Úkol 2

pátek 30. října 2020 13:42

Úloha 1. *Vajíčko je takzvané k-vajíčko, pokud se při hodu z k-tého patra nerozbije, ale z $(k+1)$-ního se rozbije. Zjistěte na co nejmenší počet pokusů číslo k, pokud máte k dispozici dvě naprosto totožná k-vajíčka a házíte z budovy o n patrech.*

Na cvičení jsme si ukázali, že v případě 1 vajíčka musíme projít v nejhorším případě n pater. Pokud by se totiž vajíčko na l-tém patře rozbilo, nemáme šanci zjistit jestli bylo 1-vajíčko nebo $l-1$ vajíčko.

Díky druhému vajíčku získáváme možnost dál pracovat a zkoušet v intervalu $1 \dots l$. Zde ale opět nemáme už více pokusů a musíme projít všech l pater.

Záleží tedy na zvolení l. Pokud bychom se rozhodli použít $l-1$, v nejhorším případě půjdeme všech n pater, pokud se jedná o $n, n+1 ..$ vajíčko.

Pokud zvolíme l příliš vysoké (třeba n), při troše smůly stále zůstaneme na n patrech.

Ideální volba hodu 1. vajíčka je tedy na $\frac{n}{2}$ tém patře. V nejhorším případě se rozbije a pak musíme projít $\frac{n}{2}$ pater. Jinak pokračujeme na $\frac{n}{2}+\frac{n}{4}$ tém patře apod... s tím, že pokračujeme od posledního "checkpointu".

Úloha 2. *Navrhněte algoritmus, který dostane na vstupu částečně seřazenou posloupnost takovou, že každý prvek je nejvýše k pozic od místa, kam patří v seřazené posloupnosti. Výstupem je pak doseřazená posloupnost. Dokažte korektnost a konečnost algoritmu. Ukažte jeho časovou složitost. Pozn.: Naivní algoritmus to zvládne za* $\mathcal{O}(n \cdot k)$*, zkuste to rychleji.*

Naivní pro každé $i \in 1 \dots n$ zkontroluje všechny prvky k pozic od něj napravo, nalezne hledaný prvek, přehodí ho na pozici i.

Korektnost: prvek jistě na k pozicích najdu. Stačí se dívat jen doprava, jelikož vlevo máme již seřazené pole.

Konečnost: algoritmus obsahuje dva cykly s přesným počtem opakování $n \cdot x$, $x \in 1 \dots k$, vždy tedy skončí, nejhorší složitost $O(n \cdot k)$

Úloha 3. *Uvažujme nafukovací pole tak, jak bylo probíráno na cvičeních, pouze s tou výjimkou, že při vkládání do zaplněného pole zvětšujeme o pětinásobek. Bude amortizovaná složitost jednoho vkládání stále* $\Theta(1)$*? Své tvrzení dokažte.*

6x

ANO, bude.

Pro n prvků dojde k realokaci vždy při prvku 2., (1→6), 7. (6→36), 37. (36→216 ...)

vkládání tedy budou $\underbrace{1 + \dots + 1}_{6x} + 6 + \underbrace{1 + \dots + 1}_{30x} + 36 + \dots$

CENA 6, MÁM 36 CENA 36, MÁM 180

Pokud dáme tedy každému prvku 6 (konstantně) "penízků", dokáže při realokaci zaplatit veškerý náklad.

Úloha 4. *Uvažujme nafukovací pole tak, jak bylo probíráno na cvičeních, pouze s tou výjimkou, že při vkládání do zaplněného pole zvětšujeme vždy pole o 100 prvků. Bude amortizovaná složitost jednoho vkládání stále $\Theta(1)$? Své tvrzení dokažte.*

NE, zde nám "konstantní" počet penízků nepomůže.

