Experimenting with networkx

The **Going further** section is not thought to be done in class. Check it only if you have finished earlier or if you want to know more.

1. Simple network analysis with python and networkx.

   `networkx` package has a good documentation. To get started, do the beginning of the tutorial: `https://networkx.org/documentation/stable/tutorial.html`. After that, the best way to find what you're searching for is to ask google. For instance, if you wonder how to compute the betweenness centrality with networkx, just search *betweenness networkx* in google and your first result will certainly contain the answer.

   I highly recommend to use Jupyter Notebook to make those experiments. Notebooks allow to run pieces of codes without reloading the whole data everytime. If you don't have it installed yet, you can use online notebooks, such as `https://colab.research.google.com`.

   The airport dataset is a network of connections between airports, i.e., an edge exist between two airports if a company offer a direct flight between the two. It is available on the webpage of the class.

   (a) Using networkx, load the airport dataset using `read_graphml`. (in google colab, you can first retrieve it with `!wget http://cazabetremy.fr/Teaching/CN2020/airportsAndCoord.graphml`

   (b) Compute the number of nodes and edges of your graph. A simple way to go is to use `len` on `g.nodes` and `g.edges`

   (c) Compute the density and the clustering coefficent (`nx.density(g)` , `nx.transitivity(g)`)

   (d) Compute the average shortest path length and the diameter of the graph. You'll encounter a connected component issue. Can you understand why? As a solution, you need to apply those methods on the largest connected component, that you can extract with
   `cc = g.subgraph(sorted(nx.connected_components(g), key=len, reverse=True)[0])`

   (e) Obtain the list of the 20 nodes of highest and lowest degrees. You can use `degree` and `sorted(X, key=lambda x: x[1])` for instance.

   (f) Compute the list of the 20 nodes of highest and lowest Pagerank. You might need to use `items()` to transform a dictionary in a list of pair. Observe the differences.

   (g) Do the same for the betweenness (You might need to check parameter `k` of the function). Where in hell is **Anchorage**? And **Port_Moresby**? Investigate a little to understand what is going on. How many neighbors do these nodes have, and who are they?

   (h) Check with other typical node centralities.

   (i) Check edge betweenness.

   (j) Would you say that the network is a *small world* network ? To compare with a random network, you can generate one with `gnm_random_graph`.

   (k) Plot the network using `draw_networkx`. Check the documentation to see how you could improve your plot (node colors, size, layouts, etc.). Default plots are of poor quality. You could also use `https://pyvis.readthedocs.io/en/latest/` to create interactive, tunable graphs in python (but takes a bit more effort)

(l) Use the `pos` argument to plot nodes according to their geographical position. You can access node attributes either by using `get_node_attributes`, or by simply accessing the node, i.e.: `G.node['node_name']` gives existing attributes of this node, `G.node['node_name']["attribute1"]` gives the value of an attribute for a node

(m) It can be useful to export a graph with computed node or edges properties. Add their PageRank score to nodes as an attribute using `set_node_attributes`.

(n) Save the graph in graphml format using `write_graphml`. Check that you can open this file with Gephi and that the PageRank score is available as a node property.

2. Going further : Assortativity, homophily, friendship paradox

(a) During the class, we mentioned notions of assortativity, homophily, and friendship paradox. Check the page `https://networkx.org/documentation/networkx-1.10/reference/algorithms.assortativity.html` of networkx, which contains functions related to these questions. On the airport dataset, explore these questions. You can use the `country` attributes, in particular.