Cvičení 11

Algoritmické hádanky

Úloha 1. Mějme RAM pracující s reálnými čísly a konstantní cenou operací $+, *, /, <, >, \leq, \geq$. Jak rychle v tomto modelu dokážete pro zadané $x \in \mathbb{R}$ najít jeho dolní celou část $\lfloor x \rfloor$?

Úloha 2. Je dána posloupnost celých čísel $x_1, \ldots, x_n$. Jak v ní najít úsek (souvislou podposloupnost) s maximálním součtem?

Úloha 3. Je dána posloupnost *kladných* celých čísel $x_1, \ldots, x_n$ a číslo s. Jak najít indexy $i \leq j$, tž. $x_i + x_{i+1} + \ldots + x_j = s$ (nebo poznat, že taková dvojice není)?

*** Úloha 4.** Co když v předchozí úloze povolíme i záporná čísla?

*** Úloha 5.** Žijeme ve městě, jehož mapa tvoří *strom* (vrcholy jsou křižovatky, hrany ulice). Máme k policistů a chceme každého postavit do jedné ulice tak, že pokud se něco stane v libovolném vrcholu, cesta nejbližšího policisty do tohoto vrcholu (počet překonaných hran) je minimální.

Dīvide et imperā

Úloha 6. Vyřešte rekurenci $T(1) = 1,\ T(n) = \sqrt{n} \cdot T(\sqrt{n}) + \Theta(n)$.

Úloha 7. Vícecestný Mergesort s k „cestami“ dělí posloupnost na k stejně velkých částí, které rekurzivně setřídí a výsledky slije.
Nejprve ukažte, jak slévat k setříděných polí celkové délky n v čase $\mathcal{O}(n \log k)$.
Pak analyzujte časovou složitost vícecestného Mergesortu. Může takový algoritmus být rychlejší než standardní Mergesort?
Čemu odpovída n-cestný Mergesort?

Dynamické programování

Úloha 8. *(Nejdelší rostoucí podposloupnost)* Mějme zadanou posloupnost čísel $x_1, x_2, \ldots, x_n$. Chceme co nejméně čísel vyškrtat tak, aby nám zbyla rostoucí posloupnost.
Co když nás zajímá i *kolik různých* nejdelších rostoucích podposloupností zadaná posloupnost má?
Najděte algoritmus pracující v $\mathcal{O}(n^2)$. Dynamickým programováním je možné úlohu vyřešit i v $\mathcal{O}(n \log n)$.

Úloha 9. Mějme neorientovaný ohodnocený graf. Najděte DP algoritmus, který pro každou dvojici vrcholů najde jejich vzdálenost (což taky umožňuje v čase $\mathcal{O}(m+n)$ zrekonstruovat příslušnou cestu).
Pro porovnání, jak rychle by to dokázalo několikanásobné použití Dijkstrova algoritmu?

Úloha 10. *Asociativní násobení:*
Přímočaré násobení matic $A \in \mathbb{R}^{m \times k}$ a $B \in \mathbb{R}^{k \times n}$ vyžaduje $m \cdot k \cdot n$ součinů čísel.
Násobení matic je asociativní, tedy pokud násobíme více matic mezi sebou, výsledek nezáleží na uzávorkování. Časová složitost (počítaná pomocí počtu součinů čísel) ale na uzávorkování rozhodně záleží!
Vymyslete algoritmus, který stanoví, jak násobení několika matic uzávorkovat tak, abychom minimalizovali složitost.