Arithmetic subword complexity of automatic sequences

Clemens Müllner

TU Wien

Tuesday, November 19, 2024

Automatic Sequences

Definition (Automaton - DFA)

$$A = (Q, \Sigma = \{0, \ldots, k-1\}, \delta, q_0, \tau)$$

Example (Thue-Morse sequence)

$$n = 22 = (10110)_2, \qquad t_{22} = 1$$

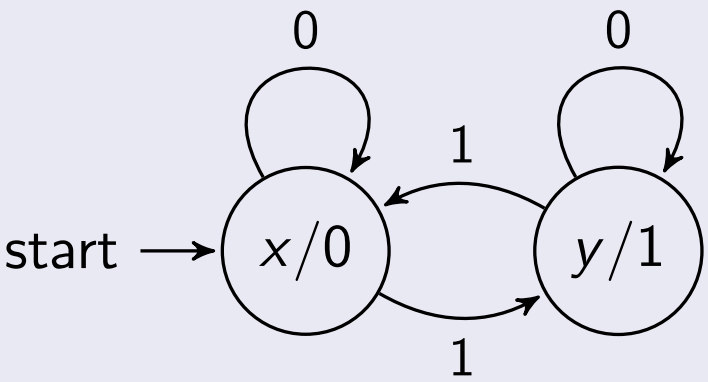
$$(t(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$$

Automatic Sequences

Definition (Automaton - DFA)

$$A = (Q, \Sigma = \{0, \ldots, k-1\}, \delta, q_0, \tau)$$

Example (Thue-Morse sequence)



$$n = 22 = (10110)_2, \qquad t_{22} = 1$$

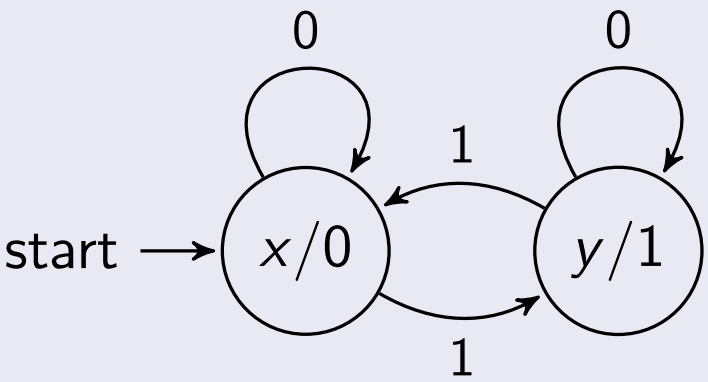
$$(t(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$$

Automatic Sequences

Definition (Automaton - DFA)

$$A = (Q, \Sigma = \{0, \ldots, k-1\}, \delta, q_0, \tau)$$

Example (Thue-Morse sequence)



$$n = 22 = (10110)_2, \qquad t_{22} = 1$$

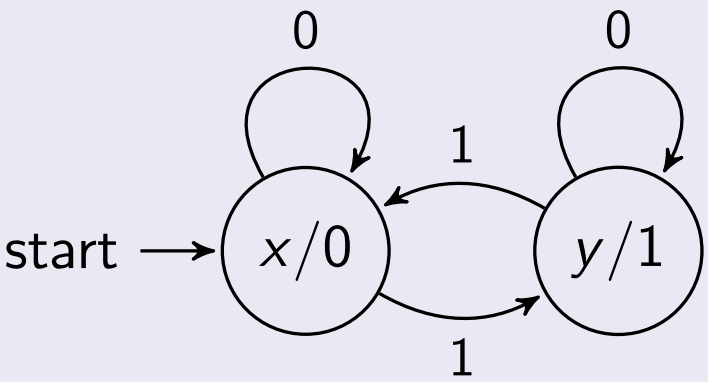
$$(t(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$$

Automatic Sequences

Definition (Automaton - DFA)

$$A = (Q, \Sigma = \{0, \ldots, k-1\}, \delta, q_0, \tau)$$

Example (Thue-Morse sequence)



$$n = 22 = (10110)_2, \qquad t_{22} = 1$$

$$(t(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$$

Different Points of View I

$(u(n))_{n\geq 0} = 011010011001011010010110011101001\ldots$

Automaton (Computer Science)

Substitution (Dynamics)

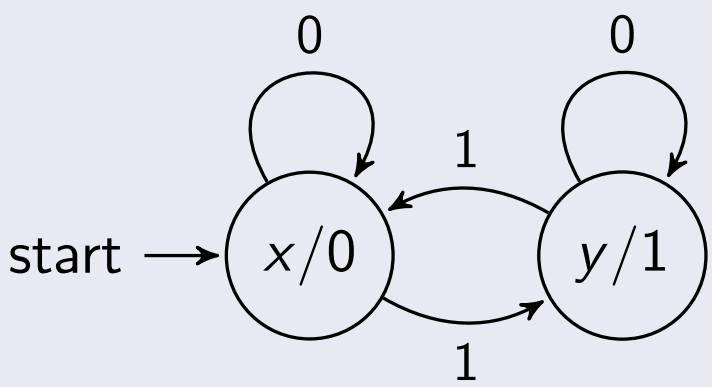
Coding of the fixpoint of a substitution:

$$x \to xy \qquad x \mapsto 0$$
$$y \to yx \qquad y \mapsto 1$$

Different Points of View I

$(u(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$

Automaton (Computer Science)



Substitution (Dynamics)

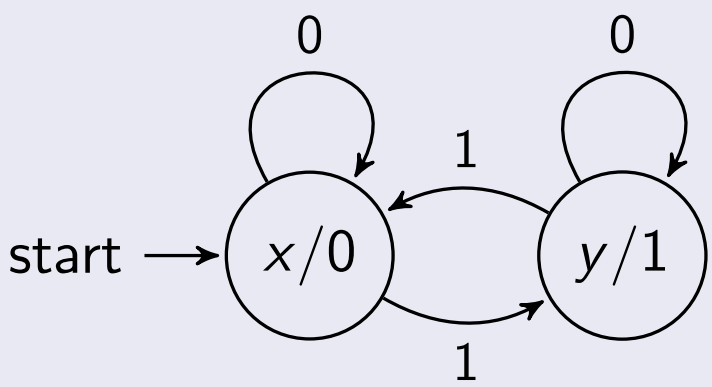
Coding of the fixpoint of a substitution:

$$x \to xy \qquad x \mapsto 0$$
$$y \to yx \qquad y \mapsto 1$$

Different Points of View I

$(u(n))_{n\geq 0} = 01101001100101101001011001101001\ldots$

Automaton (Computer Science)



Substitution (Dynamics)

Coding of the fixpoint of a substitution:

$$x \to xy \qquad x \mapsto 0$$
$$y \to yx \qquad y \mapsto 1$$

Different Points of View II

$(t(n))_{n\geq 0} = 0110100110010110100101100110100 1\ldots$

Formal Power Series (Algebra)

Algebraicity over $\mathbf{F}_q(X)$.

$$t(X) := \sum_{n\geq 0} a(n)X^n$$

$$X + (1+X)^2 t(X) + (1+X)^3 t(X)^2 = 0$$

Finite Kernel

The *k-kernel* of a sequence $a(n)$ is defined as

$$\{(u(nk^\lambda + r))_{n\geq 0} : \lambda \geq 0, 0 \leq r < k^\lambda\}.$$

Different Points of View II

$(t(n))_{n\geq 0} = 0110100110010110100101100110100 1\ldots$

Formal Power Series (Algebra)

Algebraicity over $\mathbf{F}_q(X)$.

$$t(X) := \sum_{n\geq 0} a(n)X^n$$

$$X + (1+X)^2 t(X) + (1+X)^3 t(X)^2 = 0$$

Finite Kernel

The *k-kernel* of a sequence $a(n)$ is defined as

$$\{(u(nk^\lambda + r))_{n\geq 0} : \lambda \geq 0, 0 \leq r < k^\lambda\}.$$

Different Points of View II

$(t(n))_{n\geq 0} = 0110100110010110100101100110100 1\ldots$

Formal Power Series (Algebra)

Algebraicity over $\mathbf{F}_q(X)$.

$$t(X) := \sum_{n\geq 0} a(n)X^n$$

$$X + (1+X)^2 t(X) + (1+X)^3 t(X)^2 = 0$$

Finite Kernel

The *k-kernel* of a sequence $a(n)$ is defined as

$$\{(u(nk^\lambda + r))_{n\geq 0} : \lambda \geq 0, 0 \leq r < k^\lambda\}.$$

Properties of Automatic Sequences

- Relatively easy to define (structured).
- Complex enough that interesting phenomena appear.
- Every subsequence $(u(xn+y))_{n\geq 0}$ along an arithmetic progression of an automatic sequence $\mathbf{u}$ is again automatic.

Properties of Automatic Sequences

- Relatively easy to define (structured).
- Complex enough that interesting phenomena appear.
- Every subsequence $(u(xn+y))_{n\geq 0}$ along an arithmetic progression of an automatic sequence $\mathbf{u}$ is again automatic.

Properties of Automatic Sequences

- Relatively easy to define (structured).
- Complex enough that interesting phenomena appear.
- Every subsequence $(u(xn+y))_{n\geq 0}$ along an arithmetic progression of an automatic sequence $\mathbf{u}$ is again automatic.

Subword Complexity

Let $\mathcal{A}$ be a finite alphabet and $\mathbf{u} = (u(n))_{n\in\mathbb{N}} \in \mathcal{A}^{\mathbb{N}}$.

