KØBENHAVNS UNIVERSITET

Introduktion til diskret matematik og algoritmer: Exam April 10, 2024

Submission: Please write your solutions with ample margins on all sides, and make sure your handwriting is legible. *Start your solution of every new problem on a new page. Please mark every page with your name, exam number or something else that uniquely identifies your exam,* so that it is easy to see for every page which exam submission it is part of.
Exposition: Please try to be precise in your solutions and refrain from vague statements. Never, ever just state an answer, but always make sure to *explain why* the answer is what it is. Provide clear references to any facts in the course literature used. *Write your solutions in such a way that a fellow student of yours could read, understand, and verify your solutions.*
Collaboration: All problems should be solved individually. No communication or collaboration is allowed, and solutions will be checked for plagiarism.
Reference material: Textbooks and handwritten notes (including lecture notes) are allowed. Other typewritten material, including (but not limited to) problem sets or previous exams with solutions, is not allowed. Please note that you deviate from definitions and algorithm descriptions in the course material at your own risk! If, however, slight variations of algorithms were presented in the textbooks and/or in class, such minor details do not matter.
Grading: A score of 200 points is guaranteed to be enough to pass the exam.
About the problems: Note that the problem are of quite different types, and are *not sorted in increasing order of difficulty. Please read through the whole exam first,* before you start working on any single problem, so that you can plan which order of dealing with the problems makes most sense to you. Note that this is a fairly large exam, and so you can definitely get a top grade without solving all problems. Also, partial answers to problems can sometimes give substantial amounts of points. Please do not hesitate to alert the exam administrators if any problem statement is unclear. ***Good luck!***

1 (70 p) In the following snippet of code `A` is an array indexed from 1 to n containing elements that can be compared and (totally) ordered using the operators $=$, $>$, and $<$.

```
i    := 1
good := TRUE
while (i < n and good)
    j := n
    while (j > i and good)
        if (A[i] != A[j])
            j := j - 1
        else
            good := FALSE
    if (good)
        i := i + 1
if (good)
    return TRUE
else
    return (i, j)
```

1a Explain in plain language what the algorithm above does. When and why does the algorithm return *TRUE*? What is the meaning of the pair (i, j) returned instead of *TRUE*?

1b Provide an asymptotic analysis of the running time as a function of the array size n.

1c Can you improve the code to run faster while retaining the same functionality? Note that you can compare elements with operators $=$, $>$, and $<$, but other than that it is not known what the elements are. If you are able to provide a more efficient algorithm, how much faster can you get it to run? Analyse the time complexity of any new algorithm.

Suppose that we are guaranteed that all elements in the array `A` are integers between $-n$ and n. Can you then improve the code to run faster while retaining the same functionality? If you are able to provide a more efficient algorithm, how much faster can you get the algorithm to run? Analyse the time complexity of any new algorithm. Can you prove that it is asymptotically optimal?

2 (60 p) Provide formal proofs of the following claims using proof techniques that we have learned during the course.

2a For all $K \in \mathbb{Z}^+$ it holds that $\sum_{s=1}^{K} s \cdot s! = (K+1)! - 1$.

2b For all $q \in \mathbb{N}$ it holds that $2^{4q+2} + 3^{q+2}$ is a multiple of 13.

3 (60 p) Let us return to the array $A = [5, 6, 4, 7, 3, 8, 2, 9, 1, 10]$ that was considered in problem set 2. Just as in that problem set, we would like to sort A in increasing order.

3a Run heap sort by hand on this array for the first few steps of the algorithm. Show how the original heap is built, and how the first two numbers in the array are sorted into their correct final positions. Make sure to explain what calls to the heap are made from the sorting algorithm and what comparisons are made, and illustrate how the heap and array have changed at the end of every "heapify" or "bubble" call (after all recursive subcalls have been taken care of).

Remark: Please note that you do *not* have to run the whole sorting algorithm, since this might be quite time-consuming. Just showing the initial steps as described above is fully sufficient.

3b Suppose that we build a max-heap from any given array A. Is it true that once the array A has been converted to a correct max-heap, then considering the elements in reverse order (i.e., $A[n], A[n-1], \ldots, A[2], A[1]$) yields a correct min-heap? Prove this or give a simple counter-example.

4 (100 p) The busy life as a computer science professor does not leave much time for exercise, but Jakob has finally enlisted the help of a personal trainer (PT) to do something about this. The PT has devised an ambitious program in which Jakob will need to lift dumbbells of weights in all increments of $3, 4, 5, \ldots, 9$ kilos.[1]

The local fitness store sells weights in the range $1, 2, 3, \ldots, 9$ kilos. Jakob would like to buy as few weights as possible so that they can be combined to yield all weight sums $3, 4, 5, \ldots, 9$ kilos in the program suggested by the PT. Just to give a couple of examples, with two 3-kilo weights the possible weight sums are 3 and 6 kilos, while with one 2-kilo weight and one 3-kilo weight the possible weight sums are 2, 3, and 5 kilos. Jakob has given up on trying to get away with buying two weights only, but would really prefer not having to buy more than three weights.

Note: All the subproblems described below can be solved independently of one another! You do *not* need to solve Problem 4a in order to solve Problem 4b, or Problem 4b in order to solve Problem 4c, for instance.

4a How many different combinations of three weights in the range $1, 2, 3, \ldots, 9$ are there to consider as Jakob tries to figure out how to buy only three weights that can sum to all desired total weights?

Remark: Please do not just state a number, but also provide an explanation of what this number means and how you know it is the right number. In fact, you do not even have to compute the exact number—answering with a combinatorial expression containing products or quotients of integers and/or factorials of integers, together with an explanation of why this expression is correct, is sufficient.

4b Prove that if Jakob wants to buy weights that can be combined to yield all weight sums in the range $3, 4, 5, \ldots, 9$, then unfortunately he will have to buy more than three weights.

Remark: Note that a solution to this problem will have to show conclusively that none of all the combinations in Problem 4a works. Since this is a fairly large number of combinations, the approach will probably have to be something smarter than exhaustive case analysis.

4c Show that it is sufficient for Jakob to buy four weights in order to be able to combine them to yield all weight sums $3, 4, 5, \ldots, 9$.

4d Jakob also wants to minimize the *total weight* of all the weights he buys, i.e., the sum of the weights, in order not to have to carry around too heavy a training bag. What is the smallest possible total weight of a collection of four weights that can be combined to yield all weight sums $3, 4, 5, \ldots, 9$ as in Problem 4c?

[1] Well, OK, ambitious or not so ambitious, but you have to start *somewhere*, no?

5 (60 p) During the stressful mornings when Jakob is teaching the IDMA course in Copenhagen, he needs to successfully plan the following actions:

Coffee Have a morning latte.

Dress Put on clothes.

Lecture Give the IDMA lecture in Copenhagen.

Shower Take a shower.

Train Take the train from Lund to Copenhagen.

Wake up The hardest task of them all...

Just a quick look at this list reveals that the correct sequencing of actions is crucial in order to avoid potentially catastrophic consequences. Jakob is aware of the following constraints:

- He has to take the **train** to Copenhagen before **lecturing**.
- He also needs to have **coffee** before **lecturing**.
- He has to **dress** before taking the **train**.
- He has to **shower** before **dressing**.
- He has to **wake up** before taking a **shower**.
- He definitely has to **dress** before **lecturing**.
- Very annoyingly, he has to **wake up** before having **coffee** (though the other way round would be more helpful).
- Finally, ideally he should also **wake up** before **lecturing**.

5a Model this problem as a graph, with the constraints corresponding to edges between actions. Make sure to explain concretely what graph you get for Jakob's problem.

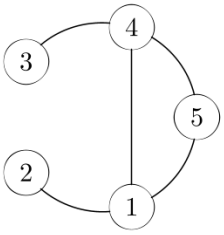
5b Propose a suitable graph algorithm to solve Jakob's problem. Explain what this algorithm is and why it is the right choice for this problem. Make a dry-run of the algorithm and explain the relevant steps in the execution (similarly to what has been done in class and in the lecture notes). Let your graph process all actions in alphabetical order as listed above. What morning schedule can you propose for Jakob based on what your algorithm says?

6 (60 p) In this problem we focus on relations. In particular, suppose that $A = \{e_1, e_2, \ldots, e_6\}$ is a set of 6 elements and consider the relation R on A represented by the matrix

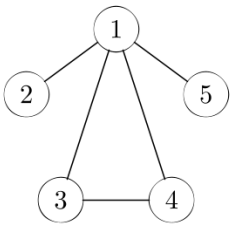
$$M_R = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(where element e_i corresponds to row and column i).

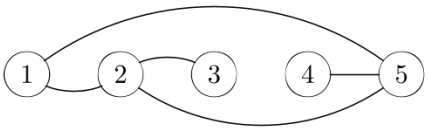
6a Let us write T to denote the transitive closure of the relation R. What is the matrix representation of T in this case? Can you explain clearly in words what the relation T is?



