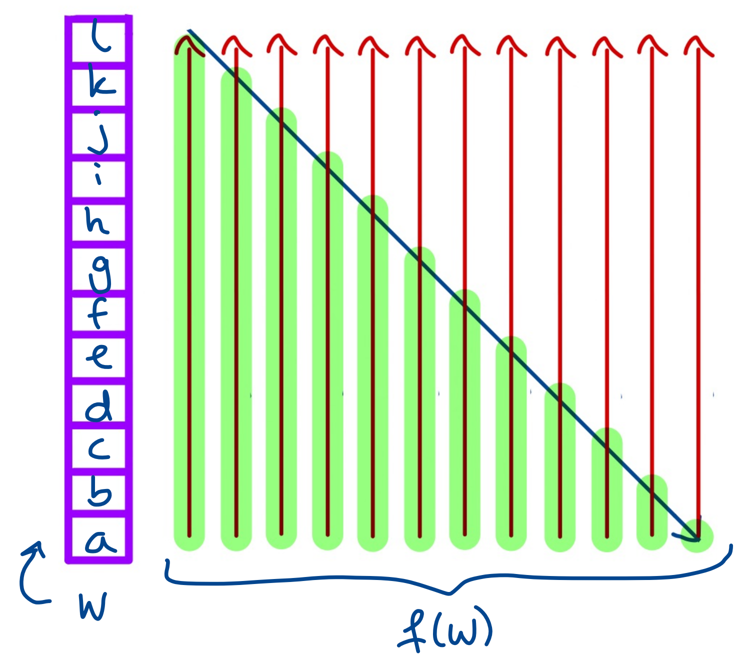
Regular functions = functions computed by deterministic 2-way transducers

For regular functions, the output length is always at most linear in the input length: $f(|w|) = \mathcal{O}(|w|)$ ⇒ linear growth rate

How can we modify the model to go <u>beyond</u> linear growth?

We equip the 2-way transducers with multiple reading heads, which can also serve as markers ("pebbles").

→ Tito's pebble transducer simulation ←



Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}: w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

Output:

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output:

Pebble transcducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output:

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

Output:

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \#w_n \longmapsto (w_0)^n\# \ldots \#(w_n)^n$

$\Downarrow$

$\downarrow$

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

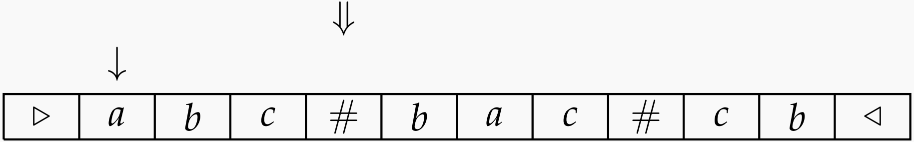
Output:

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad$ $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



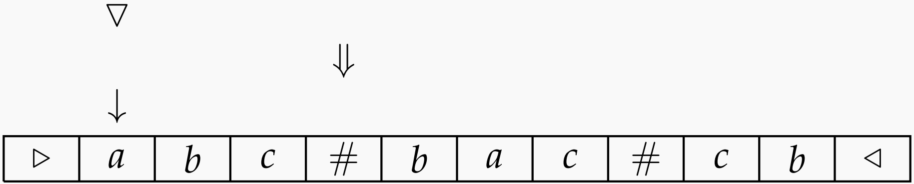
Output:

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



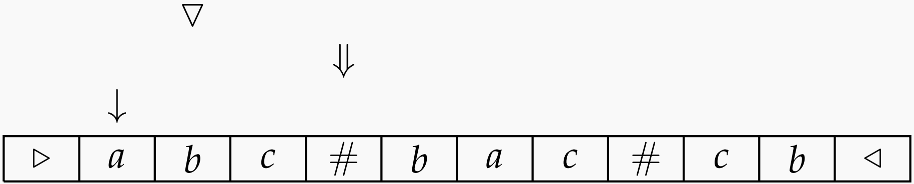
Output:

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

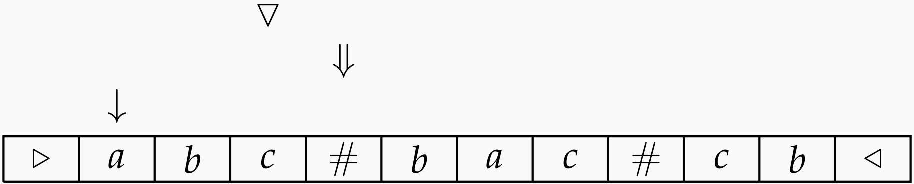


Output: a

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



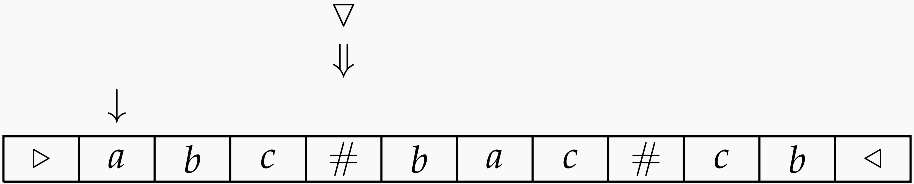
Output: $\quad ab$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq} \colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: *abc*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \mathtt{innsq} \colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

Output: $\quad abc$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $\quad abc$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad$ `innsq`: $w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

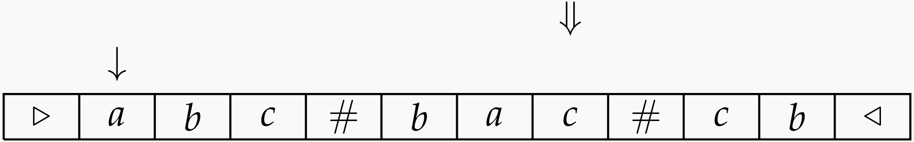
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abc$

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: *abc*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

	$\downarrow$							$\Downarrow$			
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

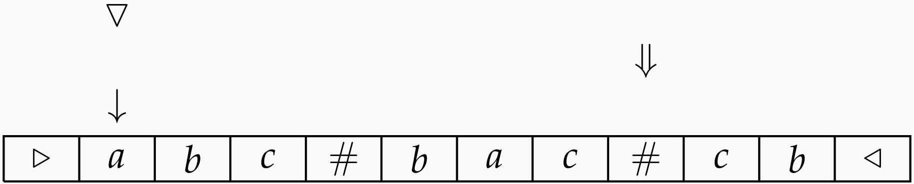
Output: $\quad abc$

Pebble transducers

Polyregular functions = computed by <ins>k-pebble transducers</ins> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



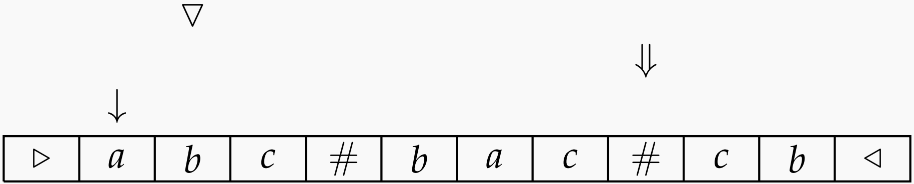
Output: $\quad abc$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



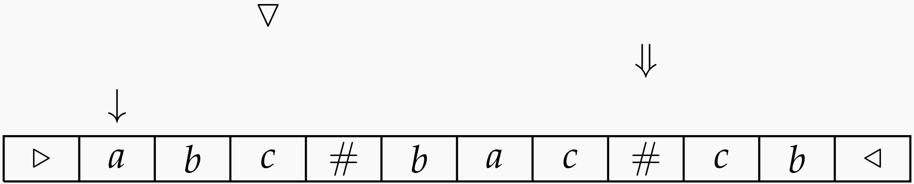
Output: *abca*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



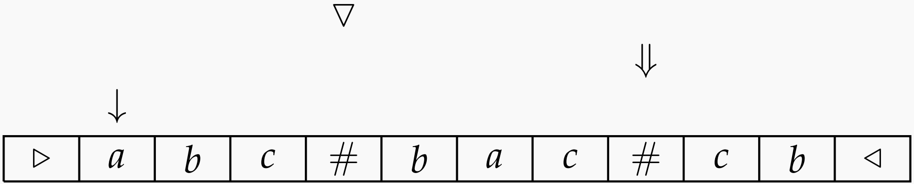
Output: *abcab*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: *abcabc*

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

	$\downarrow$							$\Downarrow$			
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

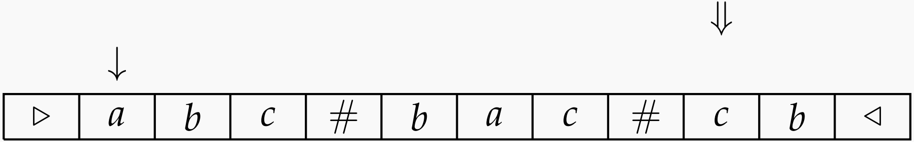
Output: $\quad abcabc$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



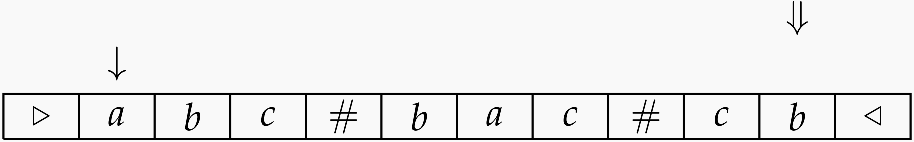
Output: *abcabc*

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



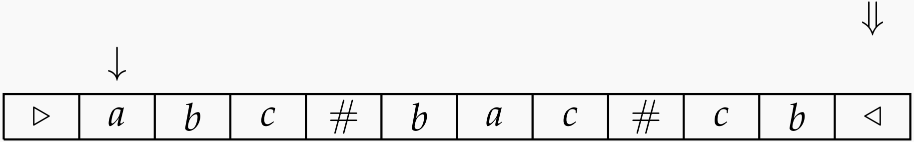
Output: $\quad$ *abcabc*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $\quad abcabc$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \#w_n \longmapsto (w_0)^n\# \ldots \#(w_n)^n$

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

Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

Output: $abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

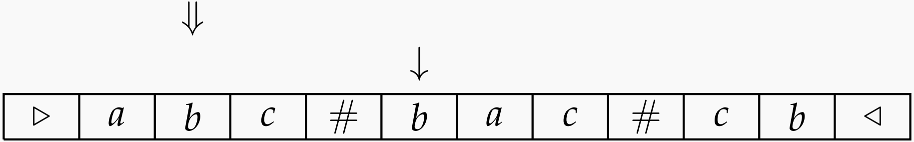
Output: $abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

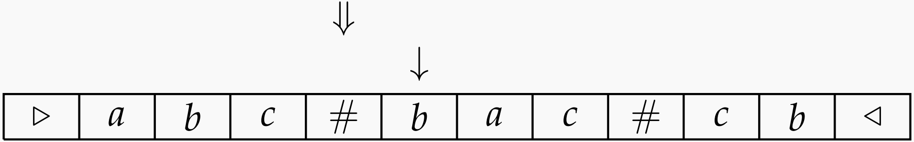
Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



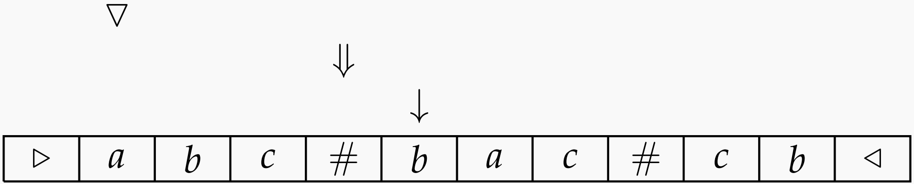
Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: *abcabc#*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

