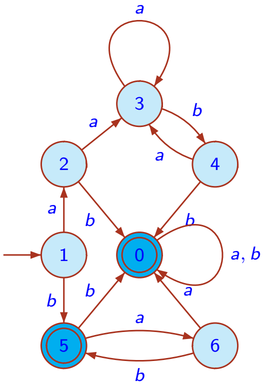
Minimization of Automata: Hopcroft's Algorithm revisited

Jean Berstel, Luc Boasson, Olivier Carton

Institut Gaspard-Monge, Université Paris-Est
Liafa, Université Paris VII

June 8, 2009

Outline



Each state q defines a language $L_q = \{w \mid q \cdot w \text{ is final}\}$.

The automaton is minimal if all languages L_q are distinct.

Here $L_2 = L_4$. States 2 and 4 are (Nerode) equivalent.

The Nerode equivalence is the coarsest partition that is compatible with the next-state function.

Refinement algorithm

Starts with the partition into two classes 05 and 12346.
Tries to refine by splitting classes which are not compatible with the next-state function.
A first refinement: $12346 \rightarrow 1234|6$ because $6 \cdot a$ is final.
A second refinement: $05 \rightarrow 0|5$ because of $0 \cdot a$ is final.

Moore equivalence

The Moore equivalence of order h is the equivalence $\sim_h$ defined for $h \geq 0$ by

$$p \sim_h q \iff L_p^{(h)}(\mathcal{A}) = L_q^{(h)}(\mathcal{A}), \quad \text{with} \quad L_p^{(h)}(\mathcal{A}) = \{w \in A^* \mid |w| \leq h,\ q \cdot w \in F\}.$$

Computation rule

For two states p, q and $h \geq 0$

$$p \sim_{h+1} q \iff p \sim_h q \quad \text{and} \quad p \cdot a \sim_h q \cdot a \ \text{ for all } a \in A.$$

Depth

- The depth of a finite automaton $\mathcal{A}$ is the smallest h such that the Moore equivalence $\sim_h$ equals the Nerode equivalence $\sim$.
- The depth is the smallest h such that $\sim_h$ equals $\sim_{h+1}$.
- It is at most $n-2$, where n is the number of states of $\mathcal{A}$.

Moore's algorithm

$\mathcal{P} \leftarrow \{F, F^c\}$ ▷ the initial equivalence $\sim_0$
repeat
  $\mathcal{Q} \leftarrow \mathcal{P}$ ▷ $\mathcal{Q}$ is the old partition, $\mathcal{P}$ is the new one
  for all $a \in A$ **do**
    $\mathcal{P}_a \leftarrow a^{-1}\mathcal{P}$ ▷ action of the letter a
  $\mathcal{P} \leftarrow \mathcal{P} \wedge \bigwedge_{a \in A} \mathcal{P}_a$ ▷ the new partition
until $\mathcal{P} = \mathcal{Q}$

Remarks

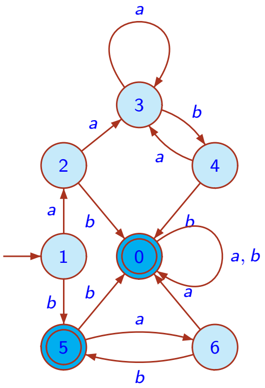
- $a^{-1}\mathcal{P}$ is the partition (equivalence) defined by

$$p \equiv q \mod (a^{-1}\mathcal{P}) \iff p \cdot a \equiv q \cdot a \mod \mathcal{P}$$

- If $\mathcal{P}$ is the partition (equivalence) $\sim_h$, then $\mathcal{P}' = \mathcal{P} \wedge \bigwedge_{a \in A} \mathcal{P}_a$ is $\sim_{h+1}$.
- The computation of $\mathcal{P}' = \mathcal{P} \wedge \bigwedge_{a \in A} \mathcal{P}_a$ can be done in time $O(n \,\mathrm{Card}\, A)$ for an automaton with n states, by a bucket sort.

Proposition

The complexity of Moore's algorithm on an n-state automaton $\mathcal{A}$ is $O(dn)$, where d is the depth of $\mathcal{A}$.



	0	1	2	3	4	5	6
a	0	2	3	3	3	6	5
b	0	5	0	4	0	0	0
$\mathcal{P} = \sim_0$	red	yellow	yellow	yellow	yellow	red	yellow
$a^{-1}\mathcal{P}$	red	yellow	yellow	yellow	yellow	yellow	red
$b^{-1}\mathcal{P}$	red	red	red	yellow	red	red	red
$\mathcal{P}' = \sim_1$	blue	red	red	yellow	red	cyan	green
$a^{-1}\mathcal{P}'$	blue	red	yellow	yellow	yellow	green	cyan
$b^{-1}\mathcal{P}'$	blue	cyan	blue	red	blue	blue	blue
$\sim_2$	blue	black	red	yellow	red	cyan	green

Average complexity

The alphabet is fixed, and the automata are accessible, deterministic and complete.

Theorem (Bassino, David, Nicaud)

For the uniform distribution over the automata of size n, the average complexity of Moore's algorithm is $O(n \log n)$.

A semi-automaton is an automaton with the final states not specified. Thus, an automaton is a pair $(\mathcal{T}, F)$, where F is the set of final states.

Proposition

For any semi-automaton $\mathcal{T}$, the average depth of Moore's algorithm on $(\mathcal{T}, F)$, for the uniform distribution over the sets F of final states, is $O(\log n)$.

