Algorithms & Data Structures 2 (1DL231) **Exam of 7 Jan 2019**

Instructions: This is a multiple-choice exam, in order to save you the time of tidying up the presentation of your answers. There is exactly ***one*** correct answer per ***assessed*** question. You can keep this questionnaire and should ***hand in only the answer sheet***: you are ***not*** expected to explain your answers. Unfortunately, the head teacher cannot attend this exam. ***Also read the instructions on the answer sheet before starting***.

0 Information About You

Question -1: (not assessed) What is your study programme?

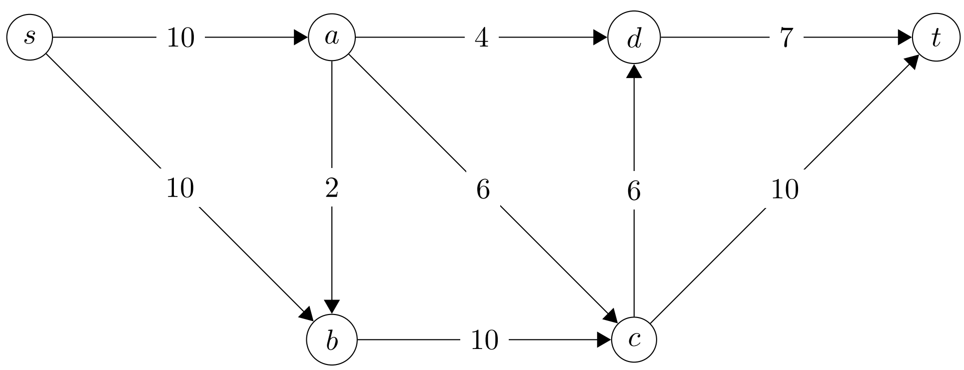
[A] DVK [B] IT [C] STS [D] master [E] exchange

Question 0: (not assessed) How many of the 12 lectures did you attend?

[A] 0 . . 1 [B] 2 . . 4 [C] 5 . . 7 [D] 8 . . 10 [E] 11 . . 12

1 Maximum Flow

Consider the following flow network with source s and sink t:



Question 1: After augmenting along the path $s \to a \to c \to t$, along $s \to a \to b \to c \to t$, and finally along $s \to b \to c \to d \to t$, what is of the following paths is an augmenting paths with the highest capacity?

[A] $s \to b \to a \to d \to t$, capacity 1

[B] none, the reached flow value is optimal

[C] $s \to b \to c \to t$, capacity 2

[D] $s \to b \to c \to d \to t$, capacity 1

[E] $s \to b \to c \to a \to d \to t$, capacity 1

Question 2: Are the flows across ***all*** cuts after the 3 augmentations of Question 1 equal and to which value?

[A] yes: 14 [B] yes: 15 [C] yes: 17 [D] yes: 18 [E] no

Question 3: What is the maximum flow value (after ***all*** possible augmentations)?

[A] 16 [B] 17 [C] 18 [D] 19 [E] 20

Question 4: What is the capacity of a minimum-capacity (s, t)-cut?

A 16 B 17 C 18 D 19 E 20

2 Greedy Algorithms

Consider lectures $\ell_1, \ell_2, \ldots, \ell_n$ for which the same classroom is requested. Each lecture ℓ_i has a start time s_i and a finish time f_i, where $0 \leq s_i < f_i < \infty$. We wish to select a largest subset of lectures of which no two overlap in time. If selected, then ℓ_i happens during the half-open time interval $[s_i, f_i)$. For example, consider the following set of lectures:

i	1	2	3	4	5	6	7
s_i	10	3	1	6	7	4	2
f_i	14	5	2	9	14	11	10

The subset $\{\ell_3, \ell_6\}$ has non-overlapping lectures but $\{\ell_1, \ell_3, \ell_4\}$ is larger. Consider the following greedy algorithm template:

```
GREEDY-LECTURE-SELECTOR(n, [s_1, ..., s_n], [f_1, ..., f_n])
1  in-place sort the lectures by some criterion
2  L := {ℓ_1}        // L is the current subset of selected non-overlapping lectures
3  for i := 2 to n
4      if lecture ℓ_i does not overlap with any lecture already in L
5          L := L ∪ {ℓ_i}
6  return L
```

Question 5: Assume we in-place sort in line 1 the lectures by monotonically increasing start time, giving $s_1 \leq s_2 \leq \cdots \leq s_n$: on which subsets of lectures taken from the set given above does the greedy algorithm above return a largest subset of non-overlapping lectures?

