CSOR W4246–Fall, 2022
Eleni Drinea

Homework 3 (130 points)

Out: Tuesday, November 8, 2022
Soft deadline: 11:59pm, Tuesday, November 22, 2022
HARD deadline: 11:59pm, Wednesday, November 23, 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, even if you don't give an entirely formal proof.
   - Give the best upper bound that you can for the running time.

   You are also encouraged to analyze the space required by your algorithm but we will not remove marks if you don't, unless the problem explicitly asks you to analyze space complexity.
2. If you give a DP algorithm, you should follow the instructions in `hw2-theoretical`.
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 your assignment as a **pdf** file on 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 hand-writing 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 and 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. You should list your collaborators on your write-up.

Homework Problems

1. (35 points) On input three sequences X, Y, Z of sizes m, n, p respectively, give an efficient algorithm to compute the length of their longest common subsequence. You should also give an algorithm to output a longest common subsequence. For example, if $X = abcde$, $Y = abde$ and $Z = abce$, then the longest common subsequence is *abe* and its length is 3. (You may think of m, p as being $\Theta(n)$.)

2. (40 points) A *flow network with demands* $G = (V, E, c, d)$ 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 *demand* d_v; if $d_v > 0$, v is a *sink*, while if $d_v < 0$, it is a *source*. Let S be the set of source nodes and T the set of sink nodes.

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

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

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

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

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

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

3. (30 points) In many applications, on top of the node demands introduced in the previous problem, the flow must also make use of certain edges. To capture such constraints, consider the following variant of the previous problem.

   You are given a *flow network with demands and lower bounds* $G = (V, E, c, d, \ell)$ where every edge e has an integer capacity c_e, *and* an integer *lower bound* $\ell_e \geq 0$. A circulation f must now satisfy $\ell_e \leq f(e) \leq c_e$ for every $e \in E$, as well as the demand constraints. Determine whether a feasible circulation exists.

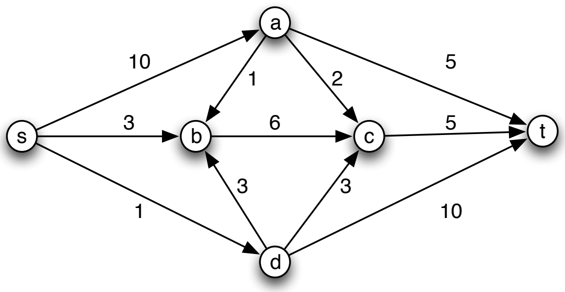
4. (25 points) Similarly to a flow network with demands, we can define a *flow network with supplies* $G = (V, E, s, c)$ where each node $v \in V$ now has an integer *supply* s_v, so that if $s_v > 0$, v is a *source* and if $s_v < 0$, it is a *sink*, and the supply constraint for every $v \in V$ is $f^{\text{out}}(v) - f^{\text{in}}(v) = s_v$.

   In a *min-cost flow* problem, the input is a flow network with supplies where each edge $(i, j) \in E$ also has a cost a_{ij}, that is, sending one unit of flow on edge (i, j) costs a_{ij}. Given a flow network with supplies and costs $G = (V, E, s, c, a)$, the goal is to find a feasible flow $f : E \rightarrow R^+$, that is, a flow satisfying edge capacity constraints and node supplies, that minimizes the total cost of the flow.

   Show that the max flow problem can be formulated as a min-cost flow problem.

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.