Algoritmy a grafy 1 (BI-AG1), Cvičení č. 9

Dynamické programování, minimální kostry

Paralelka 104, Úterý 16:15-17:45

Cvičící: Šimon Lomič
lomicsim@fit.cvut.cz

Informace: lomicsim.github.io

Fakulta informačních technologií
České vysoké učení technické v Praze
https://courses.fit.cvut.cz/BI-AG1

(Verze dokumentu: 18. 12. 2018 15:21)

9.1 Dynamické programování

- Technika návrhu algoritmů založená na **rekurzivním rozkladu na podproblémy**. Běžná zadání:
  - **Nalezněte optimální řešení**, např. největší či nejmenší možné.
  - **Nalezněte počet řešení**.
- **Obecný top-down postup:**
  1. Vytvořte **rekurzivní** řešení problému
  2. **Memoizace:** tabulka (mapa) do které budeme zapisovat výsledky každého volání rekurze, opakované volání pak čte z této tabulky - každá kombinace parametrů rekurze se tak spočte pouze jednou.
  3. **Časová složitost:** Počet různých vstupů rekurze (odpovídá počtu klíčů mapy $=$ *velikost stavového prostoru*) $\times$ čas strávený v rekurzi.

Cvičení: Dána cílová hodnota V a seznam různých hodnot mincí $s_1, s_2, \ldots, s_n$. Určete minimální počet mincí, které musíme použít, abychom složili hodnotu V (každý typ mince lze použít neomezeně).

9.2 Dynamické programování

(a) Na nákup oblečení máme k dispozici M Kč. V obchodě mají t typů oblečení (trička, mikiny, ...), každý z nich v c provedeních: i-té provedení j-tého typu stojí $C[i][j]$ Kč. Zvolte nákup tak, abyste od každého typu koupily právě jeden kus, celkovou cenou nepřesáhli M a utratili maximum. Kolik nejvíce můžeme utratit? **(0.5 b)**

(b) Mějme pole čísel $A = (a_1, \ldots, a_n)$. Najděte nejdelší rostoucí podposloupnost.

(c) Nalezněte nejdelší rostoucí podposloupnost, ve které je povolen jeden pokles.

(d) Definujme editační vzdálenost řetězců S a T jako minimální počet operací *Přidej znak*, *Smaž znak* a *Nahraď znak* provedených na řetězci S tak aby výsledek operací byl řetězec T. Spočtěte ji pro dané S a T.

- Jak to udělat v lineární paměti? **(0.5 b)**

9.3 Dynamické programování na stromech

Obecný top-down postup:

1. Zakořeňme strom a vytvořme rekurzivní řešení.
2. Použijme memoizaci.

(a) **Minimální vrcholové pokrytí**. Mějme strom $T = (V, E)$. Nalezněte nejmenší možnou množinu vrcholů S takovou, aby pro každou hranu $e \in E$ platilo $e \cap S \neq \emptyset$ (alespoň jeden její vrchol byl vybrán). **(0.5 b)**

(b) Mějme strom T. Nalezněte počet různých podstromů (dva stromy jsou různé, pokud je tvoří různé množiny vrcholů). **(0.5 b)**

(c) Souvislý graf G označíme za *housenku*, pokud má každý jeho vrchol stupeň > 2 a po odstranění listů tvoří cestu. Nalezněte největší housenku v grafu (coby podgraf). **(0.5 b)**

9.4 Minimální kostra grafu

Kostra grafu G je podgraf, který obsahuje všechny vrcholy a je to strom a má tedy $|V(G)|$-1 hran.

Definice

- Nechť $G = (V, E)$ je souvislý neorientovaný graf a $w : E \to \mathbb{R}$ **váhová funkce**, která přiřazuje hranám čísla – jejich **váhy**.
- **Váha** $w(H)$ podgrafu $H \subseteq G$ je součet vah jeho hran.
- Kostra je **minimální**, pokud má mezi všemi kostrami nejmenší váhu.

Cvičení:

(a) Změníme-li váhu jedné hrany, jak se změní minimální kostra?

(b) Jak hledat kostru grafu, v němž jsou váhy hran čísla od 1 do 5?

(c) Navrhněte všechny detaily toho, jak Jarníkův algoritmus realizovat s haldou časovou složitostí $O(m \log n)$.

9.5 Domácí úkol

Na webu `lomicsim.github.io` jsou vystaveny domácí úlohy. Za každou správně vyřešenou úlohu získáte 0.5b (maximálně však lze takto získat 5 bodů za aktivitu včetně těch získaných v semestru). Řešení domácích úloh lze až do 6.1.2019 zasílat na `slomic@fit.cvut.cz`.