Definition (Subword Complexity)

The subword complexity of a sequence $\mathbf{u} \in \mathcal{A}^{\mathbb{N}}$ is defined by

$$p_{\mathbf{u}}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists k, (u(k), \dots, u(k+L-1)) = \mathbf{w}\}.$$

$$p_{\mathbf{u}}(L) \leq |\mathcal{A}|^L$$

Subword complexity of automatic sequences

Let $\mathbf{u}$ be an automatic sequence. Then there exists $C > 0$ such that for all $L \in \mathbb{N}$

$$p_{\mathbf{u}}(L) \leq C \cdot L.$$

Subword Complexity

Let $\mathcal{A}$ be a finite alphabet and $\mathbf{u} = (u(n))_{n\in\mathbb{N}} \in \mathcal{A}^{\mathbb{N}}$.

Definition (Subword Complexity)

The subword complexity of a sequence $\mathbf{u} \in \mathcal{A}^{\mathbb{N}}$ is defined by

$$p_{\mathbf{u}}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists k, (u(k), \dots, u(k+L-1)) = \mathbf{w}\}.$$

$$p_{\mathbf{u}}(L) \leq |\mathcal{A}|^L$$

Subword complexity of automatic sequences

Let $\mathbf{u}$ be an automatic sequence. Then there exists $C > 0$ such that for all $L \in \mathbb{N}$

$$p_{\mathbf{u}}(L) \leq C \cdot L.$$

Subword Complexity

Let $\mathcal{A}$ be a finite alphabet and $\mathbf{u} = (u(n))_{n\in\mathbb{N}} \in \mathcal{A}^{\mathbb{N}}$.

Definition (Subword Complexity)

The subword complexity of a sequence $\mathbf{u} \in \mathcal{A}^{\mathbb{N}}$ is defined by

$$p_{\mathbf{u}}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists k, (u(k), \dots, u(k+L-1)) = \mathbf{w}\}.$$

$$p_{\mathbf{u}}(L) \leq |\mathcal{A}|^L$$

Subword complexity of automatic sequences

Let $\mathbf{u}$ be an automatic sequence. Then there exists $C > 0$ such that for all $L \in \mathbb{N}$

$$p_{\mathbf{u}}(L) \leq C \cdot L.$$

Subword Complexity

Let $\mathcal{A}$ be a finite alphabet and $\mathbf{u} = (u(n))_{n\in\mathbb{N}} \in \mathcal{A}^{\mathbb{N}}$.

Definition (Subword Complexity)

The subword complexity of a sequence $\mathbf{u} \in \mathcal{A}^{\mathbb{N}}$ is defined by

$$p_{\mathbf{u}}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists k, (u(k), \dots, u(k+L-1)) = \mathbf{w}\}.$$

$$p_{\mathbf{u}}(L) \leq |\mathcal{A}|^L$$

Subword complexity of automatic sequences

Let $\mathbf{u}$ be an automatic sequence. Then there exists $C > 0$ such that for all $L \in \mathbb{N}$

$$p_{\mathbf{u}}(L) \leq C \cdot L.$$

Arithmetic subword complexity

Definition (arithmetic subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{AP}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists n \geq 0, m \geq 1 : u(n+im) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem (Avgustinovich, Fon-Der-Flaass and Frid; 2003)

- A certain class of invertible automatic sequences has maximal arithmetic subword complexity. (E.g. Thue-Morse sequence)
- Certain synchronizing automatic sequences have at most linear arithmetic subword complexity. (E.g. paperfolding sequence)

Arithmetic subword complexity

Definition (arithmetic subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{AP}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists n \geq 0, m \geq 1 : u(n+im) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem (Avgustinovich, Fon-Der-Flaass and Frid; 2003)

- A certain class of invertible automatic sequences has maximal arithmetic subword complexity. (E.g. Thue-Morse sequence)
- Certain synchronizing automatic sequences have at most linear arithmetic subword complexity. (E.g. paperfolding sequence)

Arithmetic subword complexity

Definition (arithmetic subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{AP}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists n \geq 0, m \geq 1 : u(n+im) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem (Avgustinovich, Fon-Der-Flaass and Frid; 2003)

- A certain class of invertible automatic sequences has maximal arithmetic subword complexity. (E.g. Thue-Morse sequence)
- Certain synchronizing automatic sequences have at most linear arithmetic subword complexity. (E.g. paperfolding sequence)

Main Result

Definition (polynomial subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{\leq d}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists P \in \mathbb{Z}[x], P(\mathbb{N}) \subseteq \mathbb{N}, \deg P \leq d : u(P(i)) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem 1 (Konieczny, M.; 2024+)

Let $u(n)$ be an automatic sequence. Then there exists $r \in \mathbb{N}$ such that for any $d \geq 1$

$$p_{\mathbf{u}}^{\leq d}(L) = (r + o(1))^L.$$

Basically the same proof: there exist $c > 0, \eta > 0$ such that

$$r^L \leq p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot \exp(cL^{1-\eta}).$$

Main Result

Definition (polynomial subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{\leq d}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists P \in \mathbb{Z}[x], P(\mathbb{N}) \subseteq \mathbb{N}, \deg P \leq d : u(P(i)) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem 1 (Konieczny, M.; 2024+)

Let $u(n)$ be an automatic sequence. Then there exists $r \in \mathbb{N}$ such that for any $d \geq 1$

$$p_{\mathbf{u}}^{\leq d}(L) = (r + o(1))^L.$$

Basically the same proof: there exist $c > 0, \eta > 0$ such that

$$r^L \leq p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot \exp(cL^{1-\eta}).$$

Main Result

Definition (polynomial subword complexity)

Let $\mathbf{u}$ be a sequence over a finite alphabet $\mathcal{A}$.

$$p_{\mathbf{u}}^{\leq d}(L) := \#\{\mathbf{w} \in \mathcal{A}^L : \exists P \in \mathbb{Z}[x], P(\mathbb{N}) \subseteq \mathbb{N}, \deg P \leq d : u(P(i)) = \mathbf{w}(i) \text{ for } i = 0, \ldots, L-1\}.$$

Theorem 1 (Konieczny, M.; 2024+)

Let $u(n)$ be an automatic sequence. Then there exists $r \in \mathbb{N}$ such that for any $d \geq 1$

$$p_{\mathbf{u}}^{\leq d}(L) = (r + o(1))^L.$$

Basically the same proof: there exist $c > 0, \eta > 0$ such that

$$r^L \leq p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot \exp(cL^{1-\eta}).$$

Synchronizing Automata

Definition (Synchronizing Automaton / Word)

$\exists \mathbf{w}_0 : \delta(q, \mathbf{w}_0) = x \quad \forall q.$

Example

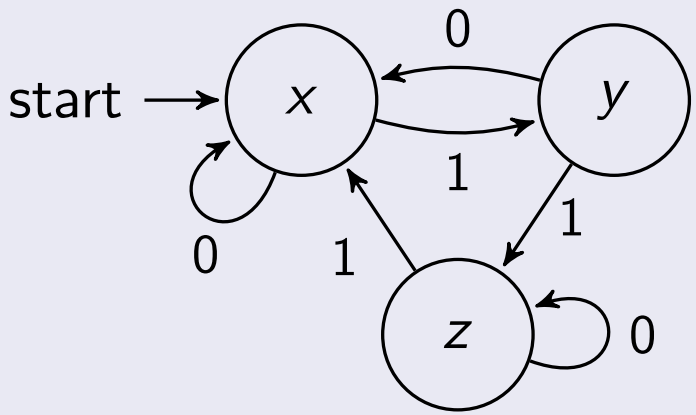
$\mathbf{w}_0 = 010.$

Synchronizing Automata

Definition (Synchronizing Automaton / Word)

$\exists \mathbf{w}_0 : \delta(q, \mathbf{w}_0) = x \quad \forall q.$

Example



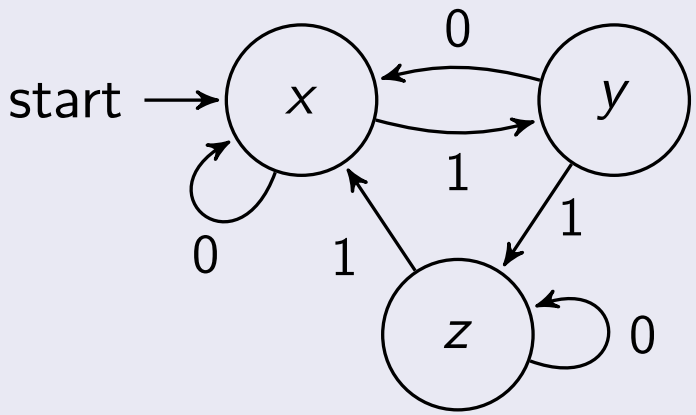
$\mathbf{w}_0 = 010.$

Synchronizing Automata

Definition (Synchronizing Automaton / Word)

$\exists \mathbf{w}_0 : \delta(q, \mathbf{w}_0) = x \quad \forall q.$

Example



$\mathbf{w}_0 = 010.$

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Synchronizing Automata

Key Property

- Any word $\mathbf{w}$ containing $\mathbf{w}_0$ is also synchronizing.
- Most words are synchronizing.
- $\mathbf{u}$ can be approximated by periodic sequences:
  Let λ be large. Most words of length λ are synchronizing.
  $u(n) = u(n \bmod k^\lambda)$ if $n \bmod k^\lambda$ is synchronizing.

"Usual" Strategy

- Understand the problem for periodic sequences.
- Transfer the result to synchronizing automatic sequences.

Minimal sets

General (primitive) automata are not synchronizing.

Definition: Minimal sets

$$c := \min_{\mathbf{w}\in\Sigma_k^*} \#\{\delta(q,\mathbf{w}) : q \in Q\} = \min_{\mathbf{w}\in\Sigma_k^*} \#\delta(Q,\mathbf{w})$$

$$\mathcal{M} := \{M \subset Q : \#M = c, \exists\, \mathbf{w} \text{ with } \delta(Q,\mathbf{w}) = M\}.$$

Lemma

We can define a "compatible" automaton where the states are $\mathcal{M}$. This automaton is synchronizing.