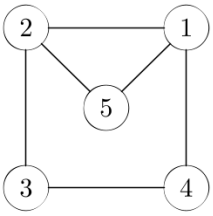
(a) Graph G_1.



(b) Graph G_2.



(c) Graph G_3.



(d) Graph G_4.

Figure 1: Graphs for Problem 7.

6b Suppose that we create a new relation S_1 from R by first taking the reflexive closure and then the symmetric closure. What does the matrix representation of S_1 look like? Can you explain clearly in words what the relation S_1 is?

6c Suppose instead that we first take the symmetric closure and then the reflexive closure to derive another relation S_2 from R. Are S_1 and S_2 different relations, or are they the same relation in the end? If the latter case holds, is it true for any relation R on a set A that relations S_1 and S_2 derived in this way will be the same, or can they sometimes be different? Give a proof or present a simple counter-example.

7 (110 p) In this problem we wish to study graphs viewed as relations.

7a Recall that two graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$ are said to be *isomorphic*, denoted $G \cong H$, if there is a bijection $f : V_G \to V_H$ such that $(u, v) \in E_G$ holds if and only if $(f(u), f(v)) \in E_H$.

Argue, briefly but clearly, why $\cong$ defines an equivalence relation.

7b Consider the graphs in Figure 1. Explain how the relation $\cong$ partitions these graphs into equivalence classes. Argue, briefly but clearly, why any pair of graphs G_i and G_j for $1 \leq i < j \leq 4$ are or are not in the same equivalence class, respectively.

7c Let A_G denote the adjacency matrix of a graph G on n vertices. Recall that for any graph G we define $G^\top$ to be the graph with adjacency matrix $A_G^\top$. Define G^i as the graph with adjacency matrix $(A_G)^i_\odot = A_G \odot A_G \odot \cdots \odot A_G$ (with the matrix A_G repeated i times). Finally, let K_n denote the complete graph on n vertices with all possible edges present, corresponding to the adjacency matrix with all entries equal to 1 except for 0s on the diagonal.

When playing around with these notions, Jakob has come up with the following claims. Explain which of them are true or false and why.

1. The edge relation on $V = V(G)$, saying that $E(u, v)$ holds precisely when $(u, v) \in E$, is a transitive relation if and only if G^2 is a subgraph of G.
2. If the edge relation is symmetric, then $G \cong G^\top$.
3. The graph G is connected if and only if $G^n \cong K_n$.
4. The edge relation is asymmetric if and only if G is a directed acyclic graph (DAG).

8 (100 p) During his years of teaching introductory discrete mathematics courses, Jakob has developed a not-always-completely-healthy interest in card games. His latest obsession is the game *Stop on Black*, which is played between a dealer and a gambler as described below.

The dealer shuffles a complete deck of 52 cards perfectly, and then deals cards face up from the deck, one card at a time. Before any card is dealt, including at the very start of the game, the gambler can place a bet. Such a bet is placed only once, and has to be made at the very latest before the last card is dealt (or, phrased differently, if the bet has not been placed before, then it is automatically placed on the last card). After the bet has been placed, the dealer deals the next card, i.e., the one on which the bet has been placed, and the game ends. The gambler wins the game if this card is black (spades or clubs) and loses if it is red (hearts or diamonds).

Jakob is interested in determining what is the highest probability of winning that can be achieved with any gambler strategy (where, to simplify matters, we say that the gambler cannot flip coins or use any other kind of randomness, but has to choose actions only based on the cards seen so far). Jakob has played this game obsessively, but has never been able to win more than half of the time on average (which is trivial to achieve by just placing the bet immediately, before even the first card is dealt).

Jakob claims to have made an exhaustive case analysis on a small deck with 2 red and 2 black cards, and says that (a) this case analysis proves rigorously that the best winning probability is exactly 50% and that (b) this should generalize to any deck of N red and N black cards. Unfortunately, his proof for (a) is pretty much unreadable, and the claim in (b) that it should generalize is even harder to understand.

Can you help Jakob by figuring out whether he is right in claiming that a 50% winning probability is best possible for the gambler, or could you provide a counter-example?

Remark: For a full score, analysing the case of a standard deck with 26 red and 26 black cards is sufficient. Partial results, including proofs or counter-examples for (much) smaller decks of cards (such as what Jakob claims to have analysed), can also yield points.