Univerzitetni programerski maraton

FINALE 2020 – rešitve nalog

Tomaž Hočevar

</upm UNIVERZITETNI PROGRAMERSKI MARATON

Kača

Simuliraj premikanje kače po mreži.

- pozicija, smer kače
- tabela premikov glede na smer
- preverjanje dotika

- rešitev: 32 vrstic (Python)

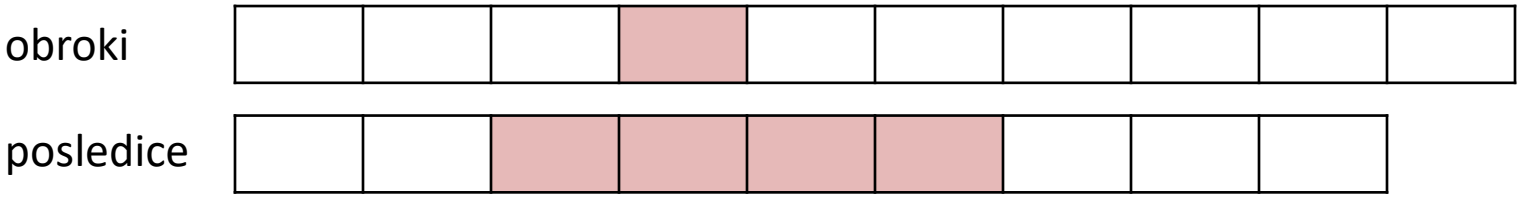
```
###########
..........#
#########.#
#.......#.#
#.*.....#.#
#.#######.#
#.........#
###########
```

</upm UNIVERZITETNI PROGRAMERSKI MARATON

Fižol

Poišči posledice vsakega obroka.

- seznam obrokov, seznam posledic/prdcev (urejeni)



- grupiranje obrokov po hrani (slovar)

a) “zlivanje” seznamov v O(n+m): kasnejši obrok ima kasnejše posledice

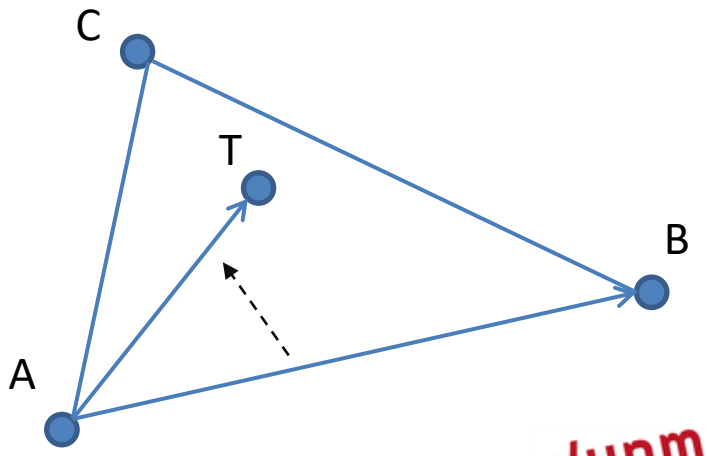
b) “bisekcija” v O(n log n): vsakemu obroku z bisekcijo poiščemo posledice

Najbermudskejši

Preštej točke znotraj trikotnikov.

- je točka T znotraj trikotnika ABC?
  - T je na isti strani vseh stranic
  - stran točke glede na stranico določimo z vektorskim produktom … $\overrightarrow{AB} \times \overrightarrow{AT}$

- napake:
  - overflow
  - ?



Odvod

Izračunajte odvod racionalne funkcije.

- implementacijska naloga (50 vrstic - Python)
- polinom
  - seštevanje, odštevanje, množenje, odvod
  - deljenje

```
  (45x^4 - 36x^3 -   9x^2 -  54x - 15) : (9x^2 - 18x + 9) = 5x^2 + 6x …
- (45x^4 - 90x^3 +  45x^2)
  ------------------------------
         (54x^3 -  54x^2 -  54x – 15)
       - (54x^3 - 108x^2 +  54x)
         --------------------------
          …
```

  - veliki koeficienti (64-bitna števila)

</upm UNIVERZITETNI PROGRAMERSKI MARATON

Album

Izračunaj pričakovano število paketov, da zapolnimo album.

