Approximate notion of automata: m-LZ-78-trie-automata

The *hemisphere of radius m around n* in a graph (G, E) is

$$hs_m(n) := \{k \in G \mid E^{\leq m}(n, k)\}.$$

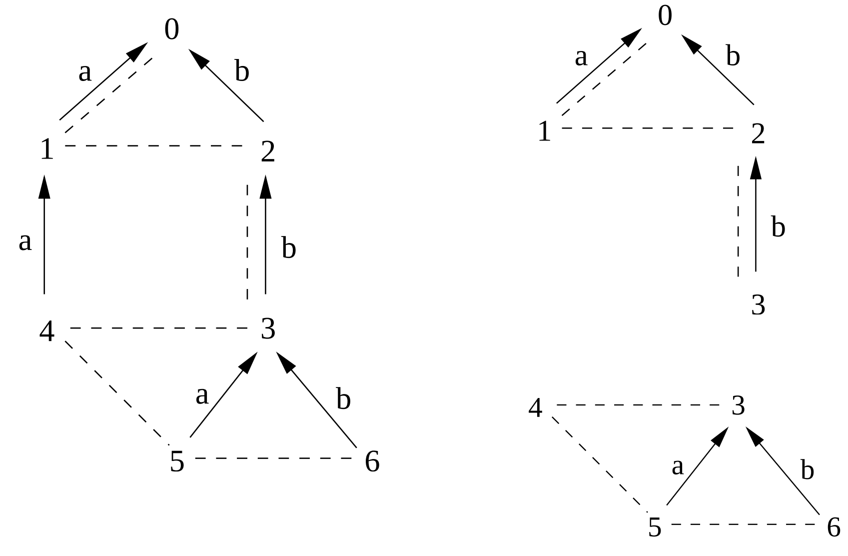
Def. 14 Let $\mathcal{T}$ be an LZ-78-trie and

$$E \quad := \quad \bigcup \{ \xleftarrow{a} \mid a \in \Sigma\} \cup \{Succ\}.$$

The *bottom-up (resp. top-down) LZ-78-hemisphere of radius m around $n \in T$* is

$$\begin{array}{lcll} bu\text{-}hs^{\mathcal{T}}_m(n) & := & \mathcal{T}\ hs_m(n) & \text{in } (T, E), \\ td\text{-}hs^{\mathcal{T}}_m(n) & := & \mathcal{T}\ hs_m(n) & \text{in } (T, \breve{E}). \end{array}$$

Expl. 15 (Cont.) The top-down resp. bottom-up 2-hemispheres of node 3 (with dashed edges for *Pred* resp. *Succ*):



m-LZ-78-trie-automata

Def. 16 A finite *bottom-up m-LZ-automaton* $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ over Σ consists of

- a finite set Q of *states*,
- sets $I, F \subseteq Q$ of *initial* and *final* states,
- a finite *transition relation* δ of pairs (P, q) (or: $P \to q$), where $q \in Q$ and P is a bottom-up *LZ*-78-hemisphere of radius m whose nodes except the root are labelled by elements of Q.

$\mathcal{A}$ is *deterministic* if $|I| = 1$ and $q = q'$ when $(P, q), (P, q') \in \delta$.

Then $r : T \to Q$ is an (*accepting*) *run* of $\mathcal{A}$ on *LZ*-78-trie $\mathcal{T}$ if

- $r(\max) \in I$ (and $r(0) \in F$),
- for each $n \in T$ there is some $(P, q) \in \delta$ such that $bu\text{-}hs_m^{\mathcal{T}}(n)$, expanded by the node-labelling r, is isomorphic to P with label q at its root,

The *class of LZ-78-tries accepted by* $\mathcal{A}$ is

$$L(\mathcal{A}) := \{\mathcal{T} \mid \mathcal{A} \text{ accepts } \mathcal{T}\}.$$

Similarly, define *top-down m-LZ-automata*.

Note: While $\mathcal{A}$ sequentially follows the enumeration of $\mathcal{T}_\alpha$, it can access some states reached at suffixes (resp. prefixes) of α.

Strictness of the automata hierarchy

The bottom-up resp. top-down hierarchies of m-LZ-automata are strict, and no level exhausts the MSO-properties of tries:

Thm. 2 For every m, there is an MSO-sentence defining a class of LZ-tries that is not accepted by any m-LZ-automaton.

