MOTIVACIJSKI PRIMJERI I ZADACI S KONTEKSTOM U NASTAVI INFORMATIKE

Goranka Nogo
PMF, Matematički odsjek, Zagreb

Državni stručni skup za učitelje i nastavnike informatike i računalstva
Informatika u obrazovanju 2022. – Info@Edu XI.
27. travnja – 28. travnja 2022.

Sadržaj

- Uvodne napomene
- Generiranje primjera
  - Od implementacije do konteksta
  - Rješavanje problema sortiranjem
  - Modificirano binarno traženje
- Mogući problemi

Uvodne napomene

- Tipični „problemi”:
  - Za zadani prirodni broj n ispitati neko njegovo svojstvo – je li
    - kvadrat nekog drugog broja
    - palindrom
    - jednak zbroju svih djelitelja bez njega samoga
    - ...
  - Za zadani realni boj x izračunati vrijednost funkcije f definirane pravilom $f(x) = \begin{cases} x - 5, & x \leq 0 \\ x^2 + 2, & x > 0 \end{cases}$
  - Odrediti najveći elemenat… (eksplicitno je zadano što se očekuje od učenika)
  - …
- Napomena. I ovakvi zadaci trebaju biti zastupljeni.

Uvodne napomene

- Primjeri:
  - lako ih je preinačiti (otežati, generalizirati,…)
  - implementacija nije zahtjevna (nekoliko linija koda).

- Domene:
  - Računalno razmišljanje i programiranje
  - Informacije i digitalna tehnologija, Digitalna pismenost i komunikacija (proračunske tablice, pretraživanje,…).

Generiranje primjera

- Implementacija 1:

```
if(n <= 30):
    return 25*n
else:
    return 0.8*25*n
```

  - Cijene ulaznica:
    - pojedinačna karta – 25 kn
    - grupe s više od 30 osoba ostvaruju pravo na popust od 20 %

```
if(n <= 30):
    return 25*n
else:
    b = n//35
    bp = 2*b
    return 0.8*25*(n - bp)
```

    - dva pratitelja na 35 osoba imaju pravo na besplatnu ulaznicu.

Generiranje primjera

- Veza s proračunskim tablicama:

	A	B	C
1	Škola	Broj učenika	Cijena
2			
3			
4			

  - Primjena jednostavnih funkcija za izračun cijene
  - Analiza i prikaz podataka

Generiranje primjera

- Implementacija 2:

```
if(abs(malo - kat) <= abs(veliko - kat)):
    print("malo")
else:
    print("veliko")
```

  - Dva dizala:
    - `malo`, `veliko` i `kat` – cjelobrojne oznake katova na kojima se nalaze dizala i osoba koja je pozvala dizalo
    - dolazi ono dizalo koje je bliže pozivatelju; u slučaju da su dizala jednako udaljena od osobe koja ih je pozvala dolazi manje dizalo.

- Naredba grananja – ideja:
  - provjera je li zadana godina prijestupna.

Generiranje primjera

- Implementacija 3:

```
s = 0
k = 1
for i in range(12):
      s = s + k%4*list[i]
      k = k + 2
return (10 - s%10)%10
```

```
>>> list = [9, 7, 8, 9, 5, 3, 1, 9, 7, 3, 9, 6]
>>> g(list)
0
```

- Kontrolna znamenka (*International Standard Book Number*)

ISBN
978-953-197-396-0

Generiranje primjera

- Primjeri znakovnih nizova (*stringova*):
  - JMBAG – 1191*, 0036*,…
  - oznaka proizvoda (903.012.25, 058*,…)
  - lozinka
  - …
- Implementacija 4:

```
x = False
if(len(s) == 8 and s[0:2].isalpha() and s[2:8].isdigit()):
      x = True
return x
```

  - Provjera ispravnosti lozinke – je li oblika dva slova nakon kojih slijedi šest znamenki? [1]

Generiranje primjera

- „Implementacija 5”:
  - pronalaženje najmanjeg ili najvećeg elementa u nekoj listi.

