Proiectarea Algoritmilor 2011-2012

Laborator 2-3 - Aplicații de laborator

Greedy & Programare Dinamica

Laborator 2

1. Problema rucsacului (varianta continuă) [2 pct]

Un camion poate transporta T tone de material. Există n tipuri de materiale disponibile, fiecare caracterizat de greutatea G_i disponibila și de valoarea V_i adusă de transportarea sa. Să se decidă ce cantități din fiecare material vor fi transportate pentru a aduce o valoare maximă.

2. Descompunerea numerelor raționale în fracții egiptene [2 pct]

O fracție egipteană este o fracție ireductibilă scrisă ca sumă de fracții unitare (1/n). Scrieți un algoritm care realizează automat această descompunere, într-o manieră eficientă din punctul de vedere al timpului de execuție.

Hint: algoritmul lui Fibonacci

$$\frac{x}{y} = \frac{1}{\lceil y/x \rceil} + \frac{(-y) \bmod x}{y\lceil y/x \rceil}$$

3. Selecția activităților [3 pct]

Trudi este un robot foarte conștiincios, dar are programul foarte încărcat și trebuie să-și facă ordine în activități (din păcate are o arhitectură mai veche, fiind single-tasking). Astfel, pentru o mulțime S de taskuri definite printr-un timp de start și unul de finish, scrieți un algoritm optim care selectează un număr maxim de taskuri care nu se suprapun și pe care Trudi să le execute.

4. Problema rucsacului (varianta discreta) [3 pct]

Un camion poate transporta T tone de mobilă. Există n piese de mobilă, fiecare caracterizată de greutatea G_i și de valoarea V_i adusă de transportarea sa. Să se decidă ce piese de mobilă vor fi transportate pentru a aduce o valoare maximă.

Laborator 3

5. Parantezarea unei expresii booleene [3 pct]

Fiind dată o expresie booleană exprimată prin stringurile “true”, “false”, “and”, “or”, “xor”, identificați numărul de parantezări complete (moduri în care se pot aşeza paranteze astfel încât rezultatul final să fie “true”).

Spre exemplu, pentru expresia:

```
true and false xor true
```

există două modalități de parantezare astfel încât rezultatul să fie “true”.

6. Înmulțirea optimă a unui lanț de matrici [2 pct]

Întrucât înmulțirea de matrici este asociativă, determinați modul optim de realizare al înmulțirilor unui şir A1…An de matrici astfel încât complexitatea înmulțirilor să fie minima.

Ex. Fie A1 - 10 × 30; A2 - 30 × 5; A3 - 5 × 60:

$(A_1 \text{ X } A_2) \text{ X } A_3 = (10 \times 30 \times 5) + (10 \times 5 \times 60) = 1500 + 3000 = 4500$ operații

$A_1 \text{ X } (A_2 \text{ X } A_3) = (30 \times 5 \times 60) + (10 \times 30 \times 60) = 9000 + 18000 = 27000$ operații

7. Word wrapping [5 pct]

O problemă clasică a editoarelor de text este facilitatea de text wrapping. Textul este considerat un vector de N cuvinte word[1..N], iar lungimea unui rand este L. Pentru ambele variante este necesar să afișați textul formatat.

Primul lucru pe care trebuie sa-l implementați este abordarea Greedy utilizată în majoritatea editoarelor (MS Words, suita Open Office). **[2 pct]**

Acum că avem o versiune etalon, ne dorim să îmbunătțim aspectul său estetic prin utilizarea unei funcții de cost mai restrictive şi anume pătratul numărului de spații libere rămase la sfârşitul fiecărei linii. Implementați această logică utilizată în TeX pentru a rafina designul unui control de tip TextArea. **[3 pct]**

Ex. Fie textul inițial **aaa bb cc ddddd** şi diminesiunea rândului **6**.

O abordare Greedy clasică ar întoarce următorul rezultat:

```
aaa bb     Spații: 0 (cost1 = 0 cost2 = 0)
cc         Spații: 4 (cost1 = 4 cost2 = 16)
ddddd      Spații: 1 (cost1 = 1 cost2 = 1)
           cost1 = 5 cost2 = 17
```

dar versiunea cu un aspect estetic îmbunătățit ar trebui să întoarcă următorul rezultat:

```
aaa        Spații: 3 (cost1 = 3 cost2 = 9)
bb cc      Spații: 1 (cost1 = 1 cost2 = 1)
ddddd      Spații: 1 (cost1 = 1 cost2 = 1)
           cost1 = 5 cost2 = 11
```