Problem statement

This assignment deals with **link/edge prediction** in networks. A network is represented as an undirected graph. Such a graph is viewed as $G(V, E)$, where V is the set of nodes and E is the set of edges; *there are no self edges or multiple edges.* Let degree of node $x \in V, d(x)$ be given by

$$d(x) = |e : e \in E \; and \; x \; is \; an \; end \; vertex \; of \; e|.$$

Let U be the universal set containing all $\frac{|V|.(|V|-1)}{2}$ possible edges, where $|V|$ is the number of vertices. So, $U - E$ is the set of **nonexistent edges** in G. We would like to predict K **important nonexistent edges** based on **ranking** the elements of $U - E$. Your **implementation has to deal with two parts** as explained below:

- Part1:
  - The compressed directory is *contact-high-school-proj-graph.tar.gz* and is available for download from **https://www.cs.cornell.edu/˜arb/data/contact-high-school/index.html**. Note: Be careful about the "tilde" character while doing copy-paste.
  - There are two files in the folder/directory. You need to consider the file *contact-high-school-proj-graph.txt* only. There are 5,818 lines in the file; each line corresponds to a weighted edge of the graph in the form $i \; j \; w$ where i and j are the two nodes of the undirected edge and w is its weight.
  - This dataset corresponds to a social network of 327 high school students and two nodes have an undirected edge between them if the associated students are friends. Observe that j values are listed in non-decreasing order.
  - Note that there are 327 nodes in the graph. So, $|V| = 327$ and $|E| = 5818$ in G. Convert it into a binary graph by viewing all these w values to be equal to 1. **Use this edge information to store the given undirected graph G as an adjacency list.**

- Part2: *Use the adjacency list to rank each of the nonexistent edges*, that is elements of $U - E$, using the following **four scoring functions and print the K top-ranked edges for a given K, in $U - E$, along with their respective scores** in each case. Any edge $e \in U - E$ may be viewed as an ordered pair $< v^1, v^2 >$, where $v^1, v^2 \in V$ are the end vertices of edge e.

  1. **Jaccard's coefficient (JC):** For edge $e_i \in U - E$, where $e_i =< v_i^1, v_i^2 >$,

  $$JC(e_i) = \frac{|\Gamma(v_i^1) \cap \Gamma(v_i^2)|}{|\Gamma(v_i^1) \cup \Gamma(v_i^2)|}$$

     where $\Gamma(v)$ is the set of neighbors of the vertex v. For the edges $e_i, e_j \in U - E$, e_i is **more important** than e_j if $JC(e_i) > JC(e_j)$. Sort the edges in $U - E$ by non-increasing value of their JC.

  2. **Katz's score (KS$_\beta$):** For edge $e_i \in U - E$, where $e_i =< v_i^1, v_i^2 >$,

  $$KS_\beta(e_i) = \sum_{l=2}^{6} \beta^l \cdot |paths^l_{v_i^1, v_i^2}|$$

     where $paths^l_{v_i^1, v_i^2}$ is the set of paths of length exactly l between v_i^1 and v_i^2. Use a value of 0.1 for β in computing the KS score. Sort the edges in $U - E$ by non-increasing value of their KS_β.

3. **Hitting time (HT):** For edge $e_i \in U - E$ where $e_i =< v_i^1, v_i^2 >$, hitting time of e_i is given as

$$HT(e_i) = -R_{v_i^1, v_i^2} \tag{1}$$

Here $R(v_i^1, v_i^2)$ denotes the expected time of random walk from v_i^1 to reach v_i^2. Sort the edges in $U - E$ by non-increasing value of their HT.

4. **rooted PageRank (PR$_\alpha$):** For edge $e_i \in U - E$ where $e_i =< v_i^1, v_i^2 >$, $PR_\alpha(e_i)$ is defined as the stationary distribution weight of v_i^2 under the following random walk:

   with probability α, jump to v_i^1.

   with probability $1 - \alpha$, go to random neighbor of current node.

   Use a value of 0.2 for α in computing the PR. Sort the edges in $U - E$ by non-increasing value of their PR_α.

Reference: *David Liben-Nowell, Jon M. Kleinberg: The link prediction problem for social networks. CIKM 2003: 556-559.*