

Algoritmi in podatkovne strukture 2 — drugi izpitni rok
30. junij 2026

Ime: _____

Priimek: _____

Vpisna številka: _____

1. naloga: _____ od 20 (6 + 7 + 7)

2. naloga: _____ od 20 (5 + 5 + 5 + 5)

3. naloga: _____ od 20 (7 + 6 + 7)

4. naloga: _____ od 20 (6 + 7 + 7)

Skupaj: _____ od 80

Za reševanje imate na voljo 100 minut časa. Vse odgovore obvezno utemeljite!

- ① Podan je enosmerno povezan seznam $S = \langle x_1, x_2, \dots, x_n \rangle$, kjer so vsi elementi med seboj različni. Za vsak element je podana verjetnost iskanja tega elementa ($p(x_k)$ za $k \in \{1, \dots, n\}$). Definirajmo

$$C(S) = \sum_{k=1}^n kp(x_k).$$

- (a) Kolikšen je $C(S)$ za seznam S z n elementi, kjer so vse verjetnosti iskanja enake $1/n$?
- (b) Dokažite, da minimalno vrednost $C(\cdot)$ dosežemo tako, da elemente v seznamu uredimo po padajočih verjetnostih iskanja. (Namig: spomnite se na frizerja z vaj.)
- (c) Recimo, da seznam uredimo s sledečim algoritmom:

```
function UREDI( $S, l, r$ )           ▷ Vrne urejeno kopijo podseznama  $S[l : r]$ .
  if  $r > l$  then
     $m \leftarrow \lfloor (l + r) / 2 \rfloor$ 
     $L \leftarrow \text{URED}I(S, l, m)$ 
     $R \leftarrow \text{URED}I(S, m + 1, r)$ 
    ▷ Izdelamo urejen seznam, sestavljen iz elementov seznamov  $L$  in  $R$ . V obeh seznamih pričnemo s prvim elementom. V vsaki iteraciji nato vzamemo večjega od trenutno obravnavanih elementov seznamov  $L$  in  $R$ , ga dodamo na konec izhodnega seznama in se premaknemo na naslednji element seznama  $L$  (če smo element vzeli iz  $L$ ) oziroma  $R$  (če smo element vzeli iz  $R$ ). Postopek ponavljamo, dokler pri obeh vhodnih seznamih ne prispemo do konca.
     $T \leftarrow \text{ZLIJ}(L, R)$ 
  else
     $T \leftarrow$  nov seznam z elementom  $S[l]$ 
  return  $T$ 
```

```
function UREDI( $S$ )           ▷ Vrne urejeno kopijo seznama  $S$ .
   $n \leftarrow |S|$ 
  if  $n \geq 2$  then
     $m \leftarrow \lfloor n / 2 \rfloor$ 
     $L \leftarrow \text{URED}I(S[1 : m])$ 
     $R \leftarrow \text{URED}I(S[m + 1 : n])$ 
    ▷ Z vzporednim sprehtodom zlij seznama  $L$  in  $R$  v nov seznam in ga vrni.
     $T \leftarrow \text{ZLIJ}(L, R)$ 
  return  $T$ 
else
  return  $S$ 
```

Ker so seznam drugačni od tabel, je časovna zahtevnost tega algoritma gotovo $\Omega(n^2)$. Imam prav?

Rešitve

- (a) Mačji kašelj:

$$\begin{aligned}C(S) &= 1 \cdot \frac{1}{n} + 2 \cdot \frac{1}{n} + \dots n \cdot \frac{1}{n} \\&= \frac{1}{n} \sum_{i=1}^n i \\&= \frac{1}{n} \frac{n(n+1)}{2} \\&= \frac{n+1}{2}\end{aligned}$$

- (b) Uporabimo običajen trik pri požrešnih algoritmihi: predpostavimo, da optimalen nek drug, nepožrešen seznam $S = \langle y_1, \dots, y_n \rangle$. Naj bo $i < j$ in $p(y_i) < p(y_j)$; če seznam ni požrešen, potem tak par gotovo obstaja. Elementa y_i in y_j zamenjamo med seboj, tako da dobimo seznam S' . Računajmo:

$$\begin{aligned}C(S') &= C(S) - ip(y_i) - jp(y_j) + jp(y_i) + ip(y_j) \\&= C(S) - p(y_i)(i - j) + p(y_j)(i - j) \\&= C(S) - (i - j)(p(y_i) - p(y_j)) \\&< C(S),\end{aligned}$$

saj je $i - j < 0$ in $p(y_i) - p(y_j) < 0$. Skratka, z zamenjavo elementov, ki tvorita inverzijo, smo pridelali seznam z manjšo vrednostjo $C(\cdot)$, kar je v nasprotju s predpostavko, da je seznam S v tem smislu optimalen.

