University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
S. Faour, M. Fuchs, A. Malyusz

Algorithm Theory
Sample Solution Exercise Sheet 12

Due: Friday, 24th of January, 2025, 10:00 am

Exercise 1: Hidden numbers *(8 Points)*

(a) You are given a uniform random permutation of the numbers of $1, \ldots, n$. Prove that if we run the following algorithm

Algorithm 1 Finding the Maximum Element in a Permutation

Require: A uniform random permutation $A[1 \ldots n]$ of the numbers $1, \ldots, n$.
Ensure: The maximum element in A.

```
1: maxSoFar ← A[1]                       ▷ Initialize the maximum element as the first element
2: for i ← 2 to n do                     ▷ Iterate through the array starting from the second element
3:     if A[i] > maxSoFar then
4:         maxSoFar ← A[i]               ▷ Update the maximum element
5: return maxSoFar                       ▷ The maximum element in A
```

the maxSoFar value will, in *expectation*, be updated (line 4) at most H_n times where H_n is the n-th harmonic number defined by $H_n := \sum_{i=1}^{n} 1/i$. *(4 Points)*
Hint: Define

$$X_i := \begin{cases} 1 & \textit{if } A[i] \textit{ is larger than all values in the prefix } A[1, ..., i-1] \\ 0 & \textit{else} \end{cases}$$

and think about its expected value and how can you use it for this task?

There are n hidden integers a_i, each of them belonging to the range $[1, d]$. In a single query, you may choose two integers x and y ($1 \le x \le n$, $1 \le y \le d$) and ask the following question:

$$\text{“Is } a_x \ge y\text{?”}$$

Your goal is to determine the value of the largest element in the hidden array.

(b) Give a (deterministic) algorithm that finds the largest element in the array using $O(n \cdot \log_2 d)$ queries. *(1 Point)*

(c) In this task we want to improve the query complexity. Your objective is to modify the algorithm from b) such that, in expectation, at most $O\big(n + \ln n \cdot \log_2 d\big)$ queries are needed to find the maximum element. The algorithm itself should still be deterministic! *(3 Points)*
Hint: Use the result of task a) and the fact that $H_n \le 1 + \ln n$.

Sample Solution

(a) By the definition in the hint we have $E[X_i] = P(X_i = 1) = \frac{1}{i}$. The overall number of updates X of maxSoFar is $\sum_{i=2}^{n} X_i$ as we only have update if $A[i]$ is larger than all values in the prefix. Using the linearity of expectation we get $E[X] = \sum_{i=2}^{n} E[X_i] = \sum_{i=2}^{n} \frac{1}{i} \le H_n$.

(b) We can find the value of a_1 using binary search over all potential values in time $O(\log_2 d)$. If we do that simply for each of the n many a_i, we will get all values, and hence the max value in time $O(n \log d)$.

(c) The idea is to use the binary search from the previous task in fewer cases. Indeed, as we know our current max value, we can, when considering the next a_i ask if $a_i \geq maxSoFar$. If this is not true, we can proceed with a_{i+1}. If it is true we will use the binary search to compute the exact value of a_i. By task a), in expectation there are at most $1 + \ln n$ entries where we have to apply the binary search. Thus, the number of queries is $n + O(\log d \cdot \log n)$ in expectation.

Exercise 2: Randomized Coloring *(12 Points)*

Let $G = (V, E)$ be a simple, undirected graph with maximum degree Δ. A (node) coloring of the graph is an assignment of colors to the nodes in a way that no two adjacent nodes are assigned with the same color. More formal: A coloring is a mapping $\phi : V \to C$ of nodes in V to some color space C s.t. $\phi(u) \neq \phi(v)$ if $\{u, v\} \in E$.
Consider Algorithm 2 to assign colors from the colors pace $\{1, 2, \ldots, \Delta + 1\}$ to the nodes. Let L_v be the lists of **available** colors of v, that initially is set to the color space.

Algorithm 2 Randomized Coloring

Ensure: ϕ is a proper $\Delta + 1$ coloring
Let $L_v := \{1, 2, \ldots, \Delta + 1\}$
for each uncolored node $v \in V$ in parallel **do**
    v becomes active with probability $p = \frac{1}{2}$
    if v is active **then**
        Let v choose a color $x_v \in L_v$ uniformly at random
        if no neighbor u picked x_v as well **then**
            $\phi(v) := x_v$ ▷ v is colored now!
    if v is still uncolored **then**
        delete $\phi(u)$ from L_v for all colored neighbors u. ▷ Update L_v

Note that in every iteration, $|L_v|$ is larger than the number of uncolored neighbors of v.

(a) Show that a node v that is still uncolored will be colored in the next iteration with probability at least $1/4$. *(6 Points)*
Hint: Assume v is active and has k uncolored neighbors. What is the probability that v gets colored?

(b) After how many iterations is a node $v \in V$ colored in expectation? *(2 Points)*

(c) Show that Algorithm 2 terminates in $O(\log n)$ iterations **with high probability**.
That is for a given constant $c > 0$, all nodes are colored within $O(\log n)$ iterations with probability at least $1 - \frac{1}{n^c}$. *(4 Points)*
Hint: Use the result of a) for tasks b) and c) even if you didn't manage to come up with a solution.

Sample Solution

(a) Fix an uncolored node v and assume v is active and has k uncolored neighbors. The probability that some neighbor u is active is p. Let u decides on a color x_u, the probability that v decides on the same color $x_u = x_v$ is $1/|L_v|$ (since a color from L_v is chosen uniformly at random). Note hat by construction $|L_v| \geq k + 1$. It follows that v and u are in conflict with probability $\frac{p}{|L_v|} \leq p/(k+1)$. Hence, the probability that v is in conflict with at least one of the k neighbors is by the **union bound** at most $p \cdot \frac{k}{k+1} \leq p$. Thus, v is not in conflict with any neighbor with probability at least $(1 - p)$.

Since any node v will be colored in the current iteration if it is active and not in conflict with any neighbor, this happens with probability at least $p \cdot (1-p) = 1/4$.

(b) Let X_v be the random variable indicating the iteration $1 \leq i$ when v successfully decides on a color. Let Z be the geometrical distribution with success probability $1/4$. By task a) we know that the success probability for a node v to get colored is $\geq 1/4$, hence, we have $E[X_v] \leq E[Z]$. As the expected value of any geometrical distribution is the reciprocal of the success probability we have $E[X_v] \leq E[Z] = 4$.

(c) By a), we have that v is not colored within the first i iterations with probability at most $(3/4)^i$. Hence, by the union bound there exist at least one uncolored node after i iterations with probability $n \cdot (3/4)^i$. Contrarily, the algorithm terminates in i iterations with probability at least $1 - n \cdot (3/4)^i$. Choosing $i = (c+1)\log_{4/3}(n)$, we get the desired high probability bound through the following arithmetic:

$$P(\text{Termination after } (c+1)\log_{4/3}(n) \text{ iterations}) \geq 1 - n \cdot (3/4)^{\log_{4/3}(n^{c+1})} = 1 - \frac{n}{n^{c+1}} = 1 - 1/n^c$$