Fakulteta za Matematiko in Fiziko v Ljubljani

Najdaljši palindromski podniz in iskanje otokov

Predstavitev vprašanj pri predmetu Računalništvo 1

Jakob Valič

Ljubljana, avgust 2018

Table of Contents

Najdi najdaljši palindromski podniz

Ključne besede: Niz, Palindrom

Vir: https://www.programcreek.com/2013/12/leetcode-solution-of-longest-palindromic-substring-java/

Odgovor:

Poglejmo si algoritem s časovno zahtevnostjo `O(n^2)`.

Algoritem temelji na funkciji za premikanje centra in funkciji za preverjanje palindroma z danim centrom. Pobliže si poglejmo delovanje obeh funkcij.

Funkcija za iskanje palindroma z danim centrom

Predpostavimo, da imamo podan center palindroma. Želimo poiskati najdaljši palindrom s tem centrom. Tega se lotimo tako, da preverjamo enako oddaljene znake levo in desno od centra, vse dokler bodisi znaka nista več enaka ali ko dosežemo konec niza. Za primer vzemimo besedo `'banana'`. Recimo, da za center izberemo prvi `'n'`. Označimo ga z rdečo puščico. Z zeleno puščico označimo preverjanje znakov na levi in desni. Najprej preverimo sosednja znaka. Ker sta enaka, preverimo znaka, ki sta za 2 oddaljena od centra. Ker nista enaka, je najdaljši palindrom 'ana'. Vrnemo indeks začetka in konca najdaljšega palindroma, torej 1 in 4. Rezine nizov v Pythonu delujejo tako, da nam ukaz `'banana'[1:4]` vrne `'ana'`. Indeks začetka in konca najdaljšega palindroma z danim centrom funkcija vrne tudi tedaj, ko naleti na prvo ali zadnjo črko v besedi.

Vseeno se nam na pogled zdi, da v besedi `'banana'` obstaja daljši palindromski podniz. Ali ne bi bilo bolje, da prestavimo center v drugi `'a'`? Tu nam na pomoč pride druga funkcija.

Funkcija za prestavljanje centra

Ta funcija poskrbi za prestavljanje centra. Center je lahko sestavljen iz enega ali dveh znakov. Pri besedi `'banana'` je najboljši center sestavljen iz enega znaka. To je drugi `'a'`. S prvo funkcijo ugotovimo, da je najdaljši palindromski podniz `'anana'`.

Zakaj je nujno, da je lahko center sestavljen iz dveh črk? Zato, ker dopuščamo, da je lahko center palindroma sestavljen iz dveh črk. V primerih, ki sledijo, je najdaljši palindromski podniz s centrom iz dveh znakov zaporedoma `'ii'` in `'abba'`. Center iz enega znaka pa v obeh primerih vrne podniz dolžine 1, zaporedno `'p'` in `'a'`.

Dodatne opombe:

Obstaja Manacherjev algoritem, ki najdaljši palindromski podniz poišče v O(n). Zgoraj opisani funkciji in nakaj testnih primerov se nahajata v datoteki *najdaljsiPalindrom.py*, ki je priložena.

Število otokov v matriki

Ključne besede: Matrika, DFS

Vir: https://www.programcreek.com/2014/04/leetcode-number-of-islands-java/

Odgovor:

Imamo matriko enk in ničel, kjer 1 pomeni kopno, 0 pa vodo. Naj bo otok s stranicami povezano kopno, ki ga obdaja morje. Privzemimo, da je izven matrike tudi morje. Koliko otokov je v matriki? Na spodnjem primeru vidimo, da se v matriki pojavijo 4 otoki.

```
[0, 0, 0, 0, 1]
[0, 0, 1, 0, 0]
[0, 0, 1, 0, 1]
[0, 0, 1, 1, 0]
[1, 1, 1, 1, 0]
[0, 0, 0, 0, 0]
[0, 1, 0, 0, 0]
[1, 1, 0, 0, 0]
```

Potrebujemo 2 funkciji: glavno, ki se sprehaja čez matriko ter funkcijo, ki zbriše celoten otok.

Funkcija za brisanje otokov

Funkcija za vhodne podatke dobi referenco na matriko in kooridnate, kjer se nahaja del otoka. Funkcija z rekuzivnimi klici poskrbi, da se vse enice, ki pripadajo otoku, spremenijo v ničle. Funkciji ni potrebno vračati ničesar, saj preko podane reference spreminja matriko.

```
def dfs(graf, i, j):
    '''Enice otoka, v katerem se nahaja enica na mestu (i, j) spremeni v ničle.'''
    graf[i][j] = 0
    # Pregledamo v vse štiri smeri npr. (-1, 0) je levo
    smeri = [(-1, 0), (1, 0), (0, -1), (0, 1)]
    for (vodoravno, navpicno) in smeri:
        n = i + vodoravno
        m = j + navpicno
        if 0 <= n < len(graf) and 0 <= m < len(graf[0]) and graf[n][m] == 1:
            dfs(graf, n, m)
```

Funkcija za iskanje otokov

Funkcija sistematično išče enice v grafu. Ko jih najde, poveča števec otokov za 1. Nato pokliče funkcijo za brisanje otokov, da istega otoka ne prešteje še enkrat.

```python
def koliko_otokov(graf, izpis_grafa):
    '''Vrne število otokov, to je povezanih enic v grafu.'''
    stevilo_otokov = 0
    print("Osnovni zemljevid:")
    izpis(graf)
    for i in range(len(graf)):
        for j in range(len(graf[0])):
            if graf[i][j] == 1: # Našli smo del otoka
                stevilo_otokov += 1
                if izpis_grafa:
                    input('\n {}. Zbrisali bomo otok \n'.format(stevilo_otokov))
                dfs(graf, i, j) # Poženemo dfs, da zbrišemo cel otok
                if izpis_grafa:
                    izpis(graf)
    return stevilo_otokov
```

Primer:

Koda s testnimi primeri in generiranjem naključnega števila otokov se nahaja v priloženi datoteki *otoki.py*.