- (c) Ne, Fürst, biksaš ga: problem razdeliš na dva problema enake velikosti, za zli-vanje že urejenih seznamov pa potrebuješ $O(n)$ časa — nič več kot za tabele. Rekurenčna enačba se torej glasi

$$T(n) = 2T(n/2) + \Theta(n).$$

Po krovnem izreku je $a = 2$, $b = 2$ in $d = 1$. Ker je $a = b^d$, je časovna zahtevnost enaka $T(n) = \Theta(n \log n)$.

- ② Na n zaporednih sedežev bi radi razporedili $k \leq \lceil n/2 \rceil$ oseb, tako da je med zaporednima osebama vsaj en prost sedež. Število možnih razporeditev označimo z $R(n, k)$. Na primer, $R(7, 0) = 1$, $R(7, 1) = 7$, $R(7, 2) = 15$, $R(7, 3) = 10$ in $R(7, 4) = 1$.
- Zapišite rekurenčno enačbo za $R(n, k)$. (Namig: kateri dve možnosti imamo pri obravnavi prvega sedeža?)
 - Zapišite algoritem za računanje $R(n, k)$, ki temelji na dinamičnem programiranju po metodi od spodaj navzgor.
 - Kolikšna je časovna in kolikšna prostorska zahtevnost algoritma iz prejšnje točke?
 - Dokažite, da je časovna zahtevnost algoritma, ki $R(n, k)$ računa naivno po enačbi, enaka $\Omega\left(\binom{n-1}{k}\right)$.

Rešitve

- Za prvi sedež imamo dve možnosti: (1) nanj posadimo eno od oseb; (2) nanj ne posadimo nobene osebe. Če izberemo možnost (1), rešimo podproblem za $k - 1$ oseb in $n - 2$ sedežev (saj na drugi sedež ne smemo posaditi nikogar), v nasprotnem primeru pa rešimo podproblem za k oseb in $n - 1$ sedežev. Če upoštevamo še robne primere, dobimo

$$R(n, k) = \begin{cases} 0 & \text{v primeru } k > \lceil n/2 \rceil; \\ 1 & \text{v primeru } k = 0; \\ R(n - 2, k - 1) + R(n - 1, k) & \text{sicer.} \end{cases}$$

- function** RAZPOREDITVE(n, k)
inicializiraj tabelo $D[0 : n, 0 : k]$
for $i \leftarrow 0$ **to** n **do**
 $D[i, 0] \leftarrow 1$
 $D[1, 1] \leftarrow 1$
 for $i \leftarrow 2$ **to** n **do**
 for $j \leftarrow 1$ **to** $\min(k, \lceil i/2 \rceil)$ **do**
 $D[i, j] \leftarrow D[i - 2, j - 1] + D[i - 1, j]$
 return $D[n, k]$

- Tako časovna kot prostorska zahtevnost je enaka $\Theta(nk)$.
- Naš R je definiran rekurenčno, binomski simbol pa prav tako:

$$\binom{n-1}{k} = \binom{n-2}{k-1} + \binom{n-2}{k},$$

če zanemarimo robne primere, ki pri določanju računske zahtevnosti praviloma niso bistveni.

Induktivno predpostavimo, da velja $R(n', k') = \Omega\left(\binom{n'-1}{k'}\right)$ za vse $n' < n$ in $k' \leq k$. To lahko zapišemo kot $R(n', k') \geq c \binom{n'-1}{k'}$ za nek $c > 0$ in od nekega n'_0 in k'_0 naprej.

