Суфиксни низ и суфиксно стабло

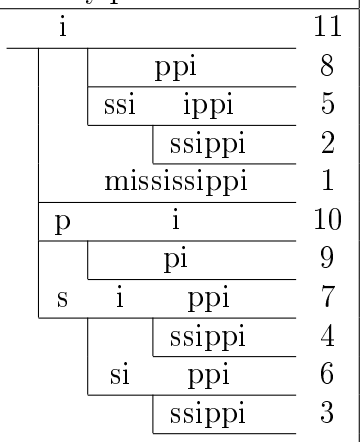
November 15, 2022

Садржај

1 Увод

Суфиксни низ и суфиксно стабло дате ниске су структуре података помоћу којих се ефикасно могу решавати проблеми као што је тражење речи у тексту, тражење понављања или палиндрома у тексту, тражење заједничке подниске за два или више текстова и слично.

Нека је задата ниска s дужине n. Нека је $s[i..j]$ део те ниске са индексима од i до j, $1 \leq i \leq j \leq n$. Специјално, нека $S_i = s[i..n]$ означава суфикс ниске s који почиње од индекса i. Суфиксни низ SA ниске s је низ индекса лексикографски сортираних суфикса: $S_{SA[1]} < S_{SA[2]} < \cdots < S_{SA[n]}$. Другим речима, ако је $SA[i] = j$, онда суфикс S_j има ранг $i = rang[j]$ међу нискама $S_1, S_2, \ldots, S_n$. Низови SA и $rang$ су међусобно

i	$SA[i]$	$rang[i]$	$S_{SA[i]}$	$LCP[i]$	Суфиксно стабло s
1	11	5	i		
2	8	4	ippi	1	
3	5	11	issippi	1	
4	2	9	ississippi	4	
5	1	3	mississippi	0	
6	10	10	pi	0	
7	9	8	ppi	1	
8	7	2	sippi	0	
9	4	7	sissippi	2	
10	6	6	ssippi	1	
11	3	1	ssissippi	3	

Табела 1: Суфиксни низ, низ LCP и суфиксно стабло ниске mississippi.

инверзне пермутације скупа $\{1, 2, \ldots, n\}$: $SA[rang[j]] = rang[SA[j]] = j$, $1 \leq j \leq n$.

Пример 1.1. *Сортирани низ суфикса ниске* $s =$ mississippi *и њен суфиксни низ приказани су у табели 1. Ниска* $S_8 =$ ippi *је суфикс* $s[8..11]$.

- *Лексикографски редослед суфикса је* $S_{11} < S_8 < \cdots < S_3$.
- *Који је по реду суфикс* S_5*? Трећи (његов ранг је* 3*), јер је* $rang(5) = 3$.
- *Који суфикс је седми по реду? То је суфикс* $S_9 =$ ppi*, јер је* $SA[7] = 9$.

Најдужи заједнички префикс или LCP (longest common preffix) за две ниске s_1 и s_2 је дужина $LCP(s_1, s_2)$ најдуже ниске која је префикс и s_1 и s_2. За фиксирану ниску s нека је $LCP(i, j) = LCP(S_i, S_j)$. Нисци s може се поред њеног суфиксног низа SA придружити и низ LCP, чији је i-ти члан једнак $LCP[i] = LCP(S_{SA[i-1]}, S_{SA[i]})$, $i = 2, 3, \ldots, n$. Другим речима, $LCP[i]$ је дужина најдужег заједничког префикса i-тог и $(i+1)$-ог суфикса ниске у сортираном редоследу.

Пример 1.2. *Низ* LCP *ниске* $s =$ mississippi *приказан је у табели 1 заједно са суфиксним низом. За ову ниску је* $LCP(4) = 4$*, јер је* $SA[3] = 5$*,* $S_5 = issippi$*,* $SA[4] = 2$*,* $S_2 = ississippi$*, и*

$$LCP[4] = LCP(issippi, ississippi) = |issi| = 4.$$

Суфиксно стабло је друга структура података за представљање интерне структуре ниске. Суфиксно стабло ниске s дужине n је коренско стабло за које важи:

- стабло има тачно n листова који су нумерисани бројевима $1, 2, \ldots, n$;
- сваки унутрашњи чвор има бар два сина;
- свака грана је означена непразном подниском ниске s;
- никоје два гране из истог чвора немају ознаке које почињу истим карактером;
- конкатенација свих ознака на путу од корена до листа са ознаком i једнака је суфиксу $S_i = s[i..n]$.

Пример 1.3. *Суфиксно стабло ниске* $s =$ mississippi *приказано је у табели 1 заједно са њеним суфиксним низом.*

Пошто је $|s| = 11$*, суфиксно стабло има* 11 *листова. Поред тога, ово стабло има корен и* 7 *унутрашњих чворова. Конкатенација ознака грана на путу од корена до листа* 7 *је* $S_7 = \mathrm{i} - \mathrm{pi} - \mathrm{ppi}$.