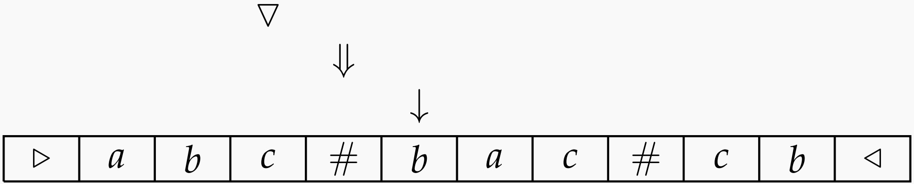
Output: $\quad abcabc\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



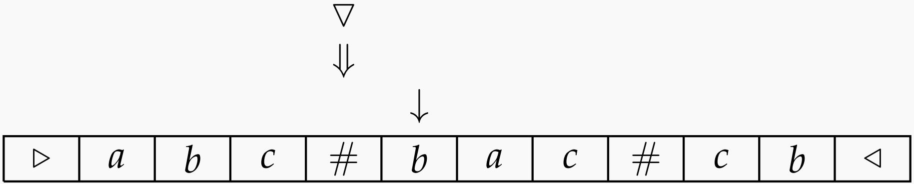
Output: *abcabc#*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



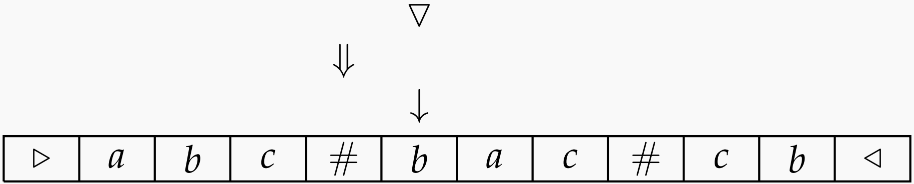
Output: $\quad$ *abcabc#*

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



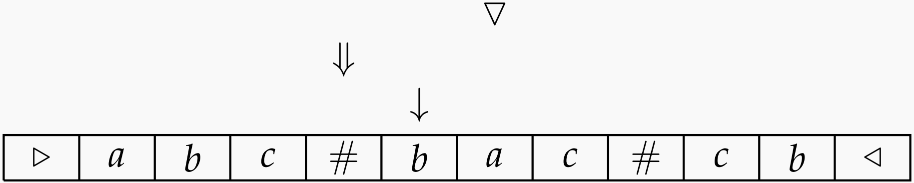
Output: *abcabc#*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



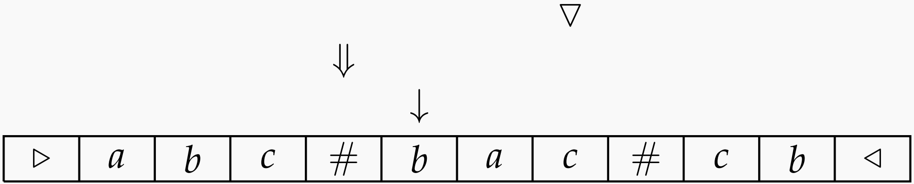
Output: *abcabc#b*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \mathtt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $\quad abcabc\#ba$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

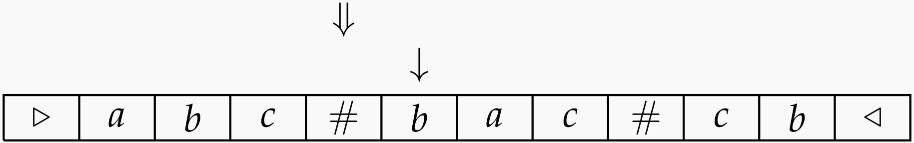
Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: *abcabc#bac*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

					$\Downarrow$ $\downarrow$						
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

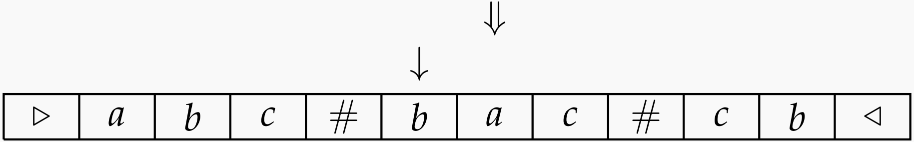
Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

							$\Downarrow$				
					$\downarrow$						
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

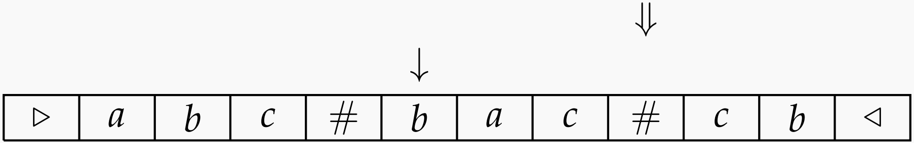
Output: $abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



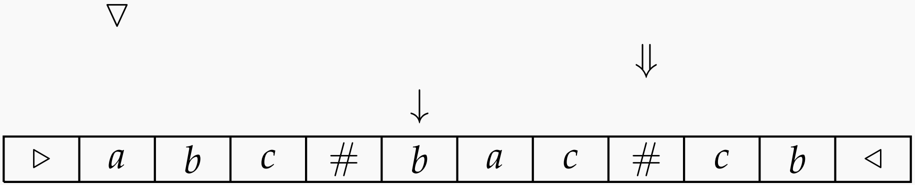
Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



Output: *abcabc#bac*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

		$\triangledown$			$\downarrow$			$\Downarrow$			
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

$\bigtriangledown$

$\Downarrow$

$\downarrow$

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

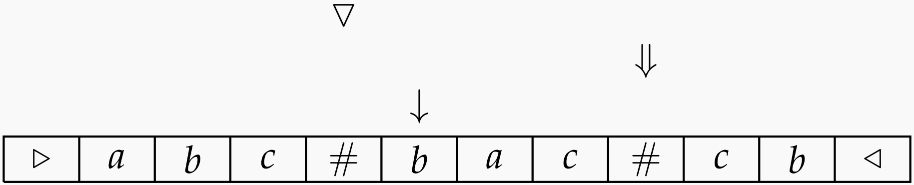
Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

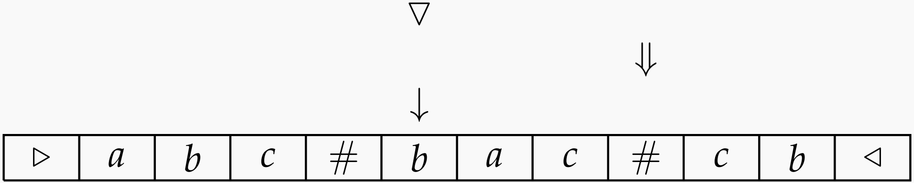


Output: $\quad abcabc\#bac$

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



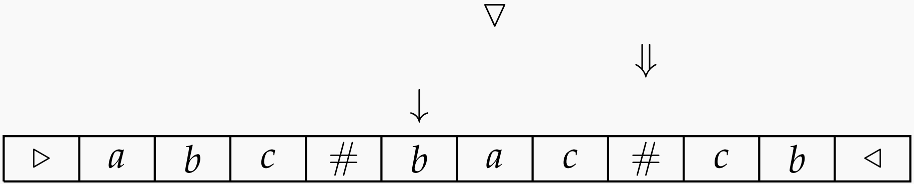
Output: $\quad abcabc\#bac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



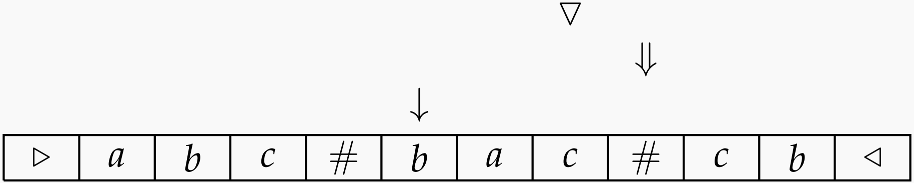
Output: $\quad abcabc\#bacb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



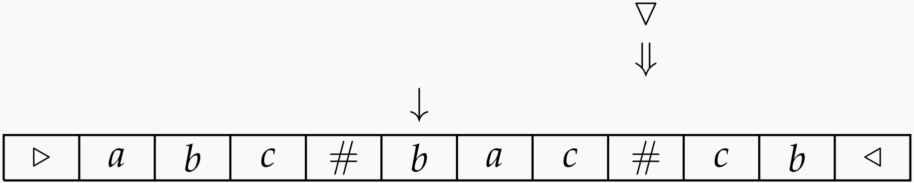
Output: $\quad abcabc\#bacba$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



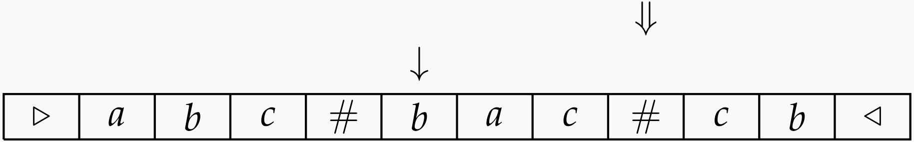
Output: $\quad abcabc\#bacbac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



Output: $\quad abcabc\#bacbac$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

Output: $abcabc\#bacbac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $\quad abcabc\#bacbac$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\;\;\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

					$\downarrow$						$\Downarrow$
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\;\; abcabc\#bacbac$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

						$\downarrow$					
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

								↓			
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

									$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

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

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

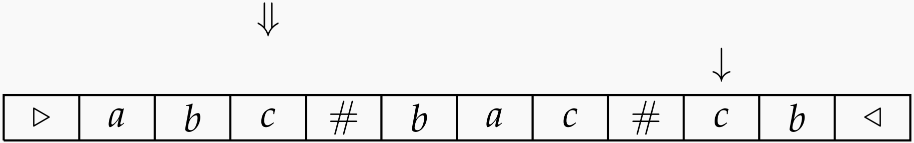
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



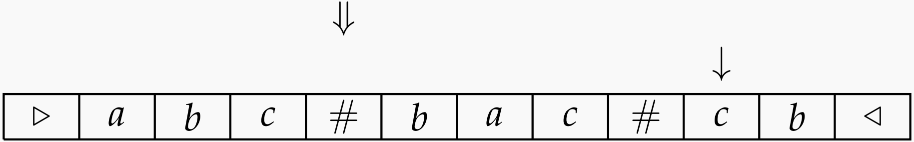
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



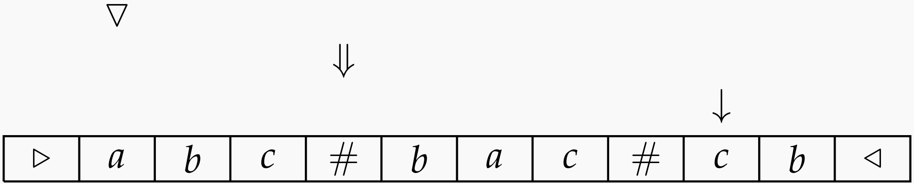
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



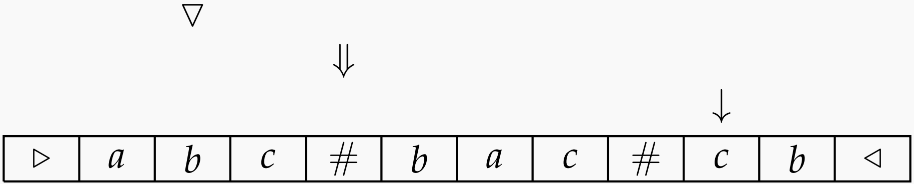
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



