Domáca úloha č. 2

2-AIN-205, Leto 2015

Termín: 23.3.2015, 22:00, M-163 (pod dvere)

Skôr ako sa pustíte do riešenia domácej úlohy, oboznámte sa so všeobecnými pokynmi, ktoré sú priložené na konci tohto dokumentu. Riešenia, ktoré odovzdáte, musia byť vaše vlastné. Neopisujte a nesnažte sa nájsť riešenia v literatúre alebo na internete!

1. [*20 bodov*] **O byrokratoch.** V tejto úlohe budeme rozdeľovať prácu medzi k úradníkov. Je potrebné vykonať n administratívnych úkonov, pričom sú dané čísla $a_1, \ldots, a_n$, kde číslo a_i reprezentuje dĺžku i-teho administratívneho úkonu. Úkony možno vykonávať v ľubovoľnom poradí.

Našou úlohou je prideliť úkony úradníkom takým spôsobom, aby všetky úkony boli dokončené čo najskôr. Napríklad ak by sme mali rozdeliť 7 úkonov s dĺžkami 1, 4, 10, 7, 14, 2, 2 medzi 3 úradníkov, tak v optimálnom riešení by prvý úradník dostal jeden úkon s dĺžkou času 14, druhý úradník tri úkony s dĺžkami 10, 2, 2 a tretí úradník tri úkony s dĺžkami 7, 4, 1. Optimálne riešenie má teda trvanie 14.

V tejto úlohe budeme uvažovať nasledujúci algoritmus A:

> Usporiadajme úkony od najdlhšieho po najkratší. V tomto poradí prideľujeme úkony úradníkom s tým, že ďalší úkon vždy priradíme úradníkovi, ktorý má najmenej práce. (Ak je takých úradníkov viac, dostane úkon ľubovoľný z nich.)

a) Algoritmus A nedáva vždy optimálne riešenie. Nájdite taký príklad. (Uveďte aj výsledok algoritmu A a optimálne riešenie pre váš príklad.)

b) Existujú však situácie, keď vieme naopak dokázať, že algoritmus A dá vždy optimálne riešenie.

Dokážte nasledujúce tvrdenie: Nech t^* je trvanie optimálneho riešenia. Potom ak pre všetky $1 \leq i \leq n$ platí $a_i > t^*/3$, tak algoritmus A vráti vždy optimálne riešenie.

c) S použitím tvrdenia z časti b) ukážte, že algoritmus A je 4/3-aproximačný.

2. [*20 bodov*] **Efektívne pokrývanie množinami.** Pripomeňme si, že pri pokrývaní množinami (Set Cover) máme daných k množín $S_1, \ldots, S_k$, kde $S_i \subseteq \{1, \ldots, n\}$ a našou úlohou je nájsť najmenší možný počet množín, ktorých zjednotením sú všetky prvky $\{1, \ldots, n\}$. $O(\log n)$-aproximačný greedy algoritmus z prednášky funguje tak, že v každom kroku vyberie množinu, ktorá pokrýva najväčší počet ešte nepokrytých prvkov.

Navrhnite dátové štruktúry a napíšte podrobný pseudokód, ktorý implementuje tento greedy algoritmus tak, aby výsledná časová zložitosť bola čo najlepšia. Časovú zložitosť odhadnite vzhľadom ku parametrom k, n a u, kde $u = \sum_i |S_i|$.

3. [*20 bodov*] **Programátorská úloha** (viď všeobecné pokyny). Na vstupe je n s osami rovnobežných štvorcov s rozmermi 2×2. Ich ľavé dolné rohy sú na celočíselných súradniciach, kde pre súradnice platí $0 \leq x \leq 1000$, $0 \leq y \leq 2$.

Vyberte z nich, čo najviac tak, aby sa žiadne dve neprekrývali (ale môžu sa dotýkať).

Formát vstupu: Na prvom riadku je počet štvorcov n. Nasleduje n riadkov, na každom z nich sú dve čísla x, y – súradnice ľavého dolného rohu príslušného štvorca.

Formát výstupu: Vypíšte jeden riadok s jedným číslom – najväčší počet štvorcov, ktorý sa dá vybrať.

Obmedzenia a bodovanie: Na zisk plného počtu bodov je nutné, aby váš program dal správnu odpoveď pre vstupy, kde n je najviac 3000. V sadách vstupov za 25% bodov platí, že $n \leq 20$. V iných sadách vstupov za 25% bodov pre každý štvorec platí $y \neq 1$.

Príklad vstupu:

```
3
0 0
1 1
2 2
```

Príklad výstupu:

```
2
```

Všeobecné pokyny

Písomné úlohy. Písomné úlohy odovzdávajte *na papieri* (či už vytlačené alebo písané rukou) pod dvere kancelárie M-163 v stanovenom termíne. **Každý príklad odovzdajte na osobitnom liste papiera,** každý príklad bude opravovať iný človek. Na neskoro odovzdané riešenia sa nebude prihliadať. Nezabudnite na každý list jasne napísať svoje plné meno a priezvisko, v prípade že riešenie jedného príkladu je na viac listov, zopnite ich pevne spinkovacím strojčekom.

Píšte riešenia takým spôsobom, aby obsahovali všetku potrebnú informáciu na pochopenie vášho riešenia, ale súčasne aby boli stručné a ľahko pochopiteľné. Všetky tvrdenia je potrebné zdôvodniť (a to aj v prípade, že to nie je explicitne napísané v zadaní).

Ak sa v zadaní požaduje vyriešenie algoritmickej úlohy, odovzdajte najlepší algoritmus, aký viete navrhnúť. Základným kritériom na hodnotenie bude *správnosť algoritmu*, druhým kritériom bude jeho *časová, prípadne pamäťová zložitosť.* Správny ale pomalý algoritmus dostane podstatne viac bodov ako algoritmus, ktorý je síce rýchly, ale nedá správnu odpoveď na každý vstup. Neefektívne algoritmy spĺňajúce podmienky zadania dostanú cca 50% bodov. Súčasťou vášho riešenia musia byť nasledujúce časti:

- Najprv popíšte hlavnú myšlienku algoritmu.
- Vyjadrite algoritmus formou pseudokódu.
- Ak to nie je zrejmé na prvý pohľad, ukážte že váš algoritmus je správny.
- Nezabudnite na analýzu zložitosti algoritmu.

Programátorské úlohy. Pri programátorských úlohách je vašou úlohou odovzdať len funkčný program, nie je vyžadované písomné riešenie. Riešenie odovzdávate cez webové rozhranie `foja.dcs.fmph.uniba.sk/eval`, kde bude okamžite otestované na niekoľkých vstupoch a dozviete sa koľko bodov získalo (body získate, keď všetky vstupy z danej sady vyriešite správne v časovom limite). Riešenie môžete odovzdávať aj viackrát, hodnotí sa posledné riešenie odovzdané v stanovenom termíne. Navyše si dajte pozor, či v systéme máte správne vyplnené meno a priezvisko (sekcia Môj účet). Podrobnosti o tom, ako má váš program vyzerať (vrátane povolených programovacích jazykov), nájdete v sekcii Návod.