Minimal sets

General (primitive) automata are not synchronizing.

Definition: Minimal sets

$$c := \min_{\mathbf{w}\in\Sigma_k^*} \#\{\delta(q,\mathbf{w}) : q \in Q\} = \min_{\mathbf{w}\in\Sigma_k^*} \#\delta(Q,\mathbf{w})$$

$$\mathcal{M} := \{M \subset Q : \#M = c, \exists\, \mathbf{w} \text{ with } \delta(Q,\mathbf{w}) = M\}.$$

Lemma

We can define a "compatible" automaton where the states are $\mathcal{M}$. This automaton is synchronizing.

Minimal sets

General (primitive) automata are not synchronizing.

Definition: Minimal sets

$$c := \min_{\mathbf{w}\in\Sigma_k^*} \#\{\delta(q,\mathbf{w}) : q \in Q\} = \min_{\mathbf{w}\in\Sigma_k^*} \#\delta(Q,\mathbf{w})$$

$$\mathcal{M} := \{M \subset Q : \#M = c, \exists\, \mathbf{w} \text{ with } \delta(Q,\mathbf{w}) = M\}.$$

Lemma

We can define a "compatible" automaton where the states are $\mathcal{M}$. This automaton is synchronizing.

Minimal sets

General (primitive) automata are not synchronizing.

Definition: Minimal sets

$$c := \min_{\mathbf{w}\in\Sigma_k^*} \#\{\delta(q,\mathbf{w}) : q \in Q\} = \min_{\mathbf{w}\in\Sigma_k^*} \#\delta(Q,\mathbf{w})$$

$$\mathcal{M} := \{M \subset Q : \#M = c, \exists\, \mathbf{w} \text{ with } \delta(Q,\mathbf{w}) = M\}.$$

Lemma

We can define a "compatible" automaton where the states are $\mathcal{M}$.
This automaton is synchronizing.

Minimal sets

General (primitive) automata are not synchronizing.

Definition: Minimal sets

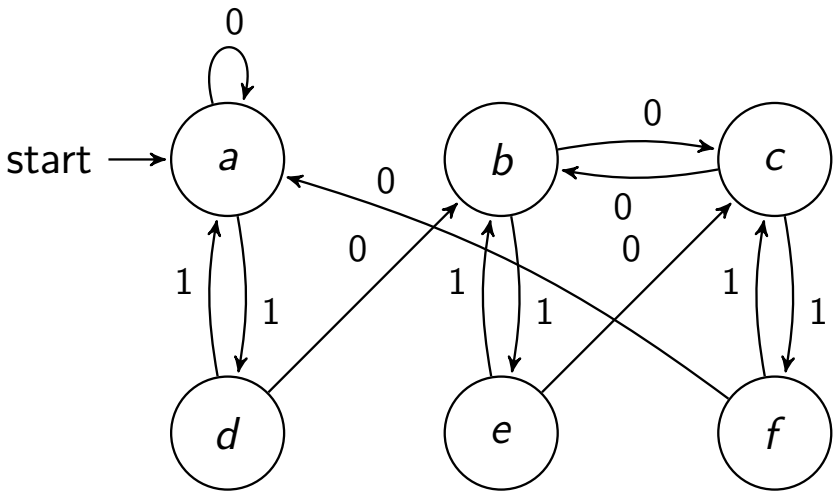
$$c := \min_{\mathbf{w}\in\Sigma_k^*} \#\{\delta(q,\mathbf{w}) : q \in Q\} = \min_{\mathbf{w}\in\Sigma_k^*} \#\delta(Q,\mathbf{w})$$

$$\mathcal{M} := \{M \subset Q : \#M = c, \exists\, \mathbf{w} \text{ with } \delta(Q,\mathbf{w}) = M\}.$$

Lemma

We can define a "compatible" automaton where the states are $\mathcal{M}$. This automaton is synchronizing.

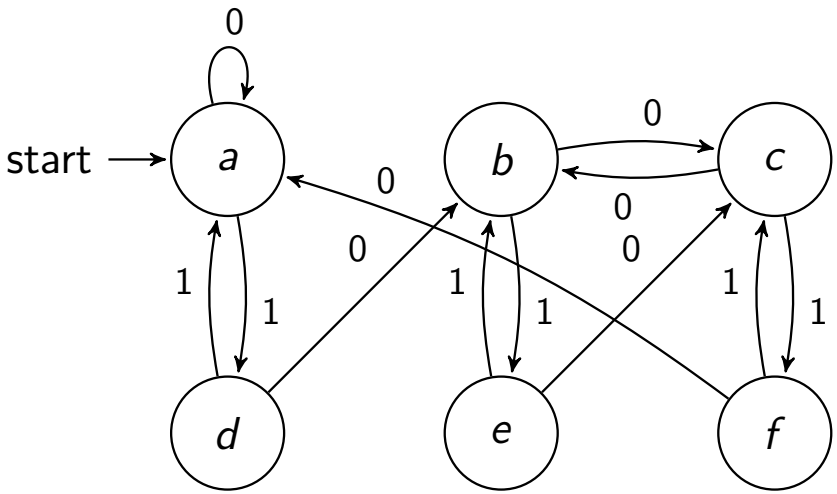
Tower of Hanoi



$c = 3, M_0 = \{a, b, c\}, M_1 = \{d, e, f\}.$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
M(n)	M_0	M_1	M_0	M_0	M_0	M_1	M_0	M_1	M_0	M_1	M_0	M_0	M_0	M_1

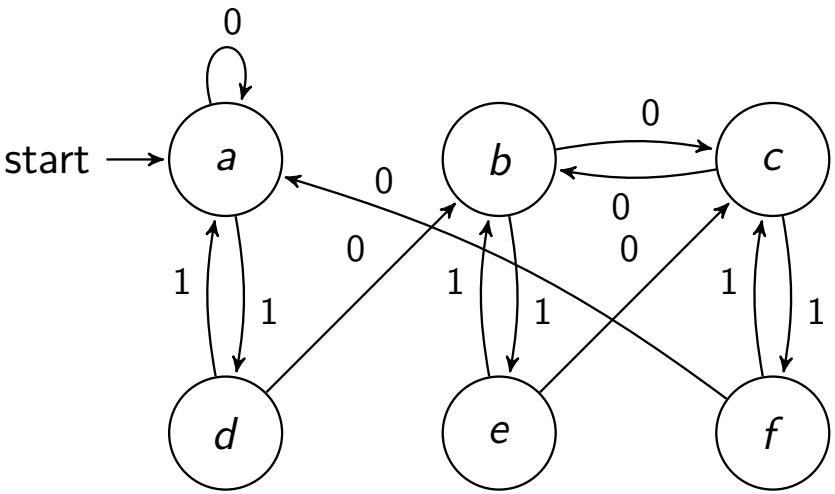
Tower of Hanoi

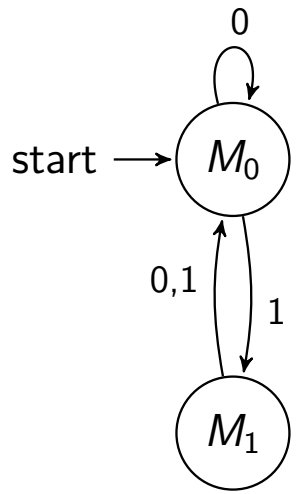


$c = 3, M_0 = \{a, b, c\}, M_1 = \{d, e, f\}.$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
M(n)	M_0	M_1	M_0	M_0	M_0	M_1	M_0	M_1	M_0	M_1	M_0	M_0	M_0	M_1

Tower of Hanoi

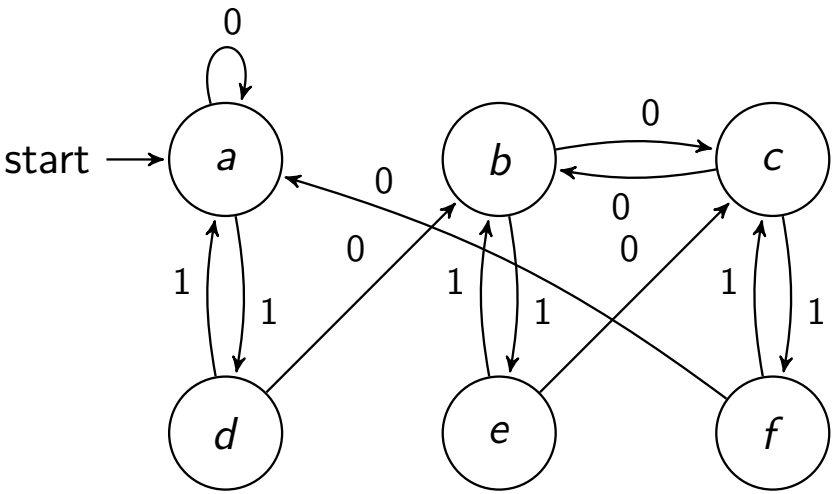


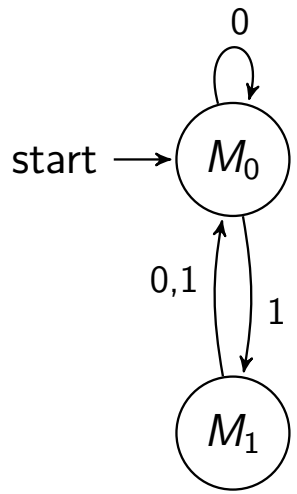


$c = 3, M_0 = \{a, b, c\}, M_1 = \{d, e, f\}.$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
M(n)	M_0	M_1	M_0	M_0	M_0	M_1	M_0	M_1	M_0	M_1	M_0	M_0	M_0	M_1

