University of Washington
Department of Computer Science and Engineering
CSE 417, Winter 2020
Yiliang Wang

February 5, 2020

Homework 4 Problem 5

Programming Problem 5 (10 points):

Implement the greedy algorithm for graph coloring discussed in class (Lecture 9). Run the algorithm on random graphs. Use values of n of 1000 (or larger). You should report results for values of p in the range 0.002 and 0.02. How many colors are needed on the average.

Answer:

```
import java.util.HashMap;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Random;

public class Graph{
    public int n;
    public int m;
    public double p;
    public HashMap<Integer, ArrayList<Integer>> vertexMap = new HashMap<>();
    public ArrayList<int[]> edgeList = new ArrayList<>();
    public int[] degrees = new int[n];

    public Graph(int n, double p) {
        this.n = n;
        this.p = p;
        m = (int) (n * (n - 1) * p / 2.0);
        getRandomGraph();
    }

    void getRandomGraph(){
        Random rand = new Random();
        for (int i = 0; i < n - 1; i++)
            for (int j = i + 1; j < n; j++)
                if (rand.nextDouble() < p){
                    AddEdge(i, j);
                }
        }
```

```
void AddEdge(int i, int j) {
    if(!vertexMap.containsKey(i)){
        vertexMap.put(i, new ArrayList<>());
    }
    vertexMap.get(i).add(j);
    //degrees[i]++;

    if(!vertexMap.containsKey(j)){
        vertexMap.put(j, new ArrayList<>());
    }
    vertexMap.get(j).add(i);
    //degrees[j]++;
}

Integer greedyColoring(){
    int[] color = new int[n];
    int maxColor = 1;
    Arrays.fill(color, -1);
    for (int i = 0; i < n; i++){
        if (color[i] == -1){
            int[] colorToChoose = new int[maxColor];
            if(vertexMap.containsKey(i)){
                for (int j:vertexMap.get(i)){
                    if(color[j]!= -1){
                        colorToChoose[color[j]] = 1;
                    }
                }
            }
            int k = 0;
            while(k < maxColor){
                if(colorToChoose[k] == 0){
                    color[i] = k;
                    break;
                }
                k++;
            }
            if(k == maxColor){
                maxColor ++;
                color[i] = maxColor-1;
            }
        }
    }
    return maxColor;
}
```

```
        public static void main(String[] args){
            for(double i = 0.002; i <= 0.021; i += 0.001){
                int color = 0;
                for(int j = 0; j <= 100; j++){
                    Graph g = new Graph(1000, i);
                    color += g.greedyColoring();
                }
                System.out.printf("n:%d,p:%.3f,avgColorNumber:%.2f\n",
                                  1000, i, color/100.0);
                }
        }
    }
/*
n:1000, p:0.002, avgColorNumber:4.22
n:1000, p:0.003, avgColorNumber:5.04
n:1000, p:0.004, avgColorNumber:5.43
n:1000, p:0.005, avgColorNumber:6.06
n:1000, p:0.006, avgColorNumber:6.33
n:1000, p:0.007, avgColorNumber:6.99
n:1000, p:0.008, avgColorNumber:7.18
n:1000, p:0.009, avgColorNumber:7.67
n:1000, p:0.010, avgColorNumber:8.07
n:1000, p:0.011, avgColorNumber:8.40
n:1000, p:0.012, avgColorNumber:8.87
n:1000, p:0.013, avgColorNumber:9.20
n:1000, p:0.014, avgColorNumber:9.45
n:1000, p:0.015, avgColorNumber:9.91
n:1000, p:0.016, avgColorNumber:10.16
n:1000, p:0.017, avgColorNumber:10.45
n:1000, p:0.018, avgColorNumber:10.86
n:1000, p:0.019, avgColorNumber:11.11
n:1000, p:0.020, avgColorNumber:11.43
*/
```