EADŠ - cvičenie 8

10. novembra 2022

Dynamické programovanie - opakovanie

Prístup k riešeniu problému, kde:

- hľadáme nejako "optimálne" riešenie
- nevieme len tak jednoducho riešiť celý problém
- vieme z *menších*, ale inak *rovnakých* problémov zložiť riešenie väčšieho problému

Dynamické programovanie, "obvyklé" kroky - opakovanie

1. Nájsť podproblém
2. Zistiť, ako vypočítať podproblém z menších podproblémov (rekurentný vzťah)
3. Nastaviť/zistiť bázové prípady
4. Určiť poradie vypĺňania matice / poľa

Vykrádanie domov

Máme ulicu s n domami, ktoré stoja v rade za sebou.
Číslo c_i - peniaze, ktoré sa dajú ukradnúť v dome i.
Aby sme neboli nápadní, nechceme vykradnúť 2 domy, ktoré stoja pri sebe. Ak už vykrádame dom, chceme ho vykradnúť celý. Koľko najviac peňazí vieme ukradnúť?

Napríklad:

- domy `[1, 3, 5, 2, 8, 3]`. Chceme vykradnúť `1 + 5 + 8`.
- domy `[10, 1, 7, 9]`. Chceme vykradnúť `10 + 9`.

Bentley's problem

Máme pole s n číslami. Chceme vybrať takú súvislú podpostupnosť, ktorá má čo najväčší súčet

Napr. pole

```
[31, -41, 59, 26, -53, 58, 97, -93, -23, 84]
```

Optimum:

```
[--, ---, 59, 26, -53, 58, 97, ---, ---, --]
```

Memoizácia

Fibonacciho čísla:

```
F(0) = 0
F(1) = 1
F(n) = F(n-2) + F(n-1)
```

Memoizácia

Fibonacciho čísla:

```
F(0) = 0
F(1) = 1
F(n) = F(n-2) + F(n-1)
```

```
def fib(n):
  if n==0:
    return 0
  if n==1:
    return 1
  return fib(n-2) + fib(n-1)
```

Memoizácia

Funkcia v matematickom zmysle slova: závisí *iba od vstupu*.

Pre konkrétny vstup dá vždy rovnaký výstup.

Ak si zapamätáme/nakešujeme/namemoizujeme jej výsledok, nemusíme ju počítať viac krát.

Memoizácia - ako na to?

1. pole/matica/veľarozmerné pole
2. dictionary
   `{(vstup1): výstup1, (vstup2): výstup2, ...}`

Wine selling problem

Máme n fliaš vína (napr. `[2,4,6,2,5]`).
Každý rok musíme predať jednu fľašu vína. Ak fľašu s cenou c_i predáme v roku r, dostaneme za ňu $r \cdot c_i$ peňazí.
Fľaše vieme predávať iba tie (dve), ktoré sú na kraji.

Editačná vzdialenosť (Levenshtein distance)

Máme dve slová (dva kusy textu) `T` a `S`. Za poplatok 1 môžeme:

- zmazať jeden znak
- pridať jeden znak
- nahradiť (substituovať) jeden znak

A chceme prerobiť jedno slovo na druhé.

Napr. ak `T = "KITTEN", S = "SITTING"`, tak `dist(T, S) = 3`, lebo:

1. `KITTEN` $\rightarrow$ `SITTEN`
2. `SITTEN` $\rightarrow$ `SITTIN`
3. `SITTIN` $\rightarrow$ `SITTING`