- Denote by $\mathcal{F}^{\geq \ell}$ the set of set of states F such that the depth $d(\mathcal{T}, F)$ of Moore's algorithm on $(\mathcal{T}, F)$ is $\geq \ell$. The authors show that

$$\mathrm{Card}(\mathcal{F}^{\geq \ell}) \leq n^4 2^{n-\ell}\,.$$

- It follows that

$$\frac{1}{2^n} \sum_{F \in \mathcal{F}^{\geq \ell}} d(\mathcal{T}, F) \leq n^5 2^{-\ell} \quad \text{and} \quad \frac{1}{2^n} \sum_{F \in \mathcal{F}^{\leq \ell}} d(\mathcal{T}, F) \leq \ell\,.$$

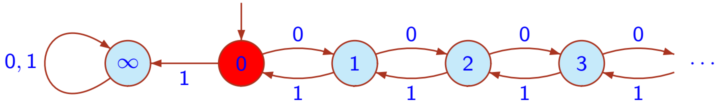
- The estimation is obtained by choosing $\ell = \lceil 5 \log n \rceil$.

Definition

- An infinite automaton is slow (for Moore) iff each Moore equivalence $\sim_h$ has $h+2$ classes.
- An finite automaton with n states is slow iff each Moore equivalence $\sim_h$, for $h \leq n-2$, has $h+2$ classes.

Example

The Dyck automaton is slow. The minimal automaton of the Dyck language is the following.



The Moore equivalences of this automaton

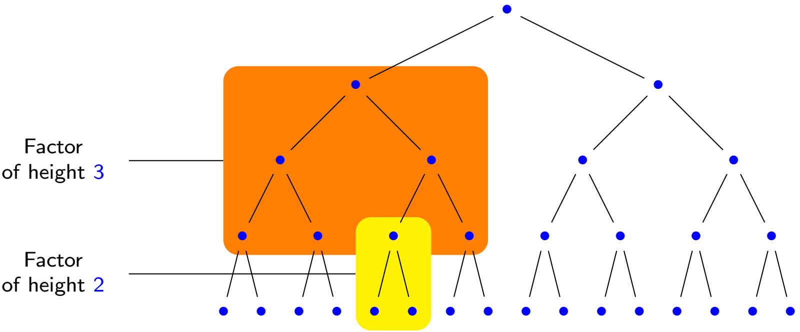
$\sim_0$: $0 \mid 1, 2, 3, 4, \ldots \infty$

$\sim_1$: $0 \mid 1 \mid 234 \ldots \infty$

$\sim_2$: $0 \mid 1 \mid 2 \mid 3, 4, \ldots \infty$

$\sim_3$: $0 \mid 1 \mid 2 \mid 3 \mid 4, \ldots \infty$

- We consider infinite binary trees t labeled with two colors.
- To each deterministic automaton $\mathcal{A}$ over two letters corresponds an execution tree t defined as follows
  - Each word labels a path in the tree
  - A node is colored red (black) if the state is accepting (not accepting)
- A factor of height h of a tree t is a subtree of height h that occurs in t.



Proposition (Carpi et al)

A complete tree t is rational if there is some integer h such that t has at most h distinct factors of height h.

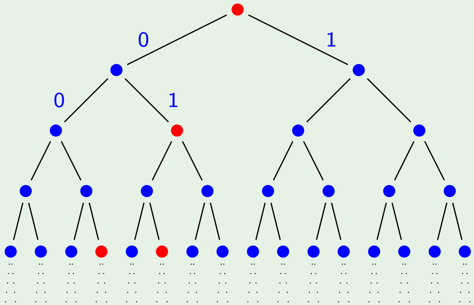
Definition

A tree is Sturmian if the number of its factors of height h is $h+1$ for each h.

Example (Dyck tree)

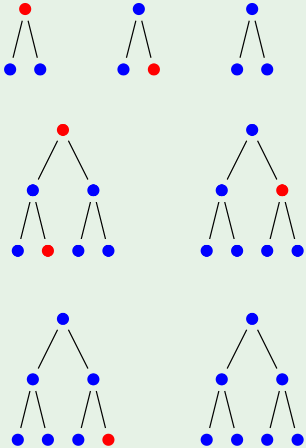
A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



$D_2^* = \{\varepsilon, 01, 0101, 0011, \ldots\}$

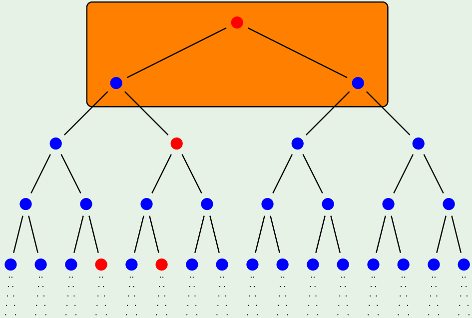
Its factors



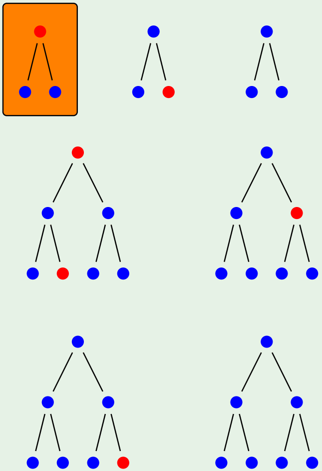
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



