Task 11: Betweenness

Node betweenness centrality

The **betweenness centrality** of a node is based on the number of shortest paths which pass through that node. Where multiple shortest paths between a pair of nodes exist, the proportion of those paths that go via the node is considered.

In general, computing the betweenness centrality is expensive. In this practical, you should implement Brandes' algorithm, which is relatively efficient. You can find the description of the algorithm (including pseudocode) in Brandes (2008). You can use `-1` where the pseudocode says infinity.

Run your implementation on the `simple_network.edges`, as for Task 10. Your code should return betweenness centrality for each node.

What does the distribution of betweenness centralities look like? Can you see that this could give you good splitting points for clique finding in the next task?

Starred tick

Try investigating the efficiency of your implementation on some of the real networks you can download from the SNAP website, or automatically generate some random networks.

Brandes (2001) plots time against number of vertices for various networks. There's been a big improvement in computing power since that paper was written. Can your implementation cope with significantly larger networks?