Минимален суфиксен автомат

27.11.2020 г.

(Suffix Automaton)

Дадено: азбука Σ и текст $w \in \Sigma^*$.
Търси се: минимален краен (~~тотален~~) детерминиран автомат $\mathcal{A}_w = < \Sigma, Q_w, q_\varepsilon, \delta_w, F_w >$ с език $Suff(w) = \{\gamma \in \Sigma^* \mid \exists\, u \in \Sigma^* (u\gamma = w)\}$.

- не е задължително от всяко състояние да има $|\Sigma|$ прехода

Ако имаме такъв автомат $\mathcal{A}_w = < \Sigma, Q_w, q_\varepsilon, \delta_w, F_w >$ (от всяко състояние има път до F_w) $u \in Inf(w) = \{u' \mid \exists \alpha, \beta (\alpha u' \beta = w)\} \Leftrightarrow \delta^*(q_\varepsilon, u)$ е дефинирано!

Алгоритъм на Blumer et al., 1983(Ehrenfeucht, Haussler and Warmuth (1985), 1983-обобщение).

- on-line алгоритъм за $\mathcal{A}_w$ със сложност $O(|w| \log |\Sigma|)$ с брой състояния $\leq 2|w| + 1$, а броя на преходите $\leq 3|w| - 4$.

Дефиниция. За езика $\mathcal{L} \subseteq \Sigma^*$, релацията на Майхил-Нероуд се дефинира по следния начин: $u \equiv_{\mathcal{L}} v \overset{def}{\Leftrightarrow} \forall \gamma \in \Sigma^* (u\gamma \in \mathcal{L} \Leftrightarrow v\gamma \in \mathcal{L})$.

Известно е от теорията по дискретни структури, че $\mathcal{L}$ е регулярен точно тогава, когато гореописаната релация на еквивалентност има краен индекс. Ако знаем какви са класовете на еквивалентност на тази релация, тогава може да построим минимален краен детерминиран автомат със състояния съответните класове на еквивалентност.

Еквивалентна формулировка на релацията на Майхил-Нероуд:

За дума $u \in \Sigma^*$, $\mathcal{L} \subseteq \Sigma^*$: $\quad u^{-1}\mathcal{L} \overset{def}{=} \{\gamma \in \Sigma^* \mid u\gamma \in \mathcal{L}\}$.
Това което лесно се вижда е следното:
Забележка:
$u \equiv_{\mathcal{L}} v \Leftrightarrow u^{-1}\mathcal{L} = v^{-1}\mathcal{L}$

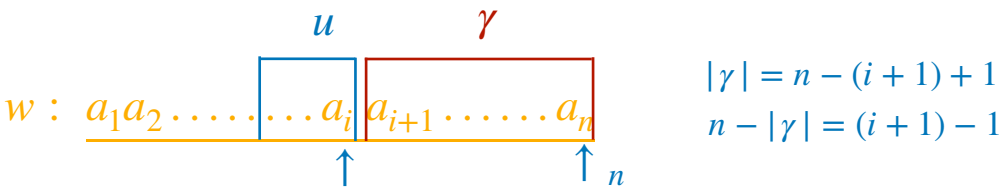
Забележка:
Ако $u \in \Sigma^*$, $a \in \Sigma$

Следствие: $u \equiv_{\mathcal{L}} v \Rightarrow \forall a \in \Sigma (ua \equiv_{\mathcal{L}} va)$
Естествено няма да има никакво значение дали сме взели буква $a \in \Sigma$ или цяла дума $a \in \Sigma$. Разсъжденията ще са аналогични.

Тоест посланието от теорията е: Намерете релацията на Майхил-Нероуд за суфиксите на думата w и поне от теоритична точка - въпросът ще е решен.

Разглеждаме езика $\mathcal{L} = Suff(w)$.
Нека $u \in \Sigma^*$ е произволна дума. За да намерим класа на еквивалентност $[u]_{\equiv_{Suff(w)}}$ е достатъчно да намерим $u^{-1}Suff(w)$.

$u^{-1}Suff(w) \overset{def}{=} \{\gamma \in \Sigma^* \mid u\gamma \in Suff(w)\} = \{\gamma \in \Sigma^* \mid \gamma \in Suff(w)\ \&\ u\gamma \in Suff(w)\} =$
$= \{\gamma \in \Sigma^* \mid u$ завършва на позиция $|w| - |\gamma|$ в $w\}$



Дефиниция: Нека $w, u \in \Sigma^*$. Тогава $end - pos_w(u) = \{i \mid u = a_{i-|u|+1} \ldots a_i\}$, където $w = a_1 a_2 \ldots a_n$.

Сега може да продължим горното равенство като приложим дефиницията:

$u^{-1}Suff(w) \stackrel{def}{=} \{\gamma \in \Sigma^* \mid u\gamma \in Suff(w)\} = \{\gamma \in \Sigma^* \mid \gamma \in Suff(w) \ \& \ u\gamma \in Suff(w)\} =$
$= \{\gamma \in \Sigma^* \mid u$ завършва на позиция $|w| - |\gamma|$ в $w\} = \{\gamma = a_{i+1} \ldots a_n \mid i$ е от множеството $end - pos_w(u)\}$

Следствие:
$u \equiv_{Suff(w)} v \Leftrightarrow end - pos_w(u) = end - pos_w(v)$.

Пример:
$\mathbf{w} = \underset{\mathbf{1\,2\,3\,4\,5}}{\mathbf{ababc}}$. Интерес представляват думите които се срещат като инфикси в думата w, т.е. тези които обхващат някоя дума от вида „от i-та до j-та позиция в думата w“. Това е така, защото, ако имаме дума, която не се среща в рамките на w, то каквато и дума да и залепим от дясната страна, тя няма как да се превърне магически във суфикс на w.

Например:

$u = d$ $\quad d^{-1}Suff(w) = \{\gamma \in \Sigma^* \mid d\gamma \in Suff(w)\} = \emptyset$
$\quad \mathbf{end - pos_w(d) = \emptyset}$

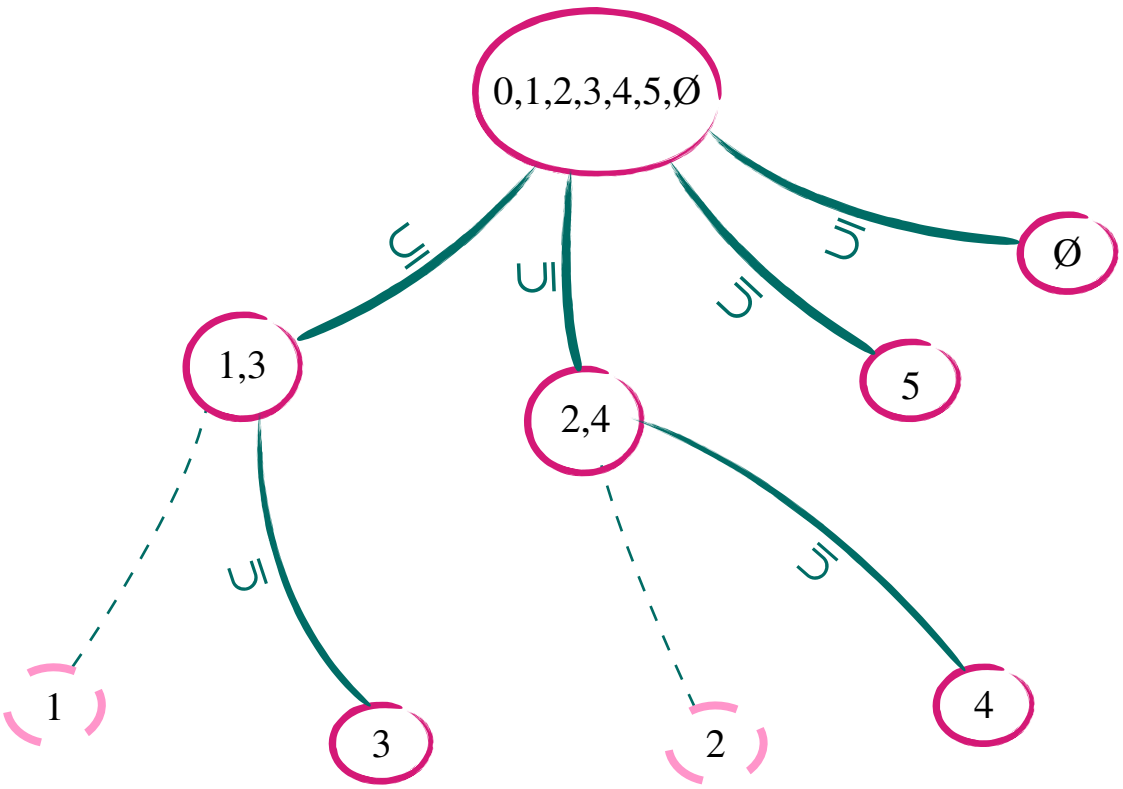
$u = bb$ $\quad (bb)^{-1}Suff(w) = \{\gamma \in \Sigma^* \mid (bb)\gamma \in Suff(w)\} = \emptyset$
$\quad \mathbf{end - pos_w(bb) = \emptyset}$

$\Rightarrow \mathbf{d} \equiv_{\mathbf{Suff(w)}} \mathbf{bb}$

Инфикси (поддуми на $w = \underset{1\,2\,3\,4\,5}{ababc}$): $Inf(w) = \{\beta \mid \exists\ \alpha, \gamma\ (\alpha\beta\gamma = w)\}$:

Inf(w)	end – pos_w(u)	$\cdot^{-1}$ Suff(ababc)
ε	{0, 1, 2, 3, 4, 5}	{ababc, babc, abc, bc, c, ε}
a	{1, 3}	{babc, bc}
b	{2, 4}	{abc, c}
c	{5}	{ε}
ab	{2, 4}	{abc, c}
ba	{3}	{bc}
bc	{5}	{ε}
aba	{3}	{bc}
bab	{4}	{c}
abc	{5}	{ε}
abab	{4}	{c}
babc	{5}	{ε}
ababc (w)	{5}	{ε}

Имаме еднозначност между позициите, на които завършва един инфикс в дума от w и съответно суфиксите (десния език), които могат да го следват.



Твърдение: Две множества от $end - pos$ или нямат общи елементи или едното е подмножество на другото.

