Procházení stromem, zásobník a fronta

Základy algoritmizace – 5. cvičení

Zuzana Petříčková

21. března 2020

Složitost procházení stromem do hloubky a do šířky

Zatím jsme probírali průchod binárním vyhledávacím stromem do hloubky a do šířky s využitím rekurzivní funkce

Průchod do hloubky:

- do každého vrcholu vstoupím právě jednou ze shora a vrátím se do něho právě jednou z každého syna, tedy $T(n) = O(n)$.
- podrobný výpočet časové složitosti: skripta, kapitola 2.2.5, strana 47
- hloubka rekurze = hloubka stromu, tedy až n (v průměru $O(log_2 n)$)
  $\rightarrow$ hodně nám narůstá časová i prostorová složitost

Složitost procházení stromem do hloubky a do šířky

Průchod do šířky:

- zpracování i-té hladiny:
  - velikost podstromu o hloubce i bude nejvýše: $n_i \leq \sum_{j=1}^{i} 2^{i-1} = 2^i - 1$
  - zpracování i-té hladiny = průchod podstromem o hloubce i, tj. $T_i(n) = O(n_i) \sim 2^i - 1$
- celkem pro strom o hloubce h: $T(n) = \sum_{i=1}^{h} T_i(n) = \sum_{i=1}^{h}(2^i - 1) \leq -h + \sum_{i=1}^{h} 2^i = -h - 1 + \sum_{i=0}^{h} 2^i = 2^{h+1} - h - 2$
  - pro úplný strom o hloubce $h = log_2(n+1)$: $T(n) = 2 \cdot 2^{log_2 n} - log_2(n+1) - 2 = 2n - log_2(n+1) - 2 \sim 2n = O(n)$
  - pro degenerovaný strom o hloubce $h = n$: $T(n) = \sum_{i=1}^{h} i = \sum_{i=1}^{n} i = O(n^2)$
- rekurzivní průchod do hloubky se opakuje pro každou hladinu stromu $\rightarrow$ opět nám hodně narůstá časová i prostorová složitost kvůli rekurzi

Procházení stromem do hloubky a do šířky

Můžeme se nějak rekurze zbavit?

- ano, pokud použijeme jednu z následujících pomocných datových struktur (kontejnerů): zásobník a fronta

Zásobník (stack, LIFO)

- viz skripta, kapitola 2.3.2, strana 63-66
- pro průchod do hloubky

Fronta (queue, FIFO)

- viz skripta, kapitola 2.3.3, strana 67-68
- pro průchod do šířky

Zásobník

Základní operace:

- vlož (push) - vložíme další hodnotu
- vyjmi (pop) - vyjmeme hodnotu, kterou jsme vložili jako poslední
- top - podíváme se na hodnotu na vrchu zásobníku (tj. na tu, kterou jsme vložili jako poslední)

Implementace:

- lineární spojový seznam:
  - vlož (push) - vložíme prvek na začátek
  - vyjmi (pop) - vyjmeme první prvek
- pole: musíme si pamatovat index vrcholu zásobníku i (nevýhoda: musíme vědět předem, jak velké pole potřebujeme)
  - vlož (push) - vložíme prvek na index i, posuneme index i o jedna doprava (i++)
  - vyjmi (pop) - podíváme se na prvek na indexu i, posuneme index i o jedna doleva (i−−)

Fronta

Základní operace:

- vlož (push) - vložíme další hodnotu
- vyjmi (pop) - vyjmeme hodnotu, kterou jsme vložili jako první
- top - podíváme se na hodnotu na vrchu fronty (tj. na tu, kterou jsme vložili jako první)

Implementace:

- lineární spojový seznam:
  - vlož (push) - vložíme prvek na konec
  - vyjmi (pop) - vyjmeme první prvek
- pole: musíme si pamatovat indexy začátku fronty i a konce fronty j (nevýhoda: musíme vědět předem, jak velké pole potřebujeme, budeme muset indexy v poli ,,cyklit")
  - vlož (push) - vložíme prvek na index j, posuneme index j o jedna doprava (j++)
  - vyjmi (pop) - podíváme se na prvek na indexu i, posuneme index i o jedna doprava (i++)

Procházení binárním stromem do hloubky (BACKTRACKING) s využitím zásobníku

použijeme zásobník

- PREORDER (nejjednodušší a nejčastější případ)
  - na zásobník budeme vkládat podstromy (tj. např. ukazatele na vrcholy stromu)
- INORDER, POSTORDER (složitější případy)
  - na zásobník budeme vkládat dva typy hodnot:
    1. podstromy (tj. např. ukazatele na vrcholy stromu)
    2. data / samotné vrcholy bez synů (při implementaci pozor, abychom si nerozházeli strom)

Procházení binárním stromem do hloubky (BACKTRACKING) s využitím zásobníku

PREORDER

zásobník ... pro uložení podstromů
vlož na zásobník celý strom (není-li prázdný)
while zásobník není prázdný **do**
  vyjmi ze zásobníku podstrom s kořenem k
  zpracuj vrchol k
  vlož na zásobník **pravý** podstrom k (není-li prázdný)
  vlož na zásobník **levý** podstrom k (není-li prázdný)

