Forming a graph of Wikidepia topics

Graph Theory

Christina Koutsou
github: @kchristin22

1 About

The purpose of this project is to implement a network of Wikipedia articles related to a specific subject. In more details, the Wikipedia page with the subject as a title serves as the root node and is searched for links that reference other correlated articles. Afterwards, each such link is also scanned for articles related to it, forming a graph reminiscing a tree.

2 Algorithm and Implementation details

The program's parameters include:

- **Theme**: The central topic around which the graph is constructed.
- **Minimum correlation**: The threshold that determines whether a link of a page is included in the graph, based on their semantic correlation.
- **Tree depth**: The level of depth to which the tree search extends.

In order to speed up the process, due to work being download-heavy, multiple processes are utilized. Processes are preferred over threads to utilize more CPUs, as threads in Python are restrained to a single CPU. In addition, to mitigate potential system hang-ups caused by simultaneous network requests, a delay is introduced after each request to minimize the risk of deadlock.

Each process gets a chunk of subjects linked to the current parent node and returns the children that meet the correlation requirement. In more detail, Wikipedia articles are searched based on the name of the provided child. The relation of the parent and the child pages is determined based on their summaries' semantic correlation. Due to the summary containing a multitude of sentences, the correlation limit should be lees strict to acquire a broader range of subjects. Provision has also been taken for exceptions occurring in retrieving the summary of the matched article. In particular:

- Disambiguation errors are treated by matching the first option containing the exact character sequence in its title. If no suitable option is found, it has been shown that the subject refers to a number and thus it is converted to one.
- Subjects causing page errors are ignored

After returning, the algorithm tries to identify instances of the children nodes with different casings already present in the graph, aiming to prevent the inclusion of duplicate themes. The equivalent edges between themes are then appended to the list, if they're not already included, with their correlation metric serving as their weight. If the current tree level is below the user-specified depth and the child node hasn't been visited previously as a parent node, the function is invoked recursively with updated inputs. Consequently, the algorithm falls under the Depth-First category.

Afterwards, the formed graph can be studied and analyzed. Due to the edges being directed and the nature of the task, the following metrics are calculated and stored in a dictionary:

Metric	Dictionary name
average clustering coefficient	avg clustering
degree centrality for each node	degree centrality
in degree centrality for each node	in degree centrality
out degree centrality for each node	out degree centrality
eigenvector centrality	eigenvector centrality
node influence based on VoteRank	voterank influential nodes
number of strongly connected components (scc)	num of strongly connected components
node influence based on PageRank	pagerank influential nodes
correlation of all nodes to the root node	all nodes correlation with the main theme

Table 2.1: *Metrics provided by NetworkX that are included in the code*

3 Analysis of a specific graph: Quantum computing

Since the relation between two nodes of the graph is based on their semantic similarity, it is evident that topics are more interesting to study than people.

Such an example could be Quantum computing. The graph shown in Figure 3.1 is the result of the algorithm run with a correlation limit of 0.5 and a tree depth of 2. The visualization technique used is Force Atlas which helps in identifying communities.

