Mix-Automatic Sequences

Jörg Endrullis   Clemens Grabmayer   Dimitri Hendriks

Fields Workshop on Combinatorics on Words
Toronto, April 22, 2013

Zipping sequences

Zipping sequences

$$u = a_0 : a_1 : a_2 : \ldots$$
$$v = b_0 : b_1 : b_2 : \ldots$$

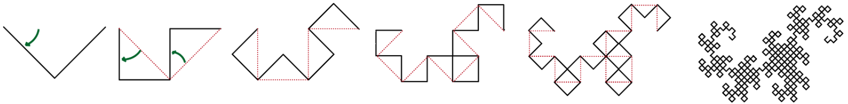
results in

$$\mathrm{zip}(u, v) = a_0 : b_0 : a_1 : b_1 : a_2 : b_2 : \ldots$$

Operationally:

$$\mathrm{zip}(a : u, v) \to a : \mathrm{zip}(v, u)$$

Zip-specifications



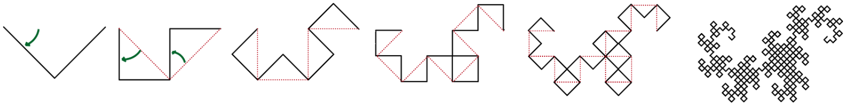
$$\text{Peaks} = \wedge : \text{Peaks}$$

$$\text{Valleys} = \vee : \text{Valleys}$$

$$\text{Tyrol} = \text{zip}(\text{Peaks}, \text{Valleys})$$

$$\text{Folds} = \text{zip}(\text{Tyrol}, \text{Folds})$$

Zip-specifications



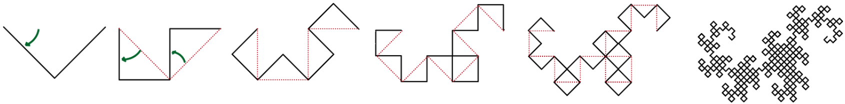
$$\text{Peaks} = \wedge : \text{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\text{Valleys} = \vee : \text{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\text{Tyrol} = \text{zip}(\text{Peaks}, \text{Valleys})$$

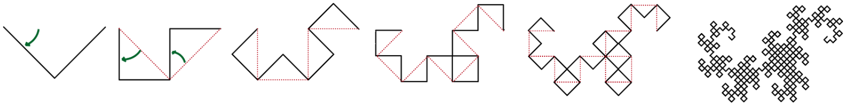
$$\text{Folds} = \text{zip}(\text{Tyrol}, \text{Folds})$$

Zip-specifications



$$\begin{aligned}
\text{Peaks} &= \wedge : \text{Peaks} && = \wedge : \wedge : \wedge : \ldots \\
\text{Valleys} &= \vee : \text{Valleys} && = \vee : \vee : \vee : \ldots \\
\text{Tyrol} &= \text{zip}(\text{Peaks}, \text{Valleys}) && = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots \\
\text{Folds} &= \text{zip}(\text{Tyrol}, \text{Folds})
\end{aligned}$$

Zip-specifications



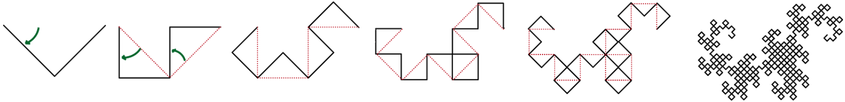
$$\text{Peaks} = \wedge : \text{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\text{Valleys} = \vee : \text{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\text{Tyrol} = \text{zip}(\text{Peaks}, \text{Valleys}) \qquad = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots$$

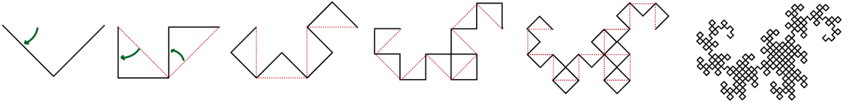
$$\text{Folds} = \text{zip}(\text{Tyrol}, \text{Folds}) \qquad = \wedge : \quad : \vee : \quad : \wedge : \quad : \vee : \quad : \wedge : \quad : \vee : \ldots$$