- $f(x)$ = pričakovano št. paketov, če potrebujemo še x sličic
  - verjetnost, da v naslednjem paketu dobimo y novih sličic

$$p(x,y) = \binom{x}{y}\binom{n-x}{k-y} / \binom{n}{k}$$

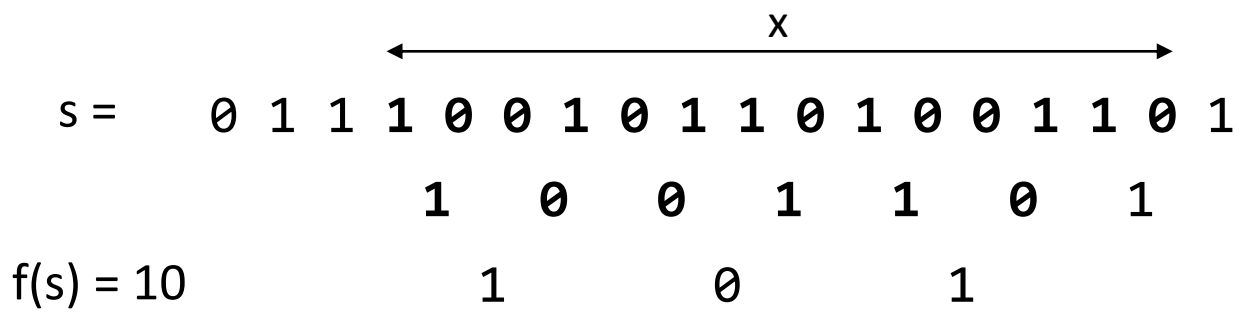
$$f(x) = 1 + p(x,0)f(x) + \sum_{y=1}^{k} p(x,y)f(x-y)$$

  - $O(n^2)$

- O(n): http://www.unige.ch/math/folks/velenik/Vulg/Paninimania.pdf

Podvojitve

Sestavi niz s čim manj potezami dodajanja števke 0 ali 1 na začetek ali konec in podvajanja vseh števk (0-->01, 1-->10).



- podvojitev x mora biti “maksimalna” (možnih je več)
- podproblemi so dolžine <= n/2
- vsota dolžin podproblemov je <= n (n/2 na lihih in n/2 na sodih indeksih)
- O(n log n)

</upm UNIVERZITETNI PROGRAMERSKI MARATON

Sedežni red

Pripravi optimalen sedežni red glede na preference oseb, ki želijo sedeti v različnih delih avtobusa (spredaj ali zadaj).

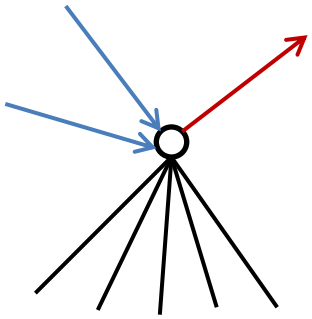
- obstaja opt. rešitev, kjer so sprednje osebe v prvih vrstah, zadnje v zadnjih, vmes pa prosti sedeži
  - sprednje ($S_i \geq Z_i$), zadnje osebe ($S_i < Z_i$)
  - dva podobna podproblema ($x_i = |S_i - Z_i|$)

- sprednje sedeže polnimo požrešno od zadaj
  - med kandidati posedemo najboljšega (max x_i); prioritetna vrsta

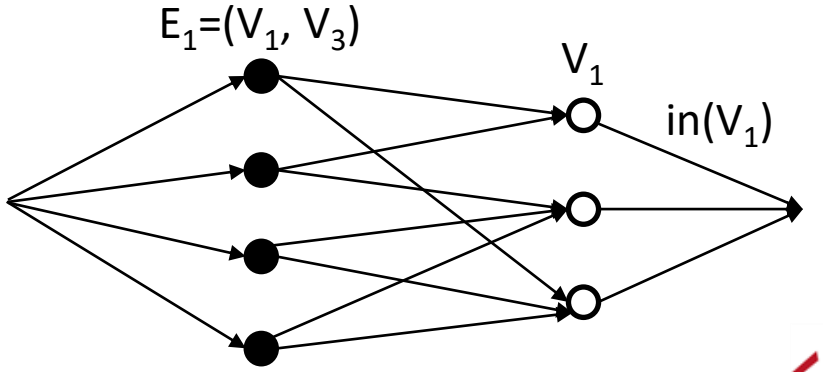
Utaja davkov

Usmeri neusmerjene povezave tako, da bo v grafu obstajal Eulerjev obhod in ga poišči.

