Technical University of Crete

School of Electrical and Computer Engineering

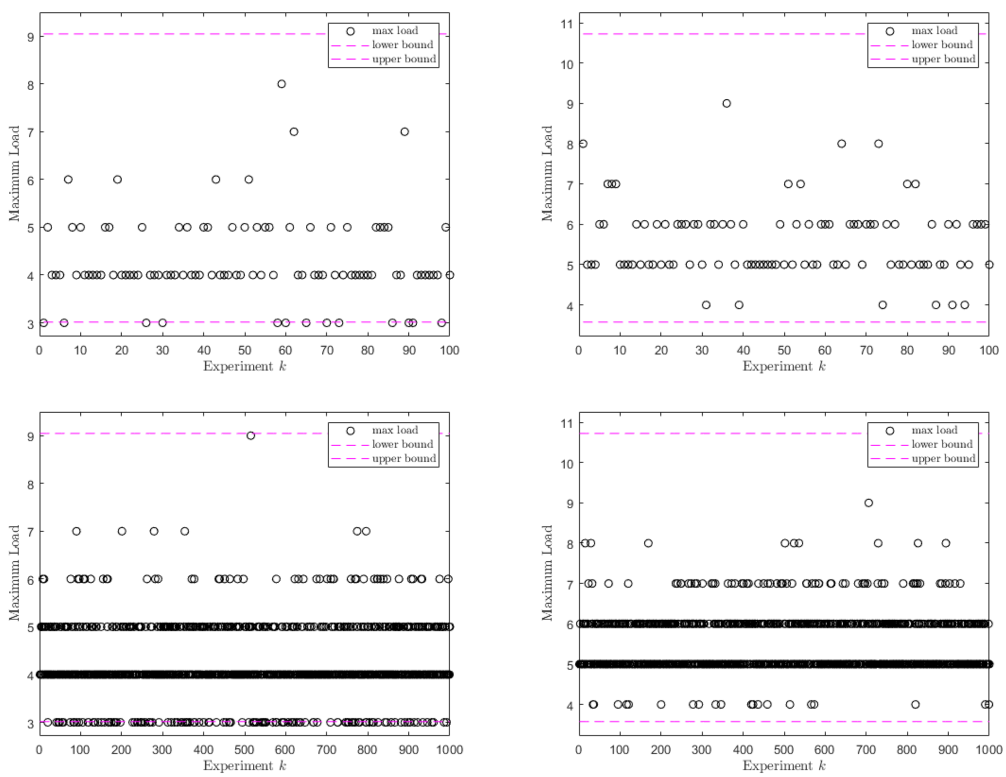
Course: **Randomized Algorithms**

Exercise 2

Angelopoulos Dimitris - 2020030038

Maximum Load in n-Balls and n-Bins

In this experiment we will simulate the problem Balls & Bins. More specifically, we will compute the maximum load of balls in some bin. Initially, we perform the experiment for $n = 100, 1000$ and repeat it $K = 100$ times. Then, we repeat the experiment $K = 1000$ times. The respective figures generated are the following.



Observations First and foremost we can see that no amount of maximum load surpasses either of the bounds. Simultaneously, for the simulations for $n = 100$, we can see that the most common max load is primarily 4 or 5 with little amount of outliers (we had to run multiple trials of repeating the experiment for $K = 1000$ to manage and get one value to lie to the upper bound). As for the experiments of using $n = 1000$ balls and bins, most of the observations above still hold, the only difference being in the value of the maximum load.

Validation To check that the results are reasonable, we plotted the lower and upper bounds $L = \frac{\ln n}{\ln \ln n}$, $M = 3\frac{\ln n}{\ln \ln n}$.

Hamiltonian Cycles

In this problem we will study and implement the modified algorithm for finding hamiltonian cycles in random graphs $G_{n,p}$. To create such a (undirected) graph we choose $n = 500, 1000$ vertices and a probability $p \geq \frac{40 \log n}{n}$. This p represents the probability of some edge (v_i, v_j) existing in the graph.

In the algorithm we created, we have to keep track of the edges we used in each step. To do this we implement two lists, `used_edges`(v) and `unused_edges`(v) for some vertex v. If we denote the second list as a random variable X_v we know that $X_v \sim B(n-1, q)$ where q is computed by solving $p = 2q - q^2$. Meaning that some edge exists in the (initial) unused edge list with probability q.

After initializing everything we implement the algorithm in two steps. First we run the algorithm until it finds a path $\mathcal{P}$ with the exception that it fails if it reaches $2n \log n$ iterations. Then we run it again, for completing the cycle, using the existing lists and path from the previous step, again terminating if it reaches $n \log n$ iterations. Some path is valid if its elements are distinct (they always are because of the nature of the algorithm) and it contains every vertex of the graph and thus having size equal to n. A valid cycle is returned if there exists a path and if the last element of the path is connected to the first (meaning that the cycle is closed). One more condition that holds for both steps is that for a head vertex v if $|$`unused_edges`$(v)| = 0$ then a hamiltonian path or cycle does not exist. Last but not least we count the amount of iterations of each step and return their values.

Validation To evaluate whether the results are reasonable we construct a list that contains either 0 or 1. Then we iterate through $\mathcal{P}$ and check if two consequent vertices are connected, if they are we append 1. If the algorithm is successful then this list contains only values of 1. By implementing this idea we verify the correctness of the Hamiltonian Path.