Algoritmi in podatkovne strukture 2 — tretji izpitni rok
17. avgust 2026

Ime: _____

Priimek: _____

Vpisna številka: _____

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

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

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

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

Skupaj: _____ od 80

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

(1) V tabelo A, ki ima dovolj prostora, da nam je nikoli ne bo treba »raztegniti«, na zaporedne indekse (1, 2, 3, ...) shranjujemo medsebojno različne elemente. Kadarkoli število elementov v tabeli doseže 2^k (za $k \geqslant 1$), s sledečim algoritmom uredimo celotno tabelo:

```
function UREDI(A, n)
    ▷ n je število elementov v tabeli A, indeksi pa se pričnejo z 1 ◁
    for i ← 2 to n do
        j ← i
        while j ⩾ 2 ∧ A[j] < A[j − 1] do
            zamenjaj A[j] in A[j − 1] med seboj
            j ← j − 1
```

Dogovorimo se, da ima vpis novega elementa v tabelo ($A[i] \leftarrow x$) ceno 1. Medsebojna zamenjava dveh elementov tabele naj ima tudi ceno 1, vse ostale elementarne operacije pa so brezplačne. Potencial tabele definirajmo kot število inverzij, torej kot število parov (i, j), kjer je $i < j$ in $A[i] > A[j]$.

(a) Največ za koliko se lahko poveča potencial tabele z n elementi ob dodajanju novega elementa, če število elementov po tej operaciji (torej $n + 1$) *ni* potenca števila 2?

(b) Kolikšen je maksimalni možni potencial tabele z $n = 2^k$ elementi ($k \geqslant 1$) tik pred urejanjem? Upoštevajte, da je prva polovica tabele že urejena!

(c) Dokažite, da je — glede na potencial, ki smo ga definirali — računovodska cena dodajanja elementa enaka $O(n)$. Upoštevati morate obe možnosti: da število elementov po tej operaciji ni potenca števila 2 in da je potenca števila 2. (Namig: uporabite formulo $\hat{c} = c + \Delta\Phi$.)

Rešitve

(a) Če na konec tabele dodamo element, ki je manjši od vseh obstoječih n elementov, bomo potencial povečali za n, saj smo uvedli toliko novih inverzij.

(b) Prva polovica ne vsebuje inverzij, druga pa vsebuje kvečjemu $\frac{1}{2}(n/2)(n/2-1) = \frac{1}{8}n^2 - \frac{1}{4}n$ inverzij. Če prištejemo še $(\frac{n}{2})^2 = \frac{1}{4}n^2$ inverzij, ki jih tvori prva polovica tabele z drugo polovico, dobimo skupno $\frac{3}{8}n^2 - \frac{1}{4}n$ inverzij. Toliko torej znaša maksimalni možni potencial.

(c) V skladu z namigom uporabimo formulo $\hat{c} = c + \Delta\Phi$. Če dolžina tabele po dodajanju elementa ni potenca števila 2, je $c = 1$ in $\Delta\Phi \leqslant n$, od koder sledi $\hat{c} = O(n)$. Če pa je dolžina tabele po dodajanju enaka $n = 2^k$, je dejanska cena enaka $1 + z$, kjer je z število zamenjav, ki jih opravi algoritem urejanja. No, število zamenjav pa je natanko enako številu inverzij pred urejanjem, zato se bo potencial zmanjšal ravno za z. Računovodska cena je v tem primeru torej 1, to pa je seveda prav tako $O(n)$.

(2) Osebe 1, 2, ..., n bi radi razporedili v ravni vrsti, in to tako, da bomo minimizirali število sovražnosti, torej število parov sosedov, ki se med seboj ne marajo. Sovražni pari so podani v simetrični dvojiški matriki S velikosti $n \times n$ z ničelno diagonalo. Naj bo $S[i, j] = 1$ natanko tedaj, ko se osebi i in j med seboj ne marata.

Na primer, pri $n = 4$ in

$$S = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

je optimalna razporeditev $\langle 2, 3, 1, 4 \rangle$ z eno samo sovražnostjo (to je par $(2, 3)$).

(a) Koliko časa potrebujemo, če se problema lotimo tako, da preizkusimo vsako možno razporeditev in izberemo optimalno?

(b) Naj bo $f(M, k)$ število sovražnosti v optimalni razporeditvi, ki zajema množico oseb $M \subseteq \{1, \ldots, n\}$ in se prične z osebo $k \in M$. Zapišite rekurenčno enačbo za izračun $f(M, k)$.

(c) Vrednost $f(M, k)$ bi radi izračunali s pomočjo dinamičnega programiranja. Koliko časa in koliko prostora potrebujemo za to? Kaj predstavlja element $D[i, j]$ v memoizacijski tabeli?