- Eulerjev obhod obstaja, če imajo vsa vozlišča enako vhodno in izhodno stopnjo.
  - pazi: nepovezana vozlišča, nepovezane komponente
  - pretok/ujemanje: $O(m^2)$



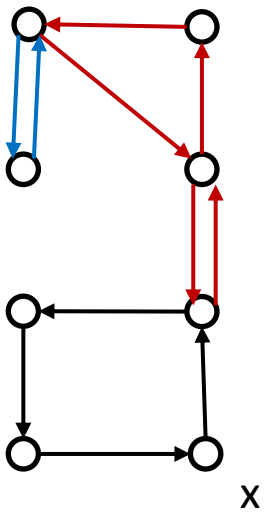
in(V) = 2



</upm UNIVERZITETNI PROGRAMERSKI MARATON

Utaja davkov

- iskanje Eulerjevega obhoda
  - Hierholzer
  - združuješ cikle, O(E)

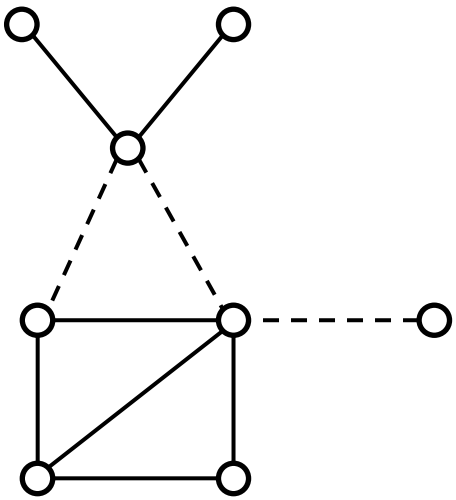


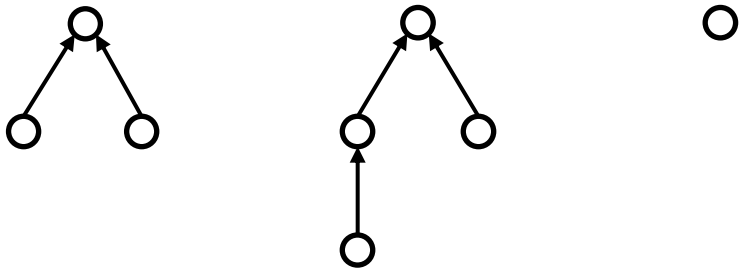

```
vector<int> path;
void euler(int x) {
    while (!out[x].empty()) {
        int y=out[x].back();
        out[x].pop_back();
        euler(y);
        path.push_back(y);
    }
}
```

Teroristične celice

Preštej povezane komponente ob upoštevanju povezav z utežjo na določenem intervalu.

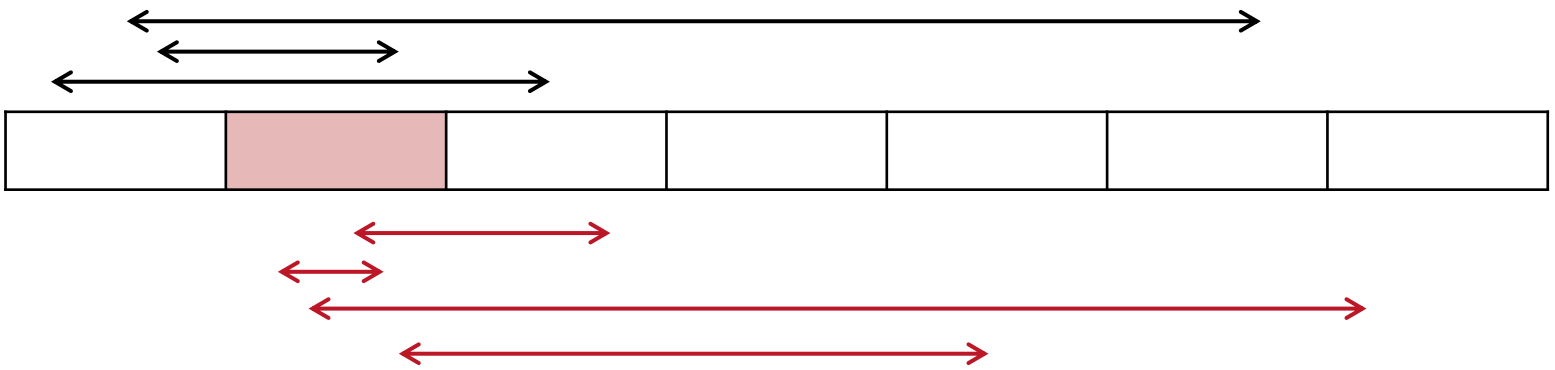
- povezave uredimo po utežeh
- poenostavitve
  - samo ena poizvedba: bfs/dfs
  - enaka spodnja meja X_j:
    - poizvedbe uredimo po Y_j
    - disjunktne množice (union-find)





Teroristične celice

- korenska dekompozicija poizvedb



  - povezav ne moremo odstranjevati!
    - razveljavimo operacije iz trenutnega koša v union-find strukturi

  - $O(\sqrt{m}(k+m)\alpha), \qquad \alpha = O(\log n)$

Zaključek

- počasno reševanje
  - 3 osebe
- geometrija
- precej napak
- prepoznavanje lahkih nalog