Из дефиниције не следи постојање суфиксног стабла за произвољну ниску s. Проблем се појављује када је неки суфикс S_i једнак префиксу неког другог суфикса S_j, јер се тада пут који одговара суфиксу S_i завршава унутар пута који одговара суфиксу S_j, тј. не завршава се у листу стабла. На пример, не постоји суфиксно стабло за ниску cxabxa, јер се пут који одговара суфиксу xa, као префикс суфикса xabxa, не завршава у листу. Уобичајени начин да решавања овог проблема реши је да се на крај ниске дода знак који се не појављује унутар ње (обично је то знак \$; претпоставља се da знак \$ лексикографски претходи свим осталим знацима).

Лема 1.1. *Укупан број грана у суфиксном стаблу ниске дужине* n *је највише* $2n - 1$.

Доказ. Можемо да замислимо да суфиксно стабло настаје од корена и само једне гране која води до јединог листа додавањем нових суфикса. Пошто додавање сваког унутрашњег чвора повећава број листова бар за 1, а број листова у коначном стаблу је n, број унутрашњих чворова је највише $n - 1$. Број грана ка листовима једнак је n, а број грана ка унутрашњим чворовима је највише $n-1$. Према томе, укупан број грана у суфиксном стаблу је највише $2n - 1$. □

Ако би се у суфиксном стаблу уз сваку грану чувала њена комплетна ознака, онда би просторна сложеност стабла за ниску дужине n била у најгорем случају $O(n^2)$. Међутим, произвољна подниска $s[i..j]$ познате ниске s одређена је са два индекса i и j. Пошто је на основу доказане леме број грана у суфиксном стаблу ниске дужине n највише $O(n)$, заменом ознака на гранама са по два броја постиже се да је простор који заузима суфиксно стабло $O(n)$.

2 Алгоритам линеарне сложености за конструкцију суфиксног низа

Без смањења општости може се претпоставити се да су знакови ниске $s = s[1..n]$ кодирани природним бројевима из скупа $\{1, 2, \ldots, n\}$. Упознаћемо се са алгоритмом линеарне сложености за конструкцију суфиксног низа ниске чији су аутори Каркаинен и Сандерс [1]. Тај алгоритам је изложен у последњем издању "библије" [2]. Заснован је на специфичном разлагању (divide-and-conquer):

1. Сортирају се суфикси S_i ниске s, $i \mod 3 \neq 0$ ($i = 1, 2, 4, 5, 7, 8, 10, 11, \ldots$; у даљем тексту: А-суфикси) свођењем на рекурзивну конструкцију суфиксног низа посебно формиране ниске дужине око $2n/3$.

2. На основу тога се сортирају преостали суфикси S_i, $i \mod 3 = 0$ ($i = 3, 6, 9, \ldots$; у даљем тексту: Б-суфикси).

3. Обједињавањем сортираних низова А-суфикса и Б-суфикса одређује се суфиксни низ ниске s.

Прелазимо на детаљнији опис три основна корака алгоритма, илуструјући поступак обрадом ниске $s =$ mississippi.

1. Полазећи од ниске s, формира се ниска P над азбуком од мета знакова — трословних блокова полазне азбуке; при томе суфикси ниске P имају исти лексикографски редослед (одређен суфиксним низом):

   А Формирати ниске P_1 и P_2 од мета знакова (тројке одвајамо заградама):

   - $P_1 = (s[1..3])(s[4..6])(s[7..9]) \cdots (s[n'..n'+2])$, где је n' највећи број са остатком 1 по модулу 3, $n' \leq n$. При томе је ниска

s продужена специјалним знаковима $\varnothing$ са кодом 0. За $s = \text{mississippi}$ добија се $P_1 = (mis)(sis)(sip)(pi\varnothing)$.

- $P_2 = (s[2..4])(s[5..8])(s[8..10]) \cdots (s[n''..n''+2])$, где је n'' највећи број са остатком 2 по модулу 3, $n'' \leq n$. За ниску s је $P_2 = (iss)(iss)(ipp)(i\varnothing\varnothing)$.

Ако је n дељиво са 3, на крај P_1 додаје се мета знак $(\varnothing\varnothing\varnothing)$. Тиме се постиже да се P_1 увек завршава мета знаком који садржи $\varnothing$ (ово не мора да важи за ниску P_2).

B Нека је P конкатенација ниски P_1 и P_2. Чињеница да се P_1 завршава мета знаком који садржи $\varnothing$ обезбеђује да су два суфикса P, од којих један почиње унутар P_1, а други унутар P_2, увек различита. Од индекса i' мета знака $P[i'] = s[i..i+2]$, одговарајући индекс $i = g(i')$, $i \equiv 1, 2 \pmod 3$, унутар ниске s добија се на следећи начин:

$$i = g(i') \begin{cases} 3i' - 2, & i' \leq |P_1| \\ 3i' - 1 - 3|P_1|, & i' > |P_1| \end{cases} \tag{1}$$

Обрнуто:

$$i' = g^{-1}(i) = \begin{cases} (i+2)/3, & i \equiv 1 \pmod 3 \\ (i+1)/3 + |P_1|, & i \equiv 2 \pmod 3 \end{cases} \tag{2}$$

У табели 2 приказана је ниска P. При томе је на пример

- $P[3] = \text{sip} = s[7..9]$, јер је $7 = g(3) = 3 \cdot 3 - 2$ $(3 \leq |P_1| = 4)$, односно $3 = (7+2)/3$ $(7 \equiv 1 \pmod 3)$;
- $P[6] = \text{iss} = s[5..7]$, јер је $5 = g(6) = 3 \cdot 6 - 1 - 3|P_1|$ $(5 > |P_1| = 4)$, односно $6 = (5+1)/3 + |P_1|$ $(5 \equiv 2 \pmod 3)$.