Zip-specifications



$$
\begin{aligned}
\text{Peaks} &= \wedge : \text{Peaks} && = \wedge : \wedge : \wedge : \ldots \\
\text{Valleys} &= \vee : \text{Valleys} && = \vee : \vee : \vee : \ldots \\
\text{Tyrol} &= \text{zip}(\text{Peaks}, \text{Valleys}) && = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots \\
\text{Folds} &= \text{zip}(\text{Tyrol}, \text{Folds}) && = \wedge : \wedge : \vee : \quad : \wedge : \quad : \vee : \quad : \wedge : \quad : \vee : \ldots
\end{aligned}
$$

Zip-specifications



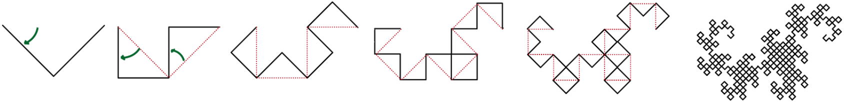
$$\text{Peaks} = \wedge : \text{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\text{Valleys} = \vee : \text{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\text{Tyrol} = \text{zip}(\text{Peaks}, \text{Valleys}) \qquad = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots$$

$$\text{Folds} = \text{zip}(\text{Tyrol}, \text{Folds}) \qquad = \wedge : \wedge : \vee : \wedge : \wedge : \quad : \vee : \quad : \wedge : \quad : \vee : \ldots$$

Zip-specifications



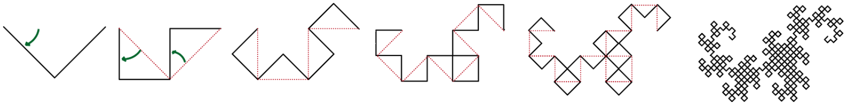
$$\mathsf{Peaks} = \wedge : \mathsf{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\mathsf{Valleys} = \vee : \mathsf{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\mathsf{Tyrol} = \mathsf{zip}(\mathsf{Peaks}, \mathsf{Valleys}) \qquad = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots$$

$$\mathsf{Folds} = \mathsf{zip}(\mathsf{Tyrol}, \mathsf{Folds}) \qquad = \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \quad : \wedge : \quad : \vee : \ldots$$

Zip-specifications



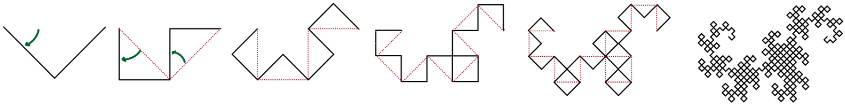
$$\text{Peaks} = \wedge : \text{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\text{Valleys} = \vee : \text{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\text{Tyrol} = \text{zip}(\text{Peaks}, \text{Valleys}) \qquad = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots$$

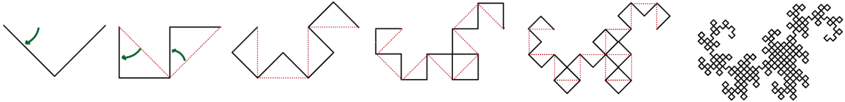
$$\text{Folds} = \text{zip}(\text{Tyrol}, \text{Folds}) \qquad = \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \quad : \vee : \ldots$$

Zip-specifications



$$\begin{aligned}
\text{Peaks} &= \wedge : \text{Peaks} && = \wedge : \wedge : \wedge : \ldots \\
\text{Valleys} &= \vee : \text{Valleys} && = \vee : \vee : \vee : \ldots \\
\text{Tyrol} &= \text{zip}(\text{Peaks}, \text{Valleys}) && = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots \\
\text{Folds} &= \text{zip}(\text{Tyrol}, \text{Folds}) && = \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \wedge : \vee : \ldots
\end{aligned}$$

Zip-specifications



$$\mathsf{Peaks} = \wedge : \mathsf{Peaks} \qquad = \wedge : \wedge : \wedge : \ldots$$

$$\mathsf{Valleys} = \vee : \mathsf{Valleys} \qquad = \vee : \vee : \vee : \ldots$$

$$\mathsf{Tyrol} = \mathsf{zip}(\mathsf{Peaks}, \mathsf{Valleys}) \qquad = \wedge : \vee : \wedge : \vee : \wedge : \vee : \ldots$$

$$\mathsf{Folds} = \mathsf{zip}(\mathsf{Tyrol}, \mathsf{Folds}) \qquad = \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \wedge : \vee : \ldots$$

A **zip-specification** over $\langle A, \mathcal{X} \rangle$ is a system of equations $\mathsf{X} = t$ where the right-hand sides t are terms defined by the grammar

$$t ::= \mathsf{X} \mid a : t \mid \mathsf{zip}(t, t) \qquad (\mathsf{X} \in \mathcal{X},\ a \in A)$$

Well-definedness of zip-specifications

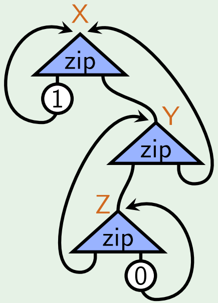
Productivity (implies unique solvability) for a zip-specification is easy to check: at least one guard on every leftmost cycle.