Tower of Hanoi





$c = 3, M_0 = \{a, b, c\}, M_1 = \{d, e, f\}.$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
M(n)	M_0	M_1	M_0	M_0	M_0	M_1	M_0	M_1	M_0	M_1	M_0	M_0	M_0	M_1

Height

The height of a primitive k-automaton with $\delta(q_0, 0) = q_0$ and automatic sequence u is given by

$$h := \max\{m : \gcd(m, k) = 1, \\ m \mid n \text{ for all } n \geq 0 \text{ with } \delta(q_0, (n)_k) = q_0\}.$$

This allows for a cyclical decomposition using for $0 \leq j < h$

$$C_j = \{q \in Q : \exists n \in \mathbb{N}, n \equiv j \bmod h, \delta(q_0, (n)_k) = q\}.$$

Height

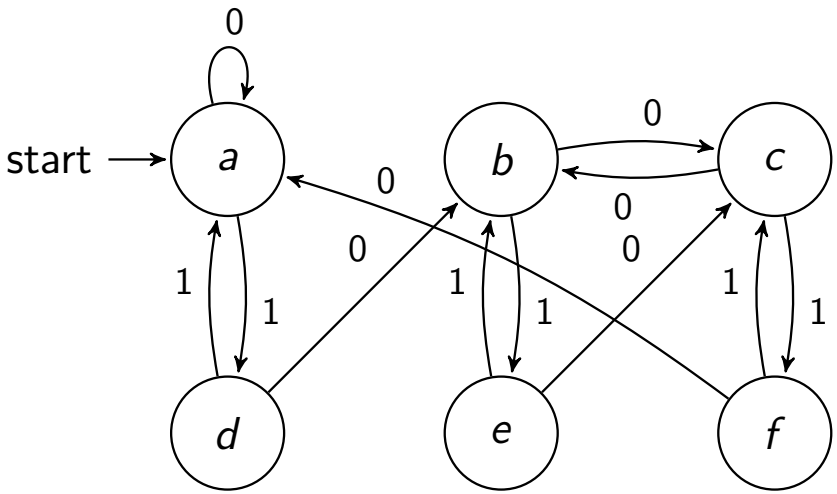
The height of a primitive k-automaton with $\delta(q_0, 0) = q_0$ and automatic sequence u is given by

$$h := \max\{m : \gcd(m, k) = 1, \\ m \mid n \text{ for all } n \geq 0 \text{ with } \delta(q_0, (n)_k) = q_0\}.$$

This allows for a cyclical decomposition using for $0 \leq j < h$

$$C_j = \{q \in Q : \exists n \in \mathbb{N}, n \equiv j \bmod h, \delta(q_0, (n)_k) = q\}.$$

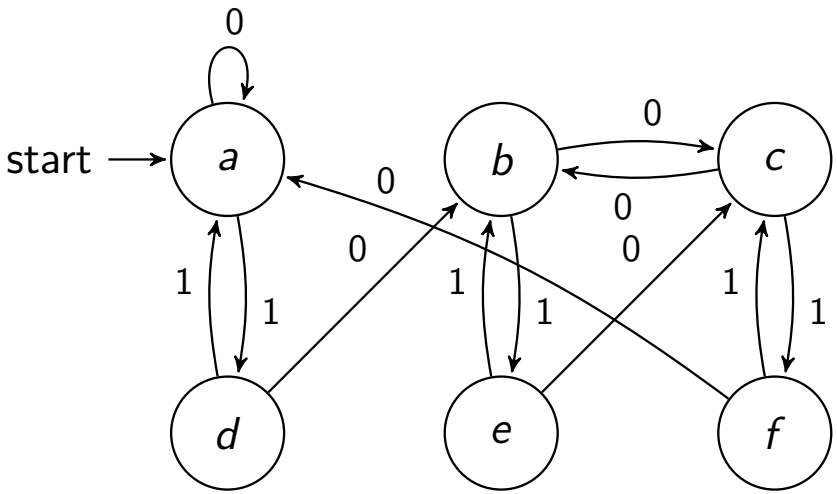
Tower of Hanoi (again)



$h = 3; C_0 = \{a, f\}, C_1 = \{c, d\}, C_2 = \{b, e\}$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
C(n)	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1

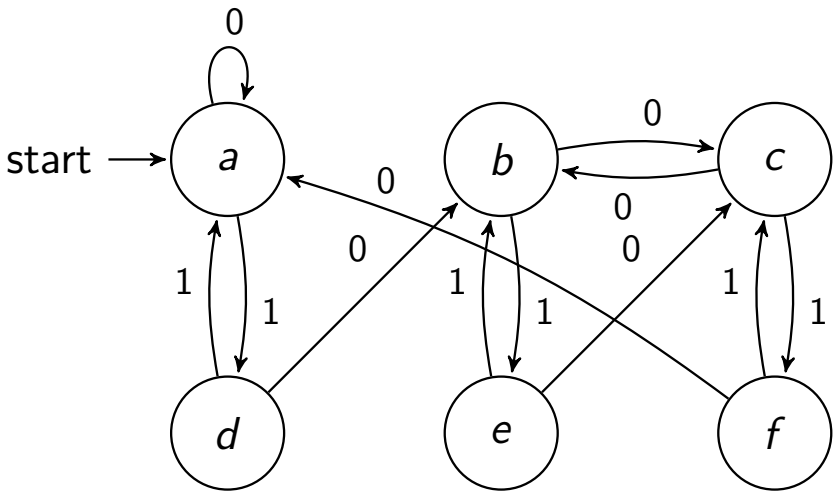
Tower of Hanoi (again)

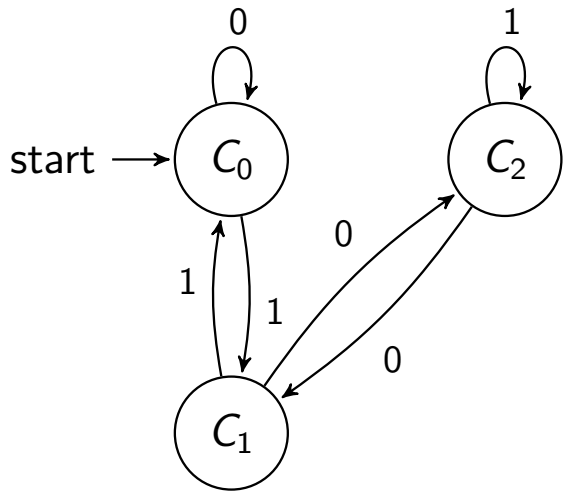


$h = 3; C_0 = \{a, f\}, C_1 = \{c, d\}, C_2 = \{b, e\}$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
C(n)	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1

Tower of Hanoi (again)

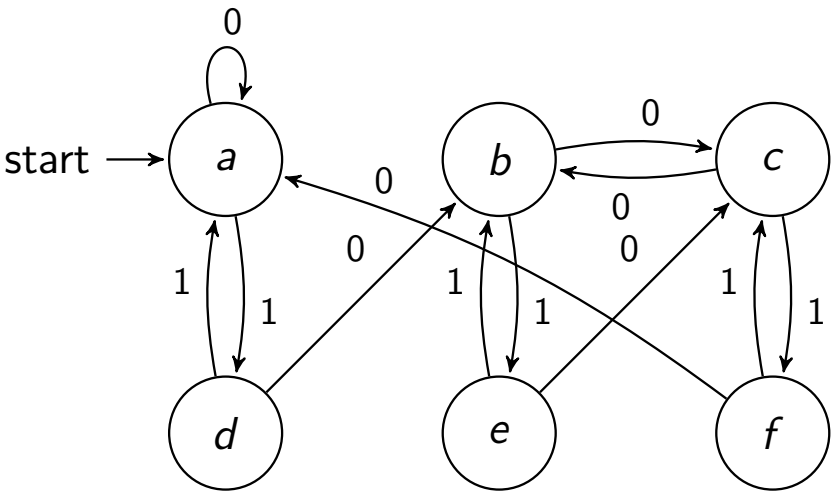


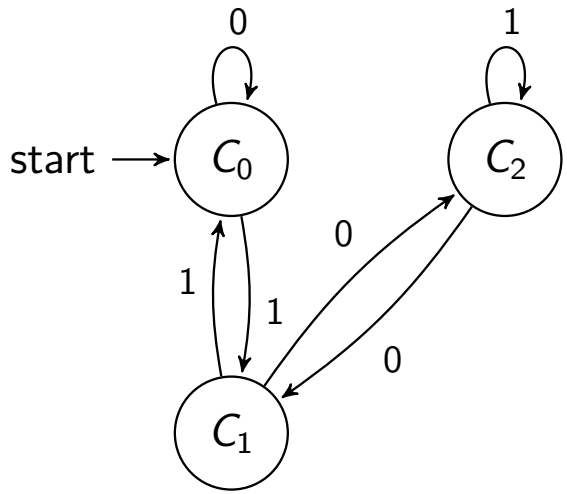


$h = 3; C_0 = \{a, f\}, C_1 = \{c, d\}, C_2 = \{b, e\}$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
C(n)	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1

Tower of Hanoi (again)





$h = 3; C_0 = \{a, f\}, C_1 = \{c, d\}, C_2 = \{b, e\}$

$u(n)$	a	d	b	a	c	e	a	d	b	f	c	b	a	d
C(n)	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_1

Structured part

Let $\mathcal{A}$ be a primitive automaton with output such that $\delta(q_0, 0) = q_0$.

- We define $S_{i,j} = M_i \cap C_j$.
- We can define a "compatible" automaton/transitions where the states are exactly the $S_{i,j}$.
- We define the output

$$\tau(S_{i,j}) = \frac{\sum_{q \in S_{i,j}} \tau(q)}{\#S_{i,j}}.$$

The corresponding automaton (with output) is called *structured part* of $\mathcal{A}$.