Свойство: Нека $u, v, w \in \Sigma^*$

1.) $u \notin Inf(w) \Leftrightarrow end - pos_W(u) = \emptyset$

2.) Ако $u, v \in Inf(w)$ и $|u| \leq |v|$, то

2.1.) $end - pos_w(u) \cap end - pos_w(v) \neq \emptyset \Leftrightarrow end - pos_w(u) \supseteq end - pos_w(v)$

2.2.) $end - pos_w(u) \cap end - pos_w(v) \neq \emptyset \Leftrightarrow u \in Suff(v)$

Доказателство:

1.) $u \in Inf(w) \Leftrightarrow \exists \gamma\ (u\gamma \in Suff(w)) \Leftrightarrow\ u^{-1}Suff(w) \neq \emptyset \Leftrightarrow end - pos_w(u) \neq \emptyset$

2.) Нека $|u| \leq |v|$ и $u, v \in Inf(w)$

1.) Да допуснем, че $end - pos_w(u) \cap end - pos_w(v) \neq \emptyset$, тогава съществува

$end - pos_w(u) \cap end - pos_w(v) = \emptyset$
$\Downarrow$
$u \in Suff(v)$
$\Downarrow$
$end - pos_w(v) \subseteq end - pos_w(u)$

$i \in end - pos_w(u) \cap end - pos_w(v) \Rightarrow$
$\Rightarrow u = \overline{a_{i-|u|+1} \ldots a_i}$ и $v = \overline{a_{i-|v|+1} \ldots a_i}$.
И тъй като u е не по-дълга дума от v, то u трябва да е суфикс на v: $|u| \leq |v| \Rightarrow i - |u| + 1 \geq i - |v| + 1 \Rightarrow$
$\Rightarrow a_{i-|u|+1} \ldots a_i \in Suff(a_{i-|v|+1} \ldots a_i) \Rightarrow u \in Suff(v)$.

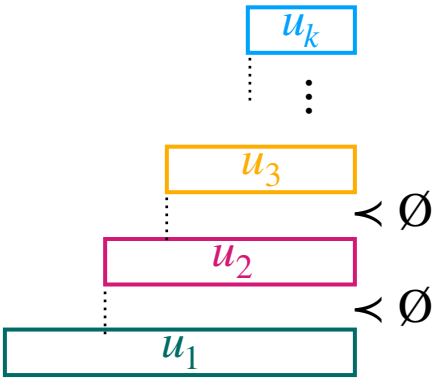
2.) Нека $u \in Suff(v)$. Тогава, ако $i \in end - pos_w(v)$, то $v = a_{i-|v|+1} \ldots a_i \underset{u=Suff(v)}{\Rightarrow}$
$\Rightarrow u = a_{i-|u|+1} \ldots a_i \Rightarrow i \in end - pos_w(u)$.

3.) Нека $end - pos_w(v) \subseteq end - pos_w(u) \overset{v \in Inf(w),\ 2.)}{\Rightarrow} end - pos_w(v) \neq \emptyset \Rightarrow$
$\Rightarrow end - pos_w(u) \cap end - pos_w(v) = end - pos_w(v) \neq \emptyset$

Тази лема вече ни помага да покажем, че броя на класовете на еквивалентност на нашата релация е не повече от два пъти дължината на думата плюс единица.

Следствие: Ако $u \in Inf(w)$ и $u \equiv_{Suff(w)} v$, то $u \in Suff(v)$ **или** $v \in Suff(u)$

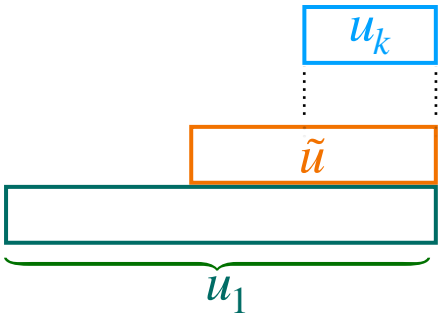
Следствие: Нека $u \in Inf(w)$, тогава $[u]_{\equiv_{Suff(w)}} = \{u_1, u_2, \ldots, u_k\}$, където $|u_i| = |u_1| - i + 1$ и u_i е суфикс на u_1.



Доказателство: Нека u_1 е най-дългата дума в класа на еквивалентност на u : $[u]_{\equiv_{Suff(w)}}$.
Тъй като $u \in Inf(w)$, то $[u]_{\equiv_{Suff(w)}} \subseteq Inf(w)$. Тъй като $Inf(w)$ е крайно, то има най-дълга дума $u_1 \in [u]_{\equiv_{Suff(w)}}$.

Нека u_1 е (една) такава дума (най-дълга), тогава от следствието всички, които са еквивалентни на u_1 са суфикси на u_1 (наистина, ако $v \equiv_{Suff(w)} u_1 \overset{def.\ u_1}{\Rightarrow}$
$|v| \leq |u_1| \Rightarrow v \in Suff(u_1)$ или $u_1 \in Suff(v)$, но u_1 е не по-къса от v
$\overset{|v|\leq|u_1|}{\Rightarrow} v \in Suff(u_1)$.

Дотук: $[u_1]_{\equiv_{Suff(w)}} = [u]_{\equiv_{Suff(w)}} \subseteq Suff(u_1)$.
Нека u_k е най-късата дума в $[u_1]_{Suff(w)}$. Тогава $end - pos_w(u_k) = end - pos_w(u_1)$.
накрая нека $\tilde{u} \in Suff(u_1)$, за който $|\tilde{u}| \geq |u_k|$. Следователно ще имаме следното:



Следователно $u_k \in Suff(\tilde{u})$ и $\tilde{u} \in Inf(w) \overset{Lema\ 1.)}{\Rightarrow}$

1. $end - pos_w(\tilde{u}) \subseteq end - pos_w(u_k) = end - pos_w(u_1)$
2. $\tilde{u} \in Suff(u_1) \Rightarrow end - pos_w(u_1) \subseteq end - pos_w(\tilde{u}) \Rightarrow$
   $\Rightarrow end - pos_w(u_1) = end - pos_w(\tilde{u}) \Rightarrow u_1 \equiv_{Suff(w)} \tilde{u}$

По този начин показахме, че $[u]_{Suff(w)} = [u_1]_{Suff(w)} = \left\{ \tilde{u} \in Suff(u_1) \,\middle|\, |u_1| \geq |\tilde{u}| \geq u_k \right\}$

Дефиниция: Нека $u \in Inf(w)$ с $\overset{w}{\leftarrow}\!\!u$ означаваме най-дългия представител в $[u]_{\equiv_{Suff(w)}}$.

Лема: Нека $\mathbf{w}$ е дума, а $\mathbf{u}$ е инфикс на $\mathbf{w}$. Тогава $\mathbf{u} = \overset{\mathbf{w}}{\leftarrow}\!\!\mathbf{u}$ тогава и само тогава, когато
1.) $\mathbf{u}$ е **префикс** на $\mathbf{w}$;
2.) има букви $\mathbf{a}$ и $\mathbf{b}$, които са **различни**, и за които $\mathbf{au}$ и $\mathbf{bu}$ са **инфикси** на $\mathbf{w}$.

Доказателство: Нека $u = \overset{w}{\leftarrow}\!\!u$. Тогава $[u]_{\equiv_{Suff(w)}} = \{\overset{w}{\leftarrow}\!\!u, u_2, u_3, \ldots, u_k\}$.
u_i - i-ти суфикс на $u = \overset{w}{\leftarrow}\!\!u$.

Да допуснем, че u не изпълнява нито 1), нито 2).
Тогава има буква a: au се среща в w.
u - суфикс на au.

Тъй като u не изпълнява нито 1), нито 2), то ако $i \in end-pos_w(u) \Rightarrow i \neq |u|$ (иначе u е префикс) $\Rightarrow a_{i-|u|}a_{i-|u|+1}\ldots a_i \in Inf(w)$ и от това, че u не изпълнява 2), $a = a_{i-|u|} \Rightarrow i \in end-pos_w(au)$. Оттук $end-pos_w(u) \subseteq end-pos_w(au)$. Оттук $end-pos_w(u) \subseteq end-pos_w(au)$, а тъй като u е суфикс на au, то $end-pos_w(au) \subseteq end-pos_w(u) \Rightarrow au \equiv_{suff(w)} u \Rightarrow u \neq \overleftarrow{u}^w$. Противоречие!

Обратно. Нека $u \in Pref(w)$. Тогава $|u| \in end-pos_w(u) \Rightarrow$
$|u| \in end-pos_w(\overleftarrow{u}^w) \Rightarrow |u| \geq |\overleftarrow{u}^w| \overset{def.}{\Rightarrow} \overleftarrow{u}^w = |u| \Rightarrow \overleftarrow{u}^w = u.$

Нека $u \in Inf(w), \quad au, bu \in Inf(w), a \neq b, a, b \in \Sigma.$

Знаем, че $u \in Suff(\overleftarrow{u}^w) \Rightarrow$ ако $u \neq \overleftarrow{u}^w$, то има буква $c \in \Sigma$, за която $cu \in Suff(\overleftarrow{u}^w)$ и $end-pos_w(cu) = end-pos_w(u)$.

Б.о.о. $c \neq a$ (или $c \neq b$). Нека $i \in end-pos_w(au) \Rightarrow \begin{cases} i \in end-pos_w(u) \\ i \neq end-pos_w(cu) \end{cases}$

$\Rightarrow u \not\equiv_{Suff(w)} cu \equiv_{Suff(w)} \overleftarrow{u}^w$. Противоречие!

Следователно $u = \overleftarrow{u}^w$.

Теоремата на Майхил-Нероуд ни дава, че автомата, който търсим е всъщност $\mathcal{A} = < \Sigma, Q_w, \overleftarrow{\varepsilon}^w, \delta_w, F_w >$, където състоянията ще са $Q_w = \{ \overleftarrow{u}^w \mid u \in Inf(w)\}$

$\delta_w(\overleftarrow{u}^w, a) = \begin{cases} \overleftarrow{ua}^w, \textbf{ if } ua \in Inf(w) \\ \textbf{else } not\ defined \end{cases}$. $\qquad F_w = \left\{ \overleftarrow{u}^w \mid \overleftarrow{u}^w \in Suff(w) \right\}$.

Искаме да построим този автомат и искаме да го построим on-line.