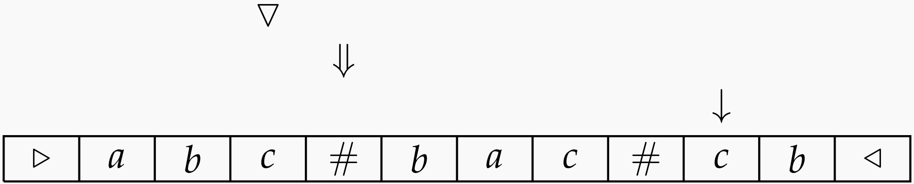
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: $abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

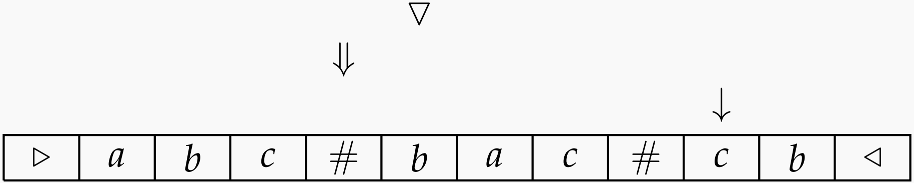
Output: $abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}: w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

$\triangledown$

$\Downarrow$

$\downarrow$

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

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

				$\Downarrow$			$\triangledown$		$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$

								$\triangledown$			
				$\Downarrow$							
									$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

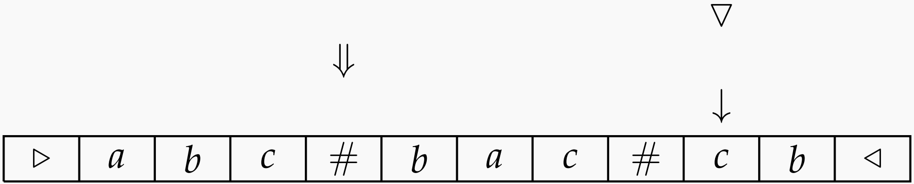
Output: $\quad abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n\# \ldots \#(w_n)^n$



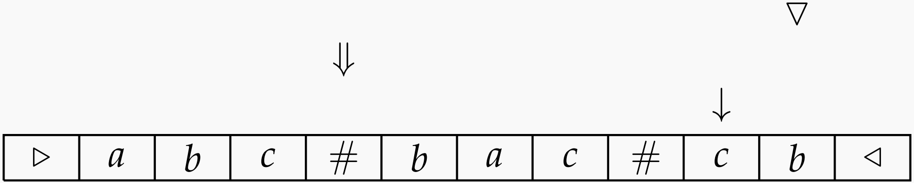
Output: $abcabc\#bacbac\#$

Pebble transducers

Polyregular functions = computed by *k*-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $abcabc\#bacbac\#c$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \#w_n \longmapsto (w_0)^n\# \dots \#(w_n)^n$

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

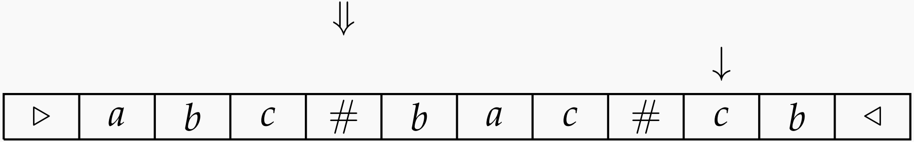
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: *abcabc#bacbac#cb*

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

					$\Downarrow$				$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

						$\Downarrow$			$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

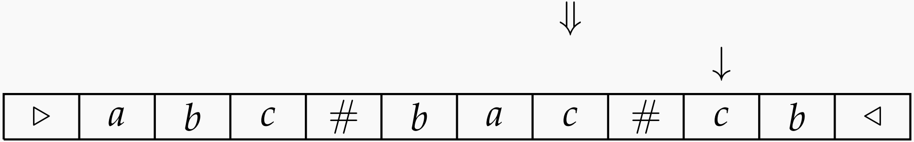
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

								$\Downarrow$	$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

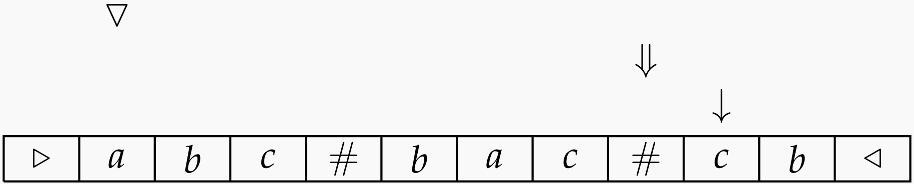
Output: $abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



Output: $abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \mathtt{innsq}\colon w_0\# \ldots \#w_n \longmapsto (w_0)^n\# \ldots \#(w_n)^n$

		∇						$\Downarrow$	$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

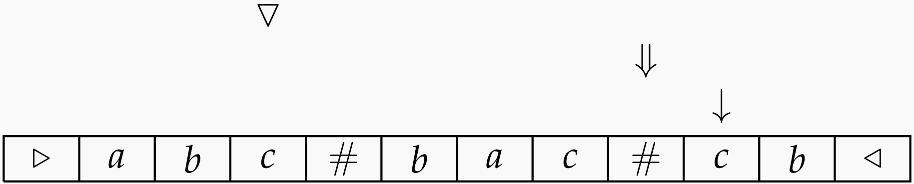
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



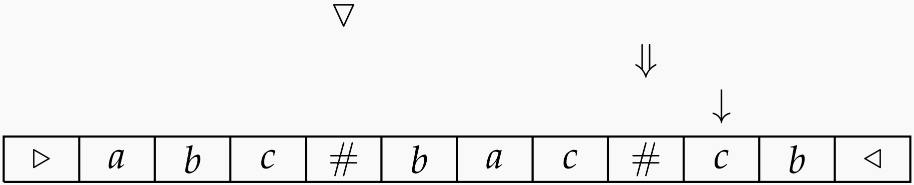
Output: $abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\mathtt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



Output: $abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

					$\triangledown$			$\Downarrow$	$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

						$\triangledown$		$\Downarrow$	$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

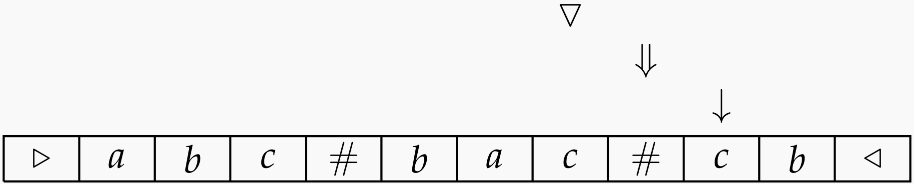
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



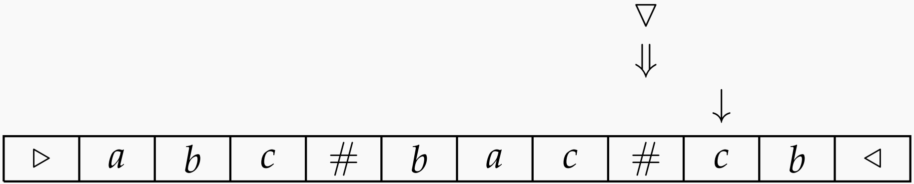
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



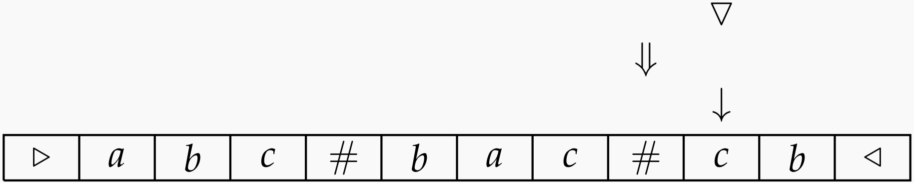
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



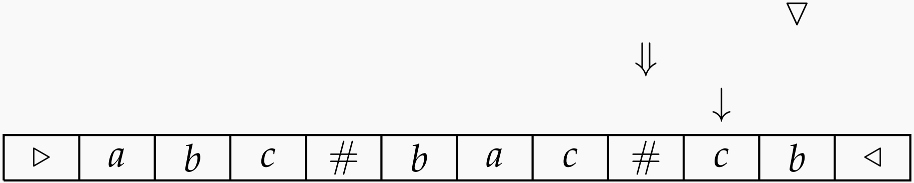
Output: $\quad abcabc\#bacbac\#cb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



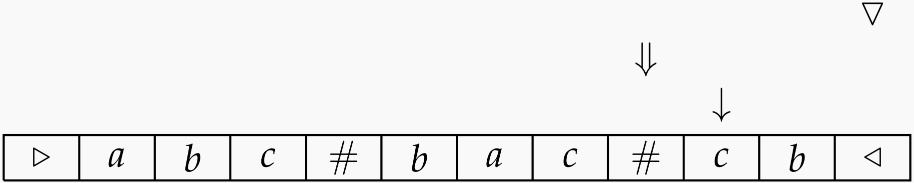
Output: *abcabc#bacbac#cbc*

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$



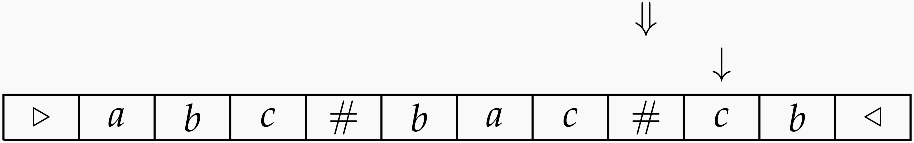
Output: $\quad abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\dots\#w_n \longmapsto (w_0)^n\#\dots\#(w_n)^n$



Output: $abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

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

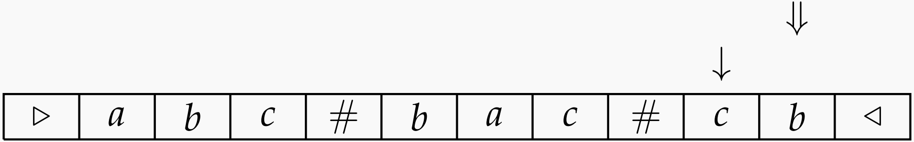
Output: $abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\texttt{innsq}\colon w_0\#\ldots\#w_n \longmapsto (w_0)^n\#\ldots\#(w_n)^n$



Output: $abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

Output: $\quad abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \dots \# w_n \longmapsto (w_0)^n \# \dots \# (w_n)^n$

									$\downarrow$		
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$

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

Output: $\quad abcabc\#bacbac\#cbcb$

Pebble transducers

Polyregular functions = computed by k-pebble transducers ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads (“pebbles”)

“Inner squaring” $\quad \texttt{innsq}\colon w_0\# \ldots \#w_n \longmapsto (w_0)^n\# \ldots \#(w_n)^n$

											$\downarrow$
$\triangleright$	a	b	c	$\#$	b	a	c	$\#$	c	b	$\triangleleft$

Output: $\quad abcabc\#bacbac\#cbcb$

From "regular" to "polyregular"

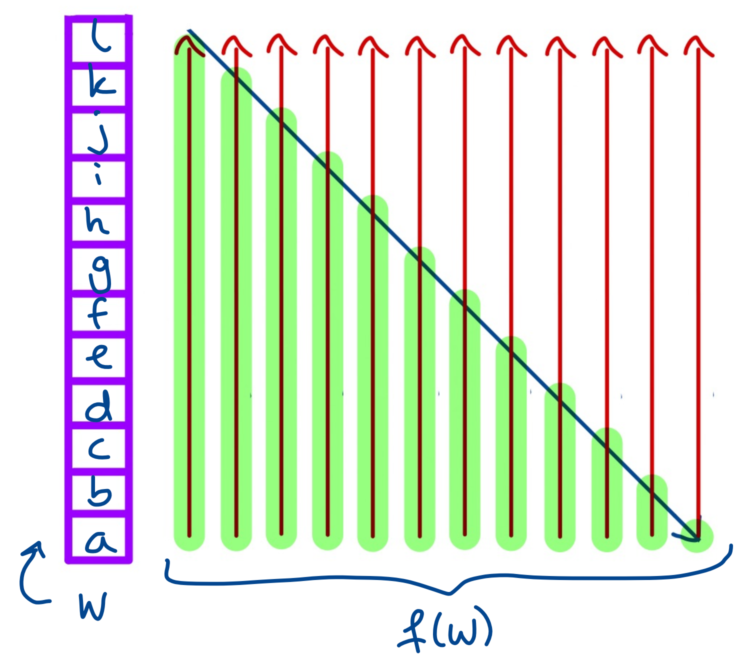
Regular functions = functions computed by deterministic 2-way transducers

For regular functions, the output length is always at most linear in the input length: $f(|w|) = O(|w|)$ ⇒ linear growth rate

How can we modify the model to go <u>beyond</u> linear growth?