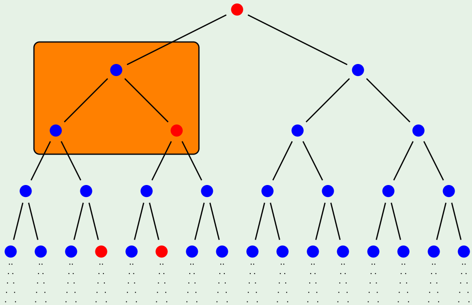
Its factors



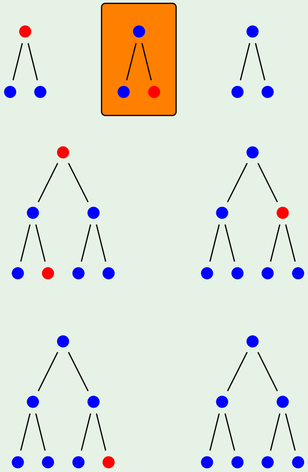
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



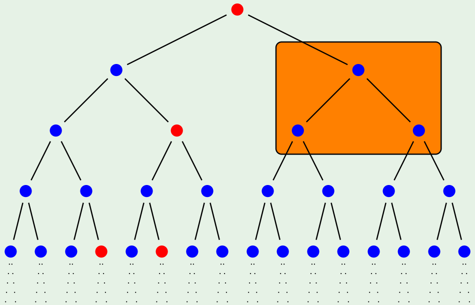
Its factors



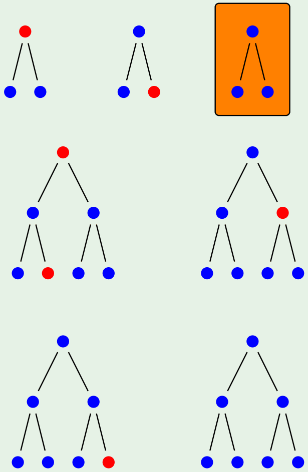
Example (Dyck tree)

A node is $\bullet$ if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



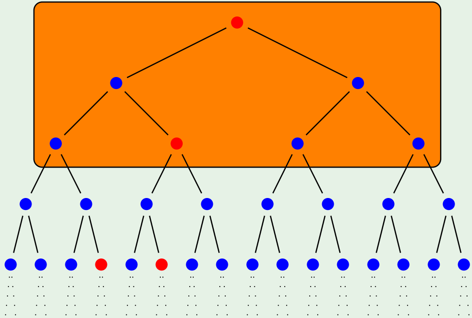
Its factors



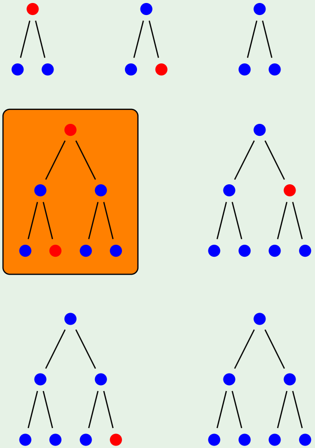
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



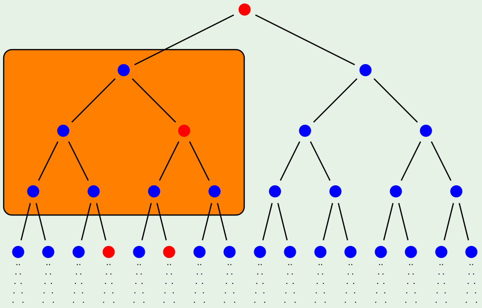
Its factors



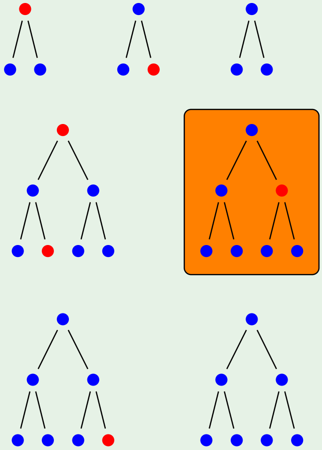
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



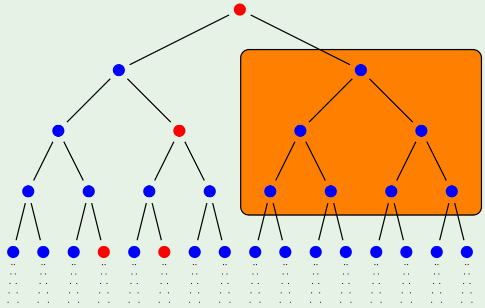
Its factors



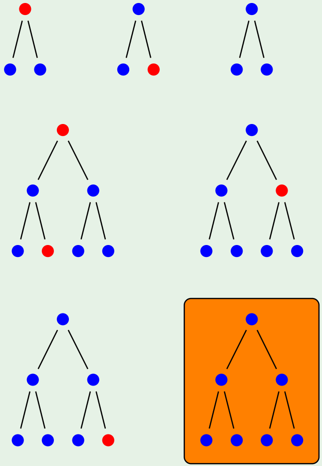
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



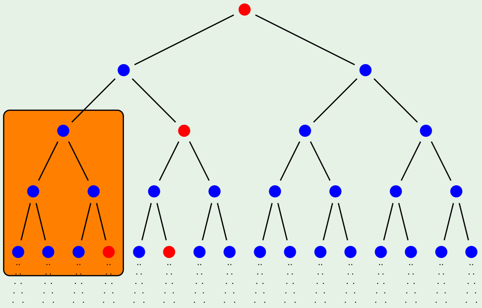
Its factors



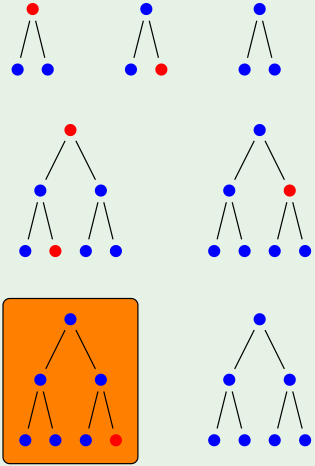
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