Structured part

Let $\mathcal{A}$ be a primitive automaton with output such that $\delta(q_0, 0) = q_0$.

- We define $S_{i,j} = M_i \cap C_j$.
- We can define a "compatible" automaton/transitions where the states are exactly the $S_{i,j}$.
- We define the output

$$\tau(S_{i,j}) = \frac{\sum_{q \in S_{i,j}} \tau(q)}{\#S_{i,j}}.$$

The corresponding automaton (with output) is called *structured part* of $\mathcal{A}$.

Structured part

Let $\mathcal{A}$ be a primitive automaton with output such that $\delta(q_0, 0) = q_0$.

- We define $S_{i,j} = M_i \cap C_j$.
- We can define a "compatible" automaton/transitions where the states are exactly the $S_{i,j}$.
- We define the output

$$\tau(S_{i,j}) = \frac{\sum_{q \in S_{i,j}} \tau(q)}{\#S_{i,j}}.$$

The corresponding automaton (with output) is called *structured part* of $\mathcal{A}$.

Structured part

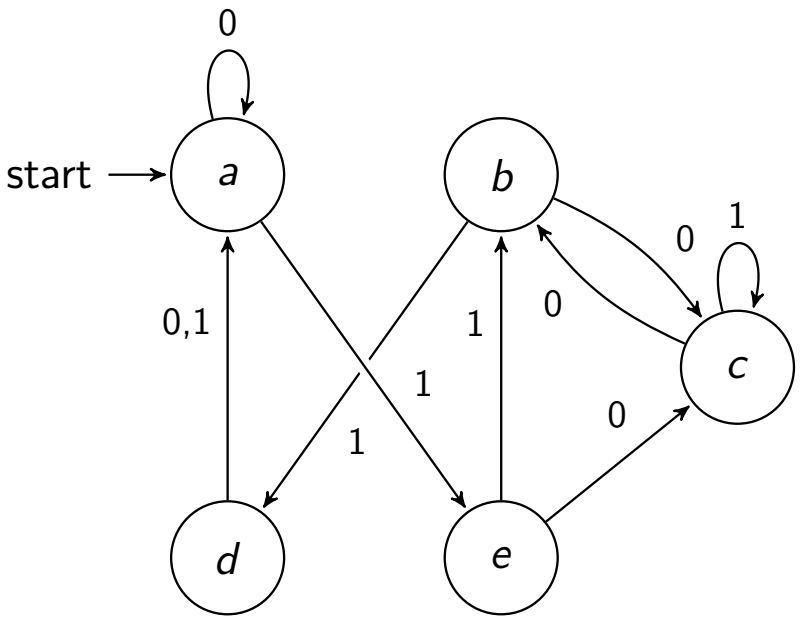
Let $\mathcal{A}$ be a primitive automaton with output such that $\delta(q_0, 0) = q_0$.

- We define $S_{i,j} = M_i \cap C_j$.
- We can define a "compatible" automaton/transitions where the states are exactly the $S_{i,j}$.
- We define the output

$$\tau(S_{i,j}) = \frac{\sum_{q \in S_{i,j}} \tau(q)}{\#S_{i,j}}.$$

The corresponding automaton (with output) is called *structured part* of $\mathcal{A}$.

Example

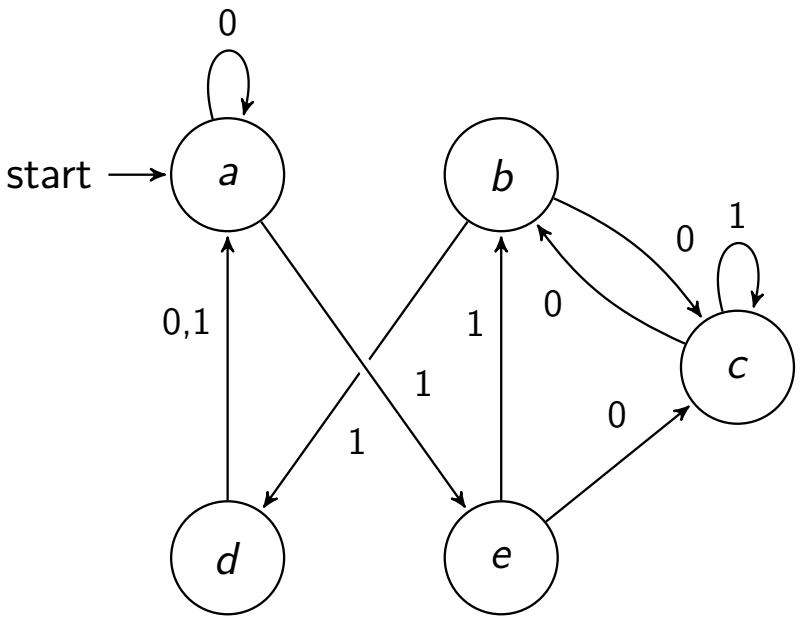


$$h = 1; C_0 = Q; \qquad c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$$

$$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

Example

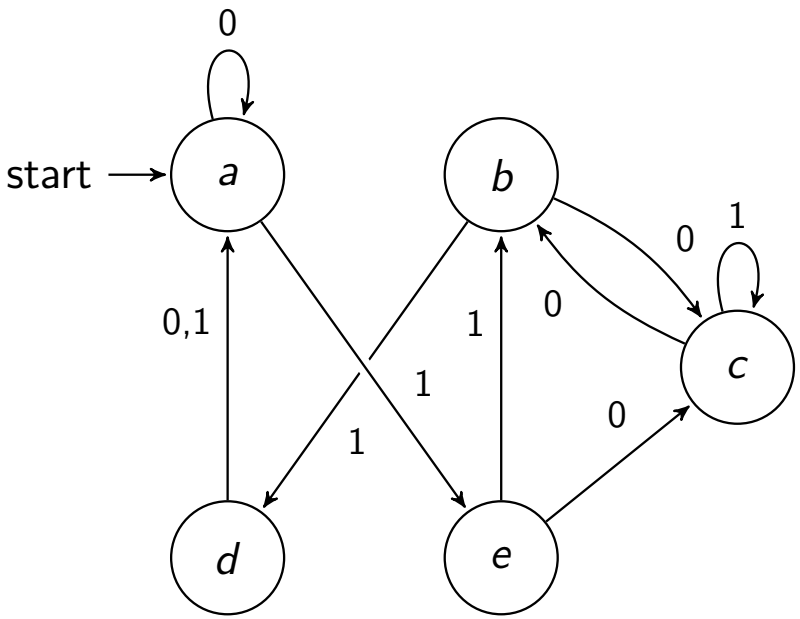


$h = 1; C_0 = Q;$ $c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$

$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

Example

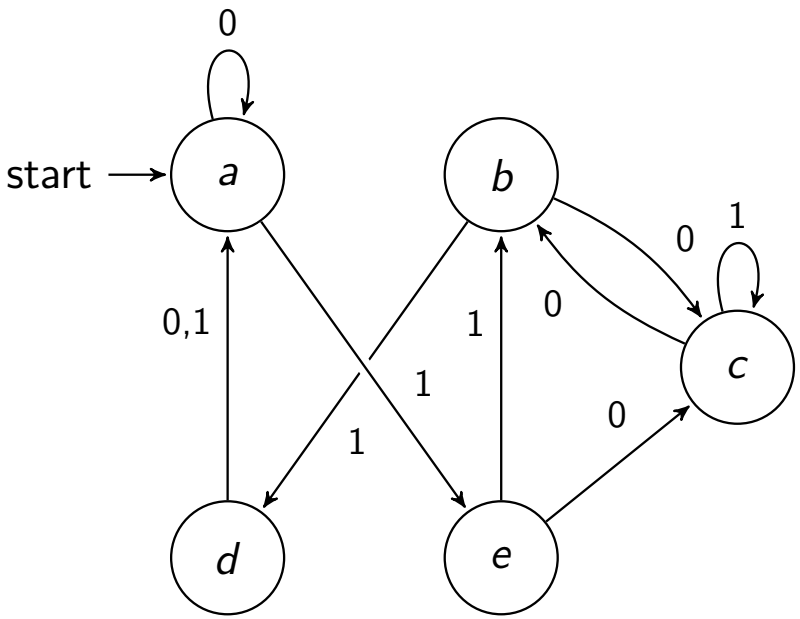


$$h = 1; C_0 = Q; \qquad c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$$

$$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

Example

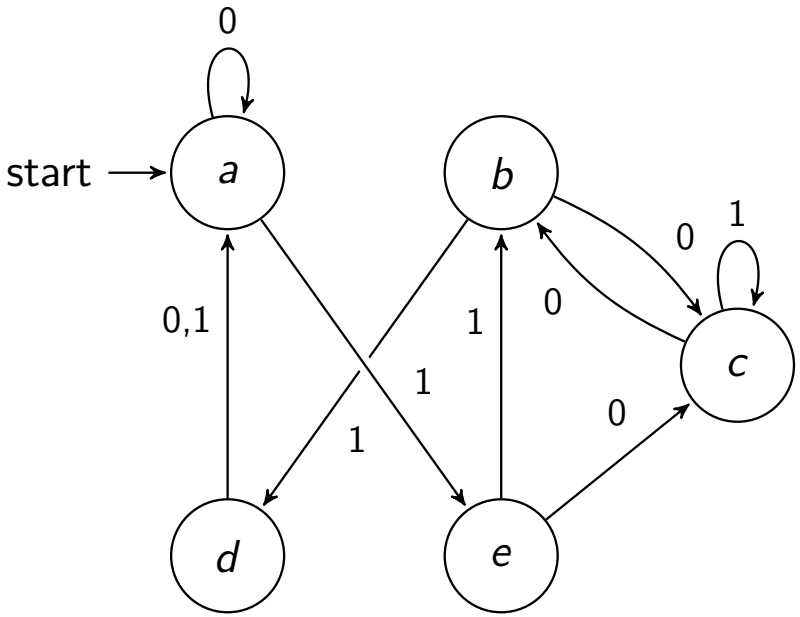


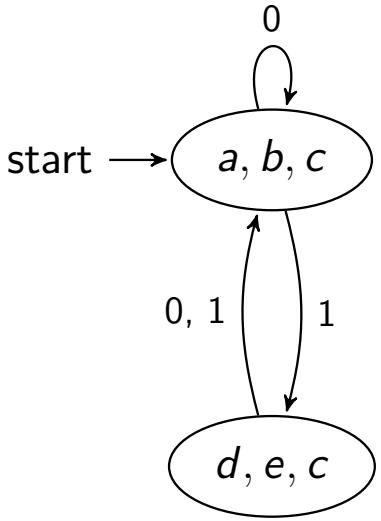
$$h = 1; C_0 = Q; \qquad c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$$

