## **Project Proposal**

You are welcome to discuss your project with others but you are expected to submit it individually. Crediting anyone who helped you with ideas or debugging is a must! In this project you should analyze a graph or graphs using Rust.

**Google web graph**

Nodes represent web pages and directed edges represent hyperlinks between them. The data was released in 2002 by Google as a part of Google Programming Contest.

http://snap.stanford.edu/data/web-Google.html

**Graphs**

The definition of a graph is a collection of nodes (also called vertices) connected by edges. Edges can be directed or undirected (i.e. A<->B or A->B. The first means that you can go from A to B and B to A, while the second means that you can go from A to B but not the inverse).

1. What data set are you planning to use and why it is interesting to you?

I am using web graph data from Google. The set has web pages and hyperlinks in the page. I found it interesting because it was an insight I have not really thought about despite the fact I heavily engage in the internet. The nodes of the graph are the web pages and the edges are hyperlinks. The graph is a directed graph directional flow of web page to hyperlink.

2. What is the problem you are solving or the question you are asking? Show that you have thought about it and share your insights.

3. What are the steps/components needed to accomplish the project? Specify the milestones with approximate dates you will accomplish them and how you plan to test the individual components.

N/A

## 

## 

## 

## **Project Ideas and implementation**

There is another post with ideas of where you can find interesting graph datasets (https://piazza.com/class/l78d5yndkuc401/post/35). You can use this data directly or you can create derivative data sets from this data. For example a dataset with people interests may be turned into a graph by converting people into vertices and connecting them if they have some interest in common. The dataset you use and how you construct it is entirely up to you.

Here are some possible ideas of projects you can pursue. Do not feel obligated to adopt one of those though you are welcome to. We will consider extra credit for original ideas 😃.

* Six degrees of separation: What is the usual distance between pairs of vertices in your graph? Is the answer very different for this versus another graph? (We will cover some algorithms that can be used for computing distances between vertices later in the class. They are called Breadth–First Search and shortest Paths)
* Degree distributions: What is the distribution of vertex degrees in your graph? What if you look at the number of neighbors at distance 2? Often people believe that such a distribution should be a power–law distribution (look up online what it means) for social networks and similar graphs. How well does the distribution you see fit this assumption (and how are you going to evaluate it)? Degree of a vertex is the number of other vertices it is connected to.
* How often are friends of my friends my friends? This is very generic question, but can you find two vertices who are friends with most similar or most dissimilar sets of connections? What is the right measure of similarity? This could involve searching for established measures of this type and or coming with your own.
* Graph clustering and partitioning: Take a network in which the identity of nodes is meaningful. Try to find k best representatives of all the graph vertices. How satisfied are you with the selection, given your previous beliets about this network?
* Densest subgraph: (The vocabulary here may not all be obvious, but I’m happy to answer all ques- tions you may have.) Let V be the set of vertices of the graph. For any V′ ⊆ V, we write E(V′) to denote the subset of edges of the graph that have both endpoints in V ′. The density of a subgraph induced by V ′ is  
    
    
   |E(V′)|/|V′|  
    
    
   In the densest subgraph problem, the goal is to find a subset V ′ ⊆ V that maximizes the density.  
    
    
   This problem can be solved exactly, but this is too complicated for this class, I think (but feel free to prove me wrong). Instead there is a relatively simple algorithm that gives a 2-approximation, i.e., a subset V ′ with density at least half the maximum possible. See, for instance, the top of page 5 in https://people.cs.umass.edu/∼barna/paper/dense-subgraph-full.pdf  
    
  Your goal could be to implement this algorithm and apply it to some graph with meaningful vertices. Are the results as expected? What happens when you look at specific subgraphs (say, people living in Massachusetts, or college students)? Are the results surprising?
* Centrality measures: Compute select centrality measures for your graph. Do the highest centrality vertices match your your intuition? What happens when you look at some meaningful subgraphs. Is this the case for them?  
   A few random links that discuss centrality measures–<https://en.wikipedia.org/wiki/Centrality> –<https://towardsdatascience.com/notes-on-graph-theory-centrality-measurements-e37d2e49550a>

In particular, look at closeness and betweenness centralities as something that is relatively easy to compute.