Example

$X = zip(1 : X, Y)$

$Y = zip(Z, X)$

$Z = zip(Y, 0 : Z)$



Well-definedness of zip-specifications

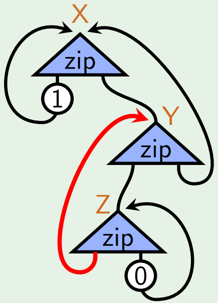
Productivity (implies unique solvability) for a zip-specification is easy to check: at least one guard on every leftmost cycle.

Example

$X = zip(1 : X, Y)$

$Y = zip(Z, X)$

$Z = zip(Y, 0 : Z)$



No guard on cycle

$Y \to Z \to Y$

Not productive!

Initial Motivation

Initial Questions

- Is equivalence of zip-specifications decidable? (L.S. Moss)
- What is the class of sequences that can be defined by zip-specifications?

Unzipping

Using 'zip-destructors'

$$\text{even}(w) = w(0) : w(2) : w(4) : \ldots$$
$$\text{odd}(w) = w(1) : w(3) : w(5) : \ldots$$

unzipping can be done:

$$\text{even}(\text{zip}(u, v)) = u$$
$$\text{odd}(\text{zip}(u, v)) = v$$

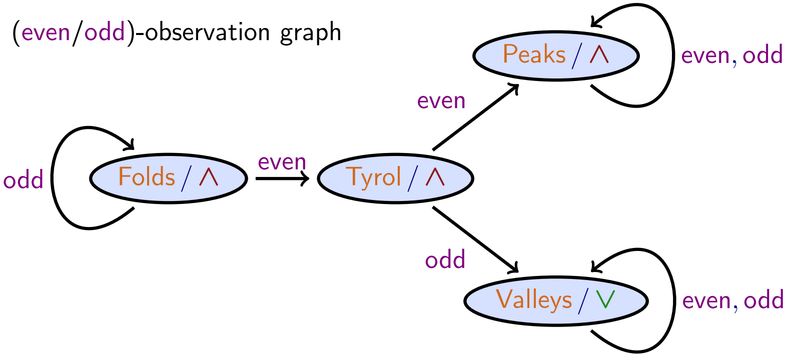
Operational definition:

$$\text{even}(a : u) = a : \text{odd}(u)$$
$$\text{odd}(a : u) = \text{even}(u)$$

Idea: use even, odd to **observe** zip-specs and check bisimilarity of the resulting graphs.

Observation graph of Folds zip-specification

(even/odd)-observation graph

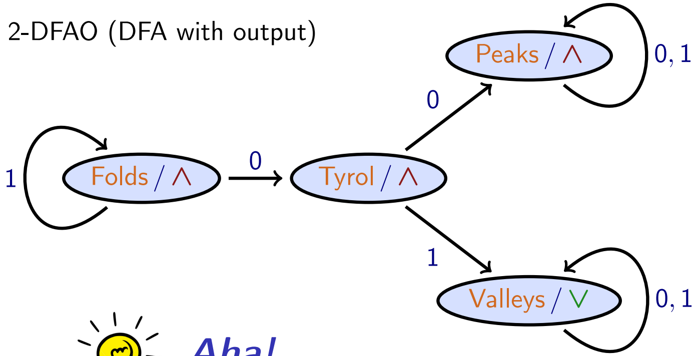


$$\mathsf{Folds} = \mathsf{zip}(\mathsf{Tyrol}, \mathsf{Folds})$$
$$\mathsf{Tyrol} = \mathsf{zip}(\mathsf{Peaks}, \mathsf{Valleys})$$
$$\mathsf{Peaks} = \wedge : \mathsf{Peaks}$$
$$\mathsf{Valleys} = \vee : \mathsf{Valleys}$$

$$\mathsf{Folds} \to^{\omega} \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \wedge : \vee : \vee : \wedge : \vee : \vee : \wedge \ldots$$