$\mathcal{T}_w = (Q_w, p_w)$, където $p_w(\overleftarrow{v}^w) = \overleftarrow{u}^w \overset{def}{\Leftrightarrow} \exists\, a \in \Sigma\ (\overleftarrow{au}^w = \overleftarrow{v}^w)$.

Вече сме готови, да докажем, че броя на на състоянията в множеството Q_w е линейно относно дължината на w и след това ще сме готови да покажем, че и броя на преходите в автомата е линейно относно броя на символите в w.

$w = ababc$

Заставаме зад някой връх и се питаме дали той се получава от някой друг с буква от ляво. Ясно е че пред празната дума няма как да добавим какъв да е елемент от азбуката, така че да се полуичи празната дума. Следователно празната дума няма да си има баща и ще бъде корен на дървото.

$\varepsilon \rightarrow \{1,2,3,4,5\}$

Но ако застанем в a ще видим, че тя се получава като конкатенираме празната дума с $a \leftarrow a \circ \varepsilon$. Следователно бащата на a ще е празната дума

$a \rightarrow \{1,3\}$

По същия начин в ab. Когато застанем в ab и отидем в ε и го конкатенираме с b, тъй като представител на b относно w е ab, то ще получим ab. Т.е. самата буква (дума) b не е във дървото $\mathcal{T}$, но нейният представител ab е.

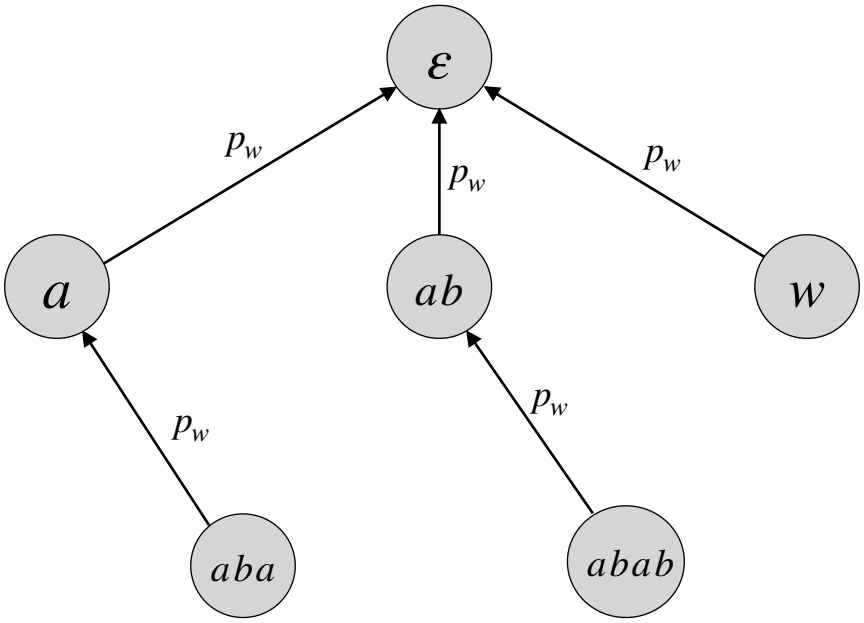
$\overleftarrow{b}^{w} = ab \rightarrow \{2,4\}$

По същия начин за w. w е еквивалентно на c така, че ако конкатенираме ε отляво с c ще получим класа на еквивалентност на c, тоест представителя на c: $\overleftarrow{c}^{w} = w$.

$w = \overleftarrow{c}^{w} = \overleftarrow{bc}^{w} = \overleftarrow{babc}^{w}$

$\overleftarrow{ba}^{w} = \overleftarrow{aba}^{w}$

Заставаме в $abab$ и се питаме, кой елемент от дървото е суфикс на $abab$. Виждаме, че това е елемента ab и добавяме към ab буквата b отпред, в резултат на което получаваме думата bab, но тя винаги се предшества от буквата a и поради тази причина, бащата на $abab$ е ab, тъй като конкатенирано отляво с b дава класа на еквивалентност на $abab$.



Наблюдение:

$p_w(v) = u \Leftrightarrow$

1.) $u \in Suff(v)$

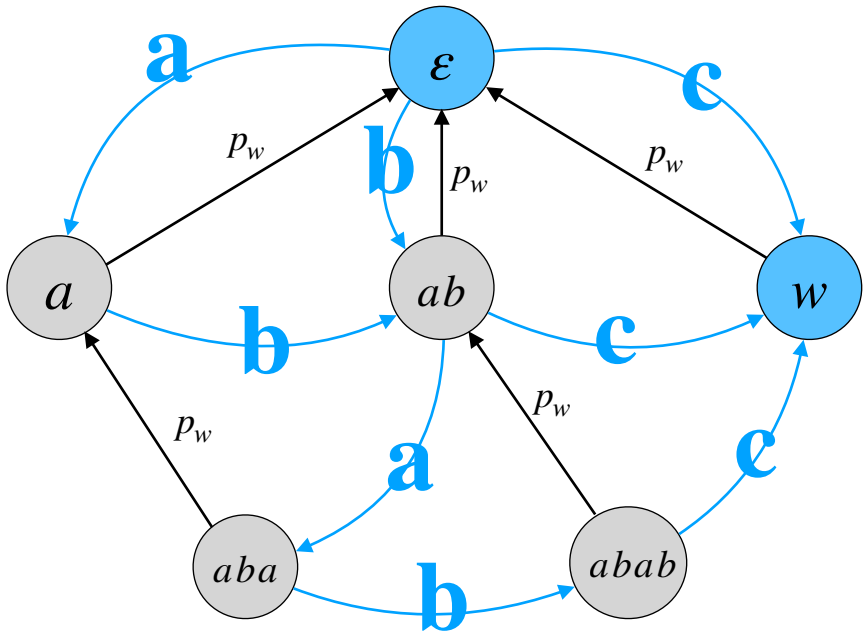
2.) u е с максимална дължина, така че $u \equiv_{Suff(w)} v$

Доказателство:

$p_w(v) = u \Leftrightarrow \exists\, a \in \Sigma\ (\overleftarrow{au}^{w} = v) \Leftrightarrow au \in Suff(v)$ и всички суфикси с дължини между $|v|$ и $|au|$ са в $[v]_{\equiv_{Suff(w)}}$ и $u \not\equiv_{Suff(w)} v$. Причината за последното твърдение е, че само по себе си u е представител в своя клас на еквивалентност и за това u не е еквивалентно на $au = v$, за никоя буква a.

(*Бащата е най-дългия суфикс на съответния връх, който не е в същия клас на еквивалентност с него*.)

Нека сега добавим към примера и **автоматните преходи**. Според дефиницията: заставаме в кое да е от състоянията и проследяваме буквите от които може да следва съответното състояние.



Необходимо е да определим и кои са финалните състояния на този автомат.
Това са w и изкачвайки се по предшествениците на w стигаме единствено до ε. Това са и двете финални състояния.

Тоеорема 1. Оценка на състоянията. За всяка дума $u \in \Sigma^*$:
$|Q_w| \leq 2|w| + 1$, като ако $w \geq 2$, то $|Q_w| \leq 2|w| - 1$.

Доказателство:
Разглеждаме $\mathcal{T}_w = (Q_w, p_w, \varepsilon)$ и нека $E_w = \{(v, p_w(v)) \mid v \in Q_w\}$

1. $|E_w| = |Q_w| - 1$
2. От $Q_w^{\geq 2} = \{u \in Q_w \mid u$ има поне **два** сина $\} = \left\{ u \in Q_w \,\middle|\, |\{v \mid (v,u) \in E_w\}| \geq 2 \right\}$.
   $Q_w^{\leq 1} = Q_w \setminus Q_w^{\geq 2}$.
3. От характеризацията на представителите $Q_w^{\leq 1} \subseteq Pref(w) \Rightarrow |Q_w^{\leq 1}| \leq |w| + 1$ (четем и празната дума като потенциален префикс).
4. От една страна
   $$|Q_w| - 1 = |E_w| = \sum_{u \in Q_w} \sum_{v:\,(v,u) \in E_w} 1 \geq \sum_{u \in Q_w^{\geq 2}} \sum_{v:(v,u) \in E_w} 1 \geq \sum_{u \in Q_w^{\geq 2}} = 2|Q_w^{\geq 2}|$$

   $$\Rightarrow |Q_w^{\geq 2}| + |Q_w^{\leq 1}| \geq 2|Q_w^{\geq 2}|; \quad |Q_w^{\leq 1}| - 1 \geq |Q_w^{\geq 2}|$$

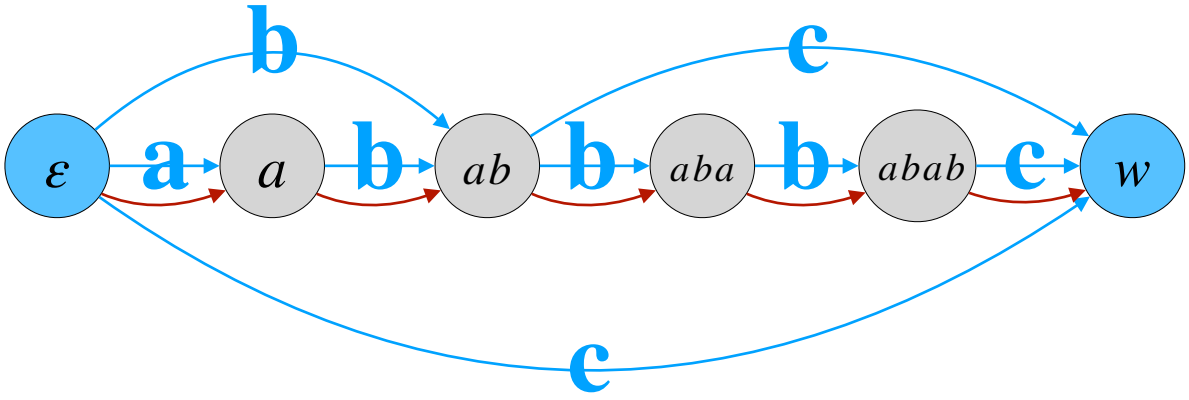
   $$\Rightarrow |Q_w| = |Q_w^{\leq 1}| + |Q_w^{\geq 2}| \leq 2|Q_w^{\leq}| - 1 \leq 2(|w| + 1) - 1 = 2|w| + 1.$$

Оценка на преходите.