Rešitve

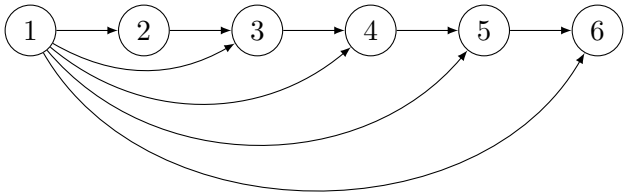
(a) Odgovor je na dlani: $\Theta(n)$ za vsako od $n!$ možnih razporeditev, kar nanese $\Theta(n!\, n)$.

(b) Za soseda osebe k izberemo eno od oseb v množici M, nato pa problem rekurzivno rešimo za preostale osebe. Katero osebo izberemo? Tisto, ki minimizira skupno vsoto sovražnosti, kakopak!

$$f(M, k) = \begin{cases} 0 & \text{v primeru } M = \varnothing; \\ \min_{a \in M \setminus \{k\}} (S[k, a] + f(M \setminus \{k\}, a)) & \text{sicer.} \end{cases}$$

(c) Potrebujemo $\Theta(2^n n)$ prostora za vzdrževanje memoizacijske tabele D. Ker izračun vsakega njenega elementa zahteva $\Theta(n)$ časa, je skupna časovna zahtevnost $\Theta(2^n n^2)$. Element $D[i, j]$ predstavlja vrednost $f(M, j)$, kjer je M množica, pri kateri je številska vrednost karakterističnega vektorja (če ga gledamo kot dvojiško število) enaka i.

(3) Podana je družina usmerjenih grafov na vozliščih 1, 2, ..., n ter povezavah $(1,2)$, $(1,3)$, ..., $(1,n)$ in $(2,3)$, $(3,4)$, ..., $(n-1,n)$. Na primer, graf za $n = 6$ izgleda takole:



Naj bo vozlišče 1 izvor, vozlišče n pa ponor. Kapacitete povezav naj bodo nenegativna cela števila (vrednost 0 je torej tudi možna). Za vsako od časovnih zahtevnosti

(a) $T(n) = O(1)$;

(b) $T(n) = \Theta(n)$;

(c) $T(n) = \Theta(n^2)$;

(d) $T(n) = \Omega(n^3)$;

določite kapacitete povezav tako, da bo Edmonds-Karpov algoritem na opisani družini grafov potreboval $T(n)$ časa, ali pa pokažite, da takšne časovne zahtevnosti ni mogoče doseči.

Rešitve

(a) Časovne zahtevnosti $O(1)$ ni mogoče doseči, tudi če so vse kapacitete enake 0. Edmonds-Karpov algoritem bo pregledal vse sosede vozlišča 1 in že za to bo porabil $\Theta(n)$ časa.

(b) Kapacitete povezav $(1,2)$, $(2,3)$, ..., $(n-1,n)$ nastavimo na 1, vse ostale pa na 0. Algoritem bo v času $\Theta(n)$ našel in zasitil edino nezasičeno pot.

(c) Kapacitete povezav $(i, i+1)$ za $i \in \{1, \ldots, n-1\}$ nastavimo na i, vse ostale pa na 1. Algoritem bo najprej našel in zasitil pot $1 \to n$, nato pot $1 \to n-1 \to n$, nato pot $1 \to n-2 \to n-1 \to n$ itd., nazadnje pa bo našel in zasitil pot $1 \to 2 \to 3 \to \ldots \to n$. Ker algoritem za odkritje in zasičenje i-te poti porabi $\Theta(i)$ časa, je skupna časovna zahtevnost $\Theta(n^2)$.

(d) Edmonds-Karpov algoritem porabi $O(VE^2)$ časa. Ker je v našem primeru $|V| = n$ in $|E| = \Theta(n)$, je teoretična zgornja meja $O(n^3)$. Vendar pa bi to mejo lahko dosegli kvečjemu v primeru, če bi algoritem pri iskanju nezasičenih poti neko povezavo v pot vključil v nasprotni smeri, saj je število vseh poti v smeri od izvora do ponora tudi zgolj $O(n)$. Ali se to lahko zgodi? Recimo, da bi algoritem v trenutno nezasičeno pot želel dodati povezavo $(i, i+1)$ v nasprotni smeri. Ta povezava bi bila torej del nezasičene poti $p \equiv 1 \rightsquigarrow i+1 \to i \rightsquigarrow 1 \to k \rightsquigarrow n$ za nek $k > i+1$. No, če je ta pot res nezasičena, potem je nezasičena tudi njena podpot $p' \equiv 1 \to k \rightsquigarrow n$. Ker Edmonds-Karp išče nezasičene poti po njihovih naraščajočih dolžinah, bi odkril pot p', ne p, zato povezave $(i, i+1)$ v obratni smeri sploh ne bi obravnaval. Do podobnega zaključka pridemo tudi pri povezavah $(1, j)$ za $j \geqslant 3$.