$$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

Example



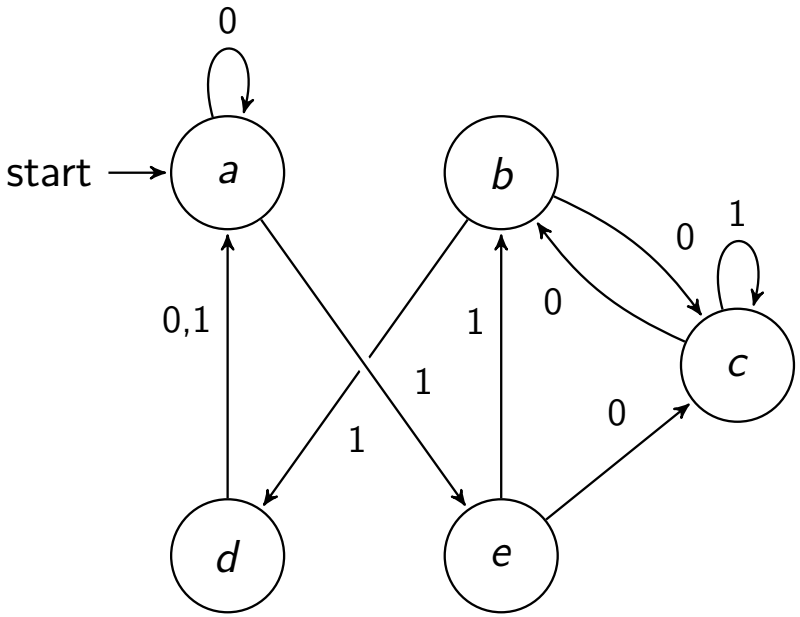


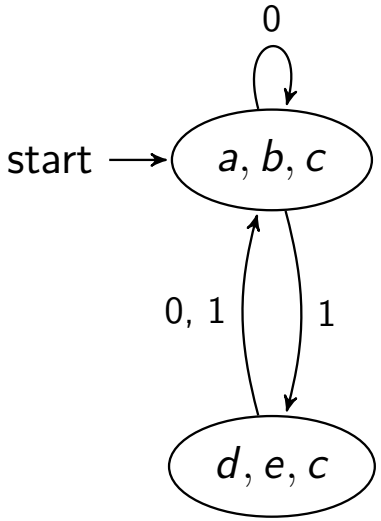
$$h = 1; C_0 = Q; \qquad c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$$

$$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

Example





$$h = 1; C_0 = Q; \qquad c = 3; M_0 = \{a, b, c\}, M_1 = \{d, e, c\}.$$

$$S_{0,0} = S_0 = \{a, b, c\}, \quad S_{1,0} = S_1 = \{d, e, c\}.$$

$u(n)$	a	e	c	b	b	c	c	d	c	d	b	c	b	c
S(n)	S_0	S_1	S_0	S_0	S_0	S_1	S_0	S_1	S_0	S_1	S_0	S_0	S_0	S_1

General philosophy

- The sequence $S(n)$ gives a "coarse picture", which is highly structured, i.e. $S(n)$ is "almost periodic".
- Which element from $S(n)$ is chosen for $u(n)$ behaves "randomly".

General philosophy

- The sequence $S(n)$ gives a "coarse picture", which is highly structured, i.e. $S(n)$ is "almost periodic".
- Which element from $S(n)$ is chosen for $u(n)$ behaves "randomly".

Effective alphabet size

We define the *effective alphabet size*

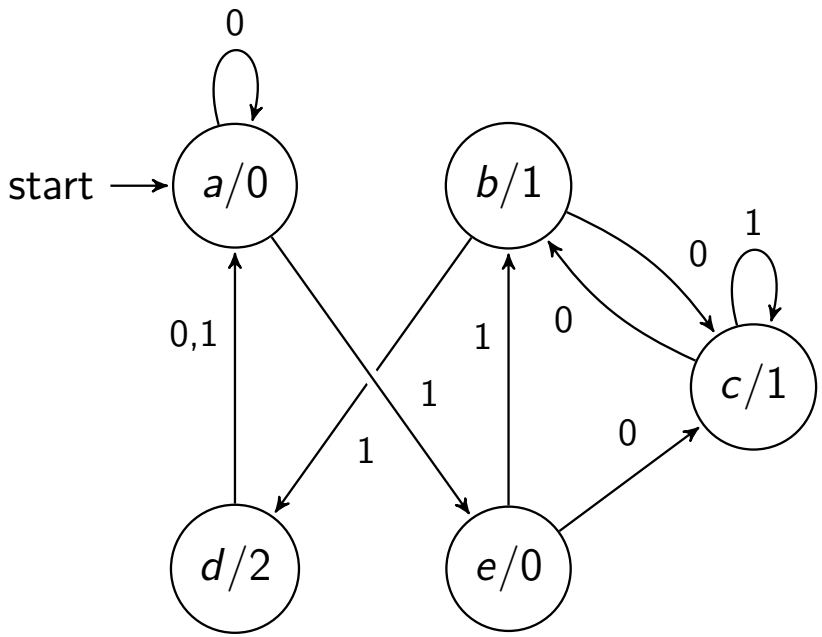
$$r = \max_{S_{i,j}} \#\{\tau(q) : q \in S_{i,j}\}.$$

$r = 3$.

Effective alphabet size

We define the *effective alphabet size*

$$r = \max_{S_{i,j}} \#\{\tau(q) : q \in S_{i,j}\}.$$

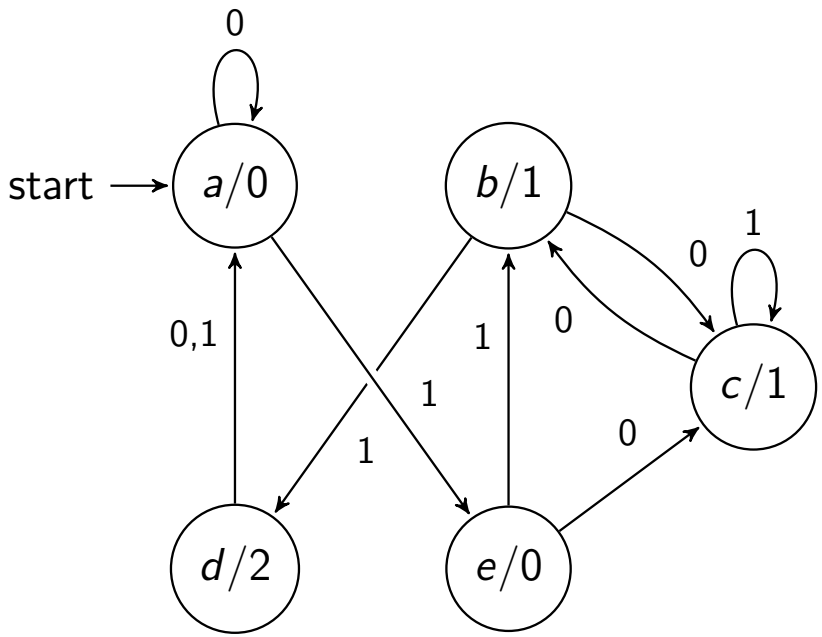


$r = 3$.

Effective alphabet size

We define the *effective alphabet size*

$$r = \max_{S_{i,j}} \#\{\tau(q) : q \in S_{i,j}\}.$$



$r = 3$.

Arithmetic regularity lemma for automatic sequences

Theorem (Byszewski, Konieczny, M.; 2023)

Let $a : \mathbb{N} \to \mathbb{C}$ be a primitive k-automatic sequence. Then it has a decomposition as $a = a_{str} + a_{uni}$, where

- a_{str} is a structured part of a.
- a_{uni} is uniform in the sense that for each $d \geq 2$ there exists $\kappa > 0$ such that

$$||a_{uni}||_{U^d[N]} \ll N^{-\kappa}.$$

Higher order Fourier analysis

Definition: Gowers norm

Let $d \geq 2$. Let $f : [N] = \{0, 1, \ldots, N-1\} \to \mathbb{C}$. Then $||f||_{U^d[N]} \geq 0$ is defined by:

$$||f||_{U^d[N]}^{2^d} = \mathbb{E}_{\mathbf{n}} \prod_{\omega \in \{0,1\}^d} C^{|\omega|} f(n_0 + \omega_1 n_1 + \ldots + \omega_d n_d),$$

where the average is taken over all parallelepipeds in $[N]$, i.e. over all $\mathbf{n} \in \mathbb{Z}^{d+1}$ such that $n_0 + \omega_1 n_1 + \ldots + \omega_d n_d \in [N]$ for all $\omega \in \{0, 1\}^d$.

