CSOR W4231–Spring, 2022
Eleni Drinea

Homework 4 (105 points)

Out: Monday, April 4, 2022
Due: 11:59pm, Monday, April 18, 2022

Homework Instructions.

1. For all algorithms that you are asked to "give" or "design", you should

   - Describe your algorithm clearly in English.
   - Give pseudocode.
   - Argue correctness; you may provide a formal proof or a convincing argument.
   - Provide, with an explanation, the best (smallest) upper bound that you can for the running time. All bounds should be **worst-case** bounds, unless explicitly stated otherwise.

   You are also encouraged to analyze the space required by your algorithm. We will not remove marks if you don't, unless the problem explicitly asks you to analyze space complexity.

2. If you give a dynamic programming algorithm, you should follow the instructions in HW2.

3. If you give a **reduction**, you should do so as we did in class, that is

   (a) Give the inputs to the two problems.

   (b) Describe in English the reduction transformation and argue that it requires polynomial time. (You do not need to give pseudocode.)

   (c) Prove carefully equivalence of the original and the reduced instances.

4. You should submit this assignment as a **pdf** file to Gradescope. Other file formats will not be graded, and will automatically receive a score of 0.

5. I recommend you type your solutions using LaTeX. For every assignment, you will earn 5 extra credit points if you type your solutions using LaTeX or other software that prints equations and algorithms neatly. If you do not type your solutions, make sure that your handwriting is very clear and that your scan is high quality.

6. You should write up the solutions **entirely on your own**. Collaboration is limited to discussion of ideas only. You should adhere to the department's academic honesty policy (see the course syllabus). Similarity between your solutions and solutions of your classmates or solutions posted online will result in receiving a 0 in this assignment, and possibly further disciplinary actions. There will be no exception to this policy and it may be applied retro-actively if we have reasons to re-evaluate this homework.

Homework Problems

1. (35 points) The following randomized algorithm returns the k-th smallest number from a set S of n distinct integers, for any k.

Algorithm 1

`k-th order statistic` (S, k)

Select an item $a_i \in S$ uniformly at random
for each item $a_j \in S$ **do**
    Put a_j in S^- if $a_j < a_i$
    Put a_j in S^+ if $a_j > a_i$
end for
if $|S^-| = k - 1$ **then** *// a_i is the k-th smallest item*
    `return` a_i
else if $|S^-| \geq k$ **then** *// the k-th smallest item lies in S^-*
    `k-th order statistic` (S^-, k)
else *// the k-th smallest item lies in S^+*
    `k-th order statistic` $(S^+, k - 1 - |S^-|)$
end if

   (a) (5 points) Show how to compute the median of S using this algorithm.

   (b) (30 points) Analyze the expected running time of `k-th order statistic` (S, k).
   Hint (feel free to ignore if you prefer to provide the analysis in a different way):
   We will say that a subproblem is of type j if its input consists of at most $n\left(\frac{3}{4}\right)^j$ items but more than $n\left(\frac{3}{4}\right)^{j+1}$ items.
   Upper bound the expected time of `k-th order statistic` *on a subproblem of type j, excluding the time spent on recursive calls. Equivalently, upper bound the expected time until an item a_i is selected such that $1/4$ of the input to the subproblem can be thrown out (thus the input shrinks by a factor of $3/4$).*
   Next obtain an upper bound to the expected running time of the entire algorithm by summing up the expected time spent on every subproblem.

2. (30 points) You are given a flow network $G = (V, E, s, t, c)$ with integer capacities, an integral maximum flow f^* in G, and the residual graph G_{f^*} upon termination of the Ford-Fulkerson algorithm.

   Alice picks a specific edge $e \in E$ and reduces its capacity by 1 unit. Show how to find a maximum flow in the resulting flow network in $O(n + m)$ time. (You may assume that f^* is acyclic, that is, there is no cycle in G on which all edges carry some positive flow.)

3. (40 points) A *flow network with supplies* is a directed capacitated graph with potentially multiple sources and sinks, which may have incoming and outgoing edges respectively. In particular, each node $v \in V$ has an integer *supply* $s(v)$; if $s(v) > 0$, v is a *source*, while if $s(v) < 0$, it is a *sink*. Let S be the set of source nodes and T the set of sink nodes.

A *circulation with supplies* is a function $f : E \to R^+$ that satisfies

(a) *capacity constraints:* for each $e \in E$, $0 \leq f(e) \leq c(e)$.

(b) *supply constraints:* For each $v \in V$, $f^{\text{out}}(v) - f^{\text{in}}(v) = s(v)$.

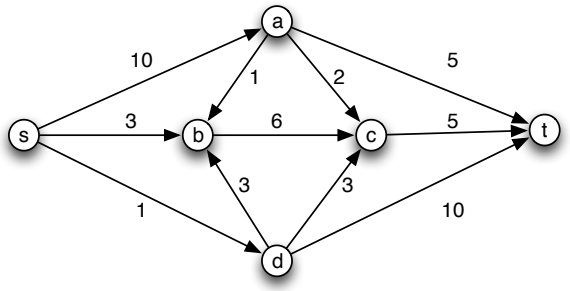
We are now concerned with a decision problem rather than a maximization one: *is there* a circulation f with supplies that meets both capacity and supply constraints?

i. (10 points) Derive a necessary condition for a feasible circulation with supplies to exist.

ii. (30 points) Reduce the problem of finding a feasible circulation with supplies to max flow.

RECOMMENDED exercises: do NOT return, they will not be graded.)

1. Run the Ford-Fulkerson algorithm on the following network, with edge capacities as shown, to compute the max s-t flow. At every step, draw the residual graph and the augmenting paths. Report the maximum flow along with a minimum cut.



2. There are many variations on the maximum flow problem. For the following two natural generalizations, show how to solve the more general problem by **reducing** it to the original max-flow problem (thereby showing that these problems also admit efficient solutions).

   - There are multiple sources and multiple sinks, and we wish to maximize the flow between all sources and sinks.
   - Both the edges *and the vertices* (except for s and t) have capacities. The flow into and out of a vertex cannot exceed the capacity of the vertex.