Fall 2021
EE 134
Christina Fragouli
Osama Hanna

EE 134: Graph Theory in Engineering

Project
Due: Oct 21 at 11:59pm

1 Project Description

In this project, you will use some of the graph theoretic concepts that you have learned to come up with solutions for a real-world problem.

Wireless sensor networks refer to networks with sensors that monitor an environment and share the data with a central node. It finds applications in many areas such as supported health, area monitoring, threat detection, air quality monitoring and many others. In such networks, finding the optimal network topology is a non-trivial and important problem. This is because issues with connectivity and routing can arise if the environment contains obstacles that impede the communication between nodes, where environments can be towns or cities. In this project, we will explore strategies that can be used to efficiently place a number of sensors in a given area such that the connectivity and coverage are maximized.

Goal. More specifically, our goal is to place the *minimum* number of sensors to *cover* a certain area while maintaining good *connectivity* between the sensors and a central node.

Input. We will be given a map that has obstructed areas (corresponding to buildings) and unobstructed areas, which we want to monitor. For simplicity, we assume that the area is divided into a grid of squares with side 1, sensors can only be placed in unobstructed areas, and each sensor covers a 9×9 "big" square.

Performance Metrics. We will use the following performance metrics.

1. The number of sensors used, N.
2. The percentage A of the unobstructed area that the sensors cover.
3. The connectivity C (we explain this in more detail next).

Connectivity. To measure connectivity we first create a weighted graph with vertices representing the sensor nodes and central node, edges representing link of communication between nodes, and weights representing the rate of communication for the link.

The weights of the edges represent the communication rate between nodes and can be calculated as follows. We assume that if two nodes lie within the squares that they cover, they can communicate with a rate $\log_2(1 + \frac{10^4}{\max\{1,d^2\}})$ if the direct line between them is unobstructed; and with a rate $\log_2(1 + \frac{10^4}{\max\{1,d^4\}})$ if the direct line between them is obstructed, where d is the distance between the nodes. We assume that the distance between any two adjacent entries of the map array in the same row or same column is unity.
On this graph, we identify a spanning tree, that has as root the sink; the sensors are going to only use this tree to forward the information, so as to avoid keeping complicated state. On this tree, we measure the following three notions of connectivity:

1. The worst case minimum cut between sensors and the central node, C_w.
2. The average minimum cut between sensors and the central node C_{av}.
3. The minimum multiple unicast cut across all subsets of the set of placed sensors C_u[1].

Tree design. To select the tree, you can come up with your own algorithm, that tries to optimize connectivity metrics. You can also consider more than one algorithms and compare them.
Comparison with random placement. A straightforward approach is to randomly place sensor nodes; this is an approach that we want to compare against.

2 Map

The maps are black and white images which we represent as 2-D arrays. The elements of the arrays take the value 0 or 1, with 1 representing a pixel of an unobstructed area and 0 representing a pixel of an obstructed area. The distance between two adjacent elements in the same row or same column of the array is assumed to be unity.

3 What to submit

Please fill in the attached notebook the following functions/cells:

- *add_nodes*: A function that adds a specified number of sensor nodes N to cover the map based on your algorithm.
- *add_one_node*: A function that adds one sensor node, given previously added nodes and the map, based on your algorithm.
- *update_graph*: A function that creates and updates the connectivity graph and corresponding maximum weight spanning tree.
- Plots: figures plotting the number of sensors vs connectivity for the described notions of connectivity for number of sensors $[0 : 1000]$. Please include a plot for each notion of connectivity. Report the minimum number of sensors required to cover 90% of the map.

[1]The multiple unicast cut from a set of sensors to the central node is defined as the minimal cut that separates the central node from the given set of sensors divided by the number of sensors in the set.

You should submit a zip file containing the following items:

1. A PDF report (maximum 2 pages) describing the algorithm/s. (10 points)
2. Report the minimum number of sensors required to cover 90% of the unobstructed areas of the map. You results should be averaged over all the attached maps. Please include this in the PDF report.
3. A figure plotting the number of sensors vs connectivity for the described notions of connectivity for number of sensors $[0 : 1000]$. Please include a subplot for each of the three described notions of connectivity. The results should be averaged over all the attached maps. Please include this in the PDF report. (5 points)
4. Compare your results in 2, 3, on the same subplots, with the algorithm that places sensors in the map uniformly at random. (5 points)
5. The python code. (10 points)

The name of the file should be: FirstName1 LastName1_FirstName2 LastName2_FirstName3 LastName3. Please include your names and UIDs at the beginning of the PDF report.