Algorithms II: Problem Set

Shashank K Mehta
IIT-Kanpur

Question 1.

Show by an example that in Kruskal's algorithm (the algorithm discussed in the class) if the edges are not selected in non-decreasing order then the resulting graph may not be an MST.

Question 2.

Suppose A is an algorithm which computes the weighted shortest path length from each vertex to every other vertex in any edge-weighted directed graph. Let G be a DAG. Using A, write an algorithm to compute the weight of the maximum-weight path (i.e., weighted longest path) in G. Prove correctness.

Question 3.

In a non-negatively edge-weighted undirected graph $G = (V, E)$, an (s, t)-cut is a vertex partition $(S, V \setminus S)$ with $s \in S$ and $t \in V \setminus S$. Weight of the cut is the sum of the weights of the edges $\{(x, y) | x \in S, y \in V \setminus S\}$. Define a suitable flow network so that maximum flow computation in it can give the weight of the minimum weight (s, t)-cut in G. Justify.

Question 4.

(i) If the LUP decomposition described in the class, is performed on a non-full matrix, then during which step(s) and how the error will be detected?

(ii) Give an efficient algorithm to compute the determinant of an $n \times n$ matrix using LUP decomposition. Justify.

Question 5.

Consider the following algorithm to compute an augmenting path. Let $G = (V, E)$ be an undirected graph, M be a matching in G and U be the set of unmatched vertices w.r.t. M. Let $G_M = (V, E_M)$ where $E_M = \cup_{(x,y) \in M} \{(x, z) | (y, z) \in E\}$. Let $Search(G', x)$ performs DFS or BFS and returns all the vertices reachable from x in G', i.e., it returns all the vertices in the connected component of x in G'. Let $u \in U$. Compute $Search(G', v)$ for each $v \in N(u)$. If any $Search(G', v)$ contains an unmatched vertex u' other than u, then can we claim that there is an augmenting path from u to u'. Prove or disprove the claim.

Question 6.

Devise an efficient algorithm to check if a rectangular n-dimensional box of size $d_1 \times d_2 \times \cdots \times d_n$ can be fitted inside another similar box of size $e_1 \times e_2 \times \cdots \times e_n$. What is the time complexity of the algorithm. Prove correctness.

Question 7.

In the analysis of Union-Find data-structure with path compression we used the fact that at any point the rank of any vertex x is strictly smaller than that of its parent. Prove this claim.

Question 8.

Compute a min-cut with minimum number of edges.

Question 9.

Compute a min-cut with maximum number of edges with positive capacity.

Question 10.

Prove that time complexity of multiplying two $n \times n$ lower-triangular matrices is same as that for general matrices.

Question 11.

Design an efficient algorithm to compute $a(mod b)$ where a is a $2n$-bit number and b is an n-bit number. Determine the time complexity.

Question 12.

Given an integer sequence S. Design an efficient algorithm to compute a longest subsequence of S which is (i) monotonically increasing, (ii) non-decreasing. What is the time complexity?

Question 13.

There are n finance companies which offer interest rates r_{ij} percent on fixed deposits for next m years, where $1 \leq i \leq n$ and $1 \leq j \leq m$. Each deposit can be placed for an integral number of years. Shifting money from one company to another costs c percent. Design an efficient algorithm which determines the investment strategy which gives maximum returns at the end of m years. Give the time complexity.

Question 14.

Let B be set of n boys and G be a set of n girls. Each boy ranks the girls 1 to n and each girl ranks the boys from 1 to n. A pairing is a one to one correspondence between boys and girls. A pairing (b_i, g_i) for $i = 1, \ldots, n$ is said to be *stable* if for each unpaired boy b_i and girl g_j following condition DOES NOT hold: b_i prefers g_j over g_i and g_j prefers b_i over b_j. (i) Prove that a stable pairing always exists, (ii) Design an efficient algorithm which takes the preference list for each boy and girl and returns a stable pairing. Give the time complexity.

Question 15.

Prove that $2^n - 1$ moves are both necessary and sufficient to solve the Tower of Hanoi problem.

Question 16.

Show that $\lceil 3n/2 - 2 \rceil$ comparisons are necessary and sufficient to find the smallest and the largest number in a set of n numbers.

Question 17.

Let x and y be strings of symbols from some alphabet. Consider the three operations: deleting a symbol from x, inserting a new symbol into x, and replacing a symbol of x by another. Describe an algorithm which transforms x into y using minimum number of such operations. Give the time complexity.

Question 18.

Suppose $p(x) = \sum_{i=0}^{9} a_i x^i$. The algorithm to compute $x^{2n-2}/p(x)$, $recip(p(x))$, assumes that n is a power of 2. So redefine $p(x)$ by setting $a_i = 0$ for $10 \leq i < 16$ and then run the algorithm. Explain why the algorithm will not work. In order to compute $x^{2.10-2}/p(x) = x^{18}/p(x)$ suggest a way to use $recip()$. Clearly state (i) what will be the input polynomial, (ii) if the output is $q(x)$, then how to get $x^{18}/p(x)$ from it $q(x)$.

Question 19.

By reducing the sorting problem on integers into the 2-dimensional convex problem show that the convex problem is $\Omega(n. \log n)$. Hint: Curve $y - x^2 = 0$ is a convex curve.