Top-down LZ-trie-automata

G.Navarro (2003) has shown that finite automata on strings can be simulated on LZ-78-compressed strings. In fact:

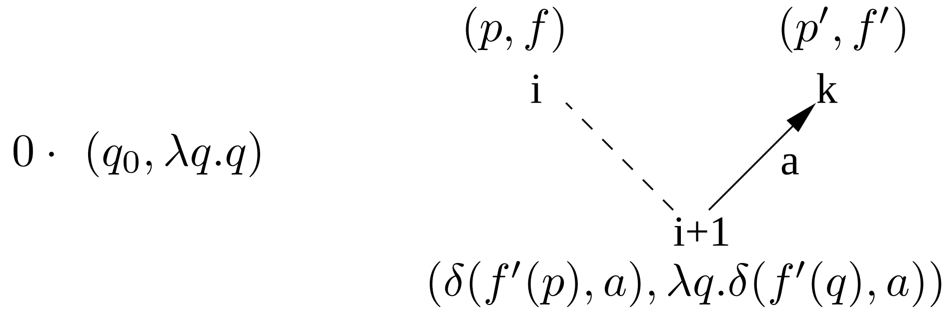
Thm. 3 The LZ-compression $\{\mathcal{T}_{LZ(w)} \mid w \in R\}$ of any regular set $R \subseteq \Sigma^+$ is accepted by a deterministic top-down 1-LZ-automaton.

Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a DFA accepting R. Define a det. top-down 1-LZ-automaton $\mathcal{A}' = (Q', \Sigma, \delta', q'_{in}, F')$ by

(i) $Q' := Q \times (Q \to Q)$,

(ii) $q'_{in} := (q_0, \lambda q.q)$, $F' := F \times (Q \to Q)$,

(iii) δ' according to the following transitions:



for each $a \in \Sigma$ (including the case $i = k$).

Suppose r' is a run of $\mathcal{A}'$ on $\mathcal{T}_{LZ(w)}$ where $w = B_0 \cdots B_{m-1}$ are the LZ-blocks. By induction we see that for each $k \leq m$,

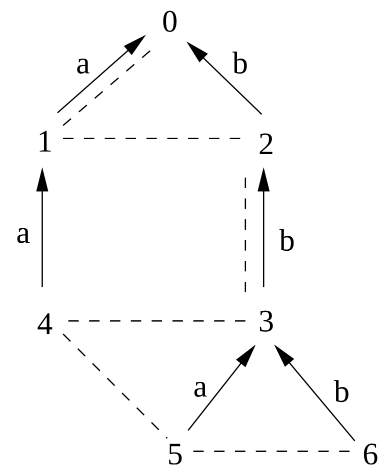
$$r'(k) \quad = \quad (\delta(q_0, B_0 \cdots B_{k-1}), \lambda p.\delta(p, B_{k-1})). \tag{4}$$

Hence $\mathcal{A}'$ accepts $\{\mathcal{T}_{LZ(w)} \mid w \in R\}$, because

$$w \in R \iff \delta(q_0, B_0 \cdots B_{m-1}) \in F \iff r'(m) \in F'.$$

Proof of (4)

Consider node 5 in our LZ-trie



Its top-down 1-hemisphere contains nodes 4 and 3. Recall that B_i labels the path from the root to node $i+1$. By induction,

$$\begin{array}{rclcccl} r'(4) & = & (& \delta(q_0, B_0 \cdots B_3) & , & \ldots &), \\ r'(3) & = & (& \ldots & , & \lambda q.\delta(q, B_2) &). \end{array}$$

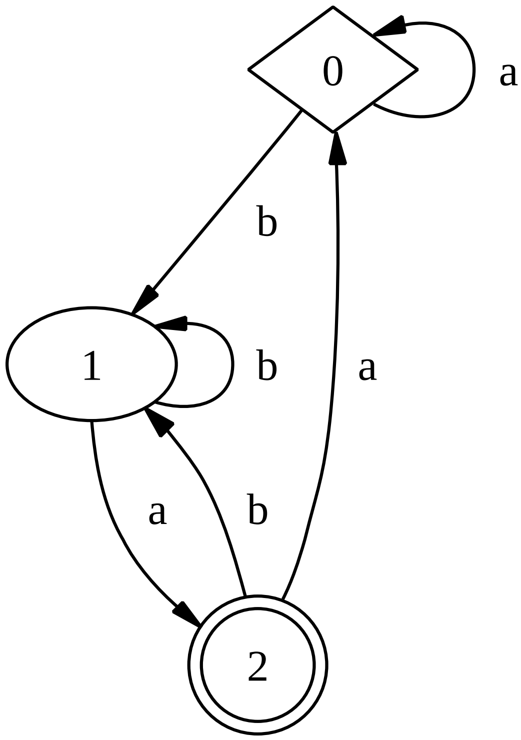
Note that B_4 labels the path to node 5, and $B_4 = B_2 a$, so

$$\begin{array}{rcl} r'_2(5) = \lambda q.\delta(q, B_4) & = & \lambda q.\delta(r'_2(3)(q), a), \\ r'_1(5) = \delta(q_0, B_0 \cdots B_4) & = & \delta(r'_1(4), B_4) = r'_2(5)(r'_1(4)) \end{array}$$