Při vkládání bude potřeba realokovat následovně:

$$\underbrace{1+\dots+1}_{100\times} + 100 + \underbrace{1+\dots+1}_{100\times} + 200 + \underbrace{1+\dots+1}_{100\times} + 300 \dots$$

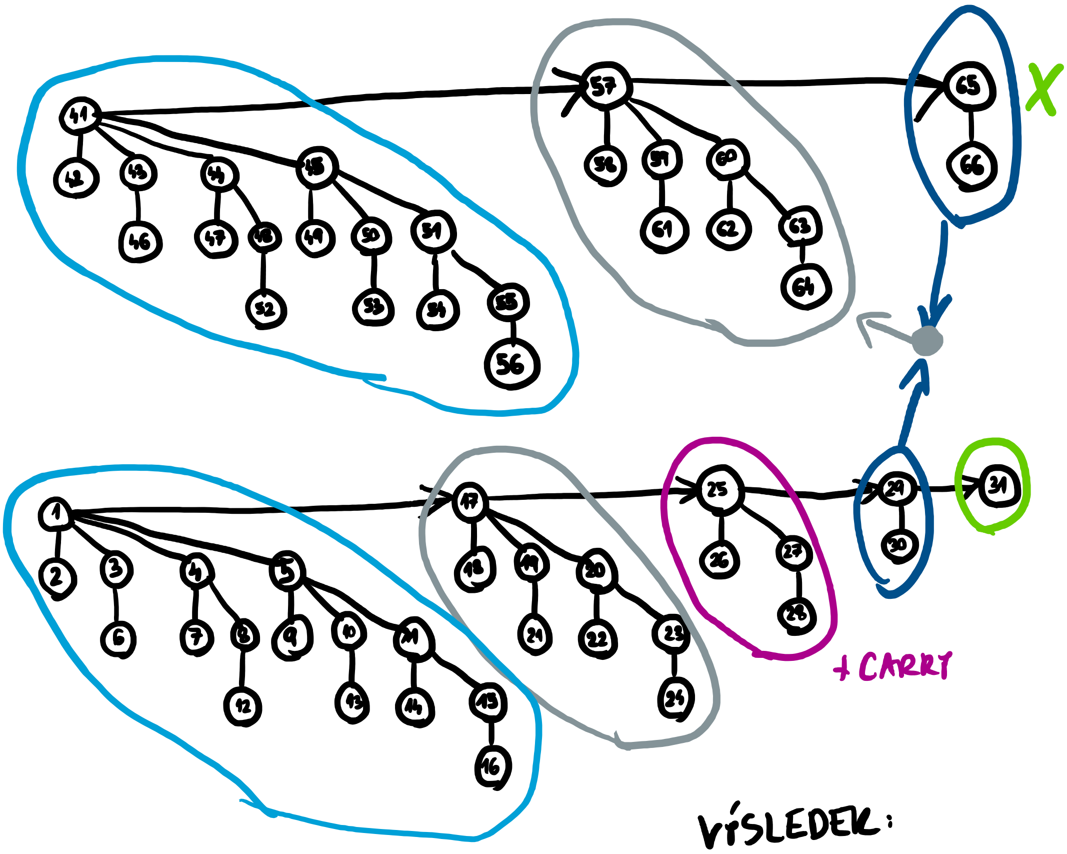
(Důkaz sporem) Kdyby měl tento algoritmus $\Theta^*(1)$, pak by byl poměr "aktuálního počtu penízků" vůči náročnosti operace konstantní (jak je tomu u př. 3).

Na 101. prvek potřebuji 100 "penízků", na 201. 200, 301. 300 ... mám však pokaždé ale pouze 100 penízků.

Poměr je tedy lineární a složitost je $O^*(n)$.

Úloha 5. *Nakreslete libovolnou binomiální haldu o 26 prvcích a proveďte merge s jinou binomiální haldou o 31 prvcích.*

$$26_{(10)} = \overset{B_4}{1}\overset{B_3}{1}\overset{B_2}{0}\overset{B_1}{1}\overset{B_0}{0}{}_{(2)} \qquad 31_{(10)} = \overset{B_4}{1}\overset{B_3}{1}\overset{B_2}{1}\overset{B_1}{1}\overset{B_0}{1}{}_{(2)}$$



PRVNÍ
(26 prvků)

DRUHÁ
(31 prvků)

VÝSLEDEK: