DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics and Computer Science

TU Delft

Exam CSE2310 Algorithm Design

January 27, 2022

- Use of the book, notes, or a calculator during this test is not allowed.
- This exam contains 8 open questions (worth a total 28 points) contributing 7/10 to your grade, and 8 multiple choice questions contributing 3/10 towards your grade. Note that the final mark for this course also consists for 60% of the unrounded score for the digital exams (if ≥ 5.0).
- The open questions require a bit more than one hour of your time:
  - Please answer in clear English and write legible (first use pencil or scrap paper).
  - Do not give any irrelevant information. This might lead to subtraction of points.
  - If an algorithm is asked, please provide the most efficient algorithm in terms of asymptotic time complexity. Providing a suboptimal algorithm can lead to a subtraction of points.
  - If asked for pseudocode, you may call algorithms from the book unless specifically asked for the respective pseudocode.
  - Before handing in your solutions, please verify that your name and student number are on every page.
- Regarding the multiple choice questions:
  - Each question has only one correct answer.
  - All questions count equally towards the grade. In computing the grade we will correct for the effects of random guessing. If you do not provide an answer, this is treated as a wrong answer. (Note: this means guessing is *never* worse than leaving your answer blank!)
  - Try to spend no more than one hour on the multiple choice questions.
- The exam material consists of modules 1–4 from the skill circuits of the course.
- Total number of pages (without this preamble): 5.

Open questions

1. (3 points) Give an asymptotic tight upper and lower bound for $T(n)$ in the following recurrence relation. Indicate the steps you use to get to your solution.

   $T(n) = 2T(n/4) + n^2$

2. (3 points) Consider the algorithm for finding the closest pair of points in a 2D-plane. Explain what role sorting plays in the algorithm and how the lack of sorting would affect the run time of the algorithm.

3. A TA can only start grading an exam after it has been scanned (and imported in a tool called *zesje*). Fortunately the TU Delft has more scanners available than there are TAs. After an exam has been scanned and graded by a TA, this is checked by the lecturer. When this has been done for all exams, the grades are announced to the students.

   Unfortunately some students write slightly more than is needed, whereas others write too little, and so there is a lot of fluctuation in how long it takes to scan and then grade and check an exam.

   Assuming we have infinitely many scanners and TAs available for scanning and grading (what a world that would be!), we propose the following algorithm to get the students their final grades as quickly as possible (and return this finish time). Here s_i, g_i, c_i represent the time for a TA to *scan* the exam, for a TA to *grade* the exam, and for the (single) lecturer to *check* the exam respectively.

   Sort exams in increasing order of their check time c_i.
   Use this new ordering for the following.
   $f_0 \leftarrow 0$
   for $i \leftarrow 1$ to n **do**
   $\quad f_i \leftarrow \max(f_{i-1}, s_i + g_i) + c_i$
   end for
   return f_n

   (a) (1 point) Provide a counterexample including at most 3 exams to prove this does *not* result in the grades being released as quickly as possible.

   (b) (1 point) What should we sort on instead?

   (c) (5 points) Prove that this returns the optimal solution using an exchange argument.

4. (2 points) Consider the following sequence of jobs in an optimal caching problem. Explain what values are cached and show how the cache changes throughout the operations.

   Initial values in the cache (cache size = 3): a, b, h

   Sequence of values retreived: $i, a, h, a, h, e, i, a, b, e$.

5. (5 points) Continuing your adventures on Taskmaster, you are now tasked to "make the best meal for the taskmaster using ingredients beginning with every letter of the alphabet."[1] Seeing how the taskmaster is a fan of food, your plan is to feed him the meal with the most calories. For each letter of the alphabet (the show is a bit strange, so they give you an alphabet of n letters not necessarily just 26), you have a collection of ingredients C_i for each letter of the alphabet i to choose from, and you must choose exactly one from each collection.

   However the task is timed, you have only T minutes to actually prepare the meal! Each ingredient has a time of $t_{i,j}$ required to prepare it (you cannot prepare multiple ingredients in parallel) and a number of calories expressed as $k_{i,j}$. Here i represents the collection C_i the ingredient is from, and j the index within the collection.

   Give a recursive formula that expresses the largest possible number of calories your meal can contain. Also show which initial parameter values to pass on to this recursive function to compute this optimal value. Return $-\infty$ if there is no feasible solution.

[1] As little Alex Horne made the constestants do in Series 1 Episode 6.

6. (2 points) Consider the following error table $e_{i,j}$ (on the left) and the memoisation table (on the right) for an instance of the segmented least squares problem where $C = 3$.