Beispiel: Suche ba in LZ-78($abbbaabbabbb$)

Methode:

(i) Konstruiere für $L = \Sigma^* ba$ einen DFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$:



Sei $\delta_c : Q \to Q$ die Funktion $\delta_c(q) := \delta(q, c)$, hier also:

$$\delta_a = \{(0,0), (1,2), (2,0)\}, \quad \delta_b = \{(0,1), (1,1), (2,1)\}.$$

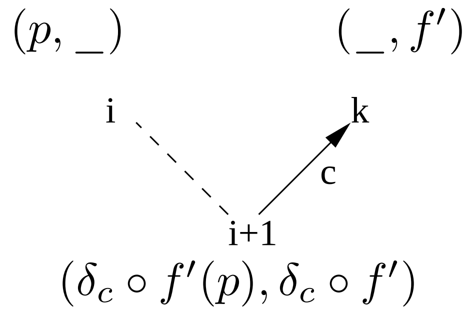
(ii) Bilde den $\mathcal{A}$ simulierenden top-down 1-LZ-78-DFA $\mathcal{A}' = (Q', \Sigma, \delta', q_0', F')$ mit

(a) $Q' := Q \times (Q \to Q)$,

(b) $q_0' := (q_0, \lambda q : Q.q)$, $F' := \{q_2\} \times (Q \to Q)$,

(c) und als δ' die top-down-1-Umgebungen, für $c \in \Sigma$:

$0 \cdot \ (q_0, \lambda q.q)$



(5)

Wegen $\delta_a = \{(0,0),(1,2),(2,0)\}$ ist

$$\delta_a \circ f' = \{(0,0),(1,0),(2,0)\} \tag{6}$$

außer für die Fälle von f' mit $1 \in \mathit{range}(f)$:

f'	$\delta_a \circ f'$
$(0,0),(1,0),(2,1)$	$(0,0),(1,0),(2,2)$
$(0,0),(1,1),(2,0)$	$(0,0),(1,2),(2,0)$
$(0,0),(1,1),(2,1)$	$(0,0),(1,2),(2,2)$
$(0,0),(1,1),(2,2)$	$(0,0),(1,2),(2,0)$
$(0,0),(1,2),(2,1)$	$(0,0),(1,0),(2,2)$
$(0,1),(1,0),(2,0)$	$(0,2),(1,0),(2,0)$
$(0,1),(1,0),(2,1)$	$(0,2),(1,0),(2,2)$
$(0,1),(1,0),(2,2)$	$(0,2),(1,0),(2,0)$
$(0,1),(1,1),(2,0)$	$(0,2),(1,2),(2,0)$
$(0,1),(1,1),(2,1)$	$(0,2),(1,2),(2,2)$
$(0,1),(1,1),(2,2)$	$(0,2),(1,2),(2,0)$
$(0,1),(1,2),(2,0)$	$(0,2),(1,0),(2,0)$
$(0,1),(1,2),(2,1)$	$(0,2),(1,0),(2,2)$
$(0,1),(1,2),(2,2)$	$(0,2),(1,0),(2,0)$

(7)

Da δ_b immer den Wert 1 liefert, ist für jedes f'

$$\delta_b \circ f' = \{(0,1),(1,1),(2,1)\} \qquad (8)$$

Durch Einsetzen dieser Funktionen in (5) sind die 1-Umgebungen für die Transition δ' bestimmt.

(iii) Lies mit $\mathcal{A}'$ die komprimierte Folge *LZ-78*(w).

$\mathcal{A}'$ bestimmt den Zustand bei $p_n = (k+1, a)$ mit a, dem Zustand bei p_{n-1} *und dem Zustand bei* p_k.