  - Implicitno nalaženje najmanjeg (najvećeg) elementa ako su zadani
    - postotci riješenosti ispita (testa) po zadacima ili brojevi bodova
    - nazivi bakterija i pripadna vremena udvostručavanja populacije
    - proizvodi i pripadna stanja zaliha na početku i na kraju mjeseca
  - a treba odrediti
    - redni broj zadatka koji je riješilo najmanje učenika
    - koja se najbrže razmnožava
    - odrediti najprodavaniji proizvod.

Generiranje primjera

- Sortiranje – primjene:
  - poznate:
    - imenici, popisi izvođača,…
    - binarno traženje
    - izbacivanje duplikata
  - manje poznate:
    - određivanje medijana
    - pronalaženje najbližeg para točaka u ravnini, konveksne ljuske,…
    - Kruskalov algoritam (Minimalno razapinjuće stablo)
    - …
- Dva problema:
  - ulaz: dvije liste jednake duljine
  - rješenje se temelji na sortiranju jedne liste i odgovarajućoj permutaciji elemenata druge liste.

Generiranje primjera

- Sortiranje 1.
  - Problem odabira aktivnosti (*Activity-Selection Problem*):
    - $S = \{a_1, a_2, \cdots, a_n\}$ – skup aktivnosti
    - $s_i \geq 0$ – početak i-te aktivnosti
    - $s_i < f_i < \infty$ – završetak i-te aktivnosti
    - $[s_i, f_i)$ – trajanje aktivnosti a_i
    - $[s_i, f_i) \cap [s_k, f_k) = \emptyset$ – aktivnosti a_i i a_k su kompatibilne.

    - Treba pronaći maksimalni podskup međusobno kompatibilnih aktivnosti. [2]

Generiranje primjera

- Primjer:

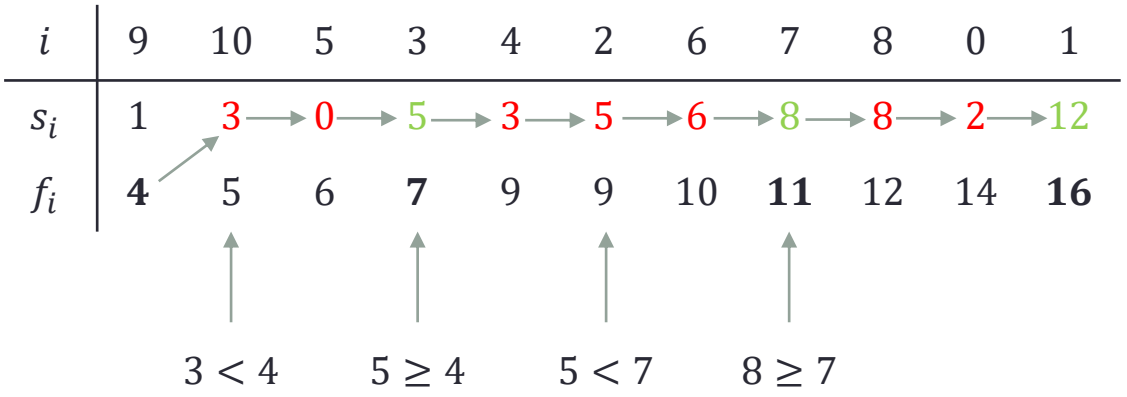
i	0	1	2	3	4	5	6	7	8	9	10
s_i	2	12	5	5	3	0	6	8	8	1	3
f_i	14	16	9	7	9	6	10	11	12	4	5

- Podskupovi međusobno kompatibilnih aktivnosti:
  - $\{a_5, a_8, a_1\}$
  - $\{a_9, a_3, a_7, a_1\}$
  - $\{a_9, a_3, a_8, a_1\}$.

- Rješenje: sortirati aktivnosti uzlazno, prema vremenu završetka f_i.

Generiranje primjera

- Primjer:



- Pronađeni podskup međusobno kompatibilnih aktivnosti:
  - $\{a_9, a_3, a_7, a_1\}$.