Finite automaton generating the paperfolding sequence

2-DFAO (DFA with output)

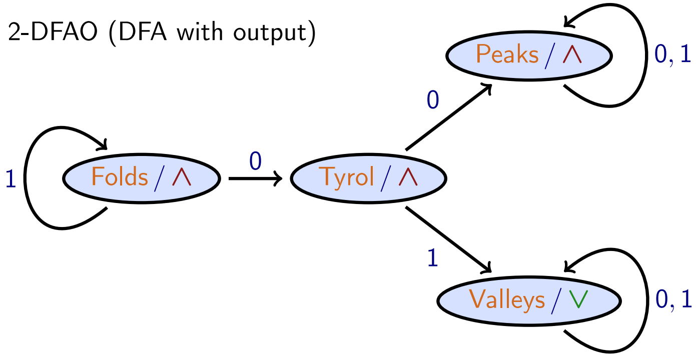


Aha!

Folds $\rightarrow^{\omega}$ $\wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \wedge : \vee : \vee : \wedge : \vee : \vee : \wedge \ldots$

Finite automaton generating the paperfolding sequence

2-DFAO (DFA with output)



$$((9)_2)_{\text{Folds}} = (1001)_{\text{Folds}} \xrightarrow{1} (100)_{\text{Folds}} \xrightarrow{0} (10)_{\text{Tyrol}} \xrightarrow{0} (1)_{\text{Peaks}} \xrightarrow{1} ()_{\text{Peaks}}$$

$$\text{Folds} \to^{\omega} \wedge : \wedge : \vee : \wedge : \wedge : \vee : \vee : \wedge : \wedge : \boxed{\wedge} : \vee : \vee : \wedge : \vee : \vee : \wedge \ldots$$

Generalization to k-automatic sequences

A **zip-k specification** over $\langle A, \mathcal{X} \rangle$ is a system of equations $\mathsf{X} = t$ where the right-hand sides t are terms defined by

$$t ::= \mathsf{X} \mid a : t \mid \mathsf{zip}_k(t, \ldots, t) \qquad (\mathsf{X} \in \mathcal{X},\ a \in A)$$

where zip_k shuffles k sequences

$$\mathsf{zip}_k(u_0, u_1, \ldots, u_{k-1})(kn + i) = u_i(n) \qquad (0 \leq i < k)$$

Operationally:

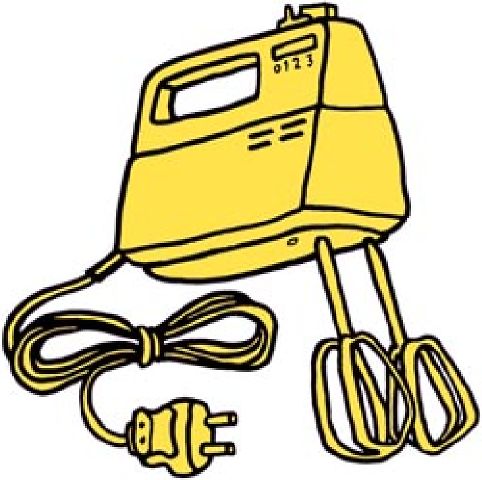
$$\mathsf{zip}_k(a : u_0, u_1, \ldots, u_{k-1}) = a : \mathsf{zip}_k(u_1, \ldots, u_{k-1}, u_0)$$

Theorem

A sequence ***k-automatic*** *if and only if it has a* ***zip-k specification****.*

Hence equivalence of zip-k specifications is decidable.

Mix-automatic sequences



Motivating question

What about zips of different arities in one specification?

Mix-automatic sequences

Zip-mix specifications: now we allow zips of different arities zip_2, zip_3, zip_4, $\dots$ in the same specification.

Mix-automatic sequences

Zip-mix specifications: now we allow zips of different arities zip_2, zip_3, zip_4, ... in the same specification.

Example

$$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$$

$$\mathsf{M} \to^{\omega} a : b : b : b : b : a : b : b : b : b : a : b : b : a : b : a : \ldots$$

Mix-automatic sequences

Zip-mix specifications: now we allow zips of different arities zip_2, zip_3, zip_4, ... in the same specification.

Example

$$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$$

$$\mathsf{M} \to^{\omega} a : b : b : b : b : a : b : b : b : b : a : b : b : a : b : a : \ldots$$

We call the corresponding sequences **mix-automatic sequences**.