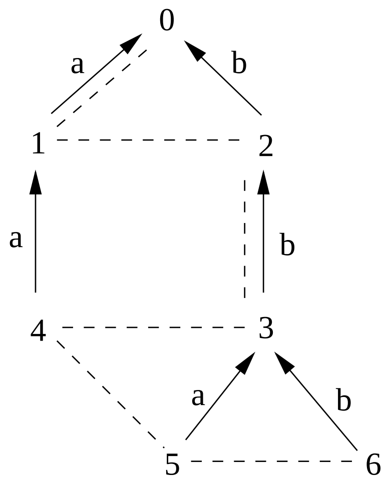
Suche von ba mit $\mathcal{A}'$ in LZ-78($abbbaabbabbb$)

Für $w = abbbaabbabbb = a.b.bb.aa.bba.bbb$ war

$$\begin{aligned} LZ\text{-}78(w) &= (0,a)(0,b)(2,b)(1,a)(3,a)(3,b), \\ &= (0,a,1)(0,b,2)(2,b,3)(1,a,4)(3,a,5)(3,b,6), \end{aligned}$$

oder als LZ-Trie

$\mathcal{T} =$



Ein Lauf $r : T \to Q'$ von $\mathcal{A}'$ auf $\mathcal{T}$ beginnt bei der Wurzel 0, die mit dem Zustand

$$r(0) = q_0' = (q_0, \lambda q.q) = (0, \lambda q.q)$$

markiert wird. Nach der Umgebung von Knoten 1 erhält er die Markierung

$$r(1) = (\delta_a \circ f'(p), \delta_a \circ f') = (\delta_a(0), \delta_a) = (0, \delta_a)$$

mit $r(0) = (p, \_)$ und $r(k) = (\_, f')$ für $k \xleftarrow{a} 1$

Analog geht es weiter:

- Da $2-1=1$ und $0 \xleftarrow{b} 2$ sind, erhält man aus

$$\begin{aligned} r(1) &= (p, \_) = (0, \_), \\ r(0) &= (\_, f') = (\_, \lambda q.q) \end{aligned}$$

$$r(2) = (\delta_b \circ f'(p), \delta_b \circ f') = (\delta_b(0), \delta_b) = (1, \delta_b)$$

- Da $3-1=2$ und $2 \xleftarrow{b} 3$ sind, erhält man aus

$$\begin{aligned} r(2) &= (p, \_) = (1, \_), \\ r(2) &= (\_, f') = (\_, \delta_b) \end{aligned}$$

$$r(3) = (\delta_b \circ f'(p), \delta_b \circ f') = (\delta_b(1), \delta_b) = (1, \delta_b)$$

- Da $4-1=3$ und $1 \xleftarrow{a} 4$ sind, erhält man aus

$$\begin{aligned} r(3) &= (p, \_) = (1, \_), \\ r(1) &= (\_, f') = (\_, \delta_a) \end{aligned}$$

$$r(4) = (\delta_a \circ f'(p), \delta_a \circ f') = (\delta_a^2(1), \delta_a^2) = (0, \lambda q.0)$$

- Da $5-1=4$ und $3 \xleftarrow{a} 5$ sind, erhält man aus

$$\begin{aligned} r(4) &= (p, \_) = (0, \_), \\ r(3) &= (\_, f') = (\_, \delta_b) \end{aligned}$$

  $$r(5) = (\delta_a \circ f'(p), \delta_a \circ f') = (\delta_a \circ \delta_b(1), \delta_a \circ \delta_b) = (2, \lambda q.2)$$

  Wegen $2 \in F$ ist das ein Treffer, d.h. wegen

  $$r(5) = (\delta(q_0, B_0 \cdots B_{5-1}), \_) = (2, \_)$$

  ist $B_0 \cdots B_4 = a.b.bb.aa.bba. \in \Sigma^* ba.$

- Da $6 - 1 = 5$ und $3 \xleftarrow{b} 6$ sind, erhält man aus

$$\begin{array}{rcl} r(5) & = & (p, \_) = (2, \_), \\ r(3) & = & (\_, f') = (\_, \delta_b) \end{array}$$

$$r(6) = (\delta_b \circ f'(p), \delta_b \circ f') = (\delta_b^2(1), \delta_b^2) = (1, \delta_b),$$

was kein Endzustand ist, also ist $w \notin \Sigma^* ba$.

Warum werden Vorkommen von *ba* übersehen?

Wir finden das erste Vorkommen von ba deshalb nicht, weil es nicht am Blockende auftrat, der Automat aber nur $\Sigma^* ba$ erkennt, also nur Endvorkommen von ba.

Der Automat zu $L' = \Sigma^* ba \Sigma^*$ würde alle Wörter erkennen, in denen ba vorkommt.

Aber: die Simulation ist zu grob, um die Vorkommen des Suchworts, die nicht am Blockende stehen, zu melden: die Funktionen $f' : Q \to Q$ in

$$r(k) = (\_, f') = (\_, \lambda q.\delta(q, B_{k-1}))$$

sind die δ-Übergänge beim Lesen ganzer Blöcke, sie melden nicht, wenn ein Endzustand beim buchstabenweisen Lesen der Blöcke erreicht würde.