INORDER

zásobník ... pro uložení dvou typů hodnot: podstromů a ,,samotných vrcholů"
vlož na zásobník celý strom (není-li prázdný)
while zásobník není prázdný **do**
  vyjmi ze zásobníku prvek k
if k je ,,samotný vrchol" **then**
  zpracuj vrchol k
else ... k je kořen podstromu
  vlož na zásobník pravý podstrom k (není-li prázdný)
  vlož na zásobník k jako ,,samotný vrchol"
  vlož na zásobník **levý** podstrom k (není-li prázdný)

Procházení binárním stromem do hloubky (BACKTRACKING) s využitím zásobníku

POSTORDER ... obdobně jako INORDER (stačí prohodit předposlední dva řádky)

zásobník ... pro uložení dvou typů hodnot: podstromů a ,,samotných vrcholů"
vlož na zásobník celý strom (není-li prázdný)
while zásobník není prázdný **do**
  vyjmi ze zásobníku prvek k
 if k je ,,samotný vrchol" **then**
  zpracuj vrchol k
 else ... k je kořen podstromu
  vlož na zásobník k jako ,,samotný vrchol"
  vlož na zásobník **pravý** podstrom k (není-li prázdný)
  vlož na zásobník **levý** podstrom k (není-li prázdný)

Procházení binárním stromem do šířky (tj. po vrstvách) ... s využitím fronty

použijeme frontu

- do fronty budeme vkládat podstromy (tj. např. ukazatele na vrcholy stromu)
- pozor, pořadí vkládání synů je opačné než u zásobníku

zpracuj (strom)
  fronta ... pro uložení podstromů
  vlož do fronty celý strom (není-li prázdný)
  while fronta není prázdná **do**
    vyjmi z fronty podstrom s kořenem k
    zpracuj vrchol k
    vlož do fronty levý podstrom k (není-li prázdný)
    vlož do fronty pravý podstrom k (není-li prázdný)

Výpis vrcholů stromu po jednotlivých vrstvách

- použijeme procházení stromem do šířky pomocí fronty

Jak ale poznáme, že jsme došli na konec hladiny a máme odřádkovat?

Jedno z možných řešení:

- do fronty budeme vkládat dvojice:
  - ukazatel na vrchol (kořen podstromu)
  - hladinu, na které se daný vrchol nachází
- při vypisování:
  - pamatujeme si aktuálně vypisovanou hladinu
  - když je aktuální vrchol na jiné hladině než ten předchozí, před jeho vypsáním odřádkujeme

Výpis vrcholů stromu v pořadí preorder odsazeně

- Chceme vypsat klíče vrcholů ve stromě v pořadí PREORDER na konzoli tak, že každý vrchol bude na jednom řádku a klíče odsazené podle hladiny, ve které se příslušný vrchol nachází
- použijeme procházení stromem do hloubky pomocí zásobníku

Jak ale poznáme, o kolik mezer máme daný vrchol odsadit?

Jedno z možných řešení:

- do fronty budeme vkládat dvojice:
  - ukazatel na vrchol (kořen podstromu)
  - hladinu, na které se daný vrchol nachází
- při vypisování:
  - vypíšeme tolik "mezer", jaká je hladina vrcholu

Výpis vrcholů stromu po jednotlivých vrstvách tak, aby byla zřejmá struktura stromu

- zde je možných řešení více

Jedno z možných řešení:

- použijeme procházení stromem do šířky pomocí fronty
- pro každý vrchol si navíc spočítáme/předpočítáme jeho sloupeček
- průběžně si pamatujeme, jakou hladinu právě vypisujeme a v kterém sloupečku se nacházel poslední vypisovaný vrchol

viz. rozšířené poznámky k minulému cvičení

Složitost procházení stromem do hloubky a do šířky s využitím zásobníku a fronty

Časová složitost

- každý vrchol vložíme do zásobníku/ fronty právě jednou (v případě INORDER a POSTORDER právě dvakrát) a přesně tolikrát ho i vyjmeme ... počet opakování cyklu bude roven počtu vrcholů stromu (v případě INORDER a POSTORDER dvojnásobku počtu vrcholů stromu) $\rightarrow$ časová složitost bude O(n)

Prostorová složitost

- budeme potřebovat nějakou paměť pro uložení zásobníku / fronty, v nejhorším případě cca. $n/2$ (v případě INORDER a POSTORDER n)
- na rozmyšlenou: pro jaký tvar stromu budeme potřebovat zásobník velikosti $n/2$ a pro jaký frontu velikosti $n/2$?

Procházení stromem - složitost operací

metoda	T(n) (časová složitost)	S(n) (prostorová složitost)
do hloubky (rekurze)	$O(n)$	O(h) (hloubka rekurze =hloubka stromu) MAX $O(n)$
do hloubky (zásobník)	$O(n)$	až $n/2$ (velikost zásobníku)
do šířky (rekurze)	MIN $O(n)$ **MAX O(n^2)**	O(h) (hloubka rekurze =hloubka stromu) MAX $O(n)$
do šířky (fronta)	**vždy O(n)**	až $n/2$ (velikost fronty)