Нетривиалния момент е да преброим сините ребра.

Има второ дърво, което е скрито във автомата и е различно от $\mathcal{T}$ дървото.

w = ababc



Нека $a \in \Sigma$, $u \in \Sigma^*$, за която $ua \in Q_w$. Тогава $u \in Q_w$.

Доказателство:

$ua \in Q_w \Rightarrow ua = \overleftarrow{ua}^{w} \overset{characterization}{\Rightarrow} ua \in Pref(w)$ или има $b \neq c : bua, cua \in Inf(w)$

$\Rightarrow bu, cu \in Inf(w) \overset{characterization}{\Rightarrow} u = \overleftarrow{u}^{w}$.

Следствие: $\mathcal{P}_w = (Q_w, \tilde{p}_w, \varepsilon)$, където $\tilde{p}_w(ua) = u$ за $a \in \Sigma$, $u \in Q_w$ е дърво.

Теорема 2. Броят на преходите в $\mathcal{A}_w$ е не по-голям от $3|w|$, като за $w \geq 2$: $\leq 3|w| - 4$.

Доказателство:

$\delta_w = \{(\overleftarrow{u}^{w}, a, \overleftarrow{ua}^{w}) \mid ua \in Inf(w)\}$.

Имаме два типа преходи. Първия тип съответстват на ребра в дървото $\mathcal{P}$:

$\delta_w^{(1)} = \{(u, a, ua) \mid ua \in Q_w \text{ и } \tilde{p}_w(ua) = u\}$

И втория тип, тези които съответстват на ребра в дървото $\mathcal{T}$:

$\delta_w^{(2)} = \delta_w \setminus \delta_w^{(1)}$.

1.) Тъй като $\mathcal{P}_w$ е дърво, то $|\delta_w^{(1)}| = |\tilde{p}_w| = |Q_w| - 1 \leq 2|w|$ от предишната теорема.

Остана да оценим $\delta_w^{(2)}$.

2.) Нека $t = (u, a, \overleftarrow{ua}^{w})$ е преход в $\delta_w^{(2)}$. Тъй като $\overleftarrow{ua}^{w} \in Q_w$, то има дума γ, за която

$\overleftarrow{ua}^{w} \underbrace{\gamma}_{\overleftarrow{\gamma(ua)}^{w}} \in Suff(w)$.

Съпоставяме на t думата $\lambda(t) = ua\gamma(\overleftarrow[w]{ua})$

2.1.) Да обърнем внимание, че
$\delta_w^*\left(\varepsilon, ua\gamma(\overleftarrow[w]{ua})\right) = \delta_w^*\left(\delta_w^*(\varepsilon, ua), \gamma(\overleftarrow[w]{ua})\right) = \delta_w^*\left(\overleftarrow[w]{ua}, \gamma(\overleftarrow[w]{ua})\right) \in F_w$, защото

$\overleftarrow[w]{ua}\,\gamma(\overleftarrow[w]{ua}) \in Suff(w) \Rightarrow ua\,\gamma(\overleftarrow[w]{ua}) \in \mathscr{L}(\mathscr{A}_w) = Suff(w) \Rightarrow \lambda(t) \in Suff(w).$

Рейнджа на функцията λ са само суфикси на w.

2.2.) λ е инекция от $\delta_w^{(2)}$ в $Suff(w)$. Това може да се докаже, като се вземат два различни прехода $t_1 = (u_1, a_1, \overleftarrow[w]{u_1a_1})$ и $t_2 = (u_2, a_2, \overleftarrow[w]{u_2a_2})$.

Втора част **(04.12.2020 г)**:

Имаме:

- Дума $w \in \Sigma^*$, $w = a_1 \ldots a_n$
- $\equiv_{Suff(w)}$ - релацията на Майхил-Нероуд за езика от суфиксите на w
- На всяка дума v от азбуката Σ, може да съпоставим множество от позиции, на които завършва тази дума в w: $end-pos_w(v) = \{i \mid v = a_{i-|v|+1} \ldots a_i\}$
- Две думи $u, v \in \Sigma^*$ са еквивалентни тогава и само тогава, когато множествата от позициите на които завършват в w съвпадат:
  $u \equiv_{Suff(w)} v \Leftrightarrow end-pos_w(u) = end-pos_w(v)$
- Нека $u, v \in \Sigma^*$ и $u, v \in Inf(w)$. Тогава $end-pos_w$ множествата им имат общ елемент, тогава и само тогава, когато е изпълнено поне едно от събитията $u \in Suff(v)$ или $v \in Suff(u)$:
  $end-pos_w(u) \cup end-pos_w(v) \neq \emptyset \Leftrightarrow \begin{cases} u \in Suff(v) \\ v \in Suff(u) \end{cases}$
- Последното ни помогна да характеризираме класовете на еквивалентност на кой да е инфикс на w по следния начин:
  $[v]_{\substack{\equiv_{Suff(w)} \\ v \in Inf(w)}} = \{v_1, \ldots, v_k\}$, където $v_1, \ldots, v_k$ са всички последователни по дължина суфикси на v_1.
- Представител на $[v]_{\substack{\equiv_{Suff(w)} \\ v \in Inf(w)}}$: $\overleftarrow[w]{v} = \arg\max\left\{ |v'| \,\middle|\, v' \equiv_{Suff(w)} v \right\}$

Имаики тази представа за класовете на еквивалентност на Майхил-Нероуд, получаваме **3** структури, които са свързани с намирането на минималния суфиксен автомат за w.

1. Автомата $\mathscr{A}_w = < \Sigma, Q_w, \varepsilon, \delta_w, F_w >$.

С автоматни състояния: $Q_w = \{ \overleftarrow[w]{v} \mid v \in Inf(w)\}$,

функция на преходите: $\delta_w(v, a) = \begin{cases} \overleftarrow[w]{va}, va \in Inf(w) \text{ и} \\ \perp \text{ не е дефинирано}, va \notin Inf(w) \end{cases}$

финални състояния: $F_w = \{v \in Q_w \mid v \in Suff(w)\}$.

2. Първо (*суфиксно*) дърво $\mathscr{T}_w = (Q_w, p_w, \varepsilon)$.

С функция на бащите: $p_w = \{(\overleftarrow{av}, v) \mid av \in Inf(w)\}$, където $a \in \Sigma$.
Едно състояние в нашия автомат (един връх) или казано по друг начин - един представител е представител на своя клас на еквивалентност $\Leftrightarrow$ е префикс в w или a се среща в различни леви контексти.
Тук за разлика от автомата, където добавяме буква от дясно и търсим представител, правим точно обратното - добавяме буква от ляво и търсим представител.

3. Второ (*префиксно*) дърво $\mathscr{P}_w = (Q_w, \tilde{p_w}, \varepsilon)$.

С функция на бащите: $\tilde{p_w} = \{(\underset{children}{\overleftarrow{va}}), \underset{parent}{v}) \mid va \in Q_w\}$, където $a \in \Sigma$.

Цел: *Online* алгоритъм, който да поддържа както автомата $\mathscr{A}_w$, така и суфиксното дърво $\mathscr{T}_w$.
Резултати: От това, че $|Q_w| \leq 2|w| + 1$ и $|\delta_w| \leq 3|w| \Rightarrow$ няма много състояния нито много преходи, т.е. си струва да се опитаме да построим такъв автомат и то сръчно.

Имаме $w = a_1 \ldots a_n$ и идва нова буква $a = a_{n+1}$. Трябва да решим как да модифицираме $\mathscr{A}_w$ и $\mathscr{T}_w$ и как да ги поддържаме.

Имаме $\mathscr{A}_w$ със състояния Q_w и функция на преходите δ_w и искаме да видим как ще получим Q_{wa}, δ_{wa};
Имаме $\mathscr{T}_w$ с функция на бащите p_w и искаме да видим как ще получим p_{wa}.

Финалните състояния няма да се поддържат online с цел да не натоварват алгоритъма. Тях ще добавим едва накрая след построяването на автомата.

Въпрос: Как се променят множествата $end - pos_w(v)$ за $v \in \Sigma^*$, $v \in Inf(w)$, когато добавим буква $a \in \Sigma$: $end - pos_w(v) \longmapsto end - pos_{wa}(v)$?

Може да се случи един от следните сценарии:

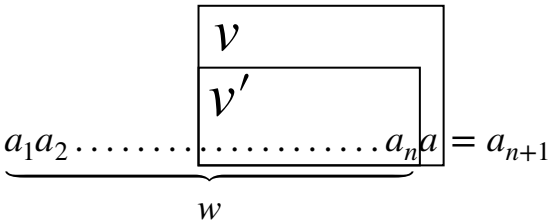
1.) $end - pos_w(v) \subseteq end - pos_{wa}(v)$

2.) $end - pos_{wa}(v) \backslash end - pos_w(v) = \begin{cases} \{n+1\}, \text{ако } v \in Suff(wa) \\ \emptyset, \text{ако } v \notin Suff(wa) \end{cases}$

Т.е. нещата, които ще се променят ще се случват по суфиксите на wa. Иначе казано, промени (нови състояния) ще има по суфиксите на wa. Но проблемът е, че ние нямаме директен достъп до суфиксите на wa.

3.) Как да получим достъп до суфиксите на wa?

$$v \in Suff(wa) \Leftrightarrow \begin{cases} v = \varepsilon \\ v = v'a,\ v' \in Suff(w) \end{cases}$$



4.) За да търсим (изброяваме) суфиксите на wa, може да процедираме по следния начин:

4.1.) използваме p_w за да изброяваме суфиксите на v'

4.2.) за да проверим дали имаме суфикс на wa може да използваме функцията на преходите δ_w. Тя ще ни каже дали съответния суфикс може да продължи с буквата a.
$\delta_w(v', a)$ ни дава суфикси на wa, които обаче са се срещали вътре в w, тоест са били инфикси на w.

Ще видим, че добавянето на позицията $\{n+1\}$ към много $end-pos_w$ множества не означава, че това ще доведе до създаването на много нови състояния (класове на еквивалентност). Всъщност се създават минимум едно и максимум две нови състояния. Защо това се случва?