Higher order Fourier analysis

- $||f||_{U^d[N]}$ is a norm for $d \geq 2$.
- $||f||_{U^1[N]} \ll ||f||_{U^2[N]} \ll ||f||_{U^3[N]} \ll \cdots$

Generalised von Neumann Theorem

Fix $d \geq 2$ and let $f_0, f_1, \ldots, f_d : [N] \to \mathbb{C}$ be 1 bounded. Then

$$\left| \mathop{\mathbb{E}}_{n,m} f_0(n) f_1(n+m) f_2(n+2m) \cdots f_d(n+dm) \right| \ll \min_{0 \leq i \leq d} ||f_i||_{U^d[N]}.$$

Higher order Fourier analysis

- $||f||_{U^d[N]}$ is a norm for $d \geq 2$.
- $||f||_{U^1[N]} \ll ||f||_{U^2[N]} \ll ||f||_{U^3[N]} \ll \cdots$

Generalised von Neumann Theorem

Fix $d \geq 2$ and let $f_0, f_1, \ldots, f_d : [N] \to \mathbb{C}$ be 1 bounded. Then

$$\left| \mathop{\mathbb{E}}_{n,m} f_0(n) f_1(n+m) f_2(n+2m) \cdots f_d(n+dm) \right| \ll \min_{0 \leq i \leq d} ||f_i||_{U^d[N]}.$$

Higher order Fourier analysis

- $||f||_{U^d[N]}$ is a norm for $d \geq 2$.
- $||f||_{U^1[N]} \ll ||f||_{U^2[N]} \ll ||f||_{U^3[N]} \ll \dots$

Generalised von Neumann Theorem

Fix $d \geq 2$ and let $f_0, f_1, \dots, f_d : [N] \to \mathbb{C}$ be 1 bounded. Then

$$\left| \mathop{\mathbb{E}}_{n,m} f_0(n) f_1(n+m) f_2(n+2m) \cdots f_d(n+dm) \right| \ll \min_{0 \leq i \leq d} ||f_i||_{U^d[N]}.$$

Proof of the Main Theorem

Theorem (Konieczny, M.; 2024+)

Let $a(n)$ be an automatic sequence. Then there exists $r \in \mathbb{N}$ such that for any $d \geq 1$

$$r^L \leq p_{\mathbf{a}}^{\leq d}(L) \leq r^L \cdot \exp(cL^{1-\eta}).$$

Lower bound

Let $u : \mathbb{N} \to \Omega$ be a primitive automatic sequence. For $x \in \Omega$, let

$$u^{(x)}(n) = \begin{cases} 1 & \text{if } u(n) = x \\ 0 & \text{otherwise.} \end{cases}$$

Lemma

For any $S_{i,j}$ there exists an arithmetic progression AP such that $S(n) = S_{i,j}$ for every $n \in AP$.

In paritcular, for each $q \in S_{i,j}$ we have $u_{str}^{(\tau(q))}$ is constant and positive along AP.

Lower bound

Let $u : \mathbb{N} \to \Omega$ be a primitive automatic sequence. For $x \in \Omega$, let

$$u^{(x)}(n) = \begin{cases} 1 & \text{if } u(n) = x \\ 0 & \text{otherwise.} \end{cases}$$

Lemma

For any $S_{i,j}$ there exists an arithmetic progression AP such that $S(n) = S_{i,j}$ for every $n \in AP$.

In paritcular, for each $q \in S_{i,j}$ we have $u_{str}^{(\tau(q))}$ is constant and positive along AP.

Lower bound

Let $u : \mathbb{N} \to \Omega$ be a primitive automatic sequence. For $x \in \Omega$, let

$$u^{(x)}(n) = \begin{cases} 1 & \text{if } u(n) = x \\ 0 & \text{otherwise.} \end{cases}$$

Lemma

For any $S_{i,j}$ there exists an arithmetic progression AP such that $S(n) = S_{i,j}$ for every $n \in AP$.
In paritcular, for each $q \in S_{i,j}$ we have $u_{str}^{(\tau(q))}$ is constant and positive along AP.

Lower bound

Let N be a large integer. We will estimate the count of L-term arithmetic progressions in $AP \cap [N]$ where w appears:

$$C = \sum_{n,m=0}^{N-1} \prod_{i=0}^{L-1} \mathbb{1}_{AP\cap[N]}(n+im)u^{(w(i))}(n+im).$$

By generalised von Neumann theorem:

$$C = \sum_{n,m=0}^{N-1} \prod_{i=0}^{L-1} \mathbb{1}_{AP\cap[N]}(n+im)u_{str}^{(w(i))}(n+im) + O(N^{2-\kappa}).$$

We recall that $u_{str}^{(w(i))}$ is constant and positive.
Therefore, $C \gg N^2$.

Upper bound

Theorem 2 (Deshouillers, Drmota, M., Shubin, Spiegelhofer; 2024+)

Let $s(n)$ be a synchronizing automatic sequence. Then for any $d \geq 1$

$$p_{\mathbf{s}}^{\leq d}(L) \leq \exp(o(L)).$$

Basically the same proof: there exist $c > 0, \eta > 0$ such that

$$p_{\mathbf{s}}^{\leq d}(L) \leq \exp(cL^{1-\eta}).$$

Upper bound

Theorem 2 (Deshouillers, Drmota, M., Shubin, Spiegelhofer; 2024+)

Let $s(n)$ be a synchronizing automatic sequence. Then for any $d \geq 1$

$$p_{\mathbf{s}}^{\leq d}(L) \leq \exp(o(L)).$$

Basically the same proof: there exist $c > 0, \eta > 0$ such that

$$p_{\mathbf{s}}^{\leq d}(L) \leq \exp(cL^{1-\eta}).$$

Consequence of Theorem 2

We recall that we have $\delta(q_0, (n)_k) \in S(n)$.

Thus, knowing $S(n)$ there are at most r different values $u(n)$ can take.

$$p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot p_{\mathbf{S}}^{\leq d}(L)$$

which proves the upper bound of Theorem 1.

Consequence of Theorem 2

We recall that we have $\delta(q_0, (n)_k) \in S(n)$.
Thus, knowing $S(n)$ there are at most r different values $u(n)$ can take.

$$p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot p_{\mathbf{S}}^{\leq d}(L)$$

which proves the upper bound of Theorem 1.

Consequence of Theorem 2

We recall that we have $\delta(q_0, (n)_k) \in S(n)$.
Thus, knowing $S(n)$ there are at most r different values $u(n)$ can take.

$$p_{\mathbf{u}}^{\leq d}(L) \leq r^L \cdot p_{\mathbf{S}}^{\leq d}(L)$$

which proves the upper bound of Theorem 1.

Proof of Theorem 2

Naive approach

- Let f be a m-periodic function. Then $p_f^{\leq d}(L) \leq m^{d+1}$.
- Approximate $s(n)$ by a k^λ-periodic function $f(n)$.
- $s(n)$ and $f(n)$ agree on most residue classes modulo k^λ.
- Problem: P can hit the "bad" residue classes very often. (Trivial example: $P(x) = k^\lambda x + r$.)

Proof of Theorem 2

Naive approach

- Let f be a m-periodic function. Then $p_f^{\leq d}(L) \leq m^{d+1}$.
- Approximate $s(n)$ by a k^λ-periodic function $f(n)$.
- $s(n)$ and $f(n)$ agree on most residue classes modulo k^λ.
- Problem: P can hit the "bad" residue classes very often. (Trivial example: $P(x) = k^\lambda x + r$.)

Proof of Theorem 2

Naive approach

- Let f be a m-periodic function. Then $p_f^{\leq d}(L) \leq m^{d+1}$.
- Approximate $s(n)$ by a k^λ-periodic function $f(n)$.
- $s(n)$ and $f(n)$ agree on most residue classes modulo k^λ.
- Problem: P can hit the "bad" residue classes very often. (Trivial example: $P(x) = k^\lambda x + r$.)

Proof of Theorem 2

Naive approach

- Let f be a m-periodic function. Then $p_f^{\leq d}(L) \leq m^{d+1}$.
- Approximate $s(n)$ by a k^λ-periodic function $f(n)$.
- $s(n)$ and $f(n)$ agree on most residue classes modulo k^λ.
- Problem: P can hit the "bad" residue classes very often. (Trivial example: $P(x) = k^\lambda x + r$.)

Reduction

We study $(s(P(0)), s(P(1)), \ldots, s(P(L-1)))$.

- Avoid trivial problems:
  $P(\ell) = k^{\lambda_0}\left(z'_d\ell^d + \ldots + z'_1\ell + z'_0\right) + r$.
  Using the kernel: $\exists b_i \in Ker_k(a)$ with $b_i(n) = s(nk^{\lambda_0} + r)$.
  $s(P(\ell)) = b_i(z'_d\ell^d + \ldots + z'_1\ell + z'_0) = b_i(P'(\ell))$.
- Remains to study $b_i(P'(\ell))$ where some z'_i $(i \geq 1)$ is not divisible by k.

Reduction

We study $(s(P(0)), s(P(1)), \ldots, s(P(L-1)))$.

- Avoid trivial problems:
  $P(\ell) = k^{\lambda_0}\left(z'_d\ell^d + \ldots + z'_1\ell + z'_0\right) + r$.
  Using the kernel: $\exists b_i \in Ker_k(a)$ with $b_i(n) = s(nk^{\lambda_0} + r)$.
  $s(P(\ell)) = b_i(z'_d\ell^d + \ldots + z'_1\ell + z'_0) = b_i(P'(\ell))$.
- Remains to study $b_i(P'(\ell))$ where some z'_i $(i \geq 1)$ is not divisible by k.