Запажа се да је лексикографски редослед суфикса ниске P исти као редослед одговарајућих A-суфикса ниске s.

C Сортирати троструким разврставањем и рангирати јединствене мета знаке ниске P (дакле, занемарујући дупликате), рачунајући рангове од 1. Сложеност овог сортирања је $O(n)$. У нашем примеру P има 7 јединствених мета знакова. Њихов сортирани редослед је (i$\varnothing\varnothing$), (ipp), (iss), (iss), (mis), (pi$\varnothing$), (sip), (sis). Рангови ових мета знакова су редом $1, 2, \ldots, 7$. Мета знак (iss) се у нисци P једини појављује два пута.

D Као у табели 2, формира се нова ниску P' кодирањем мета знакова њиховим управо одређеним ранговима. Ако P садржи k јединствених мета знакова, онда је сваки "знак" у P' неки

природни број од 1 до k. Суфиксни низови ниски P и P' су идентични.

E Одредити суфиксни низ $SA_{P'}$ ниске P'. Ако су сви знаци у P' јединствени, онда се суфиксни низ добија директно, јер редослед појединих знакова одређује суфиксни низ. У противном, одредити рекурзивно суфиксни низ P', третирајући рангове у P' као знакове ниске. У табели 2 је приказан суфиксни низ $SA_{P'}$ добијен у нашем примеру. Пошто је број мета знакова у P', а тиме и дужина P', отприлике $2n/3$, рекурзивни проблем је мањи од полазног.

F На основу $SA_{P'}$ и позиција А-суфикса у нисци s, одредити списак рангова (позиција) сортираних А-суфикса у s. Прецизније, за $i = 1, 2, \ldots, |P|$ израчунати на основу (1) индекс $j = g(SA_{P'}[i])$ и ставити $r_j \leftarrow i$. У табели 3 приказани су рангови r_j сортираних А-суфикса $s[i..i+2]$ у нашем примеру. Другим речима, резултат сортирања А-суфикса је

$$S_{11} < S_8 < S_5 < S_2 < S_1 < S_{10} < S_7 < S_4$$

2. Број Б-суфикса је око 1/3 броја свих суфикса. Користећи сортирани редослед А-суфикса, сортирати Б-суфиксе примењујући следећи поступак.

G Проширивши ниску s са два знака $\varnothing\varnothing$ (после чега је дужина ниске $n+2$), посматрајмо суфиксе $s[i..n+2]$ за $i = 1, 2, \ldots, n+2$. Доделити сваком суфиксу $s[i..n+2]$ његов ранг r_i.

- За два специјална знака $\varnothing\varnothing$ ставити $r_{n+1} = r_{n+2} = 0$.
- За А-суфиксе у нисци s рангови су додељени у кораку F.
- Рангови Б-суфикса тренутно нису дефинисани, па за њих ставити $r_i = \square$.

У табели 3 приказани су рангови за ниску $s =$ mississippi и $n = 11$.

H Сортирати Б-суфиксе двоструким разврставањем (сложености $O(n)$), замењујући их (јединственим!) паровима $(s[i], r_{i+1})$. У нашем примеру добија се

$$S_9 < S_6 < S_3,$$

јер је $(p, 6) < (s, 7) < (s, 8)$.

Индекс i' у P, P'	1	2	3	4	5	6	7	8
Индекс $i = g(i')$ у s	1	4	7	10	2	5	8	11
$P[i'] = s[i..i+2]$	(mis)	(sis)	(sip)	(pi∅)	(iss)	(iss)	(ipp)	(i∅∅)
$P'[i']$	4	7	6	5	3	3	2	1
$SA_{P'}[i'] = SA_P[i']$	8	7	6	5	1	4	3	2
Индекс i за који је ранг $r_i = i'$	11	8	5	2	1	10	7	4

Табела 2: Поступак сортирања А-суфикса ниске $s =$ mississippi у првој фази одређивања суфиксног низа.

3. Објединити сортиране низове А-суфикса и Б-суфикса. Нека су S_i и S_j нека два суфикса која треба упоредити у току обједињавања, при чему је S_i А-суфикс, а S_j Б-суфикс. У зависности од тога да ли је $i \mod 3$ једнако 1 или 2, резултат упоређивања суфикса S_i и S_j исти је као резултат упоређивања

   - парова $(s[i], r_{i+1})$ и $(s[j], r_{j+1})$, односно
   - тројки $(s[i], s[i+1], r_{i+2})$ и $(s[j], s[j+1], r_{j+2})$.