We equip the 2-way transducers with multiple reading heads, which can also serve as markers ("pebbles").

→ Tito's pebble transducer simulation ←



From "regular" to "polyregular"

Regular functions = functions computed by deterministic 2-way transducers

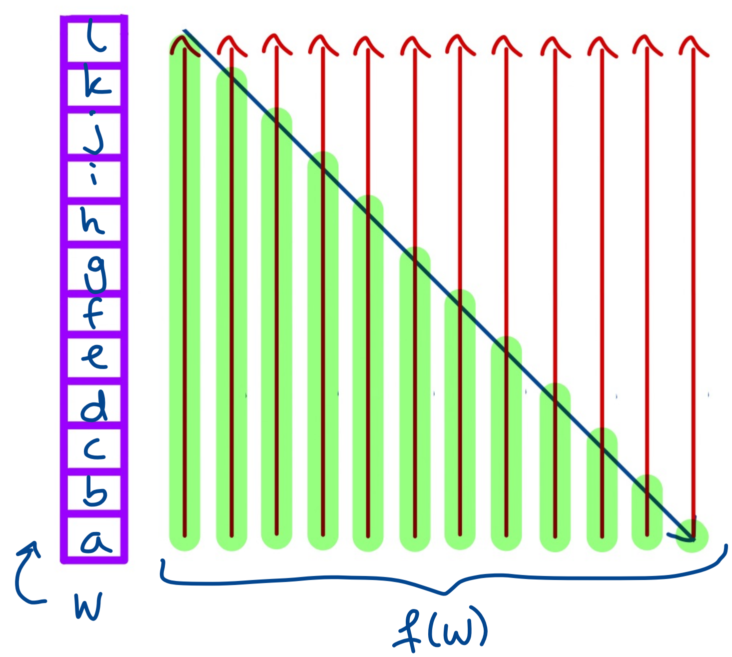
For regular functions, the output length is always at most linear in the input length: $f(|w|) = \mathcal{O}(|w|)$ $\Rightarrow$ linear growth rate

How can we modify the model to go <u>beyond</u> linear growth?

We equip the 2-way transduces with multiple reading heads, which can also serve as markers ("pebbles").

→ Tito's pebble transducer simulation ←

Configurations now depend on all k reading heads. Thus, $f(|w|) = \mathcal{O}(|w|^k)$



The history

Idea: introduce multiple reading heads to enable polynomial growth

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.

[Ibarra '71, Hartmanis '72]

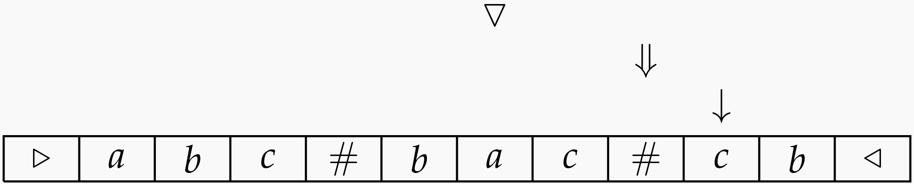
⇒ Impose stack discipline.

Pebble transducers

Polyregular functions = computed by <u>k-pebble transducers</u> ($k \geq 1$)

Finite states + *stack* of height $\leqslant k$ of two-way heads ("pebbles")

"Inner squaring" $\quad \texttt{innsq}\colon w_0\# \ldots \# w_n \longmapsto (w_0)^n \# \ldots \# (w_n)^n$



Output: $\quad abcabc\#bacbac\#cb$

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.

[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.
[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]
- String-to-string pebble transducers are closed under composition [Engelfriet, Maneth '02]

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.

[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]
- String-to-string pebble transducers are closed under composition [Engelfriet, Maneth '02]
- "Polyregular functions" [Bojańczyk '18]

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.
[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]
- String-to-string pebble transducers are closed under composition [Engelfriet, Maneth '02]
- "Polyregular functions" are characterised via [Bojańczyk '18]
  - s-to-s pebble transducers
  - closure of a certain class of s-to-s functions

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.
[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]
- String-to-string pebble transducers are closed under composition [Engelfriet, Maneth '02]
- "Polyregular functions" are characterised via [Bojańczyk '18]
  - s-to-s pebble transducers
  - closure of a certain class of s-to-s functions
  - a fragment of λ-calculus
  - for-programs

The history

Idea: introduce multiple reading heads to enable polynomial growth

Without further restrictions, the expressive power is LOGSPACE.
[Ibarra '71, Hartmanis '72]

⇒ Impose stack discipline.

- "Pebble (tree) transducers" [Milo, Suciu, Vianu '00]
- String-to-string pebble transducers are closed under composition [Engelfriet, Maneth '02]
- "Polyregular functions" are characterised via [Bojańczyk '18]
  - s-to-s pebble transducers
  - closure of a certain class of s-to-s functions
  - a fragment of λ-calculus
  - for-programs
  - s-to-s MSO interpretations

This talk!

Polyregular functions

- map strings to strings

abcd ↦ a b c d a b c a b a

Polyregular functions

- map strings to strings $\quad abcd \mapsto a\,b\,c\,d\;\; a\,b\,c\;\; a\,b\;\; a$
- positions in the output string "are" k-tuples of positions in the input string (+ finite state)

Polyregular functions

- map strings to strings $\qquad abcd \mapsto \overset{\overset{1}{4}}{a}\ b\ c\ d\ \ a\ b\ c\ \ a\ b\ \ a$

- positions in the output string "are" k-tuples of positions in the input string
  (+ finite state)

Polyregular functions

- map strings to strings

$$abcd \mapsto \overset{\substack{1\\4}}{a} \mid \overset{\substack{2\\4}}{b}\, c\, d\;\; a\, b\, c\;\; a\, b\;\; a$$

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)

Polyregular functions

- map strings to strings $\qquad abcd \mapsto \overset{\overset{1}{4}}{a} \mid \overset{\overset{2}{4}}{b} \mid \overset{\overset{3}{4}}{c} \mid \overset{\overset{4}{4}}{d} \quad a\ b\ c \quad a\ b \quad a$

- positions in the output string "are" k-tuples of positions in the input string
  (+ finite state)

Polyregular functions

- map strings to strings

$$abcd \mapsto \overset{\substack{1\\4}}{a} \mid \overset{\substack{2\\4}}{b} \mid \overset{\substack{3\\4}}{c} \mid \overset{\substack{4\\4}}{d} \mid \overset{\substack{1\\3}}{a} \mid \overset{\substack{2\\3}}{b} \mid \overset{\substack{3\\3}}{c} \quad a \quad b \quad a$$

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)

Polyregular functions

- map strings to strings

$$abcd \mapsto \begin{array}{c|c|c|c|c|c|c|c|c} \substack{1\\4} & \substack{2\\4} & \substack{3\\4} & \substack{4\\4} & \substack{1\\3} & \substack{2\\3} & \substack{3\\3} & \substack{1\\2} & \substack{2\\2} \\ a & b & c & d & a & b & c & a & b \end{array} \; a$$

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)

Polyregular functions

- map strings to strings

$abcd \mapsto$

1	2	3	4	1	2	3	1	2	1
4	4	4	4	3	3	3	2	2	1
a	b	c	d	a	b	c	a	b	a

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)

Polyregular functions

- map strings to strings

$$abcd \mapsto \begin{array}{c|c|c|c|c|c|c|c|c|c} \overset{1}{4} & \overset{2}{4} & \overset{3}{4} & \overset{4}{4} & \overset{1}{3} & \overset{2}{3} & \overset{3}{3} & \overset{1}{2} & \overset{2}{2} & \overset{1}{1} \\ a & b & c & d & a & b & c & a & b & a \end{array}$$

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)
- characterisations via : pebble transducers, for-programs, combinators, logics

Polyregular functions

- map strings to strings

$$abcd \mapsto \begin{array}{c|c|c|c|c|c|c|c|c|c} 1 & 2 & 3 & 4 & 1 & 2 & 3 & 1 & 2 & 1 \\ 4 & 4 & 4 & 4 & 3 & 3 & 3 & 2 & 2 & 1 \\ a & b & c & d & a & b & c & a & b & a \end{array}$$

- positions in the output string "are" k-tuples of positions in the input string (+ finite state)
- characterisations via: pebble transducers, combinators, for-programs, logics

for-programs are of the shape

```
for i = n to 1
    for j = 1 to n
        if j ≤ i output w(j)
```

Polyregular functions

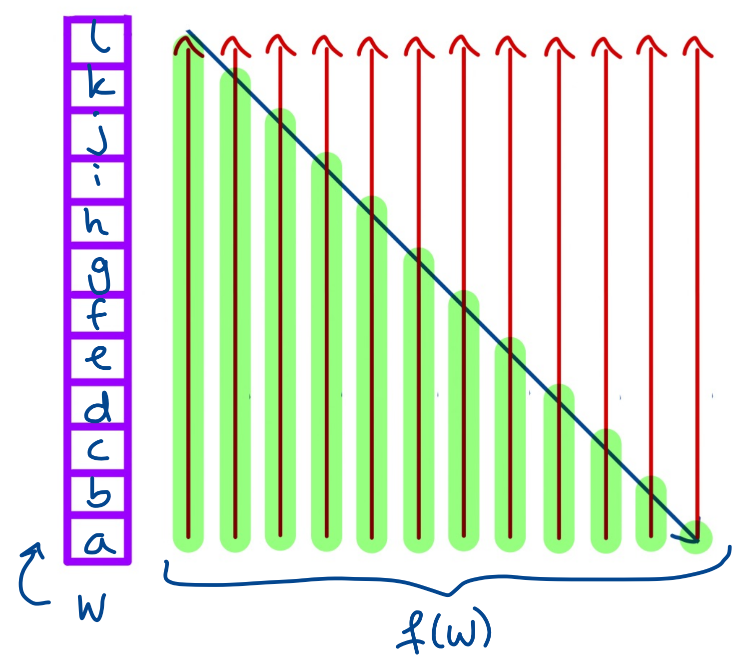
- map strings to strings $\qquad abcd \mapsto \overset{\substack{1\\4}}{a} \mid \overset{\substack{2\\4}}{b} \mid \overset{\substack{3\\4}}{c} \mid \overset{\substack{4\\4}}{d} \mid \overset{\substack{1\\3}}{a} \mid \overset{\substack{2\\3}}{b} \mid \overset{\substack{3\\3}}{c} \mid \overset{\substack{1\\2}}{a} \mid \overset{\substack{2\\2}}{b} \mid \overset{\substack{1\\1}}{a}$
- positions in the output string "are" k-tuples of positions in the input string (+ finite state)
- characterisations via: pebble transducers, combinators, for-programs, logics

for-programs are of the shape

```
for i = n to 1
    for j = 1 to n
        if j ≤ i output w(j)
```

for-loops correspond to spawned pebbles (≈ position markers) in the transducers.

