3121/9101 Final Exam Practice Problems
Solutions

Instructions: Please write the square of a number x as `x^2` and write an index such as a_i as `a_i`. Type your answers in the spaces provided.

1.

(a) Write the following complex number in the form $a + i\,b$: ω_4 (4 pts)

(b) Write the following complex number in the form $a + i\,b$: ω_8 (6 pts)

(c) Compute the convolution of sequences $(1, 0, 1)$ and $(-1, 0, -1)$. (10 pts)

Solution:

(a) $\omega_4 = e^{\frac{2\pi}{4}} = \cos\frac{\pi}{2} + i\sin\frac{\pi}{2} = 0 + i = i$

(b) $\omega_8 = e^{\frac{2\pi}{8}} = \cos\frac{\pi}{4} + i\sin\frac{\pi}{4} = \frac{\sqrt{2}}{2} + i\frac{\sqrt{2}}{2}$.

(c) Let $A = (1, 0, 1)$ and $B = (-1, 0, -1)$; then the corresponding polynomials are $P_A(x) = 1 + x^2$ and $P_B(x) = -1 - x^2$. The product of these two polynomials is $-1 - 2x^2 - x^4$ and so $A * B = (-1, 0, -2, 0, -1)$.

2. You have n items for sale, numbered from 1 to n. Alice is willing to pay $a[i] > 0$ dollars for item i, and Bob is willing to pay $b[i] > 0$ dollars for item i. Alice is willing to buy no more than A of your items, and Bob is willing to buy no more than B of your items. Additionally, you must sell each item to either Alice or Bob, but not both, so you may assume that $n \leq A + B$. You are given $n, A, B, a[1 \ldots n]$ and $b[1 \ldots n]$.

(a) Design an algorithm which correctly determines the **maximum** total amount of money you can earn and which runs in $O(n \log n)$ time. (10 pts)

(b) Justify that your algorithm is correct. (10 pts)

(c) Justify that it runs in time $O(n \log n)$. (5 pts)

Solution: Let $d[i] = |a[i] - b[i]|$; sort all $d[i]$ in a decreasing order and re-index all the items such that $|a[1] - b[1]|$ is the largest difference and so on, the i^{th} item being such that $d[i] = |a[i] - b[i]|$ is the i^{th} difference in size. We now go through the list giving the i^{th} item to Alice if $a[i] > b[i]$ and the total number of items given to Alice thus far is at most A and giving instead this item to Bob if $b[i] > a[i]$ and the total number of items given to Bob thus far is at most B. If at certain stage the number of items given to Alice reaches A we give all the remaining items to Bob and similarly if at certain

stage the number of items given to Bob reaches B we give all the remaining items to Alice. If neither A nor B are reached and there are leftover items for which $d[i] = 0$, i.e., such that $a[i] = b[i]$ we give them to either Alice or Bob making sure neither A nor B is exceeded.

To see that this algorithm is optimal, let $m[i] = \min(a[i], b[i])$. Then regardless of who gets item i you get at least the amount $m[i]$, plus you get the amount $d[i]$ if item i is given to the higher bidder. Our algorithm tries to get as many $d[i]$'s as possible, preferentially taking as large $d[i]$'s as possible, so the algorithm is optimal. (Note that here the algorithm is not trivially optimal, so we need a simple justification given.)

Computing all the differences $d[i] = |a[i] - b[i]|$ takes $O(n)$ time; sorting $d[i]$'s takes $O(n \log n)$ time and going through the list takes $O(n)$ time. Thus the whole algorithm runs in time $O(n \log n)$.

3. You are a photo reporter given an assignment to cover some events in Elbonia. You are given a map of Elbonia which looks like a connected graph with 20 cities $c_1, c_2, \ldots, c_{20}$ as vertices and with edges connecting some of the cities representing roads. The roads are quite bad and it takes one whole day to traverse any of the roads (i.e., edges of the graph) between any two connected cities. On each day t ($1 \leq t \leq 30$) in each city c_j there is an event and you are given the importance value $I(t, j)$ of that event. Your goal is to design an algorithm which determines:

   - the largest possible total sum of importances of 30 events that you can cover in 30 days;
   - the sequence of the 30 locations you will be at, such that the sum of the importances of the events in those locations at days when you visit them is as large as possible. All of these locations are among cities $c_1, c_2, \ldots, c_20$; each city can be visited multiple times. Any two consecutive locations must be two cities connected by a direct road between them (thus with an edge in the graph).