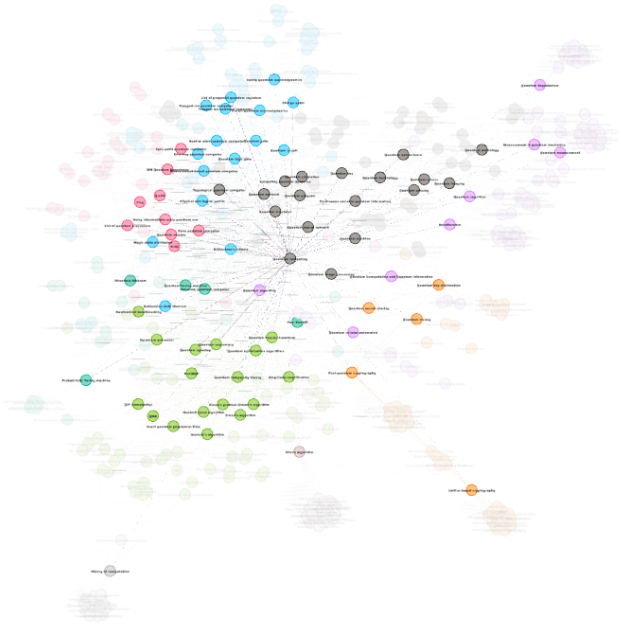

Figure 3.1: *Main node connections*

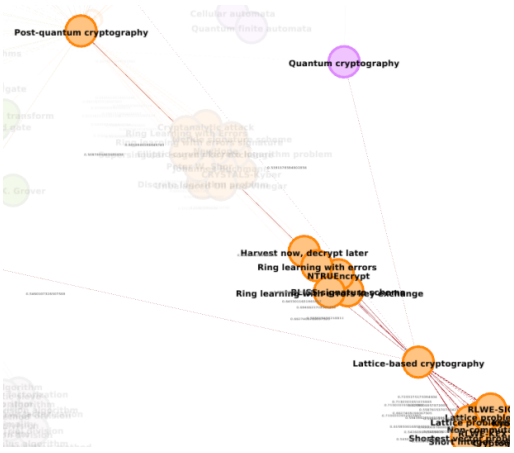

Figure 3.2: *Common ground between nodes*

It is interesting that even at the first level of the tree, independent communities are formed. That is to say that the range in themes becomes very broad and expands to different directions resulting in minimal common

ground among existing nodes. Even though the correlation limit plays a role in diminishing the importance and magnitude of these communities, the formation of independent communities cannot be avoided, only limited. There are also groups of nodes that are accessed by different parents. Such an example is the duo of Lattice-based cryptography and Post-quantum cryptography, but who also diverge at other points.

The average clustering coefficient, denoting the likelihood of neighbours of a selected node being connected too, is calculated around 0.2 (0.22 using NetworkX and 0.2 using Gephi), meaning that 1 in 5 neighbours are connected to each other. This metric decreases as the correlation limit is increased and its current value indicates that the selected theme is broad. This is also supported by the fact that the number of strongly connected components in the graph is 332, while the total nodes are 411, signifying that the network has a fragmented structure with many smaller communities rather than a few large ones.

The dispersion of the topics is also apparent in the fact that even the most central nodes are connected to less than half of the graph's nodes. In addition, the top out-degree centralities are greater than their in-degree counterparts, hinting towards the network's inclination to expand. Specifically, the five most important nodes based on degree centrality, eigenvector centrality, the VoteRank algorithm and the PageRank algorithm are displayed in the tables below.

Nodes	Degree Centrality
Quantum computing	0.39
Quantum algorithm	0.28
DiVincenzo's criteria	0.23
Quantum Turing machine	0.20
Quantum network	0.19

Table 3.1: *Top 5 nodes with highest degree centrality*

Nodes	In degree Centrality
Quantum computing	0.19
Quantum algorithm	0.12
DiVincenzo's criteria	0.11
Quantum network	0.09
Quantum Turing machine	0.09

Table 3.2: *Top 5 nodes with highest in degree centrality*

Nodes	Out degree Centrality
Quantum computing	0.20
Quantum algorithm	0.16
DiVincenzo's criteria	0.12
Quantum Turing machine	0.11
Universal quantum computer	0.11

Table 3.3: *Top 5 nodes with highest out degree centrality*

Nodes	Eigenvector Centrality
Quantum computing	0.33
Quantum algorithm	0.27
DiVincenzo's criteria	0.23
Quantum Turing machine	0.21
Quantum network	0.20

Table 3.4: *Top 5 nodes with highest eigenvector centrality*

Influential nodes based on VoteRank
Quantum computing
Quantum algorithm
DiVincenzo's criteria
Decoherence
Shor's algorithm

Table 3.5: *Top 5 influential nodes based on VoteRank*

Influential nodes based on PageRank
Quantum computing
Quantum algorithm
DiVincenzo's criteria
Quantum network
Kane quantum computer

Table 3.6: *Top 5 influential nodes based on PageRank*

The majority of the algorithms mentioned above identify the same nodes as the most influential. Upon closer examination of these nodes using Gephi, it becomes apparent that they are interconnected, forming bridges between communities (refer to community themes for more details).