Mix-automatic sequences

Zip-mix specifications: now we allow zips of different arities zip_2, zip_3, zip_4, ... in the same specification.

Example

$$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$$

$$\mathsf{M} \to^{\omega} a : b : b : b : b : a : b : b : b : b : a : b : b : a : b : a : \ldots$$

We call the corresponding sequences **mix-automatic sequences**.

- What is the relation to automatic or morphic sequences?
- What about subword complexity?
- What is the corresponding notion of automaton?

Mix-automatic extends automatic

Theorem

The class of mix-automatic sequences properly extends the class of automatic sequences.

Proof: Let u and v be 2 and 3-automatic, but not ultimately periodic. If the sequence $\mathsf{zip}(u, v)$ would be m-automatic, then so would be u and v. By Cobham's Theorem there are $a, b, c, d > 0$ such that

- $2^a = m^b$, and
- $3^c = m^d$.

But then $2^{ad} = m^{bd} = 3^{cb}$ yields a contradiction.

Theorem (Cobham's Theorem)

Let $k, \ell \geq 2$ such that $k^a \neq \ell^b$ for all $a, b > 0$. If a sequence u is both k- and ℓ-automatic, then u is ultimately periodic.

Characterization via automata

The automata corresponding to mix-automatic sequences are **mix-DFAOs** with a **state-dependent input alphabet**.

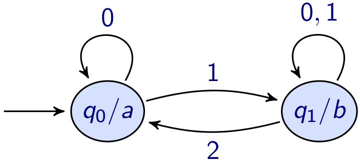
Characterization via automata

The automata corresponding to mix-automatic sequences are **mix-DFAOs** with a **state-dependent input alphabet**.

Example

$$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$$

The specification corresponds to the mix-DFAO



The input alphabet of q_0 is $\{0, 1\}$ and of q_1 is $\{0, 1, 2\}$.

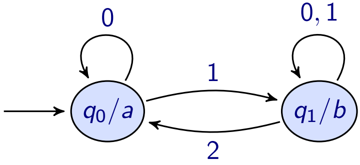
Characterization via automata

The automata corresponding to mix-automatic sequences are **mix-DFAOs** with a **state-dependent input alphabet**.

Example

$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$

The specification corresponds to the mix-DFAO



The input alphabet of q_0 is $\{0, 1\}$ and of q_1 is $\{0, 1, 2\}$.

Input: representation of $i \in \mathbb{N}$
Output: i-th element of the sequence

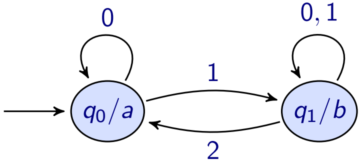
Characterization via automata

The automata corresponding to mix-automatic sequences are **mix-DFAOs** with a **state-dependent input alphabet**.

Example

$$\mathsf{M} = a : \mathsf{X} \qquad \mathsf{X} = b : \mathsf{zip}_2(\mathsf{X}, \mathsf{Y}) \qquad \mathsf{Y} = b : \mathsf{zip}_3(\mathsf{M}, \mathsf{Y}, \mathsf{Y})$$

The specification corresponds to the mix-DFAO



The input alphabet of q_0 is $\{0, 1\}$ and of q_1 is $\{0, 1, 2\}$.

Input: representation of $i \in \mathbb{N}$

Output: i-th element of the sequence

What is this representation?

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, . . .
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

For mix-automatic sequences we need another numeration system.

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

For mix-automatic sequences we need another numeration system.

Dynamic radix numeration systems

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

For mix-automatic sequences we need another numeration system.

Dynamic radix numeration systems

- generalizes usual base-k representation

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

For mix-automatic sequences we need another numeration system.

Dynamic radix numeration systems

- generalizes usual base-k representation
- generalizes Knuth's mixed radix numeration system

Number representation for mix-DFAOs

Mix-DFAOs are known:

- intensively studied by Rigo, Maes, ...
- used with abstract numeration systems

Abstract numeration systems

Let L be the language accepted as input by the automaton.
Then $i \in \mathbb{N}$ is represented by the i-th word of L in the shortlex order.

Mix-DFAOs + abstract numeration systems $=$ morphic sequences

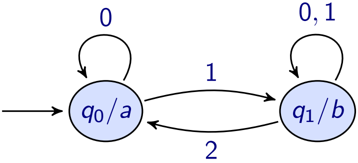
For mix-automatic sequences we need another numeration system.