I. $Q_w \longmapsto Q_{wa}$ (нови състояния)

1. $wa: \begin{cases} end-pos_w(wa) = \emptyset \\ end-pos_w(wa) = \{n+1\} \end{cases}$
$wa \in Pref(wa)$, което от характеризацията на представителите на класовете на еквивалентност на Майхил-Нероуд ни дава, че wa е представител в своя клас на еквивалентност: $\overleftarrow{wa} = wa \Rightarrow wa \in Q_{wa} \backslash Q_w$. Т.е. трябва да създадем едно ново състояние, което да съответства на wa.

2. Нека си представим суфиксите на wa, които са $n+2$ на брои:
$v_1 = wa, v_2, \ldots, v_{n+1} = a, v_{n+2} = \varepsilon$. За да видим какво се случва с тях може да видим какво се случва със съответните им $end-pos$ множества. Към техните $end-pos$ множества ще се добави $\{n+1\}$ и ще наблюдаваме следното:

$$v_1 = \underbrace{wa, v_2, \ldots\ldots\ldots\ldots}_{end-pos_{wa}=\{n+1\}} \underbrace{\ldots\ldots, v_{n+1} = a, v_{n+2}}_{|end-pos_{wa}|\geq 2} = \varepsilon$$

$$\begin{array}{lll}
v_j = & a_j \ldots . a_n a & \longleftarrow \textbf{representative } in\ [v_{i+1}]: \overleftarrow{v_{i+1}}^{wa} = v_{i+1} \\
\vdots & \vdots & \vdots \\
 & & end-pos_{wa} = \{n+1\} \cup \underbrace{end-pos_w}_{\neq\emptyset} \\
 & & \vdots \\
v_{i+1} = & a_{i+1} \ldots\ldots a_n a & \longleftarrow \textbf{representative } in\ [v_{i+1}]: \overleftarrow{v_{i+1}}^{wa} = v_{i+1} \\
v_i = & a_i \ldots\ldots\ldots a_n a & \\
\vdots & \vdots & \\
v_2 = & a_2 \ldots\ldots\ldots\ldots\ldots a_n a & end-pos_{wa} = \{n+1\} \cup \underbrace{end-pos_w}_{=\emptyset} = \\
 & & = \{n+1\} \\
v_1 = & a_1 \ldots\ldots\ldots\ldots\ldots\ldots a_n a &
\end{array}$$

$end-pos_{wa} = \{n+1\} \cup \underbrace{end-pos_w(v_j)}_{\neq\emptyset}$
разширение в ляво на v_j: $\{n+1\} \cup end-pos_w(v_{j-1})$

ако се различават, няма да е заради $\{n+1\}$, а заради множествата им $end-pos_{\underline{w}}$

Тоест за $j > i+1$ може да кажем, че
$end-pos_{wa}(v_j) = end-pos_{wa}(v_{j-1}) \Leftrightarrow end-pos_w(v_j) = end-pos_w(v_{j-1})$, което означава, че $\overleftarrow{v_j}^{wa} = v_j \Leftrightarrow \overleftarrow{v_j}^{w} = v_j$.

Извод: Освен wa, единствено v_{i+1} може да е ново състояние в Q_{wa}, т.е. $v_{i+1} \in Q_{wa} \setminus Q_w$.
Чрез дървото на суфиксите $\mathcal{T}_w$ ще намерим най-дългия суфикс wa, който се среща и в w, а всички останали класове на еквивалентност, които не засягат суфиксите на wa няма да се променят и съответно и техните представители ще си останат непроменени.

wa е едното ново състояние, а <u>най-дългия суфикс в wa, който се среща поне още веднъж в w</u> (ако има такъв) може да породи второто ново състояние. Други нови състояния няма.

Този най-дълъг суфикс ще има следната структура:

За v_{i+1} : 1 сл. $v_{i+1} = \varepsilon$, $a = a_{n+1}$ се среща за първи път в wa.
2 сл. $v_{i+1} = v'_{i+1}a$, където $v'_{i+1} \in Suff(w)$ и v'_{i+1} е най-дългия суфикс на w, за които $v'_{i+1}a$ се среща в w. $\left(v'_{i+1}a \in Inf(w)\right.$, ние ще търсим v'_{i+1} сред представителите на w, а не сред всички суфикси на $w\left.\right)$ Но v'_{i+1} е и представител на w, защото
$v'_{i+1} \equiv_{Suff(w)} = \overleftarrow{v_{i+1}}^{w} \Rightarrow v'_{i+1} \in Suff(w) \Leftrightarrow \overleftarrow{v'_{i+1}}^{w} \in Suff(w)$.

Ако $v'_{i+1}a$ се среша (завършва) на позиция j в w, то $\overleftarrow{v'_{i+1}}^{w}$ също се среща (завършва) на същата позиция j в w. $\left(\, j-1 \in end-pos_w(v'_{i+1} = end-pos_w(\overleftarrow{v'_{i+1}}^{w})\,\right)$.

Твърдение: $Q_{wa} = Q_w \cup \{wa, v'_{i+1}\}$, където:
1) $v'_{i+1} = \varepsilon \Leftrightarrow$ за всеки представител $v \in F_w$, $\delta_w(v,a)$ не е дефиниран.
2) $v'_{i+1} = va$, където $v = \arg\max\left\{ |u| \;\middle|\; u \in F_w,\ \delta_w(u,a) \text{ е дефиниран} \right\}$

Като $v'_{i+1} \notin Q_w$ $\Leftrightarrow v'_{i+1} = va \notin Q_w$
$\Leftrightarrow va \neq \overleftarrow{va}^{w}$
$\Leftrightarrow |v|+1 \neq |\overleftarrow{va}^{w}|$

Сега, преди да започнем да програмираме, ще трябва да фиксираме и някаква структура от данни, която да представя състоянията от Q_w (ясно е, че думите не са най-одачното нещо, защото ще изискват много повече памет). Ще представим състоянията с тяхната дължина и съответно за всяко състояние ще имаме баща му p_w и съответно списъка от неговите преходи δ_w в текущия автомат.
Представяне на Q_w, (δ_w, p_w):
struct $state$ {

int len // дължина на представителя на v: $|\overleftarrow{v}^{w}|$

int p // бащата на v: $p_w(\overleftarrow{v}^{w})$

$list\ \delta_w(v)$ // списък (масив) от автоматните преходи $\delta(v, \cdot)$

}

Намиране на предшественика v'_{i+1}, който е породил нуждата от второто ново състояние.

int $FindStem(Q_w, a, q, q')$ {

// q - съответства на w в масива от състояния Q_w; q' - съответства на wa

// FindStem намира състоянието v, за което $va = v'_{i+1}$, ако такова има

$v \leftarrow q$

while (v *is defined* **and** $\delta(v,a)$ *is not defined*)