   У нашем примеру приликом обједињавања се врши следећи низ упоређивања А- и Б-суфикса:

   - $S_{11} < S_9$ јер је $(i, \varnothing, 0) < (p, p, 1)$; излаз S_{11};
   - $S_8 < S_9$ јер је $(i, p, 6) < (p, p, 1)$; излаз S_8;
   - $S_5 < S_9$ јер је $(i, s, 7) < (p, p, 1)$; излаз S_5;
   - $S_2 < S_9$ јер је $(i, s, 8) < (p, p, 1)$; излаз S_2;
   - $S_1 < S_9$ јер је $(m, 4) < (p, 6)$; излаз S_1;
   - $S_{10} < S_9$ јер је $(p, 1) < (p, 6)$; излаз S_{10};
   - $S_7 > S_9$ јер је $(s, 2) < (p, 6)$; излаз S_9;
   - $S_7 < S_6$ јер је $(s, 2) < (s, 7)$; излаз S_7;
   - $S_4 < S_6$ јер је $(s, 3) < (s, 7)$; излаз S_4;
   - преостала два суфикса S_6, S_3 копирају се у излазни низ.

   Тиме је одређен сортирани редослед свих суфикса ниске s, односно њен суфиксни низ, видети табелу 1. Пошто је сложеност упоређивања $O(1)$, сложеност обједињавања је $O(n)$.

i	1	2	3	4	5	6	7	8	9	10	11	12	13
$s[i]$	m	i	s	s	i	s	s	i	p	p	i	∅	∅
$r_i = i'$	5	4	□	8	3	□	7	2	□	6	1	□	0

Табела 3: Рангови r_1 do r_{n+3} за ниску $s =$ mississippi, $n = 11$.

Сложеност алгоритма задовољава диференцну једначину $T(n) = O(n) + T(\lfloor 2n/3 \rfloor)$, чије је решење на основу мастер теореме $T(n) = O(n)$.

Ако претпоставимо да је време извршавања алгоритма приближно $T(n) = cn$, заменом у диференцној једначини

- $T(n)$ са cn,
- $O(n)$ са an, и
- $T(2n/3)$ са $2cn/3$,

долази се до једначине $cn = an + 2cn/3$, из које следи да је $a = c/3$. Закључује се да рекурзивно сортирање A-суфикса траје приближно око $2/3$ времена потребног за одређивање суфиксног низа комплетне ниске.

3 Одређивање низа LCP на основу суфиксног низа

Низ LCP дате ниске s може се одредити алгоритмом линеарне сложености који је предложило неколико аутора 2001. године [4]. Подсетимо се да је $LCP[i]$ дужина најдужег заједничког префикса $(i-1)$-ог и i-тог лексикографски најмањег суфикса $S_{SA[i-1]}$ и $S_{SA[i]}$ ниске s. По дефиницији је $LCP[1] = 0$. Приликом израчунавања низа LCP користи се низ $rang$, који садржи инверзну пермутацију пермутације SA: ако је $SA[i] = j$, онда је $rang[j] = i$. Другим речима, $rang[SA[i]] = i$ за $i = 1, 2, \ldots, n$. За суфикс S_i број $rang[i]$ је позиција тог суфикса у међу лексикографски сортираним суфиксима.

Пример 3.1. *Видели смо да је суфиксни низ ниске mississippi пермутација*

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 11 & 8 & 5 & 2 & 1 & 10 & 9 & 7 & 4 & 6 & 3 \end{pmatrix}$$

Низ rang је инверзна пермутација ове пермутације.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 5 & 4 & 11 & 9 & 3 & 10 & 8 & 2 & 7 & 6 & 1 \end{pmatrix}$$

На пример, суфикс sissippi је S_4. Његова позиција у сортираном редоследу суфикса је $rang[4] = 9$.

У току рачунања низа LCP потребно је одредити где се у лексикографском редоследу налази суфикс који се од текућег суфикса добија брисањем првог знака. За ово је користан низ $rang$. Посматрајмо i-ти најмањи суфикс, $S_{SA[i]}$. Брисањем његовог првог знака добија се суфикс $S_{SA[i]+1}$, као суфикс који почиње од индекса $SA[i] + 1$ ниске s. Позиција тог суфикса у сортираном редоследу је $rang[SA[i] + 1]$.

Пример 3.2. *Наставак претходног примера. Потребно је за суфикс* $S_4 =$ sissippi *одредити где се налази суфикс issippi (sissippi без првог знака). Суфикс sissippi је на позицији* $rang[4] = 9$ *у суфиксном низу и* $SA[9] = 4$. *Пошто је* $rang[SA[9] + 1] = rang[5] = 3$, *наредни суфикс issippi је на позицији* 3 *у сортираном редоследу.*

Низ LCP може се одредити алгоритмом чији код је приказан на следећој страни.

```
OdredjivanjeLCP(s, SA, n)
// алоцирати простор за низове rang[n] и LCP[n]
for i ← 1 to n do
  rang[SA[i]] ← i      // по дефиницији
LCP[1] ← 0      // по дефиницији
l ← 0      // текући елемент низа
for i ← 0 to n do
  if  rang[i] > 1      // тада се одређује LCP[rang[i]]
    j ← SA[rang[i] − 1]      //S_j је лексикографски претходник S_i
    m ← max{i, j}
    while  m + l < n  and  s[i + l] = s[j + l] do
      l ← l + 1      // наредни знак у заједничком префиксу
    LCP[rang[i]] ← l      // LCP(S_j, S_i)
    if  l > 0
      l ← l − 1      // уклонити први знак у заједничком префиксу S_i, S_j
return  LCP
```

Коректност алгоритма заснована је на наредној леми.