Dynamic radix numeration systems

- generalizes usual base-k representation
- generalizes Knuth's mixed radix numeration system

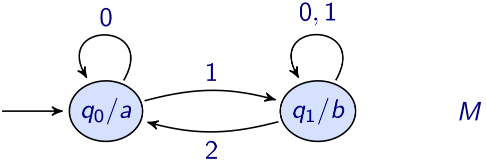
The base of a digit depends on the values of the less significant digits.

Dynamic radix numeration systems



M

Dynamic radix numeration systems

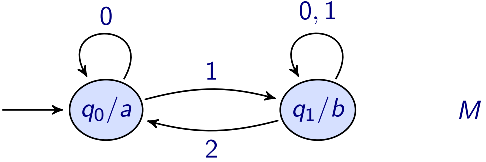


We write $(n)_M = (n)_{q_0}$ for the representation of $n \in \mathbb{N}$ as input for M.
The automaton reads the least significant digit first:

$$
\begin{aligned}
(17)_M &= (17)_{q_0} \\
&= (8)_{q_1}\, 1_2 \\
&= (2)_{q_0}\, 2_3\, 1_2 \\
&= (1)_{q_0}\, 0_2\, 2_3\, 1_2 \\
&= (0)_{q_1}\, 1_2\, 0_2\, 2_3\, 1_2 \\
&= 1_2\, 0_2\, 2_3\, 1_2
\end{aligned}
$$

(we write the base of each digit as subscript of the digit)

Dynamic radix numeration systems



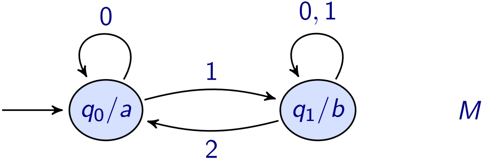
We write $(n)_M = (n)_{q_0}$ for the representation of $n \in \mathbb{N}$ as input for M.
The automaton reads the least significant digit first:

$$\begin{aligned}(17)_M &= (17)_{q_0} \\ &= (8)_{q_1}\, 1_2 \\ &= (2)_{q_0}\, 2_3\, 1_2 \\ &= (1)_{q_0}\, 0_2\, 2_3\, 1_2 \\ &= (0)_{q_1}\, 1_2\, 0_2\, 2_3\, 1_2 \\ &= 1_2\, 0_2\, 2_3\, 1_2\end{aligned}$$

$$\begin{aligned}(16)_M &= (16)_{q_0} \\ &= (8)_{q_0}\, 0_2 \\ &= (4)_{q_0}\, 0_2\, 0_2 \\ &= (2)_{q_0}\, 0_2\, 0_2\, 0_2 \\ &= (1)_{q_0}\, 0_2\, 0_2\, 0_2\, 0_2 \\ &= 1_2\, 0_2\, 0_2\, 0_2\, 0_2\end{aligned}$$

(we write the base of each digit as subscript of the digit)

Dynamic radix numeration systems



We write $(n)_M = (n)_{q_0}$ for the representation of $n \in \mathbb{N}$ as input for M.
The automaton reads the least significant digit first:

$$\begin{aligned}(17)_M &= (17)_{q_0}\\ &= (8)_{q_1}\, 1_2\\ &= (2)_{q_0}\, 2_3\, 1_2\\ &= (1)_{q_0}\, 0_2\, 2_3\, 1_2\\ &= (0)_{q_1}\, 1_2\, 0_2\, 2_3\, 1_2\\ &= 1_2\, 0_2\, 2_3\, 1_2\end{aligned}$$

$$\begin{aligned}(16)_M &= (16)_{q_0}\\ &= (8)_{q_0}\, 0_2\\ &= (4)_{q_0}\, 0_2\, 0_2\\ &= (2)_{q_0}\, 0_2\, 0_2\, 0_2\\ &= (1)_{q_0}\, 0_2\, 0_2\, 0_2\, 0_2\\ &= 1_2\, 0_2\, 0_2\, 0_2\, 0_2\end{aligned}$$

(we write the base of each digit as subscript of the digit)

Mix-DFAOs + dynamic radix numeration systems = mix-automatic

Characterization via finite kernels

For $i, k \in \mathbb{N}$ and sequences w we define

$$\pi_{i,k}(w) = w(i + 0k)\ w(i + 1k)\ w(i + 2k)\ w(i + 3k) \dots$$

the subsequence of w taking every k-th element starting from the i-th.