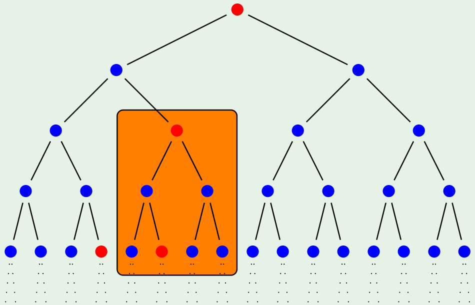
Its factors



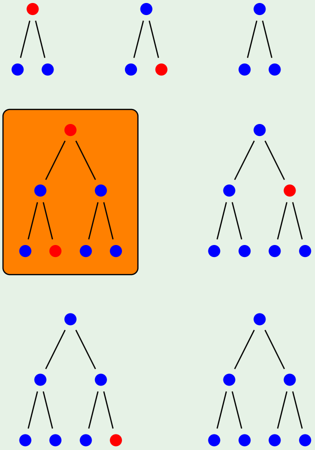
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



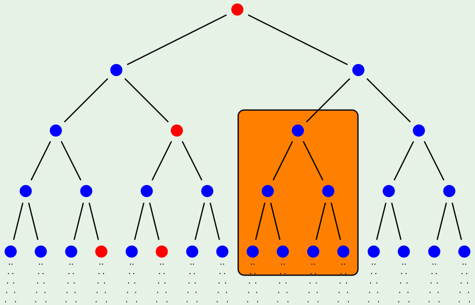
Its factors



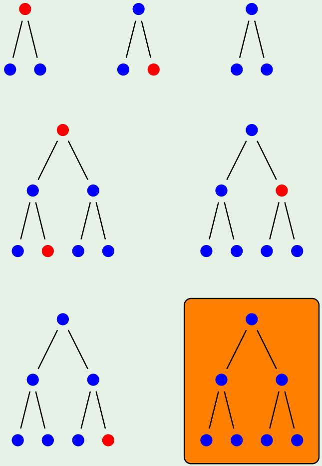
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



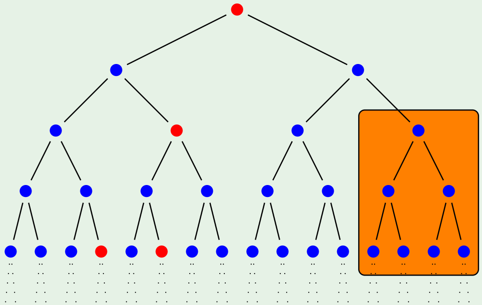
Its factors



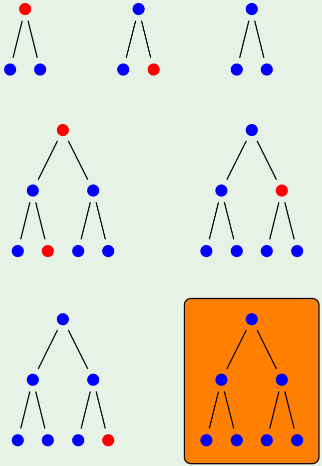
Example (Dyck tree)

A node is • if it is a Dyck word over the alphabet $\{0, 1\}$.

The Dyck tree



Its factors



Recall that an infinite automaton is slow iff each equivalence $\sim_h$ has $h+2$ classes.

Proposition

A tree t is Sturmian iff the minimal automaton $\mathcal{A}$ accepting the language of red (black) words is slow.

Indeed, a factor of height h in the tree describes the set $L_q^{(h)}(\mathcal{A})$ of words of length at most h accepted by $\mathcal{A}$ when starting in state q.

History

- Hopcroft has developed in 1970 a minimization algorithm that runs in time $O(n \log n)$ on an n state automaton (discarding the alphabet).
- No faster algorithm is known for general automata.

Question

- Question: is the time estimation sharp ?
- A first answer, by Berstel and Carton: there exist automata where you need $\Omega(n \log n)$ steps if you are "unlucky". These are related to De Bruijn words.
- A better answer, by Castiglione, Restivo and Sciortino: there exist automata where you need always $\Omega(n \log n)$ steps. These are related to Fibonacci words.
- Here: the same holds for all Sturmian words whose directive sequence have bounded geometric means.

$\mathcal{A} = (Q, i, F)$ automaton on the alphabet A. Let $\mathcal{P}$ be a partition of Q.

Definition

A splitter is a pair (P, a), with $P \in \mathcal{P}$ and $a \in A$.

The aim of a splitter is to split another class of $\mathcal{P}$.

Definition

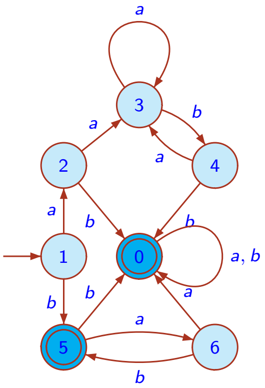
The splitter (P, a) splits the class $R \in \mathcal{P}$ if

$$\emptyset \subsetneq P \cap R \cdot a \subsetneq R \cdot a \text{ or equivalently if } \emptyset \subsetneq a^{-1}P \cap R \subsetneq R\,.$$

Here $a^{-1}P = \{q \mid q \cdot a \in P\}$.

Notation

In any case, we denote by $(P, a)|R$ the partition of R composed of the nonempty sets among $a^{-1}P \cap R$ and $R \setminus a^{-1}P$. The splitter (P, a) splits R if $(P, a)|R \neq \{R\}$.



