Hands-On Session: Data Extraction with Scrapy

In this hands-on session, we will use Scrapy in order to extract structured data from a Web site, namely the Mathematics Genealogy Project, a site that lists PhD advisors of numerous modern and past mathematicians (and computer scientists). The goal of this hands-on session will be to automatically build an (academic) genealogy tree of a given researcher. See below for an example.

1. Get familiar with this Web site to understand how it is organized.

2. Consult the `robots.txt` file: does the site allow crawling?

3. By following the Scrapy Tutorial, start building a Scrapy crawler that accesses the page of one specific researcher (for the following, we recommend to take someone from the 20th century so that he or she does not have too many descendants) and extracts his or her name, graduation date, alma mater, and dissertation title. You can use Scrapy's feed exports in order to simply produce a JSON file containing all information extracted. Use either CSS or XPath selectors, depending on which is simpler for a given data item. Do not hesitate to use Scrapy's shell as discussed in the tutorial to help debug selectors.

4. Make sure to configure Scrapy to respect a delay of at least 1 second between requests

5. We now would like to crawl all ancestors and all descendants of the chosen researcher (but not ancestors of the descendants or descendants or the ancestors!). For this, you will need to use specific *callbacks* when adding new HTTP requests to Scrapy's queue, allowing you to distinguish whether you are crawling descendants or ancestors. Again, use either CSS or XPath selector. Take into account the fact that an individual may have several advisors.

6. Make sure the same information (name, graduation year, alma mater, dissertation title) is extracted for every user crawled.

7. Write a simple script to transform the JSON produced into the input format of the Graphviz graph visualization software to display the information extracted as a graph. Use graphviz to produce an SVG image of the genealogy tree.

8. Pass as spider argument the name of a researcher to your script and have the crawler use the search engine of the Web site to automatically find the page of this researcher.

9. Add to your crawler a Downloader Middleware that is used to make sure every page accessed is saved on disk; and that every time a request is issued, one first checks whether the corresponding page has been stored on disk, and if so return it directly.