The pebbles obey a stack discipline.



Polyregular functions - Logical characterisation

Concatenation of prefixes

$$abcd \mapsto \overset{\overset{1}{4}}{a} \mid \overset{\overset{2}{4}}{b} \mid \overset{\overset{3}{4}}{c} \mid \overset{\overset{4}{4}}{d} \mid \overset{\overset{1}{3}}{a} \mid \overset{\overset{2}{3}}{b} \mid \overset{\overset{3}{3}}{c} \mid \overset{\overset{1}{2}}{a} \mid \overset{\overset{2}{2}}{b} \mid \overset{\overset{1}{1}}{a}$$

```
for i = n to 1
    for j = 1 to n
        if j ≤ i output w(j)
```

Polyregular functions - Logical characterisation

Concatenation of prefixes

$$abcd \mapsto \begin{array}{c|c|c|c|c|c|c|c|c|c} \overset{1}{4} & \overset{2}{4} & \overset{3}{4} & \overset{4}{4} & \overset{1}{3} & \overset{2}{3} & \overset{3}{3} & \overset{1}{2} & \overset{2}{2} & \overset{1}{1} \\ a & b & c & d & a & b & c & a & b & a \end{array}$$

```
for i = n to 1
    for j = 1 to n
        if j ≤ i output w(j)
```

We can describe the output via

- a domain formula $\varphi_{dom}(i,j) = j \leq i$
- a total-order formula $\varphi_{\leq}(i,j,i',j') = (i \geq i') \vee ((i = i') \wedge (j \leq j'))$
- label formulas $\varphi_a(i,j) = a(j)$

Polyregular functions - Logical characterisation

Concatenation of prefixes

$$abcd \mapsto \begin{array}{cccc|ccc|cc|c} \overset{1}{4} & \overset{2}{4} & \overset{3}{4} & \overset{4}{4} & \overset{1}{3} & \overset{2}{3} & \overset{3}{3} & \overset{1}{2} & \overset{2}{2} & \overset{1}{1} \\ a & b & c & d & a & b & c & a & b & a \end{array}$$

```
for i=n to 1
  for j=1 to n
    if j≤i output w(j)
```

We can describe the output via

- a domain formula $\varphi_{dom}(i,j) = j \leq i$
- a total-order formula $\varphi_{\leq}(i,j,i',j') = (i \geq i') \vee ((i = i') \wedge (j \leq j'))$
- label formulas $\varphi_a(i,j) = a(j)$

Together, the formulae describe a 2-dim. string-to-string interpretation.

MSO transductions for regular functions = 1-dimensional case

Polyregular functions - Logical characterisation

Concatenation of prefixes

$$abcd \mapsto$$

1	2	3	4	1	2	3	1	2	1
4	4	4	4	3	3	3	2	2	1
a	b	c	d	a	b	c	a	b	a

```
for i:=n to 1
   for j:=1 to n
      if j≤i output w(j)
```

We can describe the output via

- a domain formula $\varphi_{dom}(i,j) = j \leq i$
- a total-order formula $\varphi_{\leq}(i,j,i',j') = (i \geq i') \vee ((i=i') \wedge (j \leq j'))$
- label formulas $\varphi_a(i,j) = a(j)$

Together, the formulae describe a 2-dim. string-to-string interpretation.

MSO transductions for regular functions = 1-dimensional case

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n$ all $w_i \in \{a,b\}^*$

$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$

Polyregular functions - Logical characterisation

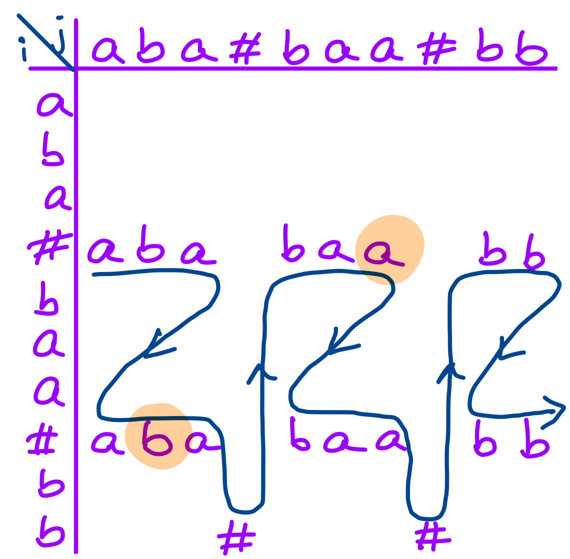
Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n$ all $w_i \in \{a,b\}^*$

$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$



Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

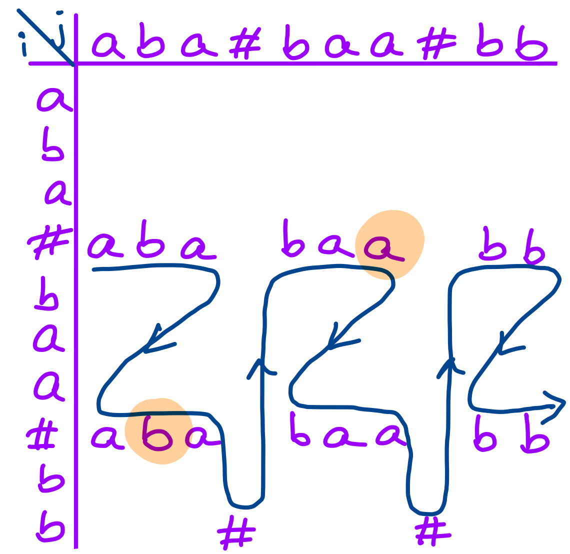
[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

$$\text{innsq}: w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n \qquad \text{all } w_i \in \{a,b\}^*$$

$$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$$

We define the corresponding MSO interpretation.



- (domain &) label formulas

$$\varphi_a(i,j) = \#(i) \wedge a(j)$$

$$\varphi_b(i,j) = \#(i) \wedge b(j)$$

$$\varphi_\#(i,j) = \max(i) \wedge \#(j)$$

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

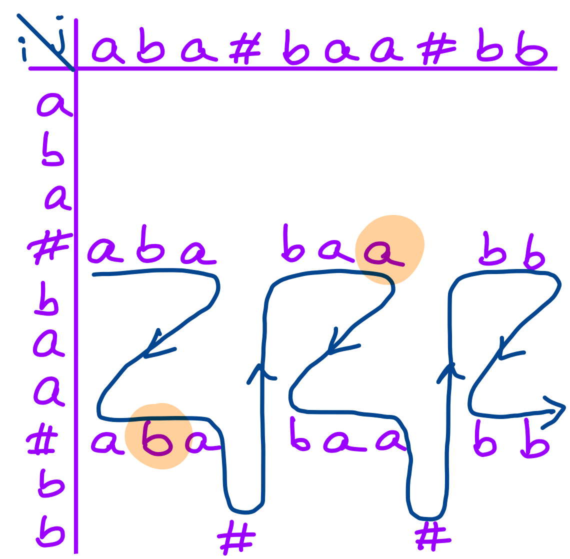
[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

$$\mathrm{innsq}: w_0\#\dots\#w_n \mapsto w_0^n\#\dots\#w_n^n \qquad \text{all } w_i \in \{a,b\}^*$$

$$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$$

We define the corresponding MSO interpretation.



- (domain &) label formulas

$$\varphi_a(i,j) = \#(i) \wedge a(j) \qquad \varphi_\#(i,j) = \max(i) \wedge \#(j)$$

$$\varphi_b(i,j) = \#(i) \wedge b(j)$$

- $\varphi_\leq(i,j,i',j') = \big(\#(j) \wedge (j \leq j')\big)$
  $\vee\ \exists s',s : \big((s \leq j \ \wedge\ s' \leq j')$
  $\wedge$ # neither between s,j nor between s',j'
  $\wedge$ neither s nor s' has a direct predecessor a or b
  $\wedge$ lexicographically $(s,i,j) \leq (s',i',j')\big)$

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

The proof employs the definition of polyregular functions via for-programs.

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

The proof employs the definition of polyregular functions via for-programs.

Then, for $\leq$: order formula $\equiv$ reachability between program states

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

The proof employs the definition of polyregular functions via for-programs.

Then, for $\leq$: order formula $\equiv$ reachability between program states

So it remains to prove:

Functions computable by for-programs $\supseteq$ Functions definable via string-to-string interpretations

That is: every MSO-definable total order implicitly respects stack discipline.

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

The proof employs the definition of polyregular functions via for-programs.

Then, for $\leq$: order formula $\equiv$ reachability between program states

So it remains to prove:

Functions computable by for-programs $\supseteq$ Functions definable via string-to-string interpretations

That is: every MSO-definable total order implicitly respects stack discipline.

To show this "domination of variables", we use:

- Simon's factorisation forest theorem (+induction)
  every string can be cut into pieces that are similar
  "blocks"

Polyregular functions - Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

The proof employs the definition of polyregular functions via for-programs.

Then, for $\leq$: order formula $\equiv$ reachability between program states

So it remains to prove:

Functions computable by for-programs $\supseteq$ Functions definable via string-to-string interpretations

That is: every MSO-definable total order implicitly respects stack discipline.

To show this "domination of variables", we use:

- Simon's factorisation forest theorem (+induction)
  every string can be cut into pieces that are similar
  "blocks"
- our Domination Lemma
  tuples from distinct blocks obey stack discipline

Domination on rationals

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

That is, for every "type" of position tuples, there is always a dominating coordinate, which determines the $<$ relation.

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

That is, for every "type" of position tuples, there is always a dominating coordinate, which determines the $<$ relation.

Consider the following toy result on a single type of tuples.

Every quantifier-free total order on
$$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
is lexicographic.

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

That is, for every "type" of position tuples, there is always a dominating coordinate, which determines the $<$ relation.

Consider the following toy result on a single type of tuples.

> Every quantifier-free total order on
> $$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
> is lexicographic.

$k=1$: $\varphi_\leq(x,y) = x \leq y$ or $\varphi_\leq(x,y) = y \leq x$

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

That is, for every "type" of position tuples, there is always a dominating coordinate, which determines the $<$ relation.

Consider the following toy result on a single type of tuples.

> Every quantifier-free total order on
> $$\{(x_1,\dots,x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
> is lexicographic.

$k=1$: $\varphi_\leq(x,y) = x \leq y$ or $\varphi_\leq(x,y) = y \leq x$

$k=2$: Case analysis of possible relations between pairs

Domination on rationals

The domination lemma is very technical, but it essentially says that every FO-definable linear order $<$ on position tuples obeys an implicit stack discipline.

That is, for every "type" of position tuples, there is always a dominating coordinate, which determines the $<$ relation.

Consider the following toy result on a single type of tuples.

Every quantifier-free total order on
$$\{(x_1,..,x_k) : x_1 < \ldots < x_k \text{ are rationals}\}$$
is lexicographic.

$k=1$: $\varphi_\leq(x,y) = x \leq y$ or $\varphi_\leq(x,y) = y \leq x$

$k=2$: Case analysis of possible relations between pairs

$k>2$: Reduction to $k=2$

Domination on rationals : $k > 2$

Every quantifier-free total order on
$$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
is lexicographic.

Domination on rationals : $k > 2$

Every quantifier-free total order on
$$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
is lexicographic.

To show domination for $k > 2$, we use the case $k = 2$.

Thus, for every pair of coordinates, there is a dominating one.

