Exam in Algorithms & Datastructures II (1DL231)

Monday 15 December 2014 from 14:00 to 17:00, in Polacksbacken

This is a multiple-choice exam, but you are first going to answer classically. The ***answer sheet*** will be made available ***at 16:00***, so that you can then identify your answers among the listed candidate answers. You ***hand in only that answer sheet***. You are ***not*** expected to explain your answers; if however you think a question is unclear or wrong, then mark its number with a $\star$ on ***the answer sheet***, and explain ***on the backside of the answer sheet*** what your difficulty with the question is ***and*** what assumption underlies the answer you have chosen or the other answer you indicate. Normally, a teacher will attend this exam from 15:00 to 16:00.

Topic 1: Maximum Flow $(5 \cdot 2 = 10$ points$)$

Consider the factories a, b, c, d as well as the shops e, f, g, h of chocolate manufacturer Chokla. Consider a supply infrastructure that can be represented as a weighted digraph with the factories and shops as the vertices, and its trucks as the weighted arcs, represented by the adjacency lists $\text{Adj}[a] = [\langle e, 5\rangle, \langle g, 3\rangle]$, $\text{Adj}[b] = [\langle e, 1\rangle, \langle f, 3\rangle]$, $\text{Adj}[c] = [\langle e, 4\rangle, \langle g, 4\rangle, \langle h, 3\rangle]$, $\text{Adj}[d] = [\langle h, 3\rangle]$, indicating for instance that a truck can supply maximum 5 tons of chocolate from factory a to shop e. The factories respectively have $3, 3, 5, 3$ tons of chocolate in store. The shop managers estimate that their customer demands respectively are $3, 4, 3, 3$ tons of chocolate. Design a flow network with source s, sink t, and the factories, shops, and trucks ***in the orders given above***, so as to model the problem of determining the maximum total customer demand that the trucks can actually supply from the factories. Answer the following questions:

1. Apply the Ford-Fulkerson method to the designed flow network. Always take the ***first shortest*** augmenting path. What is the ***last*** augmenting path that you have applied? Use the notation $n : [s, \alpha, \beta, t]$ to denote that n units flow along the augmenting path from source s to sink t via first node α and then node β.
2. What is the number of augmenting paths for computing the maximum flow for Question 1?
3. Redo Question 1, but take the ***second of the shortest*** augmenting paths at the first iteration; at all subsequent iterations, always take the ***first shortest*** augmenting path. What is the number of arcs on the ***longest*** augmenting path that you have applied?
4. In the maximum flow for Question 3, what is the net transport from factory c to shop e?
5. What is the number of augmenting paths for computing the maximum flow for Question 3?

Topic 2: P versus NP $(5 \cdot 2 = 10$ points$)$

Complete the following sentences and answer the following questions:

6. NP is the class of decision problems whose solutions take ...
7. Reducing a problem Q to an existing NP-complete problem shows ...
8. Which of the following decision problems is not NP-complete: existence of a simple path of at least a given weight? 2-SAT? 3-SAT? existence of a clique of a given size? or existence of a subset of a given sum?
9. Toward solving the optimisation problem of Topic 3, a useful decision problem is as follows: Given an array $D[1..n]$ of n integer coin denominations in strictly decreasing order, with $D[n] = 1$, and given two integers $m \geq 0$ and $b \geq 0$, is there a number c of coins, with values in D and total value m, such that ...

10. To what complexity class belongs an optimisation problem (say the one of Topic 3) if a decision problem for it (say the one of Question 9) is NP-complete?

Topic 3: Dynamic Programming $(6 \cdot 2 = 12$ points$)$

Specification: Given an array $D[1..n]$ of n coin denominations in strictly decreasing order, and a number $m \geq 0$, compute the minimum number of coins, with values in D, whose total value is m. Assume all numbers are non-negative integers and $D[n] = 1$.
Example 1: Given the $n = 4$ denominations $D = [10, 5, 2, 1]$ of Swedish coins, the minimum number of coins for $m = 9$ Swedish Crowns is 3, for $5 + 2 + 2 = 9$.
Example 2: Given the $n = 4$ denominations $D = [5, 4, 3, 1]$ of Duckburg coins, the minimum number of coins for $m = 7$ Duckburg Dollars is 2, for $4 + 3 = 7$.

Consider the following recurrence for a quantity $M[i]$, parameterised by α and β:

$$M[i] = \begin{cases} 0 & \text{if } i = 0 \\ 1 + \min\{M[\alpha] \mid 1 \leq k \leq n \wedge \beta\} & \text{if } 0 < i \leq m \end{cases}$$

Answer the following questions:

11. What is the meaning of the integer $M[i]$, with $0 \leq i \leq m$, when $M[m]$ is the value returned by an algorithm that is correct with respect to the specification above?
12. For the problem instance in Example 2 above, what is $\sum_{i=0}^{m} M[i]$?
13. What is the index expression α in the recursive case?
14. What is the formula β in the recursive case?
15. What is an ordering of the indices i under which the cells $M[i]$ of the array M can be filled without performing any redundant computations?
16. What is the run-time complexity of the dynamic program resulting from the correct answers for Questions 13 to 15? Give the ***tightest*** complexity.