Kernel

Let $k \in \mathbb{N}$ and $w \in \Delta^\omega$.

The k-kernel $\mathrm{Ker}(k, w)$ is the smallest set $K \subseteq \Delta^\omega$ such that:

- $w \in K$, and
- for all $u \in K$ and all $0 \le i < k$, we have $\pi_{i,k}(u) \in K$.

Theorem

For a sequence $w \in \Delta^\omega$ the following are equivalent:

- *w is automatic,*
- *there exists $k \in \mathbb{N}_{\geq 2}$ such that the k-kernel of w is finite.*

Characterization via finite kernels

For $i, k \in \mathbb{N}$ and sequences w we define

$$\pi_{i,k}(w) = w(i + 0k)\ w(i + 1k)\ w(i + 2k)\ w(i + 3k) \ldots$$

the subsequence of w taking every k-th element starting from the i-th.

Mix-kernel

Let $k \in \mathbb{N}$ and $w \in \Delta^\omega$.

The k-kernel $\mathrm{Ker}(k, w)$ is the smallest set $K \subseteq \Delta^\omega$ such that:

- $w \in K$, and
- for all $u \in K$ and all $0 \leq i < k$, we have $\pi_{i,k}(u) \in K$.

Theorem

For a sequence $w \in \Delta^\omega$ the following are equivalent:

- *w is automatic,*
- *there exists $k \in \mathbb{N}_{\geq 2}$ such that the k-kernel of w is finite.*

Characterization via finite kernels

For $i, k \in \mathbb{N}$ and sequences w we define

$$\pi_{i,k}(w) = w(i+0k)\ w(i+1k)\ w(i+2k)\ w(i+3k)\dots$$

the subsequence of w taking every k-th element starting from the i-th.

Mix-kernel

Let $k : \Delta^\omega \to \mathbb{N}$ and $w \in \Delta^\omega$.

The k-kernel $\mathrm{Ker}(k, w)$ is the smallest set $K \subseteq \Delta^\omega$ such that:

- $w \in K$, and
- for all $u \in K$ and all $0 \leq i < k$, we have $\pi_{i,k}(u) \in K$.

Theorem

For a sequence $w \in \Delta^\omega$ the following are equivalent:

- *w is automatic,*
- *there exists $k \in \mathbb{N}_{\geq 2}$ such that the k-kernel of w is finite.*

Characterization via finite kernels

For $i, k \in \mathbb{N}$ and sequences w we define

$$\pi_{i,k}(w) = w(i + 0k)\, w(i + 1k)\, w(i + 2k)\, w(i + 3k) \dots$$

the subsequence of w taking every k-th element starting from the i-th.

Mix-kernel

Let $k : \Delta^\omega \to \mathbb{N}$ and $w \in \Delta^\omega$.

The k-kernel $\mathrm{Ker}(k, w)$ is the smallest set $K \subseteq \Delta^\omega$ such that:

- $w \in K$, and
- for all $u \in K$ and all $0 \leq i < k(u)$, we have $\pi_{i,k(u)}(u) \in K$.

Theorem

For a sequence $w \in \Delta^\omega$ the following are equivalent:

- *w is automatic,*
- *there exists $k \in \mathbb{N}_{\geq 2}$ such that the k-kernel of w is finite.*

Characterization via finite kernels

For $i, k \in \mathbb{N}$ and sequences w we define

$$\pi_{i,k}(w) = w(i+0k)\ w(i+1k)\ w(i+2k)\ w(i+3k)\dots$$

the subsequence of w taking every k-th element starting from the i-th.

Mix-kernel

Let $k : \Delta^\omega \to \mathbb{N}$ and $w \in \Delta^\omega$.

The k-kernel $\mathrm{Ker}(k, w)$ is the smallest set $K \subseteq \Delta^\omega$ such that:

- $w \in K$, and
- for all $u \in K$ and all $0 \le i < k(u)$, we have $\pi_{i,k(u)}(u) \in K$.

Theorem

For a sequence $w \in \Delta^\omega$ the following are equivalent:

- *w is mix-automatic,*
- *there exists $k : \Delta^\omega \to \mathbb{N}_{\ge 2}$ such that the k-kernel of w is finite.*

Mix-automatic versus morphic sequences

Proposition

The class of morphic sequences is not contained in the class of mix-automatic sequences.

For the characteristic sequence $\mathsf{squares} = 1100100001\ldots$ of square numbers is morphic but not mix-automatic.