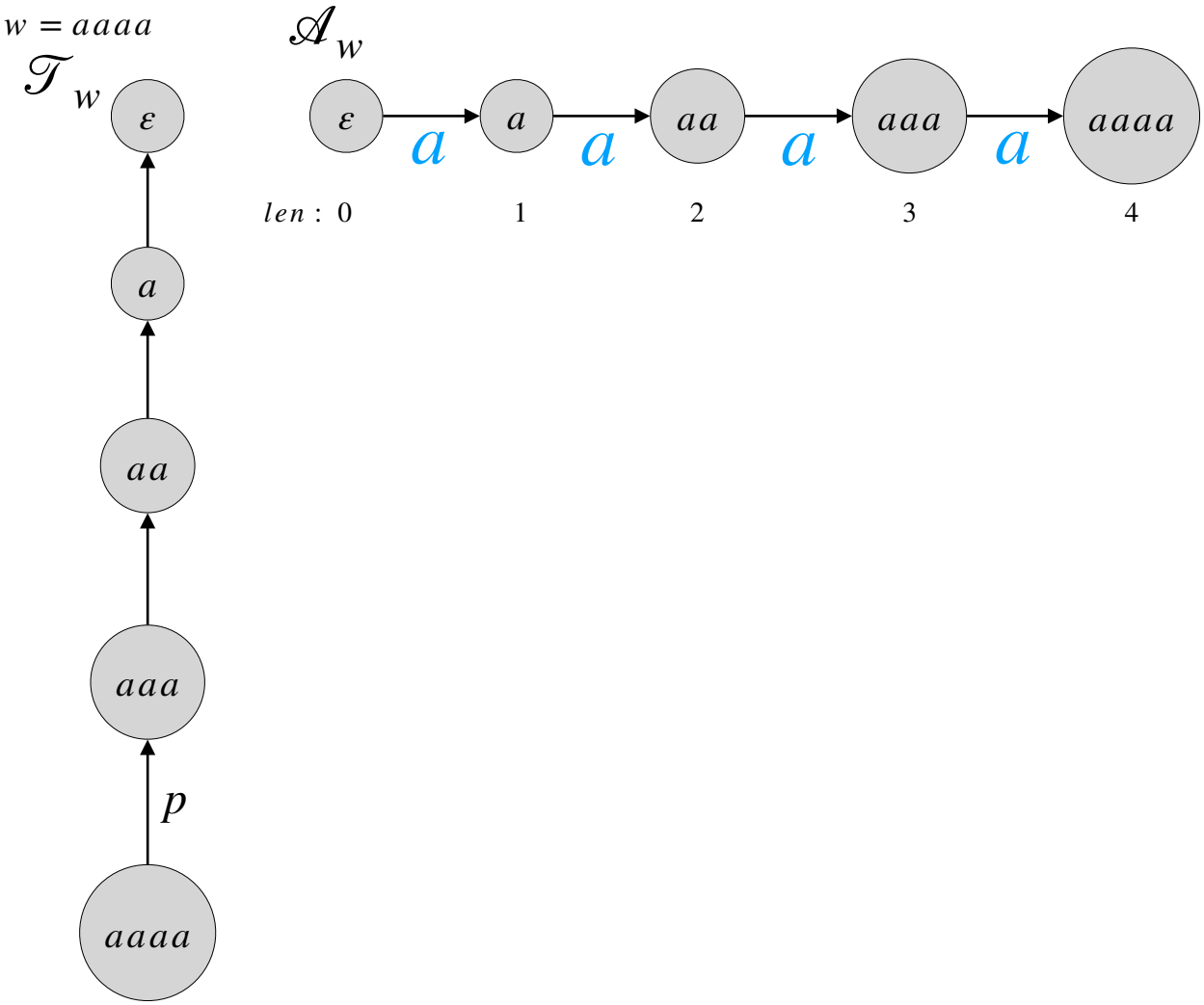
$\delta(v,a) \leftarrow q'$

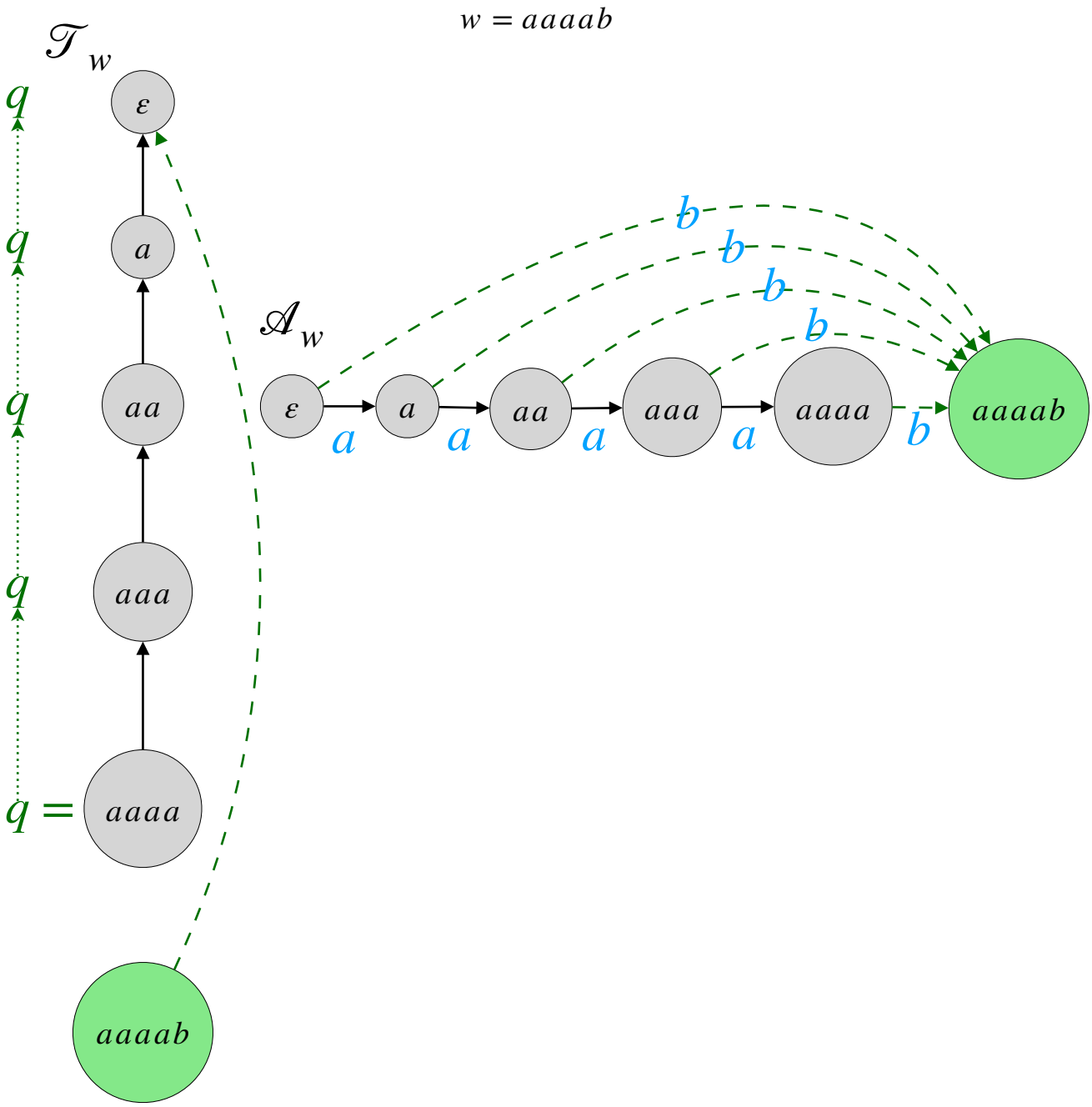
$v \leftarrow p(v)$

return v

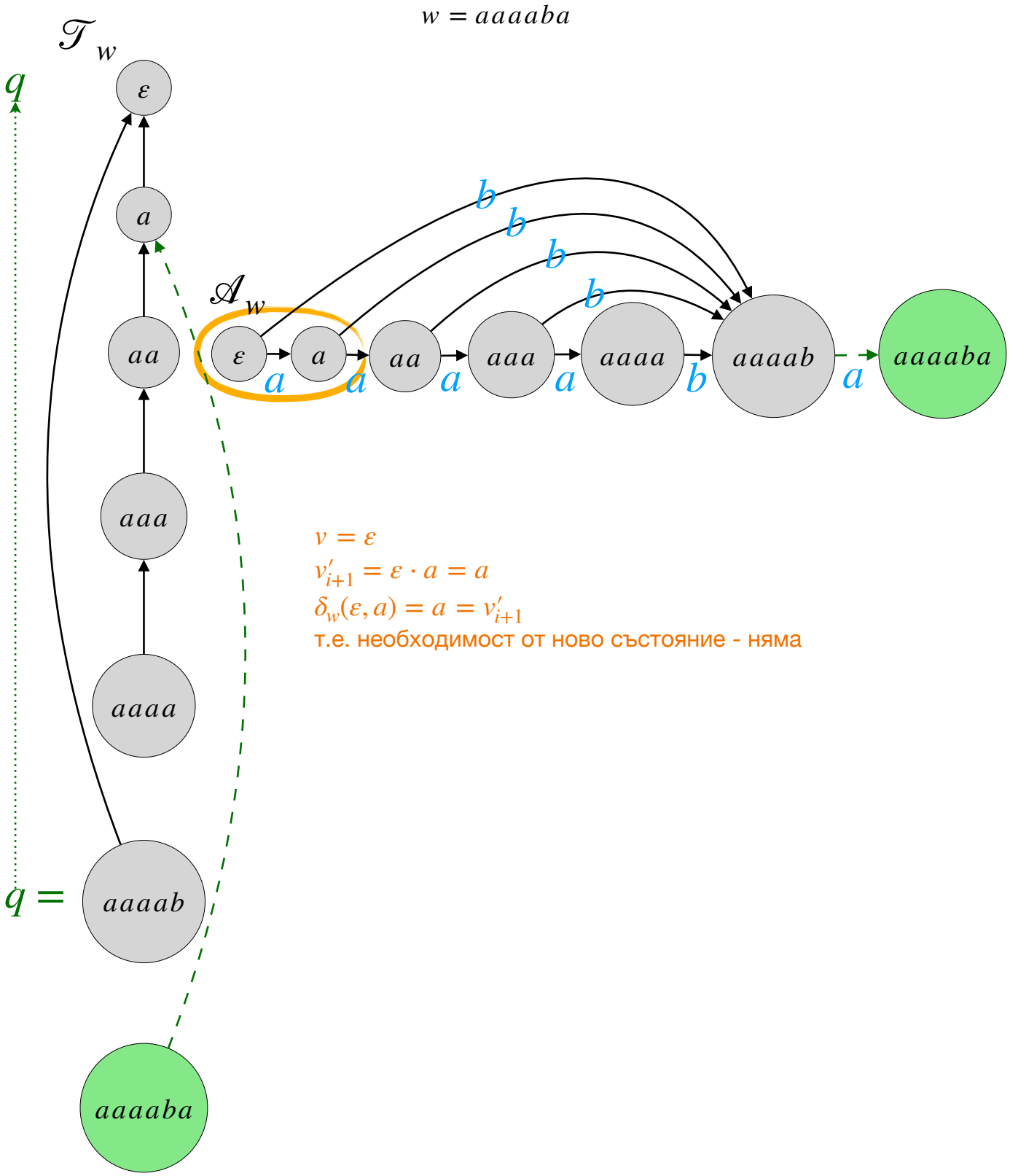
}

Пример:

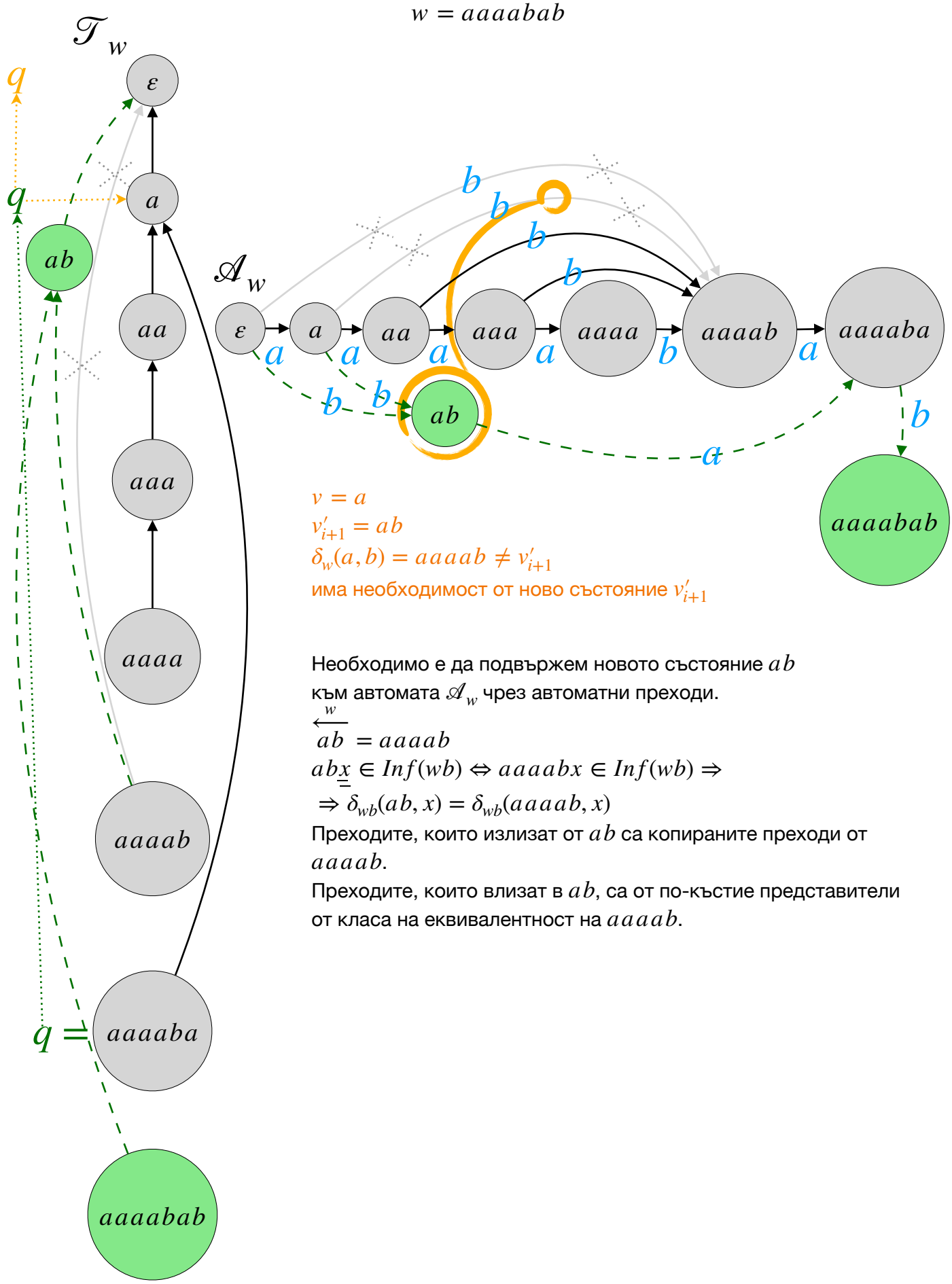




$\Rightarrow$ най-дългия суфикс в $\mathcal{T}_w$, който не е еквивалентен на $aaaab$ е ε.


