University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger

Algorithm Theory
Exercise Sheet 6

Exercise 1: Balls and Bins *(2+6+4 Points)*

Assume we have n balls and n bins. We want to throw each ball into a bin without any central coordination (like "first ball in first bin"). The aim is to distribute the balls uniformly among the bins, i.e., to keep the maximum number of balls in one bin small. We use the following randomized algorithm:

For each ball, choose a bin uniformly at random.

We want to show that the maximum bin load is $O(\log n)$, with high probability.

1. For a bin i, what is the expected number of balls in i? Proof your answer.
2. Use Chernoff's bound to show that for each bin i, the number of balls in i is $O(\log n)$, w.h.p.
3. Use a union bound to show that the bin with the maximum number of balls contains $O(\log n)$ balls, w.h.p.

Exercise 2: Finding Prime Numbers *(10 Points)*

In the lecture, we have seen a randomized primality test which for a number N, tests whether N is a prime. If N is a prime, the test always returns "yes," if N is not prime, the test returns "no" with probability at least $3/4$. The running time of the test is $O(\log^2 N \cdot \log\log N \cdot \log\log\log N)$. Your task now is to find an efficient randomized algorithm that for a given (sufficiently large) input number n, finds a prime number p between n and $2n$, w.h.p. You can assume that the number of primes between n and $2n$ is at least $\frac{n}{3\ln n}$. What is the running time of your algorithm?

Exercise 3: Minimum Cut *(8 Points)*

You are given a randomized algorithm called ALG that takes as input an undirected graph G and outputs in linear time a number k with the following property:

With probability at least $1/n$, the number k is the size of a minimum cut of G.

Someone now has the idea to increase the probability of getting the size of a minimum cut by running ALG n times and take the minimum over of outputs (which results in an $O(n^2)$ algorithm). Is this a reasonable approach and what is the main difference of the above algorithm to the contraction algorithm discussed in the lecture?