Mix-automatic versus morphic sequences

Proposition

The class of morphic sequences is not contained in the class of mix-automatic sequences.

For the characteristic sequence $\text{squares} = 1100100001\ldots$ of square numbers is morphic but not mix-automatic.

Corollary

Neither of the classes

- *mix-automatic sequences, and*
- *morphic sequences*

subsumes the other.

Subword complexity of mix-automatic sequences

Theorem

For any $k \in \mathbb{N}$ there exists a mix-automatic sequences with subword complexity in $\Omega(n^k)$.

Proof idea: For p a prime number, define the sequence $\gamma_p \in 2^\omega$ by

$$\gamma_p(n) = v_p(n) \bmod 2 \qquad \text{where} \qquad v_p(n) = \max\{e \mid p^e \text{ divides } n\}$$

Then γ_p is p-automatic: $\gamma_p = \mathsf{zip}_p(0^\omega, 0^\omega, \dots, 0^\omega, \overline{\gamma_p})$.

Let $p_1, p_2, \dots, p_k$ pairwise distinct primes. The sequence

$$\sigma = \mathsf{zip}_k(\gamma_{p_1}, \dots, \gamma_{p_k})$$

is mix-automatic. For subword complexity in $\Omega(n^k)$, it suffices that

- for all $n \in \mathbb{N}$, and
- for all factors w_1 in γ_{p_1}, …, w_k in γ_{p_k} of length n,

$\mathsf{zip}_k(w_1, \dots, w_k)$ is a factor in σ.

Subword complexity of mix-automatic sequences

Theorem

For any $k \in \mathbb{N}$ there exists a mix-automatic sequences with subword complexity in $\Omega(n^k)$.

Proof idea: For p a prime number, define the sequence $\gamma_p \in 2^\omega$ by

$$\gamma_p(n) = v_p(n) \bmod 2 \qquad \text{where} \qquad v_p(n) = \max\{e \mid p^e \text{ divides } n\}$$

Then γ_p is p-automatic: $\gamma_p = \mathrm{zip}_p(0^\omega, 0^\omega, \ldots, 0^\omega, \overline{\gamma_p})$.

Let $p_1, p_2, \ldots, p_k$ pairwise distinct primes. The sequence

$$\sigma = \mathrm{zip}_k(\gamma_{p_1}, \ldots, \gamma_{p_k})$$

is mix-automatic. For subword complexity in $\Omega(n^k)$, it suffices that

- for all $n \in \mathbb{N}$, and
- for all factors w_1 in γ_{p_1}, ..., w_k in γ_{p_k} of length n,

$\mathrm{zip}_k(w_1, \ldots, w_k)$ is a factor in σ.

Corollary

There are mix-automatic sequences that are not morphic.

Results and open questions

Results:

- Characterizations of mix-automatic sequences:
  1. via zip-mix specifications
  2. via a generalization of k-kernels
  3. via mix-DFAOs + dynamic radix numeration systems
- Novel numeration system: dynamic radix numeration systems
- For every polynomial p there exists a mix-automatic sequence whose subword complexity exceeds p.
- There exist morphic sequences that are not mix-automatic.

Questions:

- Characterize the intersection of mix-automatic and morphic sequences. (J.-P. Allouche)
- Is equality of mix-automatic sequences decidable?
  (the sequences are given in terms of their mix-DFAOs)
- Can Cobham's Theorem be generalized to mix-automatic sequences?

Bibliography

[C69] *On the Base-Dependence of Sets of Numbers Recognizable by Finite Automata*, MST, 1969

[C72] *Uniform Tag Sequences*, Theory of Computing Systems, 1972

[Rigo00] *Generalization of Automatic Sequences for Numeration Systems on a Regular Language*, TCS, 2000

[AS03] *Automatic Sequences: Theory, Applications, Generalizations*, CUP, 2003

[GEHKM12] *Automatic Sequences and Zip-Specifications*, LICS 2012

[KkR12] *On the Final Coalgebra of Automatic Sequences*, LPS, 2012

[EGH13] *Mix-Automatic Sequences*, LATA 2013

A = J.-P. Allouche, C = A. Cobham, E = J. Endrullis, G = C. Grabmayer,
H = D. Hendriks, K = J.W. Klop, Kk = C. Kupke,
M = L. Moss, Rigo = M. Rigo, R = J.J.M.M. Rutten, S = J. Shallit