Mid Sem **Advanced Algorithms (CS 315)**

Questions: 5, **Marks:** 50

Instructions: Only YOUR notebook allowed (No electronic devices, textbooks, printed/photocopys).

---

1. (10 marks) Consider the following quicksort algorithm

Algorithm 1 Quick Sort

Input: An array x of n distinct elements

Output: Sorted array

If $n = 1$ or 0 return x

Let *pivot* be randomly picked from the input.

Create an array x_1 containing all elements less than pivot (in the order they appear in x)

Create an array x_2 containing all elements greater than pivot (in the order they appear in x).

Return the array [QuickSort(x_1), pivot, QuickSort(x_2)]

The following input is given to the quicksort algorithm: $6, 2, 8, 5, 4, 1, 7, 3$. Answer the following

(a) Let us assume the random pivots picked by the algorithm are: $5, 3, 2, 7$. Find the number of comparisons done by the quicksort algorithm?

(b) Consider another sequence of random pivots: $4, 3, 2, 6, 7$. Find the number of comparisons now?

(c) Assume that the above quicksort algorithm has the following "defect". It is not able to pick any random sequence of pivots. It picks the sequence $5, 3, 2, 7$ with probability $\frac{1}{10}$ and the sequence $4, 3, 2, 6, 7$ with probability $\frac{9}{10}$. That is, all other sequences have probability 0. What is the expected number of comparisons? Justify your answer.

2. (10 marks) Consider the following Bubble sort algorithm.

Algorithm 2 Bubble Sort

Input: An array x of n distinct elements

Output: Sorted array

for $(i = 0; i < n; i++)$ **do**

    for $(j = 0; j < n; j++)$ **do**

        if $(x[j] > x[j+1])$ **then**

            swap $x[j]$ and $x[j+1]$

        end if

    end for

end for

Return the array x

Let the input to the algorithm be a random permutation of the numbers $\{1, 2, \ldots, n\}$. We are interested in finding the expected number of swaps. Justify your answers to the following questions.

(a) Let X_{ij} be the indicator random variable which takes the following values

$$X_{ij} = \begin{cases} 1, & \text{if } i < j \text{ and } x[i] > x[j]. \\ 0, & \text{otherwise.} \end{cases}$$

What is $\texttt{Prob}\,[X_{ij} = 1]$?

(b) What is the expected number of swaps.

3. (10 marks) Consider the following sorting algorithm and explain your answers to the questions below

Algorithm 3 Crazy Sort

Input: An array x of n distinct elements

Output: Sorted array

```
while true do
    Do a random permutation of the input array x and store the result in y.
    if y is a sorted array then
        return y
    end if
end while
```

(a) What is the expected number of times the while loop is executed when $n = 2$?

(b) What is the probability of the algorithm terminating for a general n?

(c) What is the expected number of times the while loop is executed for a general n?

4. (10 marks) Let $G = (V, E)$ be an undirected graph. A subset of edges, S is called an r-cut if removing all the edges in S from G results in a graph which has atleast r connected components. Our aim is to detect a minimum r-cut. In class we looked at the case when $r = 2$. Consider the following algorithm.

Algorithm 4 Min. r-cut

Input: An undirected graph, $G = (V, E)$ where $|V| \geq r$

Output: r-cut

```
while (Number of vertices in the graph G is greater than r) do
    Pick a random edge e from G.
    Compress e and let G denote the resulting graph.
end while
Return the remaining edges in G.
```

(a) Show that the algorithm always returns an r-cut.

(b) Let k be the size of a minimum r-cut. Show that the number of edges is greater than $\frac{nk}{2(r-1)}$.

(c) Compute the probability of our algorithm returning a minimum r-cut. Justify your answer.

5. (10 marks) Consider the set cover problem we discussed in class: In the problem, there is a set U of m elements and sets $S_1, S_2, \ldots, S_n$ which are subsets of U. A set cover is a collection of these sets whose union is equal to U. In the weighted set-cover problem, associated with each set S_i, there is a weight $w_i \geq 0$. The goal is to find a set cover $\mathcal{C}$ so that the total weight $\sum_{S_i \in \mathcal{C}} w_i$ is minimzed.

We consider the k-set cover problem which is a "special" case of the weighted set cover problem. In the problem, for all elements e in U, there is exactly k many S_is which contains e. Answer the following

(a) Give an algorithm which gives a k-approximation.

(b) Prove that your algorithm is a k-approximation.