Domination on rationals : $k>2$

Every quantifier-free total order on
$$\{(x_1,\dots,x_k) : x_1 < \dots < x_k \text{ are rationals}\}$$
is lexicographic.

To show domination for $k>2$, we use the case $k=2$.

Thus, for every pair of coordinates, there is a dominating one.

Then it suffices to show that domination is transitive, i.e.

$$i \to j \ \wedge \ j \to d \quad \Rightarrow \quad i \to d$$

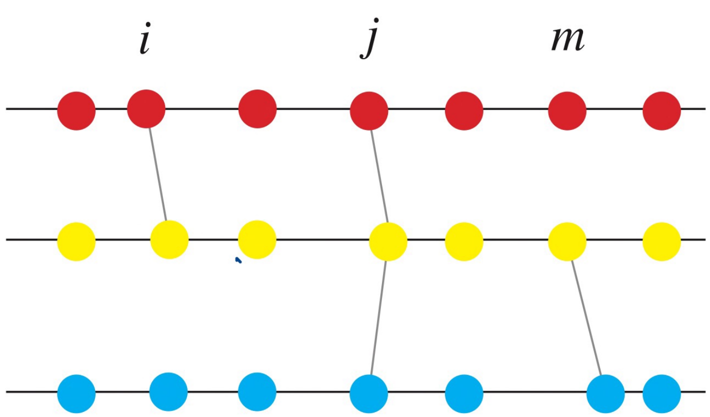
Domination on rationals : $k > 2$

Every quantifier-free total order on
$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$
is lexicographic.

To show domination for $k > 2$, we use the case $k = 2$.
Thus, for every pair of coordinates, there is a dominating one.

Then it suffices to show that domination is transitive, i.e.

$$i \to j \;\wedge\; j \to d \;\Rightarrow\; i \to d$$



Step 1. Move coordinate i to its final position, and move j so that the tuple grows

Step 2. Move coordinate j back to the initial, position and move m to its final position

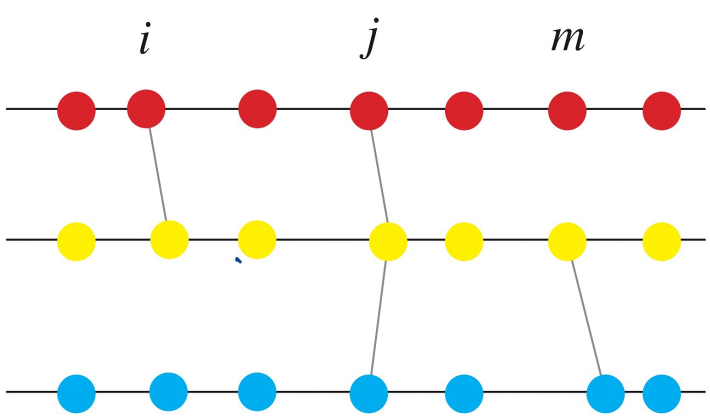
Domination on rationals : $k > 2$

Every quantifier-free total order on
$\{(x_1, \dots, x_k) : x_1 < \dots < x_k \text{ are rationals}\}$
is lexicographic.

To show domination for $k>2$, we use the case $k=2$.
Thus, for every pair of coordinates, there is a dominating one.

Then it suffices to show that domination is transitive, i.e.

$$i \to j \;\wedge\; j \to d \;\Rightarrow\; i \to d$$



Step 1. Move coordinate i to its final position, and move j so that the tuple grows

Step 2. Move coordinate j back to the initial, position and move m to its final position

$\rightsquigarrow$ One coordinate dominates globally!

□

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in O(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in \mathcal{O}(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in O(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Polyregular functions = the functions definable via string-to-string MSO interpretations

Does the growth-rate exponent coincide with the dimension?

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in \mathcal{O}(|w|^k)$
  (+ finite state) polynomial "growth rate"

What about the converse?

Polyregular functions = the functions definable via string-to-string MSO interpretations

Does the growth-rate exponent coincide with the dimension?

YES!

A polyregular function has output size $\mathcal{O}(n^k)$.

$\Longleftrightarrow$

The function can be defined via a k-dimensional MSO interpretation.

[Bojańczyk '23]

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in \mathcal{O}(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Polyregular functions = the functions definable via string-to-string MSO interpretations

Does the growth-rate exponent coincide with the dimension?

YES!

A polyregular function has output size $\mathcal{O}(n^k)$.

$\Leftrightarrow$

The function can be defined via a k-dimensional MSO interpretation.

[Bojańczyk '23]

Does the number of needed pebbles also match then?

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in \mathcal{O}(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Polyregular functions = the functions definable via string-to-string MSO interpretations

Does the growth-rate exponent coincide with the dimension?

YES!

A polyregular function has output size $\mathcal{O}(n^k)$.

$\Longleftrightarrow$

The function can be defined via a k-dimensional MSO interpretation.

[Bojańczyk '23]

Does the number of needed pebbles also match then?

Clearly, it holds that : k pebbles $\Rightarrow$ $\mathcal{O}(n^k)$ growth

Polyregular functions : Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n$ all $w_i \in \{a,b\}^*$

$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$

Polyregular functions : Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

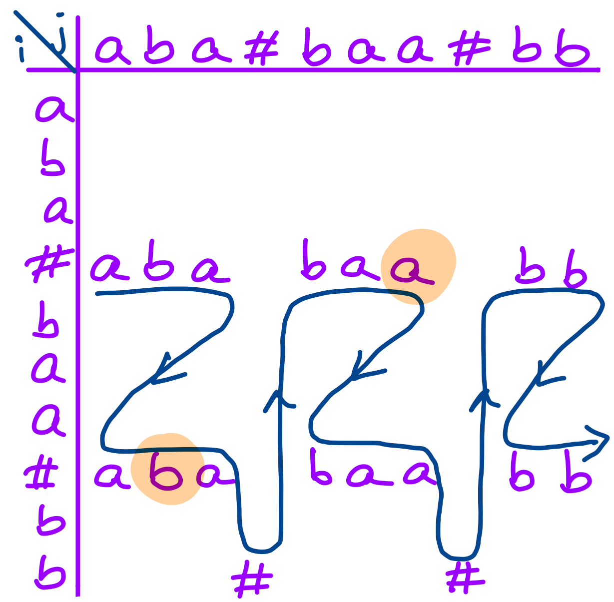
[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

$$\text{innsq}: \; w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n \qquad \text{all } w_i \in \{a,b\}^*$$

$$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$$

We define the corresponding MSO transduction.



- (domain &) label formulas

$\varphi_a(i,j) = \#(i) \wedge a(j)$

$\varphi_b(i,j) = \#(i) \wedge b(j)$

$\varphi_\#(i,j) = \max(i) \wedge \#(j)$

- $\varphi_\leq(i,j,i',j') = \big(\#(j) \wedge (j \leq j')\big)$

$\vee \; \exists s', s : \big((s \leq j \;\wedge\; s' \leq j')$

$\wedge$ # neither between s,j nor between s',j'

$\wedge$ neither s nor s' has a direct predecessor a or b

$\wedge$ lexicographically $(s,i,j) \leq (s',i',j')\big)$

Polyregular functions : Logical characterisation

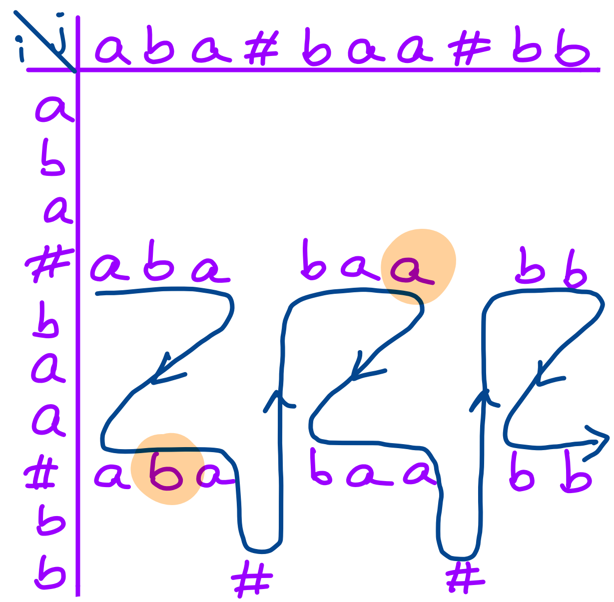
Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n$ all $w_i \in \{a,b\}^*$

$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$



Polyregular functions : Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

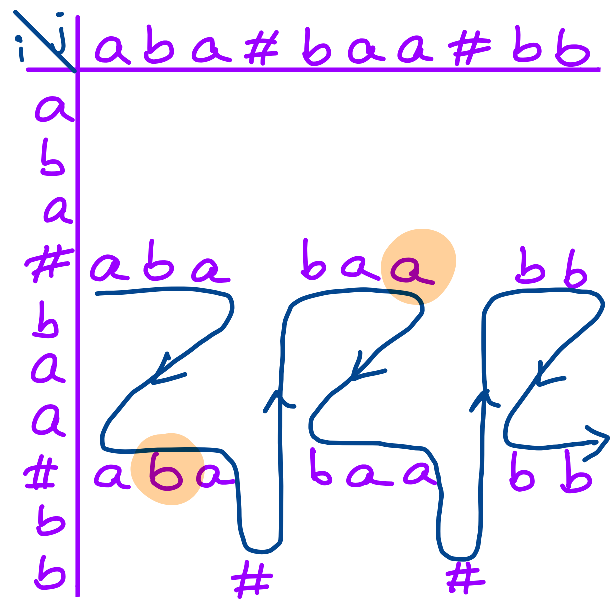
[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \cdots \# w_n \mapsto w_0^n \# \cdots \# w_n^n$ all $w_i \in \{a,b\}^*$

aba#baa#bb $\mapsto$ abaaba#baabaa#bbbb

It looks like we need 3 pebbles:



Polyregular functions : Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

innsq: $w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n$ all $w_i \in \{a,b\}^*$

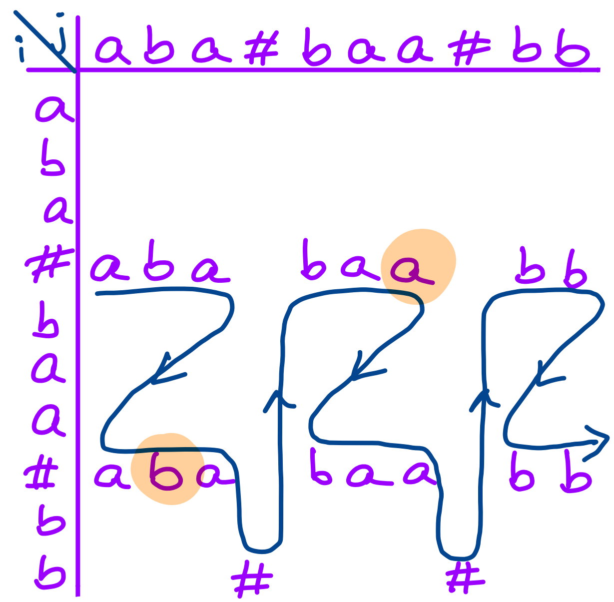
$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$

It looks like we need 3 pebbles:

1) one to mark the (beginning of the) currently copied subword w_i

2) one to count the copies that are output

3) one to actually copy the current subword w_i



Polyregular functions : Logical characterisation

Polyregular functions = the functions definable via string-to-string MSO interpretations

[Bojańczyk, K., Lhote '19]

Consider the inner squaring function

$$\text{innsq}: \; w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n \qquad \text{all } w_i \in \{a,b\}^*$$

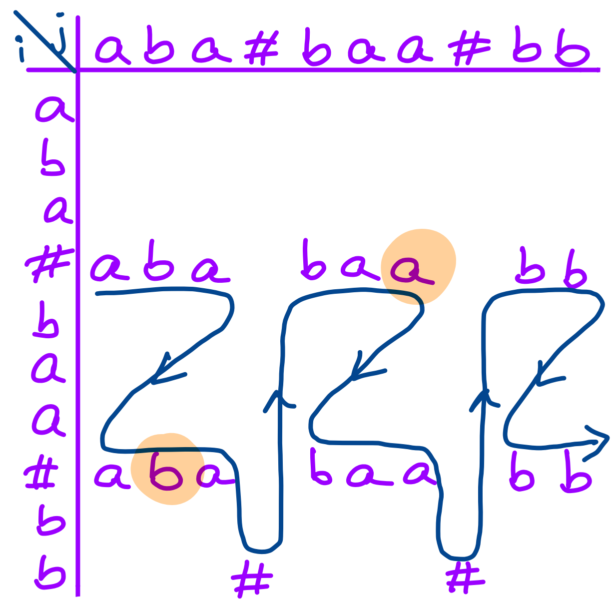
$$aba\#baa\#bb \mapsto abaaba\#baabaa\#bbbb$$

It looks like we need 3 pebbles:

1) one to mark the (beginning of the) currently copied subword w_i

2) one to count the copies that are output

3) one to actually copy the current subword w_i



Does growth-rate exponent k imply that k pebbles suffice?