   You are free to choose the city you will start from on day 1.

   (a) Formulate precisely the sub problems to be solved. (10 pts)

   (b) Formulate precisely the initial values and the recursion equations that the optimal solutions to sub problems have to satisfy. (10 pts)

   (c) Formulate precisely how the final solution to the original problem is obtained from the solutions to sub problems, both in regard to obtaining the maximal total importance of the events covered, and in regard to the sequence of cities visited. (5 pts)

Solution: Let $I(t,k)$ be the importance of the event happening in city c_k ($1 \leq k \leq 20$) on day t ($1 \leq t \leq 30$) and let $E(p,q) = true$ just in case cities c_p and c_q are connected by a direct road and $false$ otherwise.

(a) We solve the following sub problems "for every t ($1 \leq t \leq 30$) and every k ($1 \leq k \leq 20$) find the maximal total importance $\text{opt}(t,k)$ of events covered from day 1 to day t so that the event covered on day t is in city c_k", and let $\text{prev}(t,k)$ be the city visited on day $t-1$ in the sequence of cities visited in order to achieve $\text{opt}(t,k)$.

(b) Clearly, $\text{opt}(1,k) = I(1,k)$ for all k ($1 \leq k \leq 20$); the recursion is given by

$$\text{opt}(t,k) = \max_{1 \leq m \leq 20} \{\text{opt}(t-1,m) + I(t,k) \ : \ m = k \text{ OR } E(m,k)\}$$

$$\text{prev}(t,k) = \arg \max_{1 \leq m \leq 20} \{\text{opt}(t-1,m) + I(t,k) \ : \ m = k \text{ OR } E(m,k)\}$$

Note that the clause $m = k$ allows to stay for day t in the city visited at day $t-1$.

(c) Let OPT be the optimal value of the sum of the importances of events after 30 days and let $\text{CT}(t)$ be the city visited on day t of the optimal schedule; then

$$\text{OPT} = \max_{1 \leq m \leq 20} \{\text{opt}(30,m)\}$$

$$\text{CT}(30) = \arg \max_{1 \leq m \leq 20} \{\text{opt}(30,m)\}$$

The sequence of cities $\text{CT}(t)$ is obtained by backtracking starting from city $\text{CT}(30)$ and obtaining each previous city by applying recursion

$$\text{CT}(t-i) = \text{prev}(t-i, \text{CT}(t-i+1))$$

fot all $i = 1 \ldots 30$. The correct sequence of cities is $\text{CT}(1), \text{CT}(2), \ldots, \text{CT}(30)$.

4. You are running a Literature class in which students have to write essays. You have 300 students in class and have prepared 300 topics, but you want to let the students choose the topic they will write about. So each student has given you a list of a few topics which they find interesting and would like to write about. You have to design an algorithm which assigns to each student a topic from their list so that every student writes on a different topic. In case this is not possible your algorithm should output a message that the task has no solution.

Solution: This is a straightforward Max Flow problem. You form a bipartite graph with students on the left side and topics on the right side. You add

directed edges from each student to each topic this particular student want to write about. You now solve a maximal matching problem for this graph by turning it into a flow network by adding a source connected by directed edges from the source to each student. You also add a super sink and connect with directed edges from all topics to such a sink. You set the capacity of all edges to 1 and apply a polynomial time Max Flow algorithm, such as Edmonds - Karp algorithm. Finally you look for all edges from students to topics which are occupied. Since the capacity of all edges is 1 and augmenting paths always add integer amount of flow, such edges are either empty or fully occupied. The occupied edges are used to assign a maximal number of topics and if the number of such edges is less then the number of students we output message "no solution".