$w = aaaaba$
$\mathcal{T}_w$
q
ε
a
aa
aaa
$aaaa$
$aaaab$
$aaaaba$
$\mathcal{A}_w$
b
a
$q =$
$v = \varepsilon$
$v'_{i+1} = \varepsilon \cdot a = a$
$\delta_w(\varepsilon, a) = a = v'_{i+1}$
т.е. необходимост от ново състояние - няма

$w = aaaabab$



$v = a$

$v'_{i+1} = ab$

$\delta_w(a, b) = aaaab \neq v'_{i+1}$

има необходимост от ново състояние v'_{i+1}

Необходимо е да подвържем новото състояние ab към автомата $\mathcal{A}_w$ чрез автоматни преходи.

$\overleftarrow{ab}^{w} = aaaab$

$abx \in Inf(wb) \Leftrightarrow aaaabx \in Inf(wb) \Rightarrow$

$\Rightarrow \delta_{wb}(ab, x) = \delta_{wb}(aaaab, x)$

Преходите, които излизат от ab са копираните преходи от $aaaab$.

Преходите, които влизат в ab, са от по-къстие представители от класа на еквивалентност на $aaaab$.

Q_w, wa, $v \in F_w$, $\delta_w(v,a) \to def.$, v - възможно най-дълго. $\underline{v'_{i+1} = va.}$

До тук знаем, че $Q_{wq} = Q_w \cup \{wa, v_{i+1}\}$.

II. $p_w \longmapsto p_{wa}$

1. $p_{wa}(wa) = v'_{i+1}$
$v'_{i+1} \in Suff(wa)$ и v'_{i+1} е най-дълъг със свойството $v'_{i+1} \in Inf(w)$ (допускане)
$\Rightarrow |end - pos_{wa}(v'_{i+1})| \geq 2 > 1 = |end - pos_{wa}(wa)|$
$\Rightarrow v'_{i+1} \neq a \Rightarrow$ ако $bv'_{i+1} \in Suff(wa)$ е по-дълъг от $v'_{i+1} \Rightarrow bv_{i+1} \notin Inf(w) \Rightarrow$
$\Rightarrow \overleftarrow{bv'_{i+1}}^{wa} = wa \Rightarrow p_{wa}(wa) = v'_{i+1}$.

2. Ако $v'_{i+1} \in Q_w$, то $p_{wa} = p_w \cup \{(wa, v'_{i+1})\}$

3. $v'_{i+1} \neq Q_w$. Нека $v' = \overleftarrow{v'_{i+1}}^{w} \Rightarrow v' \neq v'_{i+1}$.
Нека cv'_{i+1} е суфикс на v' $(c \in \Sigma)$. Тогава може да кажем, че
$end - pos_w(cv'_{i+1}) = end - pos_w(v')$.
Още, $\begin{cases} n+1 \in end - pos_{wa}(v'_{i+1}) \\ n+1 \notin end - pos_{wa}(cv'_{i+1}), \text{ иначе } cv'_{i+1} \text{ щеше да бъде по-дълъг от } v'_{i+1} \text{ и пак} \\ n+1 \notin end - pos_{wa}(v') \end{cases}$
той да е суфикс на wa, което ще противоречи на нашето допускане.

$\Rightarrow end - pos_{wa}(cv'_{i+1}) = end - pos_{wa}(v') \Rightarrow \overleftarrow{cv'_{i+1}}^{wa} = v' \Rightarrow p_{wa}(v') = v'_{i+1}$;
$p_{wa}(v'_{i+1}) = p_w(v')$.
Аргументация:
Нека $u' = p_w(v')$ и нека du' е суфикс на v' $(d \in \Sigma)$
$\Rightarrow v'_{i+1}$ е суфикс на v', който не е в Q_w и освен това $\overleftarrow{v'_{i+1}}^{w} = v' \Rightarrow du' \in Suff(v'_{i+1})$.
$\Rightarrow \overleftarrow{du'}^{wa} = v'_{i+1}$

4. За $v \notin \{wa, v'_{i+1}, \overleftarrow{v'_{i+1}}^{w}\}$: $p_{wa}(v) = p_w(v)$.

$p_{wa}(wa) = v'_{i+1}$; $v'_{i+1} \notin Q_w$, то $\begin{cases} p_{wa}(v'_{i+1}) = p_w(\overleftarrow{v'_{i+1}}^{w}) \\ p_{wa}(\overleftarrow{v'_{i+1}}^{w}) = v'_{i+1} \end{cases}$

III. $\delta_w \longmapsto \delta_{wa}$

1. $v \in F_w$, $\delta_w(v,a)$ не е дефинирано.
Тогава $va \notin Inf(w) \Rightarrow end - pos_{wa}(va) = \{n+1\} = end - pos_{wa}(wa)$.
Следователно, това което получаваме е, че $\delta_{wa}(v,a) = wa$
2. Да разгледаме връх v'_{i+1}, в ситуацията, която е нов връх ($v'_{i+1} \in Q_{wa} \backslash Q_w$). Нека

$x \in \Sigma$ е буква. Тогава $j \in end - pos_{wa}(v'_{i+1}x) \Leftrightarrow v'_{i+1}x = a_{j-|v'_{i+1}|} \dots a_j \Leftrightarrow$
$v'_{i+1} = a_{j-|v'_{i+1}|} \dots a_{j-1}$ & $a_j = x \Leftrightarrow j-1 \in end - pos_w(v'_{i+1})$ & $a_j = x \Rightarrow$
$j-1 \in end - pos_w(\overleftarrow{v'_{i+1}})$ & $a_j = x$. Това показва, че

$$\boxed{\delta_w(v'_{i+1}, x) = \delta_{wa}(\overleftarrow{v'_{i+1}}, x)}$$

3. $v \in Q_w$, $x \neq a$
$\delta_{wa}(v, x) = \delta_w(v, x)$ - остава непроменен, тъй като $x \neq a$ и позициите
$end - pos_{wa}(\overleftarrow{vx}) = end - pos_w(\overleftarrow{vx})$

4. $v \in Q_w$, $x = a$ и $v \notin F_w$.
Тогава $end - pos_{wa}(va) = end - pos_w(va)$, $\delta_{wa}(v, a) = \delta_w(v, a)$
5. $v \in F_w$, $x = a$ и $\delta_w(w, a)$ е дефинирано

5.1. сл.) $\delta_w(v, a) \neq \overleftarrow{v'_{i+1}} \Rightarrow \overleftarrow{va}$ е суфикс (същински) на
$v'_{i+1} \Rightarrow n+1 \in end - pos_{wa}(\overleftarrow{va})$
$\Rightarrow \overleftarrow{va} \equiv \overleftarrow{va} \Rightarrow \delta_{wa}(v, a) = \overleftarrow{va} = \overleftarrow{va} = \delta_w(v, a)$

5.2. сл.) $\delta_w(v, a) = \overleftarrow{v'_{i+1}}$.

5.2.1.) Ако $|v|+1 > |v'_{i+1}|$, то $va \in Suff(\overleftarrow{v'_{i+1}}) \backslash Suff(v'_{i+1})$
$\Rightarrow n+1 \notin end - pos_{wa}(va) = end - pos_{wa}(\overleftarrow{v'_{i+1}}) \Rightarrow$
$\Rightarrow \delta_{wa}(v, a) = \overleftarrow{v'_{i+1}}$.