Question 20.

Consider the following variant of the linear program

$$\max \sum_i c_i x_i \text{ subject to } \sum_i a_{ji} x_i \leq b_j, \quad 1 \leq j \leq n.$$

Unlike the standard LP suppose the positivity of the variables is not required. Transform this LP into standard LP, in which variable are required to be non-negative, without adding many new variables. Hint: The feasible space of the given variant may not be in the positive orthant. So shift the coordinate system in the direction $(-1, -1, \ldots, -1)$ by some unknown distance. Tackle unknown by adding a variable.

Question 21.

Let $f = \sum_i a_i x^i$ and $g = b_i x^i$. Consider the following algorithm aiming to compute f/g (quotient of f divided by g):

1. compute $\vec{u} = F(\vec{f})$ and $\vec{v} = F(\vec{g})$; /* compute DFT of f, g*/

2. for all i $w_i = u_i / v_i$

3. compute $\vec{h} = F^{-1}(\vec{w})$

4. return h

Explain why this will not compute f/g. Which polynomial, closely related to f/g, can be computed using this technique? Show exactly how.

Question 22.

Show that if a and b are relatively prime, then $a.c \equiv 1 \, mod \, b$ for some c. Also show that converse also holds. Show that c is unique modulo b.

Question 23.

The *cyclic difference of a vector* $a = [a_0, a_1, \ldots, a_{n-1}]^T$, *denoted by* Δa, *is* $[a_0 - a_{n-1}, a_1 - a_0, , a_2, a_1, \ldots, a_{n-1} - a_{n-2}]^T$. *If the DFT of a is* $F(a) = [b_0, b_1, \ldots, b_{n-1}]$. *Then show that* $F(\Delta a) = [0, b_1(1-\omega), b_2(1-\omega^2), \ldots, b_{n-1}(1-\omega^{n-1})]^T$.

Question 24.

A circulant *matrix is a square matrix in which the i-th row is* $a_i \; a_{i+1} \ldots a_n \; a_1 \ldots a_{i-1}$. *Show that the DFT computation of a vector of prime length n is equivalent to multiplying it by a circulant matrix.*

Question 25.

Let c_p *be the n-vector of the coefficients of an* $n-1$ *degree polynomial* $p(x)$. *Let* v_p *be the vector in which i-th component is the evaluation of the i-th derivative* $p(x)$ *at the origin for* $0 \le i < n$. *Is* v_p *a linear transformation of* c_p *for all* $n-1$*-degree polynomials p.*

Question 26.

Give a complete algorithm that computes the GCD of n bits integers in $O(n.\log^2 n.\log\log n)$.

Question 27.

Let $M(n)$ *be the time to multiply n-bit numbers and* $Q(n)$ *be the time to compute* $\lfloor\sqrt{a}\rfloor$ *where a is an n-bit integer. Assume that* $M(bn) \ge b.M(n)$ *for any* $b > 1$ *and similarly for* $Q(n)$. *Show that* $M(n)$ *and* $Q(n)$ *are with in a constant factor of one another.*

Question 28.

Let $G = (V, E)$ *be an undirected graph. Let* $w()$ *be a non-negative edge-weight function. Let s be a specific vertex in G.*

```
1. For each x ∈ V do d(x) := ∞;
2. d(s) := 0;
3. flag := false;
4. Repeat
5.     For each (x, y) ∈ E do
6.         If d(x) > d(y) + w(x, y) Then {d(x) := d(y) + w(x, y); flag:=true;}
7.         Else {If d(y) > d(x) + w(x, y) Then {d(y) := d(x) + w(x, y); flag:=true;}}
8. Until (flag = false);
9. For each x ∈ V do print(d(x));
```

(i) Prove that this program will terminate. (ii) Show that it correctly computes the weights of the minimum weight paths from s to each x. Determine the time complexity of this algorithm.

Question 29.

(i) Let S denote the n-dimensional space, $\mathbb{R}^n$, and $\mathcal{F}$ denote the collection of independent sets of vectors belonging to $\mathbb{R}^n$ (clearly no set in $\mathcal{F}$ can have more than n vectors). Show that $(S, \mathcal{F})$ is a matroid.

(ii) Let $G = (V, E)$ be graph. Let $\mathcal{F}$ be the collection of all matchings of G (i.e., in each set of $\mathcal{F}$, no two edges share any end vertex). Does $(E, \mathcal{F})$ form a matroid? If yes, then prove it. Otherwise give an example where one of the matroid rules fails.

Question 30.

(i) In the Union-Find analysis which claim(s) depend on the fact that always smaller tree is made the child of the root of the larger tree in the union operation.

(ii) What will be the time complexity of the Union-Find problem if path compression is NOT performed. Assume that there are f find operations.

Question 31.

A box B contains n chits and each chit x has an integer $f(x)$ written on it. Consider the following process.

While $(\exists x, y \in B$ s.t. $|f(x) - f(y)| > 1)$ Do
select x, y from B with $|f(x) - f(y)| > 1$;
If $(f(x) > f(y))$ Then set $f(x) := f(x) + 1$ and $f(y) := f(y) + 2$;
Else set $f(y) := f(y) + 1$ and $f(x) := f(x) + 2$;
put x, y back into B;

Show that this process will terminate. Deduce a good upper-bound for the number of iterations.