Polyregular functions: Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in O(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Does growth-rate exponent k imply that k pebbles suffice?

Polyregular functions : Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in O(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Does growth-rate exponent k imply that k pebbles suffice?

NO!

No constant number of pebbles suffices to compute all polyregular functions with growth rate exponent $k=2$.
[Bojańczyk '23]

↯ to main result in LICS 2020 paper

Polyregular functions: Growth

- output positions are k-tuples of input positions $\Rightarrow |f(w)| \in O(|w|^k)$
  (+ finite state)
  polynomial "growth rate"

What about the converse?

Does growth-rate exponent k imply that k pebbles suffice?

NO!

No constant number of pebbles suffices to compute all polyregular functions with growth rate exponent $k=2$,
[Bojańczyk '23]

↯ to main result in LICS 2020 paper

For example, $\mathrm{innsq} \notin \mathrm{Pebble}_2$.

$\mathrm{innsq}: w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n$

all $w_i \in \{a,b\}^*$

Polyregular functions: Growth

- output positions are k-tuples of input positions (+ finite state) $\Rightarrow |f(w)| \in O(|w|^k)$
  polynomial "growth rate"

What about the converse?

Does growth-rate exponent k imply that k pebbles suffice?

NO!

No constant number of pebbles suffices to compute all polyregular functions with growth rate exponent $k=2$, [Bojańczyk '23]

↯ to main result in LICS 2020 paper

For example, innsq $\notin$ Pebble$_2$.

innsq: $w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n$

all $w_i \in \{a,b\}^*$

Our contribution: Easier proofs for the above. [K., Nguyen, Pradic '23]

This talk: Easier proof for innsq $\notin$ Pebble$_2$.

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq} : \{a, b, \#\}^* \to \{a, b, \#\}^*$$
$$w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n \qquad (\text{all } w_i \in \{a, b\}^*)$$

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq} : \{a, b, \#\}^* \to \{a, b, \#\}^*$$
$$w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

It suffices to show:

> No function in Pebble_2 coincides with innsq on $(a^* b \#)^* \#^*$.

Assume that there is such a function.

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq} : \{a,b,\#\}^* \to \{a,b,\#\}^*$$

$$w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

It suffices to show:

> No function in Pebble_2 coincides with innsq on $(a^*b\#)^*\#^*$.

Assume that there is such a function.

$\Downarrow$ using the arguments from the next slide

Then there must be $L \subseteq b\{a,b\}^*b$ that

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq}: \{a,b,\#\}^* \to \{a,b,\#\}^*$$

$$w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

It suffices to show:

> No function in Pebble_2 coincides with innsq on $(a^*b\#)^*\#^*$.

Assume that there is such a function.

$\Downarrow$ using the arguments from the next slide

Then there must be $L \subseteq b\{a,b\}^*b$ that

- is the output of some 2-way transducer (i.e. the image of a regular function)

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq} : \{a, b, \#\}^* \to \{a, b, \#\}^*$$

$$w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

It suffices to show:

> No function in Pebble_2 coincides with innsq on $(a^* b \#)^* \#^*$.

Assume that there is such a function.

$\Downarrow$ using the arguments from the next slide

Then there must be $L \subseteq b\{a,b\}^* b$ that

- is the output of some 2-way transducer (i.e. the image of a regular function)
- consists of infixes of elements in $\mathrm{innsq}((a^* b \#)^* \#^*)$

Inner squaring cannot be done with 2 pebbles

$\mathrm{innsq} : \{a,b,\#\}^* \to \{a,b,\#\}^*$

$w_0 \# \ldots \# w_n \mapsto w_0^n \# \ldots \# w_n^n$ (all $w_i \in \{a,b\}^*$)

It suffices to show:

No function in Pebble_2 coincides with innsq on $(a^*b\#)^*\#^*$.

Assume that there is such a function.

$\Downarrow$ using the arguments from the next slide

Then there must be $L \subseteq b\{a,b\}^*b$ that

- is the output of some 2-way transducer (i.e. the image of a regular function)
- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $\underbrace{ba\ldots ab \ldots ba\ldots ab}_{\text{all } a\text{-blocks have length } n \geq N}$ with at least N b's.

Inner squaring cannot be done with 2 pebbles

$$\mathrm{innsq} : \{a,b,\#\}^* \to \{a,b,\#\}^*$$

$$w_0\#\ldots\#w_n \mapsto w_0^n\#\ldots\#w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

It suffices to show:

> No function in Pebble_2 coincides with innsq on $(a^*b\#)^*\#^*$.

Assume that there is such a function.

$\Downarrow$ using the arguments from the next slide

Then there must be $L \subseteq b\{a,b\}^*b$ that

- is the output of some 2-way transducer (i.e. the image of a regular function)
- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $\underbrace{ba\ldots ab\ldots ba\ldots ab}_{\text{all } a\text{-blocks have length } n \geq N}$ with at least N b's.

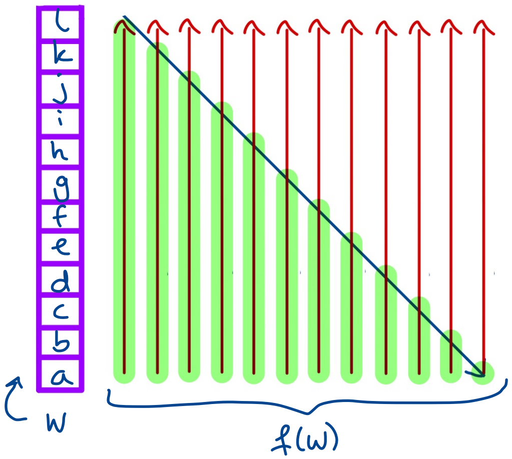
We apply a pumping argument to L to conclude that it cannot exist.

Inner squaring cannot be done with 2 pebbles

Assume there is $f \in \mathrm{Pebble}_2$ that coincides with innsq on $(a^*b\#)^*\#^*$,

Inner squaring cannot be done with 2 pebbles

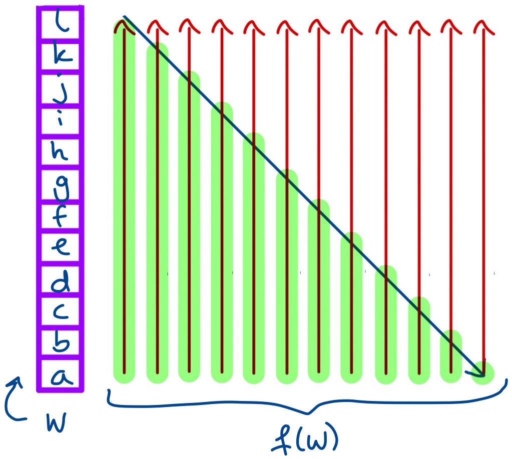
Assume there is $f \in \mathrm{Pebble}_2$ that coincides with innsq on $(a^*b\#)^*\#^*$.



f can be obtained by adequately nesting regular functions.

Inner squaring cannot be done with 2 pebbles

Assume there is $f \in \mathrm{Pebble}_2$ that coincides with innsq on $(a^*b\#)^*\#^*$,



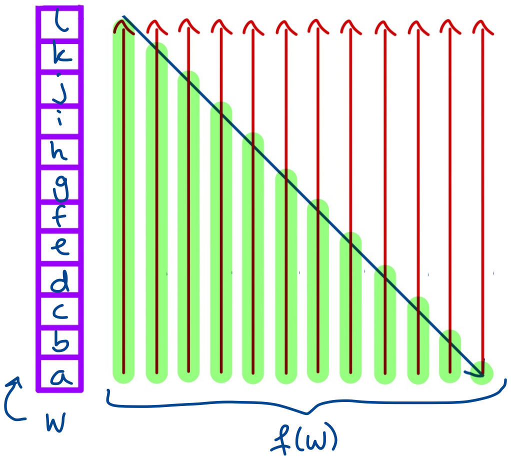
f can be obtained by adequately nesting regular functions.

These having linear growth, the "inner functions" can only produce linearly long infixes.

Consider $(a^n b\#)^n \#^{n\cdot m} \xrightarrow{\text{innsq}} ((a^n b)^{n\cdot m+n+1}\#)^n \#^{n\cdot m}$.

Inner squaring cannot be done with 2 pebbles

Assume there is $f \in \mathrm{Pebble}_2$ that coincides with innsq on $(a^*b\#)^*\#^*$,



f can be obtained by adequately nesting regular functions,

These having linear growth, the "inner functions" can only produce linearly long infixes.

Consider $(a^n b\#)^n \#^{n\cdot m} \xrightarrow{\text{innsq}} ((a^n b)^{n\cdot m+n+1}\#)^n \#^{n\cdot m}$.

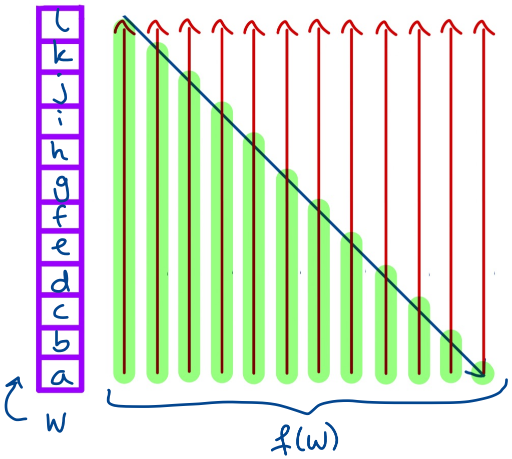
$\Rightarrow$ subcomputations produce ≤ 1 # each.

But there must be one that produces $\geq N$ b's on one side of the #.

We obtain $\cancel{a\ldots}ab(a^n b)^r\cancel{a\ldots a}$. (for some $n, r \geq N$).

Inner squaring cannot be done with 2 pebbles

Assume there is $f \in \text{Pebble}_2$ that coincides with innsq on $(a^*b\#)^*\#^*$.



f can be obtained by adequately nesting regular functions.

These having linear growth, the "inner functions" can only produce linearly long infixes.

Consider $(a^n b\#)^n \#^{n\cdot m} \xrightarrow{\text{innsq}} ((a^n b)^{n\cdot m+n+1}\#)^n \#^{n\cdot m}$.

$\Rightarrow$ subcomputations produce $\leq 1\ \#$ each.