A all B none C some D undecidable E we do not know

Question 6: Assume we in-place sort in line 1 the lectures by monotonically increasing duration, giving $f_1 - s_1 \leq f_2 - s_2 \leq \cdots \leq f_n - s_n$: on which which subsets of lectures taken from the example given above does the greedy algorithm above return a largest subset of non-overlapping lectures?

A all B none C some D undecidable E we do not know

Question 7: Assume we in-place sort in line 1 the lectures by monotonically increasing finish time, giving $f_1 \leq f_2 \leq \cdots \leq f_n$: on which inputs does the greedy algorithm above return a largest subset of non-overlapping lectures?

A all B none C some D undecidable E we do not know

Question 8: Assume we in-place sort in line 1 the lectures by monotonically increasing duration, giving $f_1 - s_1 \leq f_2 - s_2 \leq \cdots \leq f_n - s_n$: given any possible, set of lecture start and finish times, not necessarily taken from the example above, when does the greedy algorithm above return a largest subset of non-overlapping lectures?

- [A] all
- [B] none
- [C] some
- [D] un-decidable
- [E] never

Question 9: What is the *tightest* time complexity of the greedy algorithm in Question 5?

- [A] $\mathcal{O}(n)$
- [B] $\mathcal{O}(n \cdot \lg n)$
- [C] $\mathcal{O}(n^2)$
- [D] $\mathcal{O}(n^2 \cdot \lg n)$
- [E] $\mathcal{O}(n^3)$

3 Dynamic Programming

Given n objects of known volumes $v_1, \ldots, v_n$ and prices $p_1, \ldots, p_n$, as well as a bag of known volume V, we would like to determine the total price of the most expensive subset of the given objects that fits into the given bag. All numbers are positive integers. For example, given four objects of volumes $2, 1, 3, 2$ and prices $13, 10, 20, 17$ respectively, the most expensive subset fitting into a bag of volume $V = 5$ includes all except the third object; it has total volume $2 + 1 + 2 = 5 \leq V$ and total price $13 + 10 + 17 = 40$. Consider the following recurrence for a quantity $T[i, v]$, parameterised by α, β_1, $\ldots$, β_5, and γ:

$$T[i,v] = \begin{cases} 0 & \text{if } \alpha \\ T[i-1, \beta_1] & \text{if } \neg\alpha \text{ and } v < \beta_2 \\ \gamma(T[i-1, \beta_3],\ T[i-1, \beta_4] + \beta_5) & \text{if } \neg\alpha \text{ and } v \geq \beta_2 \end{cases}$$

Question 10: If $T[n, V]$ is returned by a correct algorithm for the specification above, then the integer $T[i, v]$, with $0 \leq i \leq n$ and $0 \leq v \leq V$, denotes the total price of ...

- [A] ...the most expensive subset of exactly i objects that fits into a bag of volume v.
- [B] ...the most expensive subset of at least i objects that fits into a bag of volume v.
- [C] ...the most expensive subset of at most i objects that fits into a bag of volume v.
- [D] ...the most expensive subset of the first i objects that fits into a bag of volume v.
- [E] ...the most expensive subset of i random objects that fits into a bag of volume v.

Question 11: For the instance in the example above, what is the sum $T[3, 3] + T[2, 4]$?

- [A] 23
- [B] 46
- [C] 50
- [D] 52
- [E] other

Question 12: What is the logical condition α?

- [A] $i = 0$
- [B] $v = 0$
- [C] $v = V$
- [D] $v = i$
- [E] $v + i = 0$

Question 13: What is the correct expression for β_4 in terms of v_i,p_i and v?

- [A] $v - v_i + p_i$
- [B] $v + v_i - p_i$
- [C] $v + p_i$
- [D] $v - v_i$
- [E] $v - p_i$

Question 14: What is the two-argument operator γ (written in infix form above)?

- A $+$
- B $-$
- C $\cdot$
- D max
- E min

Question 15: Assuming that recurrence $T[i, v]$ in Question 10 is implemented with an array T that is filled without performing any redundant computations, what best describes the time complexity of computing $T[n, V]$.