Лема 3.1. *Посматрајмо суфиксе* S_{i-1} *и* S_i, *чије су позиције у лексикографском редоследу суфикса редом* $rang[i-1]$ *и* $rang[i]$. *Ако је* $LCP[rang[i-1]] = l > 1$, *онда за суфикс* S_i, *који се од суфикса* S_{i-1} *добија брисањем првог знака, важи* $LCP[rang[i]] \geq l - 1$.

Доказ. Суфикс S_{i-1} је на позицији $rang[i-1]$ у сортираном редоследу суфикса. Суфикс који му непосредно претходи је у сортираном редоследу на позицији $rang[i-1]-1$, и то је суфикс $S_{rang[i-1]-1}$. По претпоставци је $LCP(S_{SA[rang[i-1]-1]}, S_{i-1}) = l > 1$. Уклањањем првог знака у овим суфиксима добијају се суфикси $S_{SA[rang[i-1]-1]+1}$ и S_i, за које је најдужи заједнички префикс дужине $l-1$.

- Ако суфикс $S_{SA[rang[i-1]-1]+1}$ непосредно претходи суфиксу S_i у сортираном редоследу, доказ је завршен.

- Претпоставимо зато да суфикс $S_{SA[rang[i-1]-1]+1}$ не претходи непосредно суфиксу S_i у сортираном редоследу. Суфикс $S_{SA[rang[i-1]-1]}$ непосредно претходи суфиксу S_{i-1}, и они имају једнаких првих $l > 1$ знакова, па суфикс $S_{SA[rang[i-1]-1]+1}$ мора да буде у сортираном редоследу пре суфикса S_i. Између њих стоји неколико суфикса. Сваки такав суфикс почиње са истих $l-1$ знакова као $S_{SA[rang[i-1]-1]+1}$ и S_i. Према томе, који год од ових суфикса стоји на позицији $rang[i]-1$ непосредно испред S_i, он има бар $l-1$ првих знакова истих као S_i. Према томе, $LCP[rang[i]] \geq l-1$.

□

Објашњење кода алгоритма. У првој **for** петљи се израчунава низ $rang$. Први елемент низа LCP је по дефиницији $LCP[1] = 0$. Друга **for** петља израчунава низ LCP пролазећи суфиксе према опадајућим дужинама.

Инваријанта петље је следеће тврђење:

- У линији $LCP[rang[i]] = l$ одређују се елементи низа LCP са индексима редом $rang[1]$, $rang[2]$, $rang[3]$, ..., $rang[n]$.

- Пре те линије суфикс S_i и суфикс S_j који му непосредно претходи имају најдужи заједнички префикс дужине бар l.

Ово тврђење је тачно у првом проласку кроз **for** петљу, јер је тада $l = 0$. Ако се претпостави да се у линији $LCP[rang[i]] = l$ одређује исправна вредност $LCP[rang[i]]$, онда се та индуктивна претпоставка због декрементирања l (ако је позитивно), одржава на основу доказане леме. Најдужи заједнички префикс суфикса S_i и S_j може бити дужи од вредности l на почетку итерације, па се тачна вредост l одређује у **while** петљи. Индекс m користи се да обезбеди да тест $s[i+l] = s[j+l]$ не изађе ван граница ниске. После изласка из **while** петље вредност l једнака је дужини најдужег заједничког префикса суфикса S_i и S_j.

Пример 3.3. *На примеру ниске mississippi могу се видети три различита случаја на које се може наићи.*

- *У нисци mississippi суфикси* $S_5 = issippi$ $S_2 = ississippi$ *су узастопни у суфиксном низу (трећи и четврти,* $SA[4] = 2$, $SA[3] = 5$*). Дужина њиховог најдужег заједничког префикса је* $LCP[4] = 4$*. Када им се обришу први знаци, добијају се суфикси* $S_6 = ssippi$ $S_3 = ssissippi$ *који су такође узастопни у суфиксном низу (десети и једанаести,* $SA[10] = 6$, $SA[11] = 3$*). Овде је* $LCP[11] = lcp[4]-1$.

- *Суфикси* $S_{10} = pi$ *и* $S_9 = ppi$ *су такође узастопни у суфиксном низу (шести и седми,* $SA[6] = 10$, $SA[7] = 9$*). Дужина њиховог најдужег заједничког префикса је* $LCP[7] = 1$*. Када им се обришу први знаци, добијају се суфикси* $S_{11} = i$ $S_{10} = pi$ *који НИСУ узастопни у суфиксном низу (први и шести,* $SA[1] = 11$, $SA[6] = 10$*). Због тога је* $1 = LCP[2] \geq LCP[7] - 1 = 0$*; запажамо да у овом примеру важи строга неједнакост* $LCP[2] = 1 > LCP[7] - 1 = 0$

- *Суфикси* $S_{11} = i$ *и* $S_8 = ippi$ *су такође узастопни у суфиксном низу (први и други,* $SA[1] = 11$, $SA[2] = 8$*). Дужина њиховог најдужег заједничког префикса је* $LCP[2] = 1$*. Када се суфиксу* $S_{11} = i$ *обрише први знак, добија се празан суфикс, кога нема у нашем сортираном редоследу суфикса. Због тога се вредност* $LCP[2] = 1$ *не може искористити на описани начин за одређивање неке друге вредности* $LCP[.]$.

Низ rang контролише редослед одређивања елемената низа LCP.