(4) S točke 0 na celoštevilski osi se odpravimo iskat ključ, ki smo ga izgubili na eni od točk iz množice $\mathbb{Z}\backslash\{0\}$.

(a) Ključ bomo poiskali tako, da bomo v i-ti iteraciji prehodili i enot, pri čemer bomo v lihih iteracijah hodili v desno smer, v sodih pa v levo smer. V prvih štirih iteracijah torej obiščemo točke 1 (prva iteracija), 0, -1 (druga iteracija), 0, 1, 2 (tretja iteracija), 1, 0, -1, -2 (četrta iteracija). Koliko enot bomo prehodili, preden bomo prvič prispeli do točke t? Lahko se omejite na primer $t > 0$.

(b) Dokažite, da konkurenčno razmerje algoritma iz prejšnje podnaloge znaša $\Theta(n)$, kjer je n oddaljenost iskanega ključa od točke 0.

(c) Dokažite, da obstaja algoritem iskanja ključa s konstantnim konkurenčnim razmerjem. Analizo si lahko smiselno poenostavite. (Namig: Če želimo ključ zanesljivo najti, nimamo druge izbire, kot da izmenično potujemo levo in desno, seveda pa lahko poskusimo z drugačnimi dolžinami posameznih iteracij.)

Rešitve

(a) Do točke $t > 0$ bomo prispeli po $2t - 1$ iteracijah, do tedaj pa bomo opravili $\sum_{i=1}^{2t-1} i = \frac{1}{2}(2t-1)(2t) = t(2t-1)$ korakov.

(b) Če je iskani ključ na položaju n, bi jasnovidni algoritem potreboval zgolj n korakov, naš algoritem pa, kot smo videli, potrebuje $n(2n-1) = \Theta(n^2)$ korakov. Konkurenčno razmerje je torej $\Theta(n^2) \,/\, \Theta(n) = \Theta(n)$.

(c) Namesto da bi v i-ti iteraciji opravili i korakov, jih opravimo 2^{i-1}. Če se vnovič omejimo na pozitivno stran osi (negativna je enakovredna v okviru konstantnega faktorja), vidimo, da po $r = 2k + 1$ iteracijah prispemo do točke

$$\begin{aligned}
P(r) &= 1 - 2 + 4 - 8 + 16 - 32 + \ldots - 2^{2k-1} + 2^{2k} \\
&= \sum_{i=0}^{k} 4^i - 2 \sum_{i=0}^{k-1} 4^i \\
&= 4^k - \sum_{i=0}^{k-1} 4^i \\
&= 4^k - \frac{4^k - 1}{3} \\
&= \frac{2 \cdot 4^k + 1}{3} \\
&= \frac{2 \cdot 2^{2k} + 1}{3} \\
&= \frac{2^{2k+1} + 1}{3} \\
&= \frac{2^r + 1}{3}.
\end{aligned}$$

No, v teh r iteracijah opravimo skupno $T(r) = \sum_{i=0}^{r} 2^{i-1} = 2^r - 1$ korakov. Če se torej ključ nahaja na točki $n \in [P(r-2)+1,\, P(r)]$, ga bomo z opisanim sprotnim algoritmom našli po največ $T(r)$ korakih (v resnici lahko še prej, a analizi to ne bo škodovalo, saj bomo dobili kvečjemu preveliko konkurenčno razmerje), medtem ko bo jasnovidni algoritem opravil samo n korakov. Pri računanju konkurenčnega razmerja moramo vzeti primer, ki je z vidika sprotnega algoritma najslabši, torej

ko je ključ tik za točko, do katere sežemo v iteraciji $r-2$. Konkurenčno razmerje je potemtakem

$$\begin{aligned} c &= \frac{T(r)}{P(r-2)+1} \\ &= \frac{2^r - 1}{\frac{1}{3}(2^{r-2}+1)+1} \\ &= \frac{3 \cdot 2^r - 3}{2^{r-2}+4} \\ &= \frac{12 \cdot 2^{r-2} - 3}{2^{r-2}+4}. \end{aligned}$$

Pri majhnih r nam člena -3 v številcu in $+4$ v imenovalcu sicer lahko mešata štrene, asimptotično pa je slika povsem jasna:

$$c = \lim_{r\to\infty} \frac{12 \cdot 2^{r-2} - 3}{2^{r-2}+4} = \lim_{r\to\infty} \frac{12 - \frac{3}{2^{r-2}}}{1 + \frac{4}{2^{r-2}}} = 12 = \Theta(1).$$