But there must be one that produces $\geq N$ b's on one side of the #.

We obtain $\cancel{a\ldots}\,ab(a^n b)^r\,\cancel{a\ldots a}$. (for some $n, r \geq N$).

This way, we construct $L \subseteq b\{a,b\}^*b$ that

- is the image of a regular function
- consists of infixes of elements in innsq$((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a $b\underbrace{a\ldots a}_{\text{all a-blocks have length } n \geq N}b\ldots ba\ldots ab$ with at least N b's.

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^* b$ is a regular image that

- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $b\underbrace{a\ldots a}b\ldots b\underbrace{a\ldots a}b$ with at least N b's.

  all a-blocks have length $n \geq N$

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^* b$ is a regular image that

- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $b\underbrace{a\ldots a}b\ldots b\underbrace{a\ldots a}b$ with at least N b's.

all a-blocks have length $n \geq N$

There are $k, K \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq K$ has a decomposition $w = u_0 v_1 u_1 \ldots v_k u_k$ with

- $v_i \neq \varepsilon$ for some $i \in \{1, \ldots, k\}$
- $|v_i| \leq K$ for all $i \in \{1, \ldots, k\}$
- $\{u_0 v_1^n \ldots u_{k-1} v_k^n u_k \mid n \in \mathbb{N}\} \subseteq L$

[Rozoy '86] **PUMPING LEMMA**

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^* b$ is a regular image that

- consists of infixes of elements in $\mathrm{insq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $b\underbrace{a\ldots a}b \ldots b\underbrace{a\ldots a}b$ with at least N b's.

all a-blocks have length $n \geq N$

There are $k, K \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq K$ has a decomposition $w = u_0 v_1 u_1 \ldots v_k u_k$ with

- $v_i \neq \varepsilon$ for some $i \in \{1, \ldots, k\}$
- $|v_i| \leq K$ for all $i \in \{1, \ldots, k\}$
- $\{u_0 v_1^n \ldots u_{k-1} v_k^n u_k \mid n \in \mathbb{N}\} \subseteq L$

[Rozoy '86] **PUMPING LEMMA**

For $N := \max\{K, 2k+2\}$, we obtain a $b(a^n b)^r \in L$ with $n \geq N \geq K$, $r \geq N \geq 2k+1$.

Hence, $b(a^n b)^r = u_0 v_1 u_1 \ldots v_k u_k$ and $\underbrace{u_0 v_1^2 u_1 \ldots v_k^2 u_k}_{=:z} \in L$.

. .

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^* b$ is a regular image that

- consists of infixes of elements in $\mathrm{innsq}((a^* b\#)^* \#^*)$
- contains for every $N \in \mathbb{N}$ a word $\underbrace{b a \ldots a b \ldots b a \ldots a b}_{\text{all } a\text{-blocks have length } n \geq N}$ with at least N b's.

There are $k, K \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq K$ has a decomposition $w = u_0 v_1 u_1 \ldots v_k u_k$ with

- $v_i \neq \varepsilon$ for some $i \in \{1, \ldots, k\}$
- $|v_i| \leq K$ for all $i \in \{1, \ldots, k\}$
- $\{u_0 v_1^n \ldots u_{k-1} v_k^n u_k \mid n \in \mathbb{N}\} \subseteq L$

[Rozoy '86] **PUMPING LEMMA**

For $N := \max\{K, 2k+2\}$, we obtain a $b(a^n b)^r \in L$ with $n \geq N \geq K$, $r \geq N \geq 2k+1$.

Hence, $b(a^n b)^r = u_0 v_1 u_1 \ldots v_k u_k$ and $\underbrace{u_0 v_1^2 u_1 \ldots v_k^2 u_k}_{=:z} \in L$.

Each v_i contains at most one b. $\Rightarrow$ There is a u_j which contains two b's.

Thus, $ba^n b$ is an infix of z. $\Rightarrow$ All a-blocks in z have fixed length n.

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^* b$ is a regular image that

- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^*)$
- contains for every $N \in \mathbb{N}$ a word $b\underbrace{a\ldots a}b\ldots b\underbrace{a\ldots a}b$ with at least N b's.

  all a-blocks have length $n \geq N$

There are $k, K \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq K$ has a decomposition $w = u_0 v_1 u_1 \ldots v_k u_k$ with

- $v_i \neq \varepsilon$ for some $i \in \{1, \ldots, k\}$
- $|v_i| \leq K$ for all $i \in \{1, \ldots, k\}$
- $\{u_0 v_1^n \ldots u_{k-1} v_k^n u_k \mid n \in \mathbb{N}\} \subseteq L$

[Rozoy '86] **PUMPING LEMMA**

For $N := \max\{K, 2k+2\}$, we obtain a $b(a^n b)^r \in L$ with $n \geq N \geq K$, $r \geq N \geq 2k+1$.

Hence, $b(a^n b)^r = u_0 v_1 u_1 \ldots v_k u_k$ and $\underbrace{u_0 v_1^2 u_1 \ldots v_k^2 u_k}_{=:z} \in L$.

Each v_i contains at most one b. $\Rightarrow$ There is a u_j which contains two b's.

Thus, $ba^n b$ is an infix of z. $\Rightarrow$ All a-blocks in z have fixed length n.

Case 1: Some v_i contains a b. ↯

Inner squaring cannot be done with 2 pebbles

$L \subseteq b\{a,b\}^*b$ is a regular image that

- consists of infixes of elements in $\mathrm{innsq}((a^*b\#)^*\#^{(*)})$
- contains for every $N \in \mathbb{N}$ a word $\underbrace{ba\ldots a}b\ldots \underbrace{ba\ldots a}b$ with at least N b's.

all a-blocks have length $n \geq N$

There are $k, K \in \mathbb{N}$ such that every $w \in L$ with $|w| \geq K$ has a decomposition $w = u_0 v_1 u_1 \ldots v_k u_k$ with

- $v_i \neq \varepsilon$ for some $i \in \{1, \ldots, k\}$
- $|v_i| \leq K$ for all $i \in \{1, \ldots, k\}$
- $\{u_0 v_1^n \ldots u_{k-1} v_k^n u_k \mid n \in \mathbb{N}\} \subseteq L$

[Rozoy '86] **PUMPING LEMMA**

For $N := \max\{K, 2k+2\}$, we obtain a $b(a^n b)^r \in L$ with $n \geq N \geq K$, $r \geq N \geq 2k+1$.

Hence, $b(a^n b)^r = u_0 v_1 u_1 \ldots v_k u_k$ and $\underbrace{u_0 v_1^2 u_1 \ldots v_k^2 u_k}_{=:z} \in L$.

Each v_i contains at most one b. $\Rightarrow$ There is a u_j which contains two b's.

Thus, $ba^n b$ is an infix of z. $\Rightarrow$ All a-blocks in z have fixed length n.

Case ①: Some v_i contains a b. ↯ Case ②: All v_i are in a^*. ↯

$\square$

Conclusion

Polyregular functions are s-to-s functions with polynomial-size output.

Conclusion

Polyregular functions are s-to-s functions with polynomial-size output.
They have various equivalent characterisations, e.g.

f is polyregular	$:\Longleftrightarrow$	f is computed by a for-program	
$\Updownarrow$	$:\Longleftrightarrow$	$\Updownarrow$	[Bojańczyk, K., Lhote '19]
f is recognised by a s-to-s pebble transducer	$\Longleftrightarrow$	f is defined by a s-to-s MSO interpretation	

Conclusion

Polyregular functions are s-to-s functions with polynomial-size output.
They have various equivalent characterisations, e.g.

f is polyregular	$\Leftrightarrow$	f is computed by a for-program
$\Updownarrow$		$\Updownarrow$ [Bojańczyk, K., Lhote '19]
f is recognised by a s-to-s pebble transducer	$\Leftrightarrow$	f is defined by a s-to-s MSO interpretation

Follow-up work and concepts:

- Polyblind functions – comparisons between pebble positions are not allowed [Nguyễn, Noûs, Pradic '21]

  → a setting where "growth $O(n^k) \Rightarrow \leq k$ pebbles" [see also Douéneau-Tabot '23]

Conclusion

Polyregular functions are s-to-s functions with polynomial-size output.
They have various equivalent characterisations, e.g.

f is polyregular	$\Leftrightarrow$	f is computed by a for-program
$\Updownarrow$	$\Leftrightarrow$	$\Updownarrow$ [Bojańczyk, K., Lhote '19]
f is recognised by a s-to-s pebble transducer	$\Leftrightarrow$	f is defined by a s-to-s MSO interpretation

Follow-up work and concepts:

- Polyblind functions – comparisons between pebble positions are not allowed [Nguyễn, Noûs, Pradic '21]
  → a setting where "growth $O(n^k) \Rightarrow \leq k$ pebbles" [see also Douéneau-Tabot '23]
- $\mathbb{Z}$-polyregular functions – functions $\mathrm{sum} \circ f$, where $f : \Sigma^* \to \{\pm 1\}^*$ is polyregular [Colcombet, Douéneau-Tabot, Lopez '23]

Conclusion

f is polyregular of growth rate $\mathcal{O}(n^k)$

$\Longleftrightarrow$

f is defined by a k-dim. s-to-s MSO interpretation

f is recognised by a k-pebble s-to-s pebble transducer

Conclusion

f is polyregular of growth rate $\mathcal{O}(n^k)$ $\Longleftrightarrow$ f is defined by a k-dim. s-to-s MSO interpretation

f is recognised by a k-pebble s-to-s pebble transducer

innsq : $\{a, b, \#\}^* \to \{a, b, \#\}^*$

$w_0 \# \dots \# w_n \mapsto w_0^n \# \dots \# w_n^n$ (all $w_i \in \{a, b\}^*$)

For example, innsq requires 3 pebbles, but $\text{innsq}(w) \in \mathcal{O}(|w|^2)$.

Conclusion

f is polyregular of growth rate $\mathcal{O}(n^k)$	$\Longleftrightarrow$	f is defined by a k-dim. s-to-s MSO interpretation

f is recognised by a k-pebble s-to-s pebble transducer

$$\mathrm{innsq} : \{a,b,\#\}^* \to \{a,b,\#\}^*$$

$$w_0\#\ldots\#w_n \mapsto w_0^n\#\ldots\#w_n^n \qquad (\text{all } w_i \in \{a,b\}^*)$$

For example, innsq requires 3 pebbles, but $\mathrm{innsq}(w) \in \mathcal{O}(|w|^2)$.

There are also quadratic-growth polyregular functions $(f_k)_{k\in\mathbb{N}}$ where $f_k \notin \mathrm{Pebble}_k$ for each $k \in \mathbb{N}$.

[see Bojańczyk '23]

Conclusion

f is polyregular of growth rate $\mathcal{O}(n^k)$ $\iff$ f is defined by a k-dim. s-to-s MSO interpretation

f is recognised by a k-pebble s-to-s pebble transducer

$$\begin{aligned} \mathrm{innsq} : \{a,b,\#\}^* &\to \{a,b,\#\}^* \\ w_0\#\dots\#w_n &\mapsto w_0^n\#\dots\#w_n^n \qquad (\text{all } w_i \in \{a,b\}^*) \end{aligned}$$

For example, innsq requires 3 pebbles, but $\mathrm{innsq}(w) \in \mathcal{O}(|w|^2)$.

There are also quadratic-growth polyregular functions $(f_k)_{k\in\mathbb{N}}$ where $f_k \notin \mathrm{Pebble}_k$ for each $k \in \mathbb{N}$. [see Bojańczyk '23]

In [K., Nguyen, Pradic '23], we show that a slightly stronger result actually follows quickly from old results due to [Engelfriet, Maneth '01].