Generiranje primjera

- Sortiranje 2.
  - Opis problema:
    - $S = \{a_1, a_2, \cdots, a_n\}$ – skup autobusa
    - $s_i \geq 0$ – dolazak i-tog autobusa na peron
    - $s_i < f_i < 24$ – odlazak i-tog autobusa s perona
    - $[s_i, f_i)$ – trajanje boravka i-tog autobusa na peronu
    - $[s_i, f_i) \cap [s_k, f_k) = \emptyset$ – autobusi a_i i a_k mogu koristiti isti peron.

    - Treba odrediti minimalni broj perona.

  - Rješenje: sortirati autobuse uzlazno, prema vremenu dolaska s_i.
  - Napomena:
    - $s_i \to x_i$, $f_i \to y_i \Rightarrow$ sortiranje prema vrijednostima koordinate x.

Generiranje primjera

- Primjer:

s_i	f_i	
0	6	novi peron (peron 0)
1	4	$1 < 6 \rightarrow$ novi peron (1)
2	14	$2 < 6$ i $2 < 4 \rightarrow$ novi peron (2)
3	5	$3 < \min\{6, 4, 14\} \rightarrow$ novi peron (3)
3	9	$3 < \min\{6, 4, 14, 5\} \rightarrow$ novi peron (4)
5	7	$5 \geq \min\{6, 4, 14, 5, 9\}$ (1)
5	9	$5 \geq \min\{6, 7, 14, 5, 9\}$ (3)
6	10	$6 \geq \min\{6, 7, 14, 9, 9\}$ (0)
8	11	(1)
8	12	novi peron (5)
12	16	$\cdots$

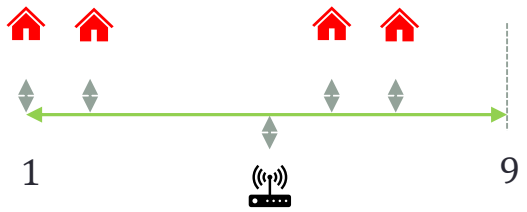
- Minimalni broj perona je 6.

Generiranje primjera

- Sortiranje 3.
  - Opis problema:
    - $S = \{x_1, x_2, \cdots, x_n\}$ – položaji kuća uz ravnu cestu zadani su koordinatom izraženom u kilometrima obzirom na neku ishodišnu točku
    - želimo postaviti antene na određene točke duž te ceste i to tako da je svaka kuća udaljena najviše 4 km od neke antene (u protivnom je signal preslab).
    - Treba odrediti najmanji potrebni broj antena i njihove pozicije y_i tako da su sve kuće pokrivene dovoljno jakim signalom.
  - Rješenje: sortirati položaje kuća uzlazno.

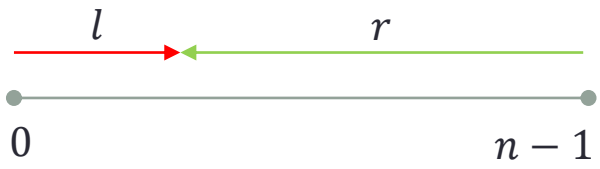
Generiranje primjera

- Sortiranje 3.
  - Strategija koja uvijek pronalazi pokrivanje s najmanjim brojem antena:
    - položaj antene (y_k) = položaj prve nepokrivene kuće $(x_i) + 4$
    - (novi) položaj prve nepokrivene kuće je prvi položaj x_j strogo veći od $y_k + 4$.
  - Primjer:
    - $S = \{1, 7, 2, 10, 6, 18\} \Rightarrow \{1, 2, 6, 7, 10, 18\}$
    - $y_1 = 5, y_2 = 14$



Generiranje primjera

- Sortiranje 4.
  - Opis problema: za zadani broj x odrediti postoje li dva elementa liste (polja) koji u zbroju daju x.
  - Rješenje: sortirati listu uzlazno i kretati se na način prikazan u programskom odsječku:




```
lista.sort()

l = 0
r = len(lista) - 1

while(l < r):
      if(lista[l] + lista[r] == x):
            return l, r
      elif (lista[l] + lista[r] < x): l = l + 1
      else:
            r = r - 1
return -1, -1
```