In order to get a more general picture of a node's degree, it is important to study the distribution of this value. In detail, most nodes exhibit a degree of less than 10, with the average degree being 4.5, a significantly low figure considering the graph's 411 nodes. Similarly, the graph's density is remarkably low, hovering around 0.011. Notably, the diameter of the graph is 4 which is equal to twice the depth of searching as expected. The radius is calculated as 0 due to self loops being present in the graph.

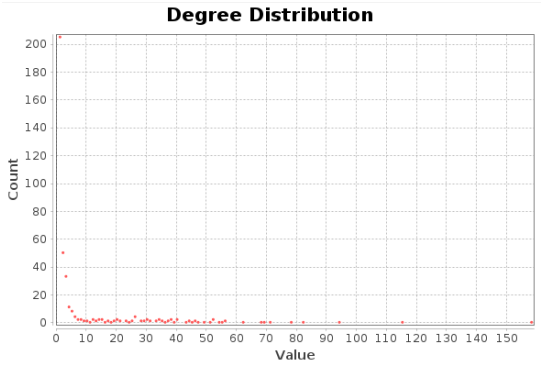

Figure 3.3: *Degree Distribution*

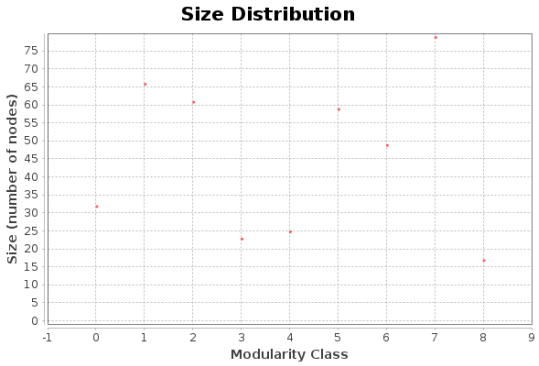

Figure 3.4: *Size of modules*

As far as the themes of the communities are concerned, the broader areas that the subjects in each community revolve around are:

- Community id=0: Quantum computing in industry
- Community id=1: Quantum Algorithm Complexity
- Community id=2: Quantum Circuits
- Community id=3: Quantum Algorithms
- Community id=4: Quantum Turing Machine
- Community id=5: Quantum Measurement
- Community id=6: Quantum Cryptography
- Community id=7: Quantum Physics and Mechanics
- Community id=8: Theory of Computation

These communities also represent the structural gaps in the network. In other words, these themes do not show a great overlap with each other, a fact evident in the elevated modularity coefficient.

Lastly, the correlation of each node to the central topic was computed. Specifically, for the nodes that were not directly connected to the main one, their correlation was computed dynamically as follows:

```
    # find parent node included in the previous depth
    cur_parent = [i for i in cur_depth_nodes[depth - 1] if i in graph.predecessors(node)]
        # list
    this_node_weight = graph.edges[cur_parent[0], node]["weight"] # this node's weight in
        regard to root

    for successor in graph.successors(node):
        this_successor_weight = graph.edges[node, successor]["weight"]
        # scale weight of this successor
        all_children.append([subject, successor, this_successor_weight * this_node_weight])
```

The nodes with the highest correlation to Quantum computing turned out to be:

- List of proposed quantum registers, 0.69, community id=2
- Quantum algorithm, 0.69, community id=7
- DiVincenzo's criteria, 0.67, community id=2
- Quantum network, 0.66, community id=5
- Quantum speedup, 0.66, community id=1

In other words, the topics with the closest relation to Quantum computing may be the ones belonging to the aforementioned communities.