- Partition $\mathcal{P} = 05 \mid 12346$.
- Splitter $(05, a)$. One has $a^{-1}05 = 06$.
- The splitter splits both 05 and 12346. (This is also seen by $05 \cap 05 \cdot a = 05 \cap 06 \neq 06$ and $05 \cap 12346 \cdot a = 05 \cap 0234 \neq 0234$)
- One gets

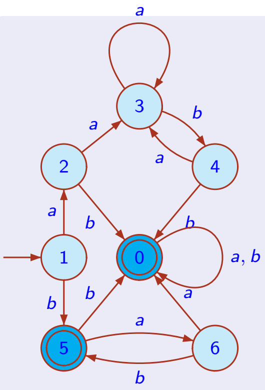
$$(05, a)|05 = 0 \mid 5 \quad \text{and} \quad (05, a)|12346 = 1234 \mid 6$$

Notation

$\mathcal{P}$ is the current partition. $\mathcal{W}$ is the waiting set.

Hopcroft's algorithm

$\mathcal{P} \leftarrow \{F, F^c\}$ ▷ The initial partition
for all $a \in A$ **do**
    $\text{ADD}((\min(F, F^c), a), \mathcal{W})$ ▷ The initial waiting set
while $\mathcal{W} \neq \emptyset$ **do**
    $(W, a) \leftarrow \text{TAKESOME}(\mathcal{W})$ ▷ takes some splitter in $\mathcal{W}$ and remove it
    for each $P \in \mathcal{P}$ which is split by (W, a) **do**
        $P', P'' \leftarrow (W, a)|P$ ▷ Compute the split
        REPLACE P by P' and P'' in $\mathcal{P}$ ▷ Refine the partition
        for all $b \in A$ **do** ▷ Update the waiting set
            if $(P, b) \in \mathcal{W}$ **then**
                REPLACE (P, b) by (P', b) and (P'', b) in $\mathcal{W}$
            else
                $\text{ADD}((\min(P', P''), b), \mathcal{W})$

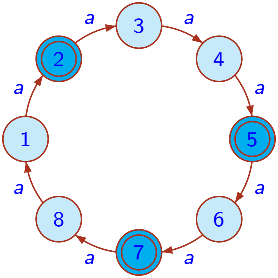


Initiale partition $\mathcal{P}$:	$05 \mid 12346$
Waiting set $\mathcal{W}$:	$(05, a), (05, b)$
Splitter chosen :	$(05, a)$
Split states :	$a^{-1}05 = 06$
First class to split:	$12346 \to 1234 \mid 6$
Splitters to add :	$(6, a)$ and $(6, b)$
Second class to split :	$05 \to 0 \mid 5$
Splitter to add:	$(5, a)$ (or $(0, a)$)
Splitter to replace:	$(05, b)$: by $(0, b)$ and $(5, b)$
New partition $\mathcal{P}$:	$0 \mid 1234 \mid 5 \mid 6$
New waiting set $\mathcal{W}$:	$(0, b), (6, a), (6, b), (5, a), (5, b)$

Basic fact

Splitting all sets of the current partition by one splitter (C, a) has a total cost of $\mathrm{Card}(a^{-1}C)$.

Cyclic automaton $\mathcal{A}_w$ for $w = 01001010$.



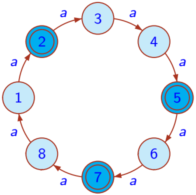
- States: $Q = \{1, 2, \ldots, |w|\}$
- One letter alphabet: $A = \{a\}$
- Transitions: $\{k \xrightarrow{a} k+1 \mid k < |w|\} \cup \{|w| \xrightarrow{a} 1\}$
- Final states: $F = \{k \mid w_k = 1\}$

Notation

Q_u is the set if starting positions of the occurrences of u in w.

Example

$Q_{010} = 146$



Initiale partition $\mathcal{P}$:	$Q_0 = 13468$, $Q_1 = 257$
Waiting set $\mathcal{W}$:	$Q_1 = 257$
States in $a^{-1}Q_1$:	146
Class to split:	$13468 \to Q_{01} = 146$, $Q_{00} = 38$
New waiting set $\mathcal{W}$:	Q_{00}
New partition $\mathcal{P}$:	$Q_{00} = 38$, $Q_{01} = 146$, $Q_1 = Q_{10} = 257$
States in inverse of Q_{00}:	27
Class to split:	$257 \to Q_{100} = 27$, $Q_{101} = 5$
New waiting set $\mathcal{W}$:	Q_{101}
New partition $\mathcal{P}$:	$Q_{001} = 38$, $Q_{010} = 146$, $Q_{100} = 27$, $Q_{101} = 5$

Definition and examples

- directive sequence $d = (d_1, d_2, d_3, \dots)$ sequence of positive integers
- standard words s_n of binary words defined by $s_0 = 1, s_1 = 0$ and
$$s_{n+1} = s_n^{d_n} s_{n-1} \quad (n \geq 1)\,.$$
- For $d = (\overline{1})$, one gets the Fibonacci words.
- For $d = (\overline{2,3})$, one gets $s_0 = 1, s_1 = 0, s_2 = 001, s_3 = 0010010010, \dots$

Proposition

A standard word is primitive. If $u01$ is a standard word, then u is a palindrome, $u10$ is standard and $u01$ and $u10$ are conjugate words.

Proposition

The standard words with directive sequence $d = (d_1, d_2, d_3, \dots)$ converge to the infinite characteristic Sturmian word with irrational slope $[0, d_1, d_2, d_3, \dots]$.

Proposition (Borel, Reutenauer)

A word w is standard if and only if it has exactly $i+1$ circular factors of length i, and exactly one circular special factor for each $i = 0, \ldots, |w| - 2$.