Generiranje primjera

- Modificirano binarno traženje 1:
  - Što tražimo? Element koji nedostaje u sortiranoj listi brojeva: $\{1, 2, \ldots, n\}$:
    - ulaz: $[1, 2, 3, 5, 6, 7, 8, 9, 10]$
    - izlaz: 4
    - ulaz: $[1, 2]$
    - izlaz: 3.

```
l = 0
r = len(lista) - 1

if (lista[l] == 2): # nedostaje 1
      return 1
elif (lista[r] == r + 1): # nedostaje n
      return r + 2
```

Generiranje primjera

- Modificirano binarno traženje 1:

```
while (l + 1 < r):
    mid = (l + r) // 2
    if ((lista[l] - l) != (lista[mid] - mid)):
        r = mid
    elif ((lista[r] - r) != (lista[mid] - mid)):
        l = mid
return lista[l] + 1
```

$$[1,2,3,5,6,7,8,9,10] \overset{4}{\Rightarrow} (1-0) \neq (6-4) \Rightarrow [1,2,3,5,6]$$

$$[1,2,3,5,6] \overset{2}{\Rightarrow} (6-4) \neq (3-2) \Rightarrow [3,5,6]$$

$$[3,5,6] \overset{3}{\Rightarrow} (3-2) \neq (5-3) \Rightarrow [3,5]$$

$$[3,5] \Rightarrow 3+1$$

Generiranje primjera

- Modificirano binarno traženje 2:
  - Što tražimo? Pomak uzlazno sortirane liste:
    - ulaz: $[6, 7, 12, 15, 3, 4]$
    - izlaz: 4 (indeks minimalnog elementa)
    - ulaz: $[2, 5, 9]$
    - izlaz: 0.

```
n = len(lista)
l = 0
r = n-1
```

```
while (l <= r):
      mid = (l + r)//2
      p = (mid - 1 + n)%n
      s = (mid + 1)%n

      if ((lista[mid] < lista[p]) and (lista[mid] <= lista[s])):
          return mid
      elif (lista[mid] < lista[l]): r = mid - 1
      elif (lista[mid] > lista[r]): l = mid + 1
      else: return 0
```

Generiranje primjera

- Modificirano binarno traženje 3 [3]:
  - Što tražimo? Indeks elementa nesortirane liste koji nije manji od svojih susjeda:
    - ulaz: $[1, 2, 3, 2, 6, 3]$
    - izlaz: 2 ili 4
    - ulaz: $[1, 2, 5, 8]$ $([7, 2, 1])$
    - izlaz: 3 (0).

```
while (l <= r):
      mid = (l + r) >> 1

      if ((mid == 0 or lista[mid - 1] <= lista[mid]) and \
         (mid == n - 1 or lista[mid + 1] <= lista[mid])):
           return mid
      elif (mid > 0 and lista[mid] < lista[mid - 1]): r = mid - 1
      else: l = mid + 1
```

Mogući problemi

- Lažni kontekst:
  - Ispisati opseg četverokuta ako su zadane duljine njegovih stranica.

- Jednostavna implementacija, kontekst neprimjeren uzrastu učenika:
  - $A = \begin{bmatrix} -2 & 4 & 3 & -1 \\ 2 & 0 & -1 & 2 \\ -5 & 0 & -2 & 1 \end{bmatrix} \begin{matrix} \rightarrow 10 \\ \rightarrow 5 \\ \rightarrow 7 \end{matrix}$, $\|A\|_\infty = 10.$

  (moguća interpretacija za $a_{ij} \geq 0$ – tri poslovnice tvrtke koja prodaje računala; koja poslovnica je prodala najviše računala u godini dana ako su za svaku poslovnicu zadani brojevi prodanih računala po mjesecima?)

- Previše teksta

Literatura

- [1] Cambridge IGSCE Computer Science: https://www.pdfdrive.com/cambridge-igcse-computer-science-e183809382.html
- [2] https://edutechlearners.com/download/Introduction_to_algorithms-3rd%20Edition.pdf
- [3] https://www.geeksforgeeks.org/greedy-algorithms/