Ker funkcija $R(n, k)$ pri fiksnem k monotonno narašča v odvisnosti od n , velja

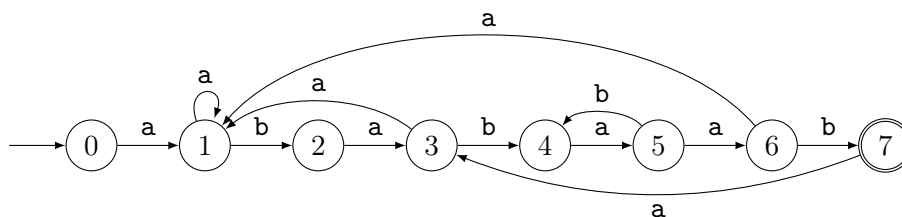
$$\begin{aligned} R(n, k) &= R(n - 2, k - 1) + R(n - 1, k) \\ &\geq R(n - 2, k - 1) + R(n - 2, k) \end{aligned}$$

$$\begin{aligned}
&\geq c\binom{n-2}{k-1} + c\binom{n-2}{k} \\
&= c\left(\binom{n-2}{k-1} + \binom{n-2}{k}\right) \\
&= c\binom{n-1}{k} \\
&= \Omega\left(\binom{n-1}{k}\right).
\end{aligned}$$

- ③ Naj bo $A(P)$ deterministični končni avtomat za iskanje vzorca P .
- Narišite (ali kako drugače zapišite) $A(\text{ababaab})$. Da bo slika preglednejša, izpušтите povezave, ki vodijo v stanje 0.
 - Formalno zapišite ali opišite množico nizov, ki avtomat $A(P)$ privedejo v končno stanje (zanimá nas torej jezik avtomata $A(P)$ za splošen P).
 - Če je $\delta(s, c) = t$ za $t < s$ in $c \in \Sigma$, potem zanesljivo vemo, da je $Z(i) = j$ (gre za Z -funkcijo, ki smo jo spoznali na vajah) za nek i in j . Kolikšna sta i in j ?

Rešitve

(a)

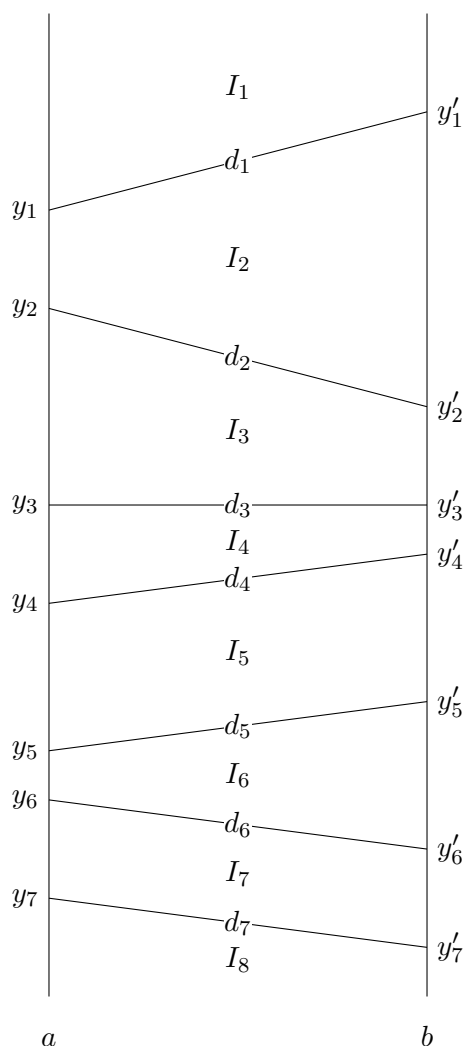


- (b) Jezik avtomata $A(P)$ je množica vseh nizov (besedil), pri katerih je zadnjih $|P|$ znakov enakih P :

$$L(A) = \{T \in \Sigma^* \mid P \sqsubseteq T\}$$

- (c) Ker je $t < s$, vemo, da c ni enak $P[s+1]$, saj bi sicer veljalo $\delta(s, c) = s+1$. Po definiciji funkcije δ pa velja $P[:t] \sqsupseteq P[:s]c$ oziroma $P[1:t] = P[s-t+2:s]c$. Zaradi $c \neq P[s+1]$ velja $P[1:t] \neq P[s-t+2:s+1]$, vendar pa je $P[1:t-1] = P[s-t+2:s]$. To pomeni, da je $Z(s-t+2) = t-1$ oziroma $i = s-t+2$ in $j = t-1$.