Theorem (Castiglione, Restivo, Sciortino)

Let w be a standard word.

- *Hopcroft's algorithm on the cyclic automaton $\mathcal{A}_w$ is uniquely determined.*
- *At each step i of the execution, the current partition is composed if the $i+1$ classes Q_u indexed by the circular factors of length i, and the waiting set is a singleton.*
- *This singleton is the smaller of the sets Q_{u0}, Q_{u1}, where u is the unique circular special factor of length $i-1$.*

Corollary

Let $(s_n)_{n\geq 0}$ be a standard sequence. Then the complexity of Hopcroft's algorithm on the automaton $\mathcal{A}_{s_n}$ is proportional to $\|s_n\|$, where $\|w\| = \sum_{u \in CF(w)} \min(|w|_{u0}, |w|_{u1})$.

Example

We compute $\|w\| = \sum_{u \in CF(w)} \min(|w|_{u0}, |w|_{u1})$ for $w = 01001010$.

u	$\|w\|_{u0}$	$\|w\|_{u1}$	min
ε	5	3	3
0	2	3	2
10	2	1	1
010	2	1	1
0010	1	1	1
10010	1	1	1
010010	1	1	1

So the number $\|w\|$ equals 10.

Notation

- Let $d = (d_1, d_2, d_3, \ldots)$ be a directive sequence.
- Let $(s_n)_{n\geq 0}$ be the sequence of standard words generated by d. and defined by

$$s_0 = 1\,, \quad s_1 = 0\,, \qquad s_{n+1} = s_n^{d_n} s_{n-1} \quad (n \geq 1)\,.$$

- Let $a_n = |s_n|_1$ be the number of letters 1 in the word s_n.
- Let c_n be the running time of Hopcroft's algorithm on the automaton $\mathcal{A}_{s_n}$.

Proposition

For any sequence d, one has $c_n = \Theta(na_n)$.

Theorem

One has $n = \Theta(\log a_n)$ and consequently $c_n = \Theta(a_n \log a_n)$ if and only if the sequence of geometric means $\left((d_1 d_2 \cdots d_n)^{1/n}\right)_{n\geq 1}$ of the directive sequence d is bounded.

Corollary

If the sequence d has bounded elements, then $c_n = \Theta(a_n \log a_n)$.

Corollary

There are directive sequences d such that $c_n = O(a_n \log \log a_n)$,

Indeed, if $d_n = 2^{2^n}$, then $a_n \geq 2^{2^n}$ and consequently $n \leq \log \log a_n$.

In fact, any running time close to a_n can be achieved by taking a rapidly growing directive sequence.

Notation

$d = (d_1, d_2, \ldots)$ directive sequence.
$(s_n)_{n \geq 0}$ standard sequence defined by d.
$a_n = |s_n|_1$.
c_n the complexity of Hopcroft's algorithm for s_n.

Definition

The generating series of length and cost are

$$A_d(x) = \sum_{n \geq 1} a_n x^n, \quad C_d(x) = \sum_{n \geq 0} c_n x^n.$$

$A_d(x) = \sum_{n \geq 1} a_n x^n$ generating series of lengths. $C_d(x) = \sum_{n \geq 0} c_n x^n$ generating series of costs.

Proposition

$$C_d(x) = A_d(x) + x^{\delta(d)} C_{\tau(d)}(x) + x^{1+\delta(T(d))} C_{\tau(T(d))}(x) \, .$$

Here

$$\tau(d) = \begin{cases} (d_1 - 1, d_2, d_3, \ldots) & \text{if } d_1 > 1 \\ (d_2, d_3, \ldots) & \text{otherwise} \, . \end{cases} \qquad \delta(d) = \begin{cases} 0 & \text{if } d_1 > 1 \, , \\ 1 & \text{otherwise}. \end{cases}$$

and

$$T(d) = \tau^{d_1}(d) = (d_2, d_3, \ldots).$$

Example

For $d = (1, 2, 3, 4, \ldots)$, one gets $\tau(d) = (2, 3, 4, \ldots)$ and $\delta(d) = 1$.

Proposition

$$C_d(x) = A_d(x) + x^{\delta(d)} C_{\tau(d)}(x) + x^{1+\delta(T(d))} C_{\tau(T(d))}(x)\,.$$

Example

For $d = (\overline{1})$ (Fibonacci), one has $\tau(d) = T(d) = d$, and $\delta(d) = 1$. The equation becomes

$$C_d(x) = A_d(x) + (x + x^2) C_d(x)\,,$$

from which we get

$$C_d(x) = \frac{A_d(x)}{1 - x - x^2}.$$

Next $a_{n+2} = a_{n+1} + a_n$ for $n \geq 0$, and since $a_0 = 1$ and $a_1 = 0$, one gets

$$A_d(x) = \frac{x^2}{1 - x - x^2}.$$

Thus

$$C_d(x) = \frac{x^2}{(1 - x - x^2)^2}\,.$$

This proves that $c_n \sim C n \varphi^n$, where φ is the golden ratio, and proves a theorem of Castiglione, Restivo and Sciortino.

