CSCI 6907: Project #1

Samuel Dooley

February 15, 2016

Our goal is to answer the following questions about the Astronomy co-authorship dataset:

1. Who is the *central person(s)* in the graph?
2. What is the *longest path* in the graph?
3. What is the *largest clique* in the graph?
4. Given a particular node/person, what is its/his/hers *ego*?
5. Given a particular node/person, what is its/his/hers *power centrality*

Note: For 4 and 5, I'm not exactly sure if these are the questions he is asking. He just writes the words 'ego' and 'power-centrality' which are properties of a node. Thus, I imagined that he intended for us to implement functions that would calculate this for a given node. What did you think about this?

In the process of answering these five questions, we have played with the following functions:

1. `degree(graph, v = V(graph), mode = c("all", "out", "in", "total"), loops = TRUE, normalized = FALSE)`

   This is a function that takes a graph as input and outputs an array of all the graph's vertices with each one's corresponding degree. The degree of a vertices is its number of 'neighbors,' i.e., how many places can one go from a given node while traversing this graph. For example:

   ```
   > degree(astrocollab)
   BIERMANN, PL    STANEV, TKGT      GOLDMAN, I      WANDEL, A
            36               1              5              4
   ....
   ```

   This array is ordered as the array of vertices is ordered. This BIERMANN, PL appears first in both this array and the array of vertices (this can be tested by `V(astrocollab)[1]`). To get the answer for the person with largest degree, or the central person, we can perform

   ```
   > sort( degree(astrocollab), decreasing = TRUE)[1]
   ```

2. woierhs
3. `largest_cliques(graph, ...)`

Answers

1. "FRONTERA, F"

Code

```
# Central person:
# input: graph
# output: array of strings with
#         names of nodes with the most neighbors
central_person <- function(graph) {

  d <-sort( degree(astrocollab), decreasing = TRUE)

  # Get the people with the most
  m <- max(d)
  n <- names(which( m == d ))
  return(n)
}
```