# Network Analytics

**Group 4**

November 21, 2016

## Question 1

(a) **Incidence Matrix**

We read the data into using the `read_edgelist()` function and get the following incidence matrix.

| | (a,b) | (a,d) | (d,b) | (d,e) | (b,e) | (b,c) | (c,d) |
|---|---|---|---|---|---|---|---|
| a | -1 | -1 | 0 | 0 | 0 | 0 | 0 |
| b | 1 | 0 | 1 | 0 | -1 | -1 | 0 |
| c | 0 | 0 | 0 | 0 | 0 | 1 | -1 |
| d | 0 | 1 | -1 | -1 | 0 | 0 | 1 |
| e | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

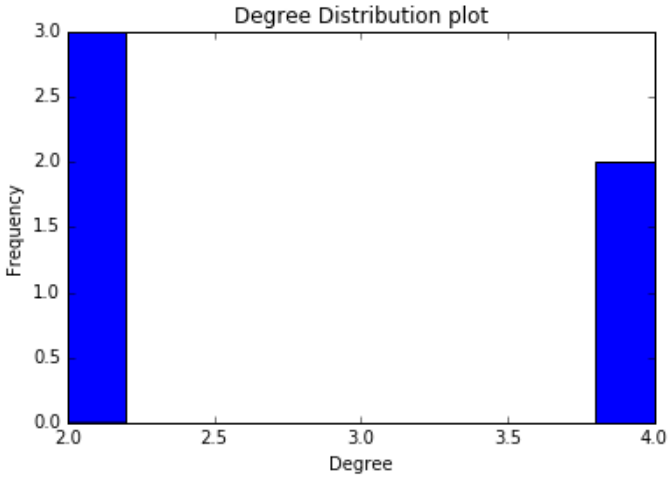
(b) **Shortest Path Matrix**

| | a | b | c | d | e |
|---|---|---|---|---|---|
| a | 0 | 5 | 15 | 3 | 6 |
| b | inf | 0 | 10 | 8 | 7 |
| c | inf | 6 | 0 | -2 | 1 |
| d | inf | 8 | 18 | 0 | 3 |
| e | inf | inf | inf | inf | 0 |

(c) **Diameter**

Using the shortest path matrix above, we iterate through all values to find the maximum value while ignoring the $\infty$ values. The diameter of the graph is 18.

(d) **Degree Distribution**



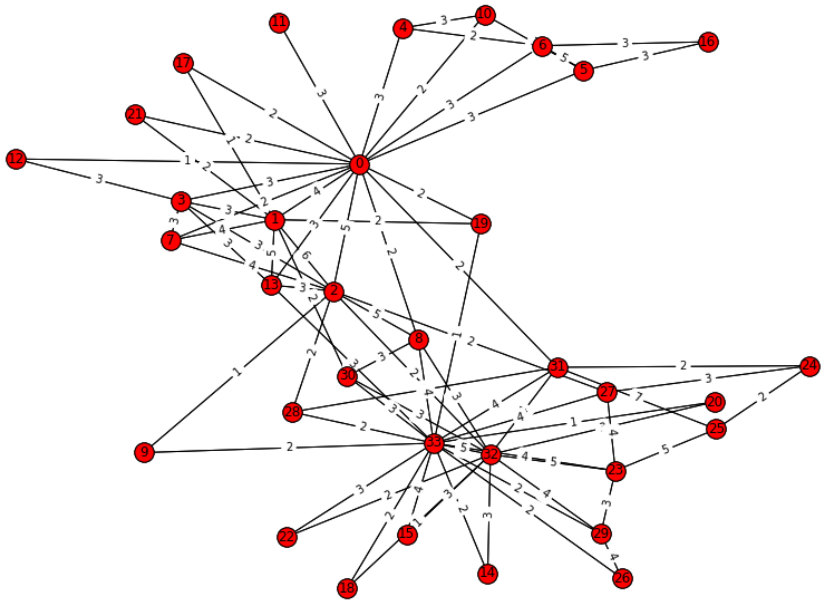
The support of the distribution is {2,4} with the probabilities,

$$\mathbb{P}(K = k) = \begin{cases} \frac{3}{5}, \text{if } k = 2 \\ \frac{2}{5}, \text{if } k = 4 \end{cases}$$

(e) **Connectedness**

Using the commands `print(nx.is_strongly_connected(G))` and `print(nx.is_weakly_connected(G))`, we check both weak and strong connectivity. We find that the graph is WEAKLY connected but not STRONGLY connected due to the directed edges.

# Question 2



Because the graph is weakly connected, it is important to visualize the network in a way that shows which nodes are connected with each other, placing nodes in their relative places to each other. The spring format for the graph shows that nodes 0, 32, and 33 in particular have high degrees and thus are considered as nodes towards the center. In a similar manner, nodes with low degrees are plotted towards the outside of the graph, which makes it easy to visualize a node's relation to other nodes. Therefore the spring layout outputs a graph that visually shows a node's degree in addition to presenting the edges between nodes without much overlap. The visualization combines the adjacency matrix to draw the presence of an edge, and also displays the edge's weight. Thus, the algorithm provides a balance between emphasizing a graph's connectedness, adjacency, edge weights, and node degrees.