5.2.2.) Ако $|v|+1 \leq |v'_{i+1}|$, тогава
$end - pos_{wa}(va) = end - pos_w(va) \cup \{n+1\} =$
$= end - pos_w(v'_{i+1}) \cup \{n+1\} = end - pos_{wa}(v'_{i+1})$.
Трябва да насочиме прехода от v с a към v'_{i+1}

Ще отбележим само промените, които трябва да се направят:

1. $v \in F_w$, $\delta_w(v, a)$ не е дефинирано. Тогава $\delta_{wa}(v, a) = wa$
2. $v \in F_w$, $\delta_w(v, a) = \overleftarrow{v'_{i+1}}$, $|v|+1 \leq |v'_{i+1}|$. Тогава $\delta_{wa}(v, a) = v_{i+1}$
3. $\delta_{wa}(v'_{i+1}, x) = \delta_{wa}(\overleftarrow{v'_{i+1}}, x)$

Псевдо код:

procedure $Redirect(v, a, old\_val, new\_val)$ {
// ще пренасочи преходите от предшествениците на v, заедно с v с буква a от old_val към new_val и ще върне първия, който няма тази стара стойност
while(v is $defined$ **and** $\delta(v, a) = old\_val$) **do** {
$\delta(v, a) \leftarrow new\_val$

$v \leftarrow p(v)$
}
return v
} // първия връх, такъв че $\delta(v, a) \neq old\_val$

$FindStem \leftarrow Redirect(v, a, NULL, new\_val)$ // възможно заместване

procedure $CopyTransitions(v\_source, v\_dest)$ {
for each $x \in I : \delta(v\_source, x)$ *is defined* **do** {
$\delta(v\_dest, x) \leftarrow \delta(v\_source, x)$
}
}

procedure $TreeModification(v\_old, v\_new)$ {
// знаем, че $v\_\overleftarrow{new}^{w} = v\_old$ и $v\_\overleftarrow{new}^{w} = v\_new$
$p(v\_new) = p(v\_old)$
$p(v\_old) \leftarrow v\_new$
}

procedure $InsertChar(Q, q, a)$ {
// $q \mapsto \overleftarrow{w}^{w}$, $Q \mapsto Q_w$, $a \in \Sigma$
$q' \leftarrow new\ state$
$len(q') \leftarrow len(q) + 1$
$v \leftarrow Redirect(q, a, NULL, q')$
if v *is not defined* **then**
$p(q') \leftarrow q_\varepsilon$ // $\overleftarrow{\varepsilon}^{w}$
else
$v' \leftarrow \delta(v, a)$ // $v' = \overleftarrow{va}^{w}$
//$v'_{i+1} = va$
if $len(v') \neq len(v + 1)$ **then** {
$v\_new' \leftarrow new\_state$
$len(v\_new') \leftarrow len(v) + 1$
$CopyTransitions(v', v\_new')$
$TreeModification(v', v\_new')$
$Redirect(v, a, v', v\_new')$
$v' \leftarrow v\_new$
}
$p(q') \leftarrow v'$
return q'
}

procedure $BuildSuffixAutomaton(w)$ {
$q_\varepsilon \leftarrow new\_state$
$Q \leftarrow \{q_\varepsilon\}$

$len(q_\varepsilon) \leftarrow 0$
$q \leftarrow q_\varepsilon$
for $i = 1$ **to** $|w|$ **do**
    $q \leftarrow InsertChar(Q, q, w[i])$
$F_w \leftarrow \emptyset$
while $(q\ is\ defined)$
    $F_w \leftarrow F_w \cup \{q\}$
    $q \leftarrow p(q)$
}

Сложност:

Имаме най-много две състояния, които се създават при всяка итерация за добавяне на буква.
Трябва да анализираме и времето, което губим при добавянето и пренасочване на преходи. Първия Redirect и CopyTransitions ще добавят преходи и то само нови. Това време ще е амортизирано точно $\leq 3w$
Времето за модификация ще е константно.
Единственото което остана неанализирано е последния Redirect. Разликата между него и първия е, че първия добавяме нови преходи, защото там имаме нещо което не е дефинирано и пренасочваме тези които не са дефинираме, докато във втория Redirect те са имали стойност и тя е била v', но сега ние я променяме на v_new' и следователно тези преходи ние ги разглеждаме за втори, трети, ... десети път и не знаем за кой път точно и не може просто да кажем, че ще загубим време колкото е общия брой на преходите в резултатния автомат.
В общия случай това не е вярно, но е вярно ако човек приеме, че азбуката е с константна дължина ($\log \Sigma$ фактор, ако използваме балансирани дървета например).

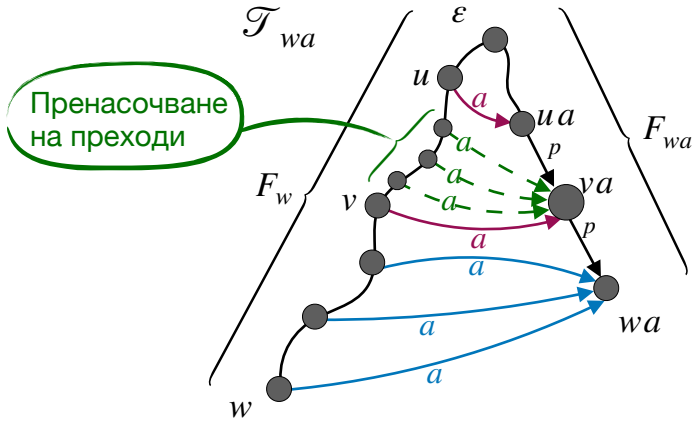
Трета част (11.12.2020 г):

До тук за времевата сложност на алгоритъма na Blumer et. al получаваме:

- добавяне на нови състояния (≤ 2 букви $\Rightarrow \leq 2|w|+1$)
- добавяне на нови преходи (общо $\leq 3|w|$)
- пренасочване на преходи (остана да изчислим времевата сложност)

Като изключим въпросите от вида: „Има ли преход от дадено състояние с дадена буква и ако има такъв върни ми къде точно отива той?“, всички останали стъпки от алгоритъма се свеждат до тези три операции.

Нетривиалния момент, който остана за анализиране е пренасочването на преходите.
Те се пренасочват от един клон, който съответства на суфикс на текущата дума, към клон на думата, която се получава след добавяне на конкретен символ.



Искаме да уточним колко на брой такива преходи ще преходи ще направим (не за всяка итерация при добавяне на нова буква, а тотално за финалната дума). Естествено ще разгледаме какво се случва и при всяка една итерация.

Да разгледаме F_{wa} и да разгледаме още $\tilde{F}_w = \{q \in Q_{wa} \mid q \text{ е предшественик на } w \text{ в } \mathcal{T}_{wa}\}$

1. За всеки връх (възел) $q \in \tilde{F}_w$, $q \in Suff(w)$ и следователно
$qa \in Suff(wa) \Rightarrow \delta_{wa}(q,a) = \overleftarrow{qa}^{wa} \in F_{wa}$

2. За всеки връх $v \in F_{wa}$:
а) $v = \varepsilon$
б) $v \in v'a$ и $\tilde{p}_{wa}(v) = v' \in \tilde{F}_w$ (т.е. $v' \in Q_{wa}$ и $\delta_{wa}(v',a) = v'a = v$)

3. Пренасочени преходи при стъпката от $w \longmapsto wa$ са:
1. най-много един преход от вида $\delta_{wa}(v,a) = va$
2. няколко прехода от вида $u \in \tilde{F}_w$, $\delta(u,a) = va$, където $|u| < |v|$

$\overset{1.\ 2.}{\Rightarrow}$ броят на тези преходи е $\leq |\tilde{F}_w| - \big(|F_{wa}| - \underbrace{1}_{\varepsilon \text{ няма предшественик}}\big)$

Общо $(w \longmapsto wa)$ сме пренасочили не повече от
$|\tilde{F}_w| - |F_{wa}| + 1 + 1 = |\tilde{F}_w| - |F_{wa}| + 2$

Остава да отбележим, че:

4. На стъпката от $(w \longmapsto wa)$ добавяме най-много 2 състояния ($wa \notin \tilde{F}_w$ със сигурност и още най-много 1 състояние), откъдето $|\tilde{F}_w| \leq |F_w| + 1$.

Окончателно, пренасочените преходи са $\leq |F_w| + 1 - |F_{wa}| + 2 = \underbrace{|F_w| - |F_{wa}| + 3}$.

Следователно за всички пренасочени преходи за n итерации са

$$\leq \sum_{i=0}^{n-1} \Big(|F_{a_1a_2\ldots a_i}| - |F_{a_1\ldots a_i a_{i+1}}| + 3 \Big) = 3n + |F_\varepsilon| - |F_w| \leq 3n$$

$\Rightarrow$ Пренасочените състояния са $\leq 3n$.

Теорема: Времевата сложност на алгоритъма на Blumer et. al. за построяване на минимален суфиксен автомат за дума w е $O\Big(|w| \,.\, c\big(|\Sigma|, |w|\big)\Big)$, където с $c\big(|\Sigma|, |w|\big)$ е времето за:

- проверка за преход $\delta(q,a)$
- достъп до преход $\delta(q,a)$
- добавяне на нов преход $\delta(q,a)$

Заявки върху дървета в контекста на минималния суфиксен автомат от алгоритъма на Blumer et. al.

1. $\mathcal{T}_w = (Q_w, p_w, \varepsilon)$

$p_w = \overleftarrow{av}^{w} \mid av \in Inf(w)\}$

Това дърво го нарекохме суфиксно. В него е смислено да правим заявки от вида LCA (най-близък общ предшественик):

$\alpha = LCA_{\mathcal{T}_w}(v, u)$, където $v, u \in Q_w$

Това е така, защото бащата на v ще бъде суфикс на v и съответно бащата на u ще бъде суфикс на u. Следователно общия прародител на u и v ще се явява суфикс и на двете думи v и u, при това най-големия такъв суфикс.
1.) α е суфикс на v и u
2.) ако $\alpha = v$ или $\alpha = u$, то α е най-дългия общ суфикс на v и u
3.) ако $\alpha \neq v$ и $\alpha \neq u$, то v и u са наследници на две различни деца на α, т.е. $v \in \mathcal{T}_w(\overleftarrow{a\alpha})$ и $v \in \mathcal{T}_w(\overleftarrow{b\alpha})$, като $\overleftarrow{a\alpha} \neq \overleftarrow{b\alpha} \Rightarrow a \neq b \Rightarrow$ $\big($т.к. $a\alpha \in Suff(v)$, а $b\alpha \in Suff(u)\big)$ α отново ще бъде най-дългия общ суфикс на v и u.

2. $\mathcal{P}_w = (Q_w, \tilde{p}_w, \varepsilon)$

$\tilde{p}_w = \{(va, v) \mid va \in Q_w\}$

Дуално, заявка за най-близък общ предшественик, но в префиксното дърво $\mathcal{P}_w$:

$\alpha = LCA_{\mathcal{P}_w}(u, v)$, където $v, u \in Q_w$

α е най-дългият общ префикс на u и v

3. В префиксното дърво още е смислено да правим заявки от вида LA (предшественик от определено ниво):
$LA_{\mathcal{P}_w}(v, d) = u \in Q_w$
Връщайки се с d стъпки нагоре, ние получаваме u - префикс на v с дължина $|u| = |v| - d.$

Има асиметрия между префиксите и суфиксите по отношение на заявката „предшественик от определено ниво“. В префиксното дърво $\mathcal{T}_w$, заявката LA има ясна семантика, но няма ясна семантика в суфиксното дърво $\mathcal{P}_w$, тъй като в $\mathcal{P}_w$ нямаме регулярност между дължината на бащата и дължината на сина в него, както има такава в префиксното дърво.