Example ($d = (\overline{2,3})$)

$$\begin{aligned}
C_{(\overline{2,3})} &= A_{(\overline{2,3})} + C_{(1,\overline{3,2})} + xC_{(2,\overline{2,3})}\\
C_{(1,\overline{3,2})} &= A_{(1,\overline{3,2})} + xC_{(\overline{3,2})} + xC_{(2,\overline{2,3})}\\
C_{(2,\overline{2,3})} &= A_{(2,\overline{2,3})} + C_{(1,\overline{2,3})} + xC_{(1,\overline{3,2})}\\
C_{(\overline{3,2})} &= A_{(\overline{3,2})} + C_{(2,\overline{2,3})} + xC_{(1,\overline{3,2})}\\
C_{(1,\overline{2,3})} &= A_{(1,\overline{2,3})} + xC_{(\overline{2,3})} + xC_{(1,\overline{3,2})}
\end{aligned}$$

In this case, the system can be replaced by

$$C_{(\overline{2,3})} = A_{(\overline{2,3})} + D_1 + xD_2\,,$$

where D_1 and D_2 satisfy the equations

$$\begin{aligned}
D_1 &= A_{(\overline{2,3})} + xA_{(\overline{3,2})} + 2xD_2 + x^2 D_1\\
D_2 &= 2A_{(\overline{3,2})} + xA_{(\overline{2,3})} + 3xD_1 + x^2 D_2\,.
\end{aligned}$$

Definition

The continuant polynomials $K_n(x_1, \ldots, x_n)$, for $n \geq -1$ are a family of polynomials in the variables $x_1, \ldots, x_n$ defined by $K_{-1} = 0$, $K_0 = 1$ and, for $n \geq 1$, by

$$K_n(x_1, \ldots, x_n) = x_1 K_{n-1}(x_2, \ldots, x_n) + K_{n-2}(x_3, \ldots, x_n)\,.$$

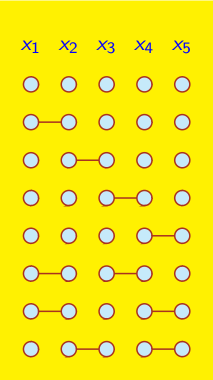
The first continuant polynomials are

$$\begin{aligned}
K_1(x_1) &= x_1 \\
K_2(x_1, x_2) &= x_1x_2 + 1 \\
K_3(x_1, x_2, x_3) &= x_1x_2x_3 + x_1 + x_3 \\
K_4(x_1, x_2, x_3, x_4) &= x_1x_2x_3x_4 + x_1x_2 + x_3x_4 + x_1x_4 + 1\,.
\end{aligned}$$

Combinatorial Interpretation

The Morse code or the “leapfrog” construction

$$K_5(x_1, x_2, x_3, x_4, x_5) = x_1x_2x_3x_4x_5 + x_3x_4x_5 + x_1x_4x_5 + x_1x_2x_5 + x_1x_2x_3 + x_5 + x_3 + x_1$$



$x_1x_2x_3x_4x_5$

$x_3x_4x_5$

$x_1x_4x_5$

$x_1x_2x_5$

$x_1x_2x_3$

x_5

x_3

x_1

Equivalent definitions

$$K_n(x_1,\ldots,x_n) = x_1 K_{n-1}(x_2,\ldots,x_n) + K_{n-2}(x_3,\ldots,x_n)\,,$$
$$K_n(x_1,\ldots,x_n) = K_{n-1}(x_1,\ldots,x_{n-1})x_n + K_{n-2}(x_1,\ldots,x_{n-2})$$

See Graham, Knuth, Patashnik, *Concrete Mathematics*, for other properties.

Let $d = (d_1, d_2, d_3, \ldots)$ be a sequence of positive numbers. The continued fraction defined by d is denoted $\alpha = [d_1, d_2, d_3, \ldots]$ and is defined by

$$\alpha = d_1 + \cfrac{1}{d_2 + \cfrac{1}{d_3 + \cdots}} .$$

The finite initial parts $[d_1, d_2 \ldots, d_n]$ of d define rational numbers

$$d_1 + \cfrac{1}{d_2 + \cfrac{1}{d_3 + \ddots + \cfrac{1}{d_n}}} = \frac{K_n(d_1, \ldots, d_n)}{K_{n-1}(d_2, \ldots, d_n)} .$$

One has

$$a_{n+2} = K_n(d_2, \ldots, d_{n+1}) \quad (n \geq -1)$$

and

$$A_d(x) = x^2 \sum_{n \geq 0} K_n(d_2, \ldots, d_{n+1}) x^n \,.$$

The series C_d also has an expression with continuants

$$C_d = x^2 \sum_{n \geq 0} (K_n(d_2, \ldots, d_{n+1}) + N_{n+1}(d_1, \ldots, d_{n+1}) + N_n(d_2, \ldots, d_{n+1})) x^n \,.$$

where

$$L_n(x_1, \ldots, x_n) = K_n(x_1, \ldots, x_n) - K_{n-1}(x_2, \ldots, x_n) \,.$$

$$N_n(x_1, \ldots, x_n) = \sum_{i=0}^{n-1} K_i(x_1, \ldots, x_i) L_{n-i}(x_{i+1}, \ldots, x_n) \,.$$

Lemma

Assume $d_2 > 1$, and let t_n be the sequence of standard words generated by $\tau T(d) = (d_2 - 1, d_3, d_4, \ldots)$. Let β be the morphism defined by

$$\beta(0) = 10^{d_1} \text{ and } \beta(1) = 10^{d_1+1}$$

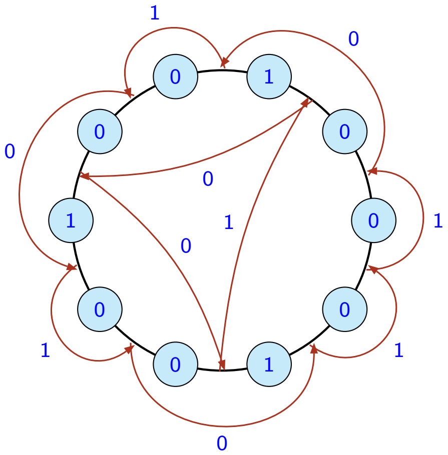
- *Then $s_{n+1}0^{d_1} = 0^{d_1}\beta(t_n)$ for $n \geq 1$.*
- *If v is a circular special factor of t_n, then $\beta(v)10^{d_1}$ is a circular special factor of s_{n+1}.*
- *Conversely, if w is a circular special factor of s_{n+1} starting with 1, then w has the form $w = \beta(v)10^{d_1}$ for some circular special factor v of t_n.*
- *Moreover, $|s_{n+1}|_{w0} = |t_n|_{v1}$ and $|s_{n+1}|_{w1} = |t_n|_{v0}$.*