- *Полази се од* $LCP[rang[1]] = LCP[5]$*; упоређују се суфикси са индексима* $SA[rang[1]] = 1$ *и* $SA[rang[1] - 1] = 2$*, тј. суфикси mississippi и ississippi. Први знаци ова два суфикса се разликују, па је* $LCP[5] = 0$.

- *Даље се редом одређују вредности* $LCP[rang[i]]$, $i = 2, 3, \ldots, 10$, *користећи као почетну вредност индекса за упоређивање* $l + 1 = LCP[rang[i] - 1] + 1$.

- *На пример, пошто је установљено да је* $LCP[rang[2]] = LCP[4] = 4$*, прелази се на одређивање* $LCP[rang[3]] = LCP[11] \geq 4 - 1 = 3$*. Упоређују се, почевши од индекса* $l + 1 = 4$ *суфикси ssissippi и ssippi са индексима редом* $SA[rang[3]] = SA[11] = 3$ *и* $SA[rang[3] - 1] = SA[10] = 6$*; упоређујући (различите) четврте знакове ова два суфикса, закључује се да је* $LCP[11] = 3$.

Сложеност алгоритма. Почетна вредност l је 0; l ни у једном тренутку не прелази n. Пошто је број декрементирања l највише $n-1$, број инкрементирања l унутар **while** петље је највише $2n$. Према томе, сложеност алгоритма је $O(n)$.

4 Примене суфиксног низа

Многе операције са нискама ефикасно се извода ако се претходно формира њихов суфиксни низ и низ LCP. Приказаћемо два таква примера примене: тражење речи у тексту и тражење најдуже заједничке подниске за две дате ниске. На основу суфиксног низа и низа LCP може се алгоритмом линеарне сложености формирати суфиксно стабло ниске, што затим омогућује сличне примене суфиксног стабла.

4.1 Тражење речи у тексту

Потребно је одредити све појаве речи P унутар текста S. Проблем је лако решити ако се одреди суфиксни низ текста S: свака појава P унутар S је почетак неког префикса S. Суфиксни низ текста S одређује сортирани редослед суфикса S, па се прва појава P унутар S може пронаћи бинарном претрагом опсега $[1..|S|]$. Индекси свих осталих појава (ако их има) налазе се на наредним узастопним позицијама у суфиксном низу.

Сложеност алгоритма је $O(n \log n)$, јер је сложеност формирања суфиксног низа $O(n)$, а сложеност сваког од $O(\log n)$ упоређивања је $O(n)$. Упоређивања се могу **извршити ефикасније** ако се користи низ LCP текста S.

Пример 4.1. *Пронаћи све појаве речи $P = lednik$ унутар текста $s = prestolonaslednikovica$. Суфиксни низ текста s (са сортираним редоследом суфикса) приказан је у следећој табели. Бинарна претрага започиње упоређивањем $P = lednik$ са $S[SA[\lceil \frac{1+22}{2} \rceil]] = S[SA[12]] = naslednikovica$. Пошто је $naslednikovica > lednik$, наставља се са бинарном претрагом опсега $[1, 11]$. После неколико корака проналази се да је индекс прве појаве P унутар s једнак 12. Пошто наредни суфикс $s[13] = ednikovica$ не започиње са P, унутар s нема других појава P.*

i	S_i	i	$SA[i]$	$S_SA[i]$	$LCP[i]$
1	prestolonaslednikovica	1	22	a	1
2	restolonaslednikovica	2	10	aslednikovica	0
3	estolonaslednikovica	3	21	ca	0
4	stolonaslednikovica	4	14	dnikovica	0
5	tolonaslednikovica	5	13	ednikovica	0
6	olonaslednikovica	6	3	estolonaslednikovica	0
7	lonaslednikovica	7	20	ica	1
8	onaslednikovica	8	16	ikovica	0
9	naslednikovica	9	17	kovica	0
10	aslednikovica	10	12	**lednik**ovica	1
11	slednikovica	11	7	lonaslednikovica	0
12	lednikovica	12	9	naslednikovica	1
13	ednikovica	13	15	nikovica	0
14	dnikovica	14	6	olonaslednikovica	1
15	nikovica	15	8	onaslednikovica	1
16	ikovica	16	18	ovica	0
17	kovica	17	1	prestolonaslednikovica	0
18	ovica	18	2	restolonaslednikovica	0
19	vica	19	11	slednikovica	1
20	ica	20	4	stolonaslednikovica	0
21	ca	21	5	tolonaslednikovica	0
22	a	22	19	vica	

4.2 Тражење најдуже заједничке подниске за две дате ниске

Потребно је одредити најдужу заједничку подниску две дате ниске s_1 и s_2. Проблем се решава тако што се најпре формира суфиксни низ ниске $s = s_1\$s_2\#$, при чему су знаци \$ и # знаци који се не појављују унутар s_1 или s_2. При томе се сваком суфиксу претходно придружује податак да ли је његов почетак унутар s_1 или s_2. У сортираном низу суфикса ове ниске потребно је пронаћи пар узастопних чланова таквих да

- одговарајући суфикси не припадају истом низу, и
- дужина њиховог најдужег заједничког префикса је **највећа у односу** на све остале овакве парове индекса (оваквих парова префикса са најдужим заједничким префиксом може бити више).

