**Network Of Research Papers**

***Goal***

The goal is to make a directed graph where each node / vertex represents a research paper, and the edge is from all the papers / nodes that have cited this paper / node to this node.

***Initial Idea***

Run two processes in a multiprocessing setup. One process will fetch research papers along with all their citations. Another process runs a network visualization package that updates its nodes and vertices when process 1 finds a new paper and its corresponding citations.

* Fetching Research Papers
* Visualizing Networks

**Fetching Papers**

***How to?***

* Many ways.
* ***Use Beautiful Soup and requests API***
* ***Scholarly API***
* ***SerpAPI***
* ***Crossref REST API***

***Tried and Failed***

* **Beautiful Soup** was a mess. Fetching only the titles involved a lot of manual scripting and workarounds for every site that hosted / published the research paper.

This happens because all information we want about a paper is not shown in the Google Scholar query result. We would have to click on the paper to get the details we want. Different papers are hosted on different sites with different html structures.

I think Beautiful Soup will be very hard to maintain in the long run, so I have given up on this for now.

* **Scholarly API**gave results which included citations, title, authors, and dates by default. Would’ve been great addition to our project but for its reputation due to which Google Scholar blocks any traffic from it. Some queries are allowed though. TOR and ProxyServices are some possible add-ons to this method. I used some Free Proxies with it but the queries were very slow. Like ~20 mins for a single result. Still on the list for future usage.
* **SerpAPI** is what I think would be best for our use, but unfortunately this option is too expensive. ~$250/month for around 30K searches a month.

**Crossref API**

This is what I’ve decided to use.

**Background**

**DOI:** Digital Object Identifier. This is a unique ID given to papers. This is accepted by all the publishers to identify their papers. DOIs are unchangeable. Even if a URL where a paper is hosted changes, DOI doesn’t change. It will update itself to point to the new URL. Clicking on DOI brings you to the paper.

**Crossref:** A DOI registration agency for scholarly articles. Has its own API. Publishers submit their metadata here. 70M DOIs. 5.5K publishers (Springer, Wiley, IEEE, …)

**Example:**

<https://link.springer.com/book/10.XXXX/XXX-X-XXXX-XXXX-3>

This is a paper linked to through Google Scholar. 10.XXXX/XXX-X-XXXX-XXXX-3 is the DOI of this paper. (X represents numbers.)

**Usage**

<https://api.crossref.org/works/DOI>

Gives a JSON metadata containing title, author, citations, and others

**Visualizing Networks**

**How to?**

* Two ways
* **Matplotlib**
* ***Pyvis***

**Tried and Left Behind**

* Easiest way to make networks is to use the **Networkx** library. The default visualization is through Matplotlib. This is okay but for the traditional non-interactive feel Matplotlib radiates. It has various functions on centrality including Eigenvector Centrality, which is a must have, so definitely on list for future usage

**PyVis**

**Background**

Interactive graphs using **JSvis**. Has integration for **Networkx**, so the problem with centrality functions is solved, however for complicated labeling and visualizations we’d have to use additional attribute dictionaries that may not be supported by traditional **Networkx** library.

**PyVis** uses its own **Network()** class.

**Unsure Of**

Centrality Functions Vs Visualizations

**Correspondingly,**

Networkx Vs PyVis

Can write my own version of Eigenvector centrality, shortest path algorithms, and degree of centrality. But I can’t make matplotlib interactive and visually appealing. **Leaning towards PyVis library**.