Example ($d = (\overline{2,3})$, so $\beta(0) = 100$, $\beta(1) = 1000$)

$$\begin{aligned} t_0 &= 1 & s_0 &= 1 \\ t_1 &= 0 & s_1 &= 0 \\ t_2 &= 001 & s_2 &= 001 \\ t_3 &= (001)^2 0 & s_3 &= (001)^3 \end{aligned}$$

$$s_3 00 = 00.100.100.1000 = 00\beta(001) = 00\beta(t_2)$$

$$t_2 = \underline{001}, s_3 00 = 00\underline{1001001}000 = 001001\underline{001000}0$$



Factorization

- Every circular word containing a 0 and a 1 has two circular factorizations: cut before each 0 and cut before each 1.
- In the case of Sturmian words, the factors are

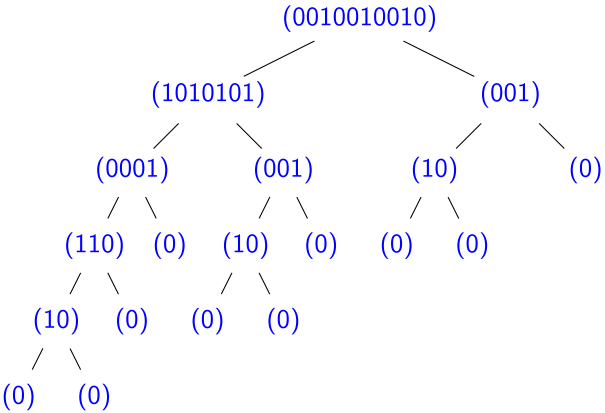
  0 and 01 and 10^p and 10^{p+1} or vice-versa.
- Moreover, the words obtained by decoding are again Sturmian!

Example

$$s = 0010010010 = 0|01|0|01|0|01|0 = 00|100|100|10 = \varphi(1010101) = \beta(001)$$

The words 1010101 and 001 are Sturmian.

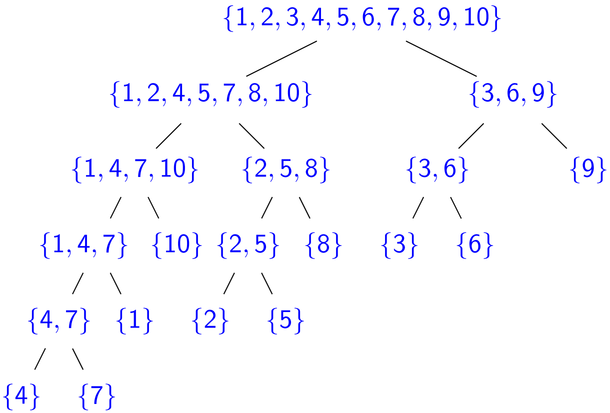
Reduction tree of Sturmian words (Castiglione, Restivo Sciortino)



Definition

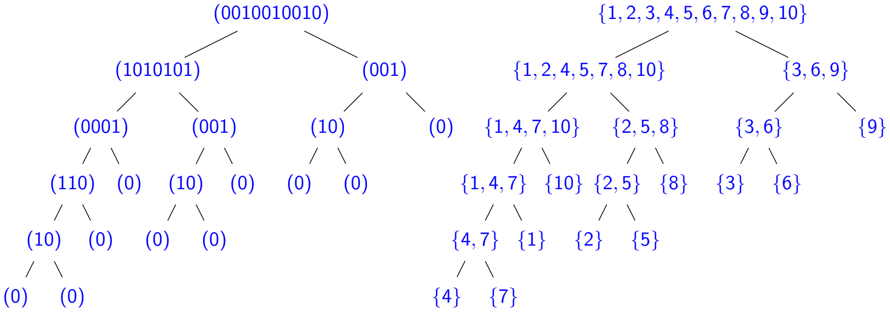
The reduction tree is the tree labeled with circular Sturmian words obtained by iterating the decoding.

Derivation tree of Sturmian words (Castiglione, Restivo Sciortino)



Definition

The derivation tree is the tree labeled with the classes of the partitions obtained by Hopcroft's algorithm.



Theorem (Castiglione, Restivo Sciortino)

The reduction tree and the derivation tree are isomorphic for circular Sturmian words.

Final remarks

Slow automata

An automaton $\mathcal{A}$ is slow for Hopcroft if, at each step of the algorithm,

- all splitters in the waiting set either do not split or split at most one class
- all splitters that split a class split the same class into the same two new classes.

Example

Whenever Hopcroft's algorithm is determined and a class is split into two new classes. This holds for cyclic automata defined by standard words, and also for a new class of automata defined by Castiglione, Restivo, Sciortino *On extremal cases of Hopcroft's algorithm*, CIAA2009.

Proposition

An automaton is slow for Moore if and only if it is slow for Hopcroft.

Although Hopcroft's algorithm seems to be a refinement of Moore's algorithm, one has:

There exist automata for which some partitions computed in Moore's algorithm are not obtained in any execution of the Hopcroft algorithm.