Ако се користи и низ LCP здружене ниске s (сложеност његовог формирања је $O(n)$; овде је $n = |s_1| + |s_2|$), онда је сложеност алгоритма линеарна, $O(n)$.

Пример 4.2. *Пронаћи најдужу заједничку подниску ниски*

$$s_1 = prestolonaslednikovica$$

и

$$s_2 = kolonizacija.$$

Суфиксни низ SA здружене ниске $s = s_1\$s_2\#$ (са сортираним редоследом суфикса) приказан је у табели на следећој страни. Највећи број у колони $LCP[i]$ је $LCP[26] = 4$. Поред тога, суфикси

$$s[SA[26]] = olonaslednikovica\$kolonizacija\#$$

и

$$s[SA[27]]olonizacija\#$$

са заједничким префиксом olon дужине четири потичу из различитих ниски. Према томе, најдужа заједничка подниска ове две ниске је ниска olon.

4.3 Проналажење најдужег палиндрома у задатој нисци

Нека је задата ниска s и нека је потребно пронаћи најдужи палиндром унутар ње. Решавање овог проблема своди се на тражење најдуже заједничке подниске ниски s и s', где је s' ниска која се од s добија "читањем уназад".

4.4 Формирање суфиксног стабла

Ако се зна суфиксни низ SA и LCP низ LCP ниске s дужине n, суфиксно стабло те ниске може се конструисати алгоритмом сложености $O(n)$ на основу следеће идеје: полазећи од парцијалног суфиксног стабла за лексикографски најмањи суфикс, уметати у њега остале суфиксе редом којим се на њих наилази лек-сикографским обиласком суфиксног стабла.

	S_i	i	низ	$SA[i]$	$s[SA[i]]$	$LCP[i]$
1	prestolonaslednikovica$ko...#	1	2	37		0
1	restolonaslednikovica$ko...#	2	2	36	#	0
1	estolonaslednikovica$ko...#	3	1	23	$ko...#	0
1	stolonaslednikovica$ko...#	4	2	35	a#	1
1	tolonaslednikovica$ko...#	5	1	22	a$ko...#	1
1	olonaslednikovica$ko...#	6	2	31	acija#	1
1	lonaslednikovica$ko...#	7	1	10	aslednikovica$ko...#	0
1	onaslednikovica$ko...#	8	1	21	ca$ko...#	1
1	naslednikovica$ko...#	9	2	32	cija#	0
1	aslednikovica$ko...#	10	1	14	dnikovica$ko...#	0
1	slednikovica$ko...#	11	1	13	ednikovica$ko...#	1
1	lednikovica$ko...#	12	1	3	estolonaslednikovica$ko...#	0
1	ednikovica$ko...#	13	1	20	ica$ko...#	1
1	dnikovica$ko...#	14	2	33	ija#	1
1	nikovica$ko...#	15	1	16	ikovica$ko...#	1
1	ikovica$ko...#	16	2	29	izacija#	0
1	kovica$ko...#	17	2	34	ja#	0
1	ovica$ko...#	18	2	24	kolonizacija#	2
1	vica$ko...#	19	1	17	kovica$ko...#	0
1	ica$ko...#	20	1	12	lednikovica$ko...#	1
1	ca$ko...#	21	1	7	lonaslednikovica$ko...#	3
1	a$ko...#	22	2	26	lonizacija#	0
1	$ko...#	23	1	9	naslednikovica$ko...#	1
2	kolonizacija#	24	1	15	nikovica$ko...#	2
2	olonizacija#	25	2	28	nizacija#	0
2	lonizacija#	26	**1**	6	**olon**aslednikovica$ko...#	**4**
2	onizacija#	27	**2**	25	**olon**izacija#	1
2	nizacija#	28	1	8	onaslednikovica$ko...#	2
2	izacija#S	29	2	27	onizacija#	1
2	zacija#	30	1	18	ovica$ko...#	0
2	acija#	31	1	1	prestolonaslednikovica$ko...#	0
2	cija#	32	1	2	restolonaslednikovica$ko...#	0
2	ija#	33	1	11	slednikovica$ko...#	1
2	ja#	34	1	4	stolonaslednikovica$ko...#	0
2	a#	35	1	5	tolonaslednikovica$ko...#	0
2	#	36	1	19	vica$ko...#	0
2		37	2	30	zacija#	

Нека је ST_i парцијално суфиксно стабло добијено уметањем првих i лексикографски најмањих суфикса, $0 \leq i \leq n$. Нека је даље $d(v)$ дужина конкатенације ознака свих грана у стаблу ST_i на путу од корена до чвора v. Полази се од ST_0, стабла које има само корен. Да би се у стабло ST_i убацио суфикс са индексом $SA[i]$, $1 \leq i \leq n$, прелази се пут од последњег уметнутог листа ка

корену, до првог чвора v за који је $d(v) \leq LCP[i]$. Могућа су два случаја:

- $d(v) = LCP[i]$ **[уметање нове гране из постојећег чвора]**: тада је дужина конкатенације ознака грана на путу од корена до чвора v једнака $LCP[S_{SA[i-1]}, S_{SA[i]} = LCP[i-1]$. У том случају суфикс са ознаком $SA[i]$ се умеће као нови лист x повезан са чвором v граном (v, x) са ознаком $S_{SA[i]+LCP[i]}$. Другим речима, ознака додате гране се састоји од преосталих знакова суфикса $SA[i]$ који нису део конкатенације ознака грана на путу од корена до чвора v. На тај начин формирано је парцијално суфиксно стабло ST_i.