- A $\mathcal{O}(n)$
- B $\mathcal{O}(n^2)$
- C Pseudo polynomial
- D $\mathcal{O}(V)$
- E $\mathcal{O}(V^n)$

4 Complexity

Question 16: To the best of our knowledge a decision problem is in NP if and only if its answer takes ...

- A ...non-polynomial time to check.
- B ...non-polynomial time to compute.
- C ...polynomial time to check.
- D ...polynomial time to compute.
- E ...possibly forever to compute.

Question 17: To the best of our knowledge a decision problem is NP-hard if and only if ...

- A ...it can be reduced in polynomial time to every problem in NP.
- B ...it can be reduced in polynomial time to every NP-complete problem.
- C ...every problem in P can be reduced in polynomial time to it.
- D ...every problem in NP can be reduced in polynomial time to it.
- E ...every NP-complete problem can be reduced in polynomial time to it.

Question 18: If the best known solution checker for a decision problem D takes $\mathcal{O}(n^{k^2})$ time on an instance of size n, for a constant $k > 0$, then what is the ***tightest*** complexity class of D, according to current knowledge?

- A P
- B NP
- C NP-complete
- D NP-hard
- E we do not know

Question 19: Given an algorithm that is implemented using dynamic programming which of the following statements best describes the complexity of the implementation.

- A Always polynomial time
- B Always NP-complete
- C Always pseudo polynomial
- D $\mathcal{O}(n^2)$
- E we do not have enough information

Question 20: Which of the following statements are true

- A Finding the longest simple path from a given source to a given target in a graph can be solved in polynomial time.
- B Finding the longest simple path from a given source to a given target in a graph can be converted into a shortest path problem by considering the complement of the input graph.
- C Finding the longest simple path from a given source to a given target in a graph is NP-complete.
- D Finding the shortest path from a given source to a given target is *not* in the complexity class NP.
- E There is a dynamic programming algorithm that finds the longest simple path from a given source that runs in polynomial time and space.

Corrected

Answer Sheet — AD2 Exam (1DL231) of 7 Jan 2019

Instructions: Do ***not*** alter the drawing above. Using a ***very dark*** colour, ***fill in*** entirely (only like this: ■) ***at most one*** answer box (A to E) per question: we will use an optical character recognition (OCR) system that ignores circles, crosses, ticks, etc. Transfer your answers from the questionnaire to this answer sheet ***just before handing in***; if an answer becomes ambiguous to an OCR system, then request another answer sheet. Every correct answer to an ***assessed*** question gives 1.5 points. Every multiple answer or incorrect answer gives 0 points. Partial credit of 0.75 point may be given in exceptional circumstances. If you think a question is unclear or faulty, then mark its number with a ⋆ on ***this sheet*** and explain ***on the backside of this sheet*** what your difficulty with the question is ***and*** what additional assumption underlies the candidate answer that you have chosen or the new answer that you indicate.

Again: Use a ***very dark*** colour to ***fill in*** your chosen boxes ***entirely*** (like this: ■)!

0 Information About You

Question -1: ■ ■ ■ ■ ■

Question 0: ■ ■ ■ ■ ■

1 Maximum Flow

Question 1: [A] [B] ■ [D] [E]

Question 2: ■ [B] [C] [D] [E]

Question 3: [A] ■ [C] [D] [E]

Question 4: [A] ■ [C] [D] [E]

2 Greedy Algorithms

Question 5: [A] [B] ■ [D] [E]

Question 6: ■ [B] [C] [D] [E]

Question 7: ■ [B] [C] [D] [E]

Question 8: [A] [B] ■ [D] [E]

Question 9: [A] ■ [C] [D] [E]

3 Dynamic Programming

Question 10: [A] [B] [C] ■ [E]

Question 11: [A] ■ [C] [D] [E]

Question 12: ■ [B] [C] [D] [E]

Question 13: [A] [B] [C] ■ [E]

Question 14: [A] [B] [C] ■ [E]

Question 15: [A] [B] ■ [D] [E]

4 Complexity

Question 16: [A] [B] ■ [D] [E]

Question 17: [A] [B] [C] ■ [E]

Question 18: [A] ■ [C] [D] [E]

Question 19: [A] [B] [C] [D] ■

Question 20: [A] [B] ■ [D] [E]

Your anonymous exam code:						