Topic 4: Greedy Algorithms $(7 \cdot 2 = 14$ points$)$

Below is a greedy algorithm, parameterised by $\alpha, \beta_1, \beta_2, \beta_3, \beta_4, \gamma, \delta$, for the specification at Topic 3: at every iteration, it uses, if possible, one coin of the largest denomination that has at most the current total value, and recurses for the remaining total value; when looking for a denomination, it avoids iterating over the entire array D, by exploiting the preconditions on D. The main call is $\textsc{Change}(1, m)$ and D is a global variable:

1: **function** $\textsc{Change}(k, i)$ $\{$**variant:** $\alpha\}$
2: **if** $i = \beta_1$ **then**
3:     **return** β_2
4: **else**
5:     **if** $i \geq \beta_3$ **then**
6:         **return** $\beta_4 + \textsc{Change}(\gamma)$
7:     **else**
8:         **return** $\textsc{Change}(\delta)$

Answer the following questions, which are totally ***independent*** of the questions for Topic 3:

17. What is the sum of the numeric expressions $\beta_1, \beta_2, \beta_3, \beta_4$ in lines 2 to 6?
18. What are the arguments γ to the recursive call in line 6?
19. What are the arguments δ to the recursive call in line 8?
20. What is a ***decreasing*** recursion variant α for line 1, establishing algorithm termination?
21. What is the run-time complexity of the greedy algorithm resulting from the correct answers for Questions 17 to 19? Give the ***tightest*** complexity.
22. When does this greedy algorithm not give the optimal answer?
23. Is there an asymptotically faster greedy algorithm when $m > n$, giving the same answer?

Algorithms & Datastructures II (1DL231): Answer Sheet

Cross over, with a ✗, ***at most one*** answer letter (A to E) per question; circles and all other indicators will be ignored. Please re-read the other instructions on the first page.

1. (A) $0.5 : [s, d, h, t]$ (B) $1 : [s, d, h, t]$ (C) $2 : [s, d, h, t]$ (D) $3 : [s, d, h, t]$ (E) another one
2. (A) 3 (B) 4 (C) 5 (D) 6 (E) 7
3. (A) 3 (B) 5 (C) 7 (D) 9 (E) 11
4. (A) 0 (B) 1 (C) 2 (D) 3 (E) 4
5. (A) 3 (B) 4 (C) 5 (D) 6 (E) 7
6. (A) …polynomial time to check. (B) …polynomial time to find.
   (C) …*non*-polynomial time to check. (D) …*non*-polynomial time to find.
   (E) …possibly forever to find.
7. (A) …that Q is in P. (B) …that Q is NP-complete. (C) …that Q is NP-hard.
   (D) …that Q is undecidable. (E) …nothing useful about Q.
8. (A) Existence of a simple path of at least a given weight (B) 2-SAT (C) 3-SAT
   (D) Existence of a clique of a given size (E) Existence of a subset of a given sum
9. (A) $\ldots c > 0$? (B) $\ldots c \geq b$? (C) $\ldots c \leq b$? (D) $\ldots c < m$? (E) $\ldots c = m$?
10. (A) Greedy (B) P (C) NP-complete (D) NP-hard (E) Undecidable
11. (A) $M[i]$ is the minimum number of coins, with values in $D[1..i]$, whose total is i.
    (B) $M[i]$ is the minimum number of coins, with values in $D[1..i]$, whose total is m.
    (C) $M[i]$ is the minimum number of coins, with values in $D[i..n]$, whose total is i.
    (D) $M[i]$ is the minimum number of coins, with values in $D[i..n]$, whose total is m.
    (E) $M[i]$ is the minimum number of coins, with values in $D[1..n]$, whose total is i.
12. (A) 7 (B) 8 (C) 9 (D) 10 (E) 11
13. (A) $i - k$ (B) $i + k$ (C) $i - D[k]$ (D) $i + D[k]$ (E) $D[k]$
14. (A) $D[k] \leq m$ (B) $D[k] + i \leq m$ (C) $k \leq i$ (D) $i + k \leq m$ (E) $D[k] \leq i$
15. (A) any (B) by increasing i (C) by decreasing i (D) middle-out (E) instance-specific
16. (A) $\mathcal{O}(m)$ (B) $\mathcal{O}(n)$ (C) $\mathcal{O}(m + n)$ (D) $\mathcal{O}(m \cdot n)$ (E) $\mathcal{O}(m^2)$
17. (A) 2 (B) $D[1] + 1$ (C) $D[k] + 1$ (D) $D[k] + i \text{ div } D[k]$ (E) $D[k] + i \bmod D[k]$
18. (A) $k,\ i \bmod D[k]$ (B) $k + 1,\ i \bmod D[k]$ (C) $k + 1,\ i \text{ div } D[k]$
    (D) $k + 1,\ i - D[k]$ (E) $k,\ i - D[k]$
19. (A) $k - 1,\ i$ (B) $k,\ i$ (C) $k + 1,\ i$ (D) $1,\ i$ (E) $1,\ m$
20. (A) i (B) k (C) $i + k$ (D) $n - k$ (E) $i + n - k$
21. (A) $\mathcal{O}(m)$ (B) $\mathcal{O}(n)$ (C) $\mathcal{O}(m + n)$ (D) $\mathcal{O}(m \cdot n)$ (E) $\mathcal{O}(m^2)$
22. (A) on odd m (B) on even m (C) on prime m (D) in ex.1 of Topic 3 (E) in ex.2 of Topic 3
23. (A) yes: $\mathcal{O}(m)$ (B) yes: $\mathcal{O}(n)$ (C) yes: $\mathcal{O}(m + n)$ (D) yes: $\mathcal{O}(m \cdot n)$ (E) no

Recall that floor division (denoted '//' in Python) and modulo (denoted '%' in Python) are often denoted 'div' and 'mod': for example, $9 \text{ div } 2 = 4$ and $9 \bmod 2 = 1$, because $9 = 2 \cdot 4 + 1$.

Grading: Your grade is as follows, when your exam mark is e points:

Grade	Condition
5	$38 \leq e \leq 46$
4	$30 \leq e \leq 37$
3	$23 \leq e \leq 29$
U	$00 \leq e \leq 22$

Identity: Your anonymous exam code (or name and personal number, if you have none):

. . . .