   As a reminder, the recursive formula for the optimal solution to the segmented least squares problem is:

$$OPT(j) = \begin{cases} 0 & \text{if } j = 0 \\ \min_{1 \le i \le j} (e_{i,j} + C + OPT(i-1)) & \text{else} \end{cases}$$

   How many line segments are used in the end and which? Explain how you derived your answer from the memoisation table.

i \ j	1	2	3	4	5	6	7
1	4	6	8	2	15	15	15
2	5	7	3	4	18	19	18
3	4	9	2	9	17	14	16
4	3	4	7	4	1	12	13
5	1	9	7	5	5	5	9
6	1	5	9	2	4	5	6
7	2	6	9	3	6	8	5

index	0	1	2	3	4	5	6	7
value	0	7	9	11	5	13	13	17

7. (2 points) Consider Figure 1, with lower bounds (if applicable) and capacities depicted on the edges and demands/supplies depicted in the nodes. Convert this network to one on which we can apply Ford-Fulkerson's algorithm. Show at least one intermediate step.

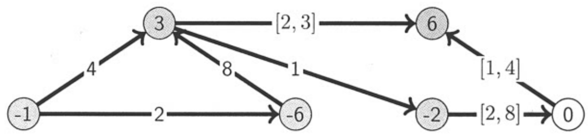


Figure 1: Network to convert

8. When sending instrumentation into space for experiments, there is always a trade-off between money spent sending the materials into space and the expected returns from the experiments. Given n experiments which will earn you b_i upon completion and m instruments which cost you c_j to send up into space, as well as sets R_i of instruments required to run an experiment i, you seek to maximise your profit. Note that if you have sent up an instrument multiple experiments can use it. The total profit P is the sum of the b values of the experiments run, minus the sum of the c values of the instruments sent into space.

   We solve this problem by building a graph G as follows and then the minimum cut tells us what instruments to send up to space.

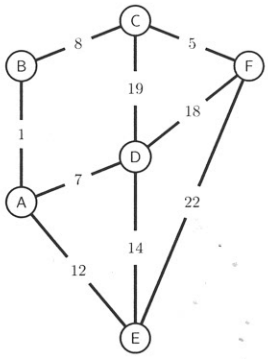
   1. Create a source node s and a sink node t.
   2. For every experiment create a node e_i.
   3. For every instrument create a node k_j.
   4. Connect s to every e_i with capacity b_i.
   5. Connect every k_j to t with capacity c_j.
   6. For every i connect e_i to all k_j corresponding to the instruments in R_i with capacity ∞.

   (a) (1 point) Explain *how* we can derive what instruments to send up to space from the minimum cut of G.

   (b) (3 points) Prove that your answer from part (a) combined with our method to building the graph G results in the optimal solution for this problem.

Multiple-Choice questions

9. Given the following graph G.



Which of the statements below is **true**?

A. There are two unique MSTs of this graph.

B. There is no MST that contains the edge $\{A, D\}$.

C. Kruskal and Prim-Jarnik would find the same MST for G.

D. Changing the weight of the $\{E, F\}$ edge to 15 would lead to it being included in the MST.

10. In this time of videostreaming, collegerama has asked for your help. They have to send n video streams sequentially, where each stream i consists of b_i bits that need to be sent (at a constant rate) over a period of t_i seconds. Furthermore, to prevent their servers from collapsing, the total number of bits sent over a time interval $[0, t]$ should never exceed $r \cdot t$ for any time t. Finally, they cannot afford any delays between streams and should start at $t = 0$. The next stream must start immediately after the previous one ends. Your job is to figure out if it is possible to do so. The algorithm you create is the following (here f keeps track of the finish time and c of the total bits sent so far).

```
Sort streams ...
Use this new ordering for the following.
f_0 ← 0
for i ← 1 to n do
    f_i ← f_{i-1} + t_i
    c_i ← c_{i-1} + b_i
    if r · f_i < c_i then
        return false
    end if
end for
return true
```

In what order should we sort the streams for this algorithm to work?

A. ascending b_i

B. descending t_i

C. ascending b_i/t_i

D. desceding $b_i + t_i$

11. Consider the following recursive formula to compute an optimal value:

$$OPT(i,j) = \begin{cases} 0 & \text{if } i = j \\ \max(v_i + OPT(2 \cdot i, j), v_j + OPT(i, j/2)) & \text{else} \end{cases}$$

We find the optimal solution from: $OPT(1,n)$ for some value n, where n is a power of 2. You may assume v contains n positive values.

Which of the following is **true** about this formula?