- ④ Med navpičnima premicama $x = a$ in $x = b$ ($a < b$) je razpetih n daljic; i -ta med njimi ima levo krajišče v točki (a, y_i) , desno pa v točki (b, y'_i) . Noben par daljic se med seboj ne seka. To pomeni, da teh n daljic razdeli pas $[a, b] \times (-\infty, \infty)$ na $n + 1$ intervalov ($I_1, \dots, I_{n+1}$). Sledeča slika prikazuje primer za $n = 7$:



Za točko $T(x, y)$, kjer je $x \in [a, b]$, bi radi ugotovili, v katerem od teh $n + 1$ intervalov se nahaja. Zaradi enostavnosti predpostavimo, da se točka ne nahaja na nobeni daljici, ampak samo strogo znotraj enega od intervalov.

- Kako bi za točko $T(x, y)$ ($x \in [a, b]$) ugotovili, ali se nahaja nad daljico med točkama $A(a, y_l)$ in $B(b, y_r)$? Pogoji zapišite v obliki matematičnega izraza.
- Interval, v katerem leži točka T , lahko učinkovito poiščemo, če pred tem zgradimo uravnoreženo dvojiško iskalno drevo. Narišite drevo za primer na gornji sliki in na kratko opišite postopek določitve iskanega intervala z uporabo takšnega drevesa.
- Opišite postopek gradnje takšnega drevesa. Kolikšna je časovna zahtevnost gradnje?

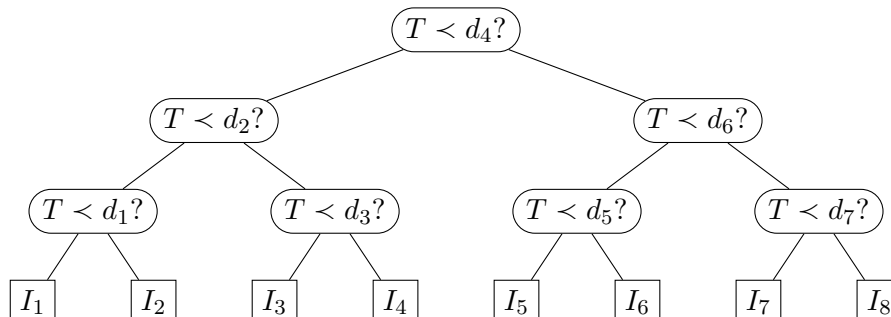
Rešitve

- Točka T se nahaja nad daljico AB natanko tedaj, ko vektorja $\overrightarrow{AB}$ in $\overrightarrow{AT}$ tvorita levi zasuk, torej ko je $\overrightarrow{AB} \times \overrightarrow{AT} > 0$. Ker je $\overrightarrow{AB} = (b - a, y_r - y_l)$ in $\overrightarrow{AT} =$

$(x - a, y - y_l)$, lahko iskani pogoj zapišemo kot

$$\overrightarrow{AB} \times \overrightarrow{AT} = (b - a)(y - y_l) - (x - a)(y_r - y_l) > 0.$$

- (b) Naj zapis $T < d$ pomeni, da točka T leži nad daljico d .



Iskanje po takšnem drevesu je enostavno, saj se zgolj premikamo od korena proti listom v skladu z odgovori na vprašanja. Ko prispemo do lista, nam oznaka v listu razkrije iskani interval.

- (c) Daljice uredimo po y -koordinatah njihovih središčih točk, za kar potrebujemo $O(n \log n)$ časa. Naj bodo $d_1, \dots, d_n$ daljice, urejene na opisani način.

V primeru $n = 0$ je drevo sestavljeno iz enega samega lista, ki predstavlja edini interval, v nasprotnem primeru pa v koren postavimo vprašanje $T < d_{\lfloor n/2 \rfloor}$, nato pa rekurzivno zgradimo levo poddrevo za daljice $d_1, \dots, d_{\lfloor n/2 \rfloor - 1}$ in desno poddrevo za daljice $d_{\lfloor n/2 \rfloor + 1}, \dots, d_n$. Gradnja torej traja

$$T(n) = O(n \log n) + U(n)$$

časa, kjer je

$$U(n) = 2U(n/2) + \Theta(1).$$

Rešitev spodnje rekurenčne enačbe je $U(n) = \Theta(n)$, tako da je $T(n) = O(n \log n)$.

Če daljic ne bi vnaprej uredili, bi v rekurenčni enačbi namesto $\Theta(1)$ imeli $\Theta(n)$ (mediano bi potem morali poiskati z algoritmom hitrega iskanja) in bi vnovič prišli do $O(n \log n)$.