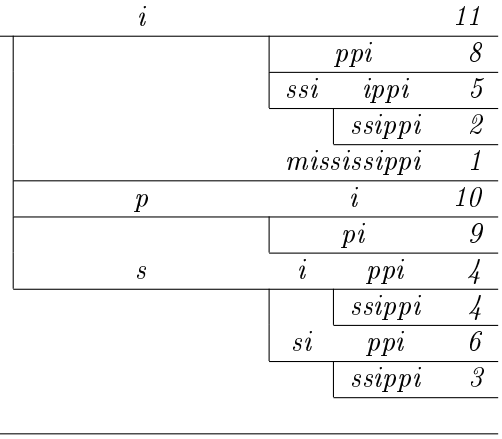
- $d(v) < LCP[i]$ **[нова грана из чвора уметнутог у постојећу грану]**: тада је конкатенација ознака грана на путу од корена до чвора v има мање знакова од $LCP[S_{SA[i-1]}, S_{SA[i]}] = LCP[i-1]$ и знакови који недостају до чвора v садржани су на почетку ознаке *најдеснијe* (последње уметнуте) гране (v, w) која излази из чвора v. Због тога се грана (v, w) мора новим унутрашњим чвором y раздвојити на две гране (v, y) и (y, w). При томе гране (v, y) и (y, w) као ознаке добијају одговарајући почетак, односно завршетак ознаке уклоњене гране (v, w). Прецизније,

  1. Уклонити грану (v, w).
  2. Додати нови унутрашњи чвор y и нову грану (v, y) са ознаком $s[SA[i-1]] + d(v), SA[i-1] + LCP[i] - 1]$. Ознака нове гране састоји се од знакова који недостају на најдужем заједничком префиксу суфикса $S_{SA[i-1]}$ и $S_{SA[i]}$. Према томе, конкатенације ознака грана на путу од корена до чвора y једнака је најдужем заједничком префиксу суфикса $S_{SA[i-1]}$ и $S_{SA[i]}$.
  3. Повезати чвор w са новокреираним унутрашњим чвором y граном (y, w) са ознаком $s[SA[i-1]] + LCP[i], SA[i-1] + d(v) - 1]$. Нова ознака састоји се од преосталих знакова обрисане гране (v, w) који нису укључени у ознаку гране (v, y).
  4. Уметнути чвор са ознаком $SA[i]$ као нови лист x и повезати га са новим унутрашњим чвором y граном (y, x) са ознаком $S_{[SA[i]]+LCP[i]}$. Према томе, ознака гране (y, x) састоји се од преосталих знакова суфикса $S_{SA[i]}$, који нису укључени у конкатенацију ознака грана на путу од корена до чвора v.
  5. На тај начин формирано је парцијално суфиксно стабло ST_i.

Пример 4.3. *На следећој слици илустрован је овај поступак формирања суфиксног стабла за ниску mississippi. У опису алгоритма претпоставља се да се суфиксно стабло формира слева удесно, тј. да се нови листови додају повезивањем са чвором на тренутно најдеснијој грани. Због једноставнијег цртања стабла, овде се нови листови прикључују стаблу испод најниже гране*

уместо десно од најдеснијe гране. Другим речима, суфиксно стабло се формира одозго наниже.

i	$SA[i]$	$s[SA[i]..n]$	$LCP[i]$
1	11	i	1
2	8	ippi	1
3	5	issippi	4
4	2	ississippi	0
5	1	mississippi	0
6	10	pi	1
7	9	ppi	0
8	7	sippi	2
9	4	sissippi	1
10	6	ssippi	3
11	3	ssissippi	



Лако је видети да је амортизована сложеност овог алгоритма $O(n)$. Чворови који се пролазе у кораку i на путу ка корену најдеснијим путем у стаблу ST_i (осим последњег чвора v) уклањају се из најдеснијег пута кад се $A[i]$ дода у стабло као нови лист. Ови чворови се никад не пролазе поново у наредним корацима $j > i$. Према томе, укупан број чворова који се пролазе у току извршења алгоритма је највише $2n$.

5 Примене суфиксног стабла

Исти проблеми sa које смо видели да се могу решавати применом суфиксног низа (тражење речи у тексту, тражење најдуже заједничке подниске две или више ниски, тражење најдужег палиндрома у нисци) могу се још једноставније решавати применом суфиксног стабла.

Литература

[1] J. Kärkkäinen, P. Sanders, Simple Linear Work Suffix Array Construction, *Lecture Notes in Computer Science* 2719: 943–955.

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms, 4. издање, The MIT Press, 2022.

[3] D. Gusfield, Algorithms on Strings, Trees, and Sequences, Cambridge University Press, 1997.

[4] T. Kasai, G. Lee, H. Arimura, S. Arikawa, K. Park, Linear-Time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications. У

зборнику A. Amir (eds) Combinatorial Pattern Matching CPM 2001. *Lecture Notes in Computer Science* 2089. Springer, 2001.