Reduction

We study $(s(P(0)), s(P(1)), \ldots, s(P(L-1)))$.

- Avoid trivial problems:
  $P(\ell) = k^{\lambda_0}\left(z'_d\ell^d + \ldots + z'_1\ell + z'_0\right) + r$.
  Using the kernel: $\exists b_i \in Ker_k(a)$ with $b_i(n) = s(nk^{\lambda_0} + r)$.
  $s(P(\ell)) = b_i(z'_d\ell^d + \ldots + z'_1\ell + z'_0) = b_i(P'(\ell))$.
- Remains to study $b_i(P'(\ell))$ where some z'_i $(i \geq 1)$ is not divisible by k.

Reduction

We study $(s(P(0)), s(P(1)), \ldots, s(P(L-1)))$.

- Avoid trivial problems:
  $P(\ell) = k^{\lambda_0}\left(z'_d\ell^d + \ldots + z'_1\ell + z'_0\right) + r$.
  Using the kernel: $\exists b_i \in Ker_k(a)$ with $b_i(n) = s(nk^{\lambda_0} + r)$.
  $s(P(\ell)) = b_i(z'_d\ell^d + \ldots + z'_1\ell + z'_0) = b_i(P'(\ell))$.
- Remains to study $b_i(P'(\ell))$ where some z'_i $(i \geq 1)$ is not divisible by k.

New Approach

- Problem: We still only hit few residue classes modulo k^λ. (E.g. $P'(\ell) = 5 \cdot 3^\lambda \cdot \ell \bmod 6^\lambda$.)
- "low" digits: $P'(\ell)$ might still not equidistribute mod k^λ.
- "high" digits work: $\exists \varepsilon(k) > 0$ such that for any $\mathbf{w} \in \mathcal{A}^{\varepsilon\lambda}$ we have $\#\{\ell < k^\lambda : (P'(\ell) \bmod k^\lambda)_k \text{ starts with } \mathbf{w}\} \approx k^{\lambda(1-\varepsilon)}$.

New Approach

- Problem: We still only hit few residue classes modulo k^λ. (E.g. $P'(\ell) = 5 \cdot 3^\lambda \cdot \ell \bmod 6^\lambda$.)
- "low" digits: $P'(\ell)$ might still not equidistribute mod k^λ.
- "high" digits work: $\exists \varepsilon(k) > 0$ such that for any $\mathbf{w} \in \mathcal{A}^{\varepsilon\lambda}$ we have $\#\{\ell < k^\lambda : (P'(\ell) \bmod k^\lambda)_k \text{ starts with } \mathbf{w}\} \approx k^{\lambda(1-\varepsilon)}$.

New Approach

- Problem: We still only hit few residue classes modulo k^λ. (E.g. $P'(\ell) = 5 \cdot 3^\lambda \cdot \ell \bmod 6^\lambda$.)
- "low" digits: $P'(\ell)$ might still not equidistribute mod k^λ.
- "high" digits work: $\exists \varepsilon(k) > 0$ such that for any $\mathbf{w} \in \mathcal{A}^{\varepsilon\lambda}$ we have $\#\{\ell < k^\lambda : (P'(\ell) \bmod k^\lambda)_k \text{ starts with } \mathbf{w}\} \approx k^{\lambda(1-\varepsilon)}$.

Equidistribution of high digits

- Detection of digits: The digits of ℓ in base k between positions μ and λ coincide with the digits of $m < k^{\lambda-\mu}$ iff

$$\left\{\frac{\ell}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right).$$

- Expand the indicator function into a Fourier series.

$$\sum_{\ell<k^\lambda} \mathbf{1}_{\left[\left\{\frac{P'(\ell)}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right)\right]} \approx \sum_{|h|<H} c_h \sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right).$$

- Use classical estimates for

$$\sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right),$$

depending on $\min_{1\le j\le d} \gcd(z_j', k^\lambda)$.

Equidistribution of high digits

- Detection of digits: The digits of ℓ in base k between positions μ and λ coincide with the digits of $m < k^{\lambda-\mu}$ iff

$$\left\{\frac{\ell}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right).$$

- Expand the indicator function into a Fourier series.

$$\sum_{\ell<k^\lambda} \mathbf{1}_{\left[\left\{\frac{P'(\ell)}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right)\right]} \approx \sum_{|h|<H} c_h \sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right).$$

- Use classical estimates for

$$\sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right),$$

depending on $\min_{1\le j\le d} \gcd(z_j', k^\lambda)$.

Equidistribution of high digits

- Detection of digits: The digits of ℓ in base k between positions μ and λ coincide with the digits of $m < k^{\lambda-\mu}$ iff

$$\left\{\frac{\ell}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right).$$

- Expand the indicator function into a Fourier series.

$$\sum_{\ell<k^\lambda} \mathbf{1}_{\left[\left\{\frac{P'(\ell)}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right)\right]} \approx \sum_{|h|<H} c_h \sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right).$$

- Use classical estimates for

$$\sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right),$$

depending on $\min_{1\le j\le d} \gcd(z_j', k^\lambda)$.

Equidistribution of high digits

- Detection of digits: The digits of ℓ in base k between positions μ and λ coincide with the digits of $m < k^{\lambda-\mu}$ iff

$$\left\{\frac{\ell}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right).$$

- Expand the indicator function into a Fourier series.

$$\sum_{\ell<k^\lambda} \mathbf{1}_{\left[\left\{\frac{P'(\ell)}{k^\lambda}\right\} \in \left[\frac{m}{k^{\lambda-\mu}}, \frac{m+1}{k^{\lambda-\mu}}\right)\right]} \approx \sum_{|h|<H} c_h \sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right).$$

- Use classical estimates for

$$\sum_{\ell<k^\lambda} \mathrm{e}\left(\frac{h \cdot P'(\ell)}{k^\lambda}\right),$$

depending on $\min_{1\le j\le d} \gcd(z_j', k^\lambda)$.

Putting everything together

- Approximate $b_i(n)$ with a k^λ-periodic function $f(n)$.
- $b_i(P'(\ell)) \neq f(P'(\ell))$ only when $P'(\ell) \bmod k^\lambda$ is not synchronizing.
- This happens rarely ($\ll Lk^{-\varepsilon\lambda}$).
- $p_f^{\leq d}(L) \leq \left(k^\lambda\right)^{d+1}$.
- (Optional: optimize λ as a function of L.)

Putting everything together

- Approximate $b_i(n)$ with a k^λ-periodic function $f(n)$.
- $b_i(P'(\ell)) \neq f(P'(\ell))$ only when $P'(\ell) \bmod k^\lambda$ is not synchronizing.
- This happens rarely ($\ll Lk^{-\varepsilon\lambda}$).
- $p_f^{\leq d}(L) \leq \left(k^\lambda\right)^{d+1}$.
- (Optional: optimize λ as a function of L.)

Putting everything together

- Approximate $b_i(n)$ with a k^λ-periodic function $f(n)$.
- $b_i(P'(\ell)) \neq f(P'(\ell))$ only when $P'(\ell) \bmod k^\lambda$ is not synchronizing.
- This happens rarely ($\ll Lk^{-\varepsilon\lambda}$).
- $p_f^{\leq d}(L) \leq \left(k^\lambda\right)^{d+1}$.
- (Optional: optimize λ as a function of L.)

Putting everything together

- Approximate $b_i(n)$ with a k^λ-periodic function $f(n)$.
- $b_i(P'(\ell)) \neq f(P'(\ell))$ only when $P'(\ell) \bmod k^\lambda$ is not synchronizing.
- This happens rarely ($\ll Lk^{-\varepsilon\lambda}$).
- $p_f^{\leq d}(L) \leq \left(k^\lambda\right)^{d+1}$.
- (Optional: optimize λ as a function of L.)

Putting everything together

- Approximate $b_i(n)$ with a k^λ-periodic function $f(n)$.
- $b_i(P'(\ell)) \neq f(P'(\ell))$ only when $P'(\ell) \bmod k^\lambda$ is not synchronizing.
- This happens rarely ($\ll Lk^{-\varepsilon\lambda}$).
- $p_f^{\leq d}(L) \leq \left(k^\lambda\right)^{d+1}$.
- (Optional: optimize λ as a function of L.)

Open questions

- What is the minimal (non-trivial) growth rate of $p_u^{\leq d}(L)$ for automatic sequences/general sequences?
- Period-doubling sequence seems to satisfy $p_u^{\leq 2}(L) \ll L^2$.
- What is the maximal growth rate of $p_s^{\leq 1}(L)$ for synchronizing automatic sequences? (It is probably super-polynomial.)

Thank you for your attention!

Open questions

- What is the minimal (non-trivial) growth rate of $p_u^{\leq d}(L)$ for automatic sequences/general sequences?
- Period-doubling sequence seems to satisfy $p_u^{\leq 2}(L) \ll L^2$.
- What is the maximal growth rate of $p_s^{\leq 1}(L)$ for synchronizing automatic sequences? (It is probably super-polynomial.)

Thank you for your attention!

Open questions

- What is the minimal (non-trivial) growth rate of $p_u^{\leq d}(L)$ for automatic sequences/general sequences?
- Period-doubling sequence seems to satisfy $p_u^{\leq 2}(L) \ll L^2$.
- What is the maximal growth rate of $p_s^{\leq 1}(L)$ for synchronizing automatic sequences? (It is probably super-polynomial.)

Thank you for your attention!

Open questions

- What is the minimal (non-trivial) growth rate of $p_u^{\leq d}(L)$ for automatic sequences/general sequences?
- Period-doubling sequence seems to satisfy $p_u^{\leq 2}(L) \ll L^2$.
- What is the maximal growth rate of $p_s^{\leq 1}(L)$ for synchronizing automatic sequences? (It is probably super-polynomial.)

Thank you for your attention!