Algoritmy a datové struktury

Petr Švarný 2021

Rekurze je potvora

Rekurzivní volání

A university has 10 mathematicians, each one so proud that, if she learns that she made a mistake in a paper, no matter how long ago the mistake was made, she resigns the next Friday. To avoid resignations, when one of them detects a mistake in the work of another, she tells everyone else but doesn't inform the mistake-maker. All of them have made mistakes, so each one thinks only she is perfect.

One Wednesday, a super-mathematician, whom all respect and believe, comes to visit. She looks at all the papers and says: "Someone here has made a mistake."

What happens then? Why?

Rekurze na katedře

1 matematik = rezignuje hned první pátek.

2 matematici A, B = A ví, že B udělal chybu a tedy první pátek A nerezignuje. To samé, ale B. Tedy také nerezignuje. Druhý pátek tedy rezignují oba.

…

Klasický příklad rekurze

```
3! = 3 * 2! = 3 * 2 * 1! = 3 * 2 * 1
```

Klasický příklad rekurze

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n-1)
```

Základní složky rekurze

?

Základní složky rekurze

Jako u indukce:

- “stopping term” - něco, co ukončuje rekurzi
  (v nejhorším případě paměť PC nebo přinejhorším existence vesmíru)

- rekurzivní volání sebe sama s **novou hodnotou**

- (!) Může být i nepřímá (A -> B -> A -> …)

Co je hlavní přínos rekurze?

A) Šetří množství napsaného kódu.
B) Rozděluje problém na lehčí problémy.
C) Vytváří vždy výpočetně méně náročný kód než bez rekurze.

Co je hlavní přínos rekurze?

A) Šetří množství napsaného kódu.
B) Rozděluje problém na lehčí problémy. (Viz i fraktály)
C) Vytváří vždy výpočetně méně náročný kód než bez rekurze.

Rekurze a rekurence

Vyjádříme složitost rekurzivního algoritmu “rekurencí”:

$$T(n) = \begin{cases} \Theta(1), & \text{if } n = 1, \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n), & \text{if } n > 1. \end{cases}$$

Zaokrouhlení dolu

Složitost = zanedbáme detaily, ale které?

A) Zaokrouhlení by mohla být nepodstatná.
B) Cokoliv, co je konstantně složité je zanedbatelné.
C) Součet je zanedbatelný.

Složitost = zanedbáme detaily, ale které?

A) **Zaokrouhlení by mohla být nepodstatná.** (někdy!)
B) **Cokoliv, co je konstantně složité je zanedbatelné.**
C) Součet je zanedbatelný.

Zjednodušená podoba

$$T(n) = \begin{cases} \Theta(1), & \text{if } n = 1, \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n), & \text{if } n > 1. \end{cases}$$

$$T(n) = 2T(n/2) + \Theta(n)$$

Jak složitý je tedy výpočet dle vzorce?

$$T(n) = 2T(n/2) + \Theta(n)$$

A) Omega(n),
B) Theta(n + 2)
C) Nelze ještě určit.

Jak složitý je tedy výpočet dle vzorce?

$$T(n) = 2T(n/2) + \Theta(n)$$

A) Omega(n),
B) Theta(n + 2)
C) Nelze ještě určit.

Převod rekurence na přímé vyjádření

Přímé vyjádření, tj. bez rekurence/opětovného výskytu

- **Substitucí**
  Odhadneme řešení a induktivně dokážeme
- **Rekurzivním stromem**
  Spočítáme složitost rekurzivního stromu
- **Použitím “kuchařky”**
  Mistrovská věta (Master theorem), některé tvary mají známá řešení

DU Hanojské věže

Úloha: přemístit disky z tyče vlevo na tyč vpravo

Omezení:

• disky přesouváme pouze jednotlivě, z tyče na tyč

• větší disk nesmí nikdy ležet na menším disku

DU:

1. Zkuste vyjádřit složitost rekurentním vztahem.
2. Zkuste získat přímý vztah.