A. This problem cannot be solved in $O(n)$ time.

B. A divide & conquer approach for this problem requires $O(n)$ time.

C. The time complexity is pseudopolynomial, meaning it depends on the $\max_i(v_i)$ value.

D. The time complexity can be decreased from $O(2^n)$ without memoisation to a tight bound of $O(n)$ when using memoisation.

12. Consider the following recursive solution to a DP problem:

$$OPT(i,j) = \begin{cases} 1 & \text{if } i = 0 \\ OPT(i-1, j) & \text{if } v_i > m - j \\ \max(OPT(i-1,j), v_i \cdot OPT(i-1, j+v_i)) & \text{else} \end{cases}$$

The optimal value can be found by calling $OPT(n, 0)$.

Suppose we want to optimise for space, whilst maintaining the same optimal time complexity. What is the tightest bound on space we can use? You may assume both n and m to be positive integers.

A. $O(1)$

B. $O(n)$

C. $O(m)$

D. $O(nm)$

13. We have previously seen a method for solving the knapsack problem where $OPT(n, W)$ gives us the correct answer. Which of the following solves the problem so that $OPT(1,0)$ gives us the answer instead?

A. $$OPT(i,w) = \begin{cases} 0 & \text{if } i > n \\ OPT(i+1, w) & \text{if } w_i > W - w \\ \max(OPT(i+1,w), v_i + OPT(i+1, w+w_i)) & \text{else} \end{cases}$$

B. $$OPT(i,w) = \begin{cases} 0 & \text{if } i > n \\ OPT(i+1, w) & \text{if } w_i > W \\ \min(OPT(i+1,w), v_i + OPT(i+1, w+w_i)) & \text{else} \end{cases}$$

C. $$OPT(i,w) = \begin{cases} 0 & \text{if } i > n \\ OPT(i-1, w) & \text{if } w_i > w \\ \max(OPT(i-1,w), v_i + OPT(i-1, w-w_i)) & \text{else} \end{cases}$$

D. $$OPT(i,w) = \begin{cases} 0 & \text{if } i > n \\ OPT(i+1, w) & \text{if } w_i > w \\ \min(OPT(i+1,w), v_i - OPT(i+1, w-w_i)) & \text{else} \end{cases}$$

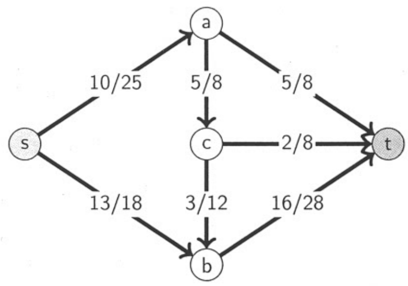


Figure 2: The numbers on the edges represent flow/capacity.

14. Consider the graph G depicted in Figure 2 with some flow f. Which of the following statements about this graph is **true**?
    - A. The minimimum cut of G has a capacity of 32.
    - B. The residual graph G' has exactly 2 edges with a capacity of 5.
    - C. This flow f can be be augmented one or more times to get to a flow with a value of 34.
    - D. The flow f is invalid, as it violates the flow-conservation conditions on one or more nodes.

15. Consider the project selection problem, where we are given n projects with a revenue p_i which is either positive or negative, and a set of prerequisites in the form of tuples (i, j) to indicate that i depends on j. The goal is to select the projects (including their prerequisites) that maximise the sum of the revenues.

    A student has submitted the following model to solve this problem:

    1. Create source s, a sink t, and a node for every project r_i.
    2. Create an edge from r_j to r_i with a capacity of $\sum_i |p_i|$ iff j is a prerequisite for i.
    3. If project i has a positive revenue, connect it to the sink with capacity p_i (so an edge (r_i, t)).
    4. If project i has a negative revenue, connect the source to it with capacity $-p_i$ (so an edge (s, r_i)).
    5. Return the value of the max-flow as the maximum obtainable profit.

    Which of the following statements about this formulation is **true**?
    - A. This is a correct model to solve the problem.
    - B. Step 2 of the model is incorrect, the capacity of these edges should be ∞.
    - C. Step 3 and 4 of this model are incorrect, positive revenues should be connected from the source and negative revenues to the sink.
    - D. Step 5 is is incorrect, we should instead return $\sum_{i,\text{ such that } p_i>0} p_i - v(f)$ where $v(f)$ is the value of the maximum flow.

16. Which of the following is a tight bound on the time complexity for solving the bi-partite matching problem given n items in set X and n items in set Y?
    - A. $O(n)$
    - B. $O(n^2)$
    - C. $O(n^3)$
    - D. $O(n^4)$

End of exam questions