# Class Assignment
# Web and Social Computing (IT750)

**Submitted by: Prajwal M P**
**Roll Number: 192IT015**
**Submitted to : Dr. Sowmya Kamath**

## Web Crawling

For finding information on the thousands of millions of Web pages that exist on the internet, a search engine employs special software bots, called spiders, to build lists of the words found on Websites. When a spider is building its lists, the process is called Web crawling.

One of the open source crawlers available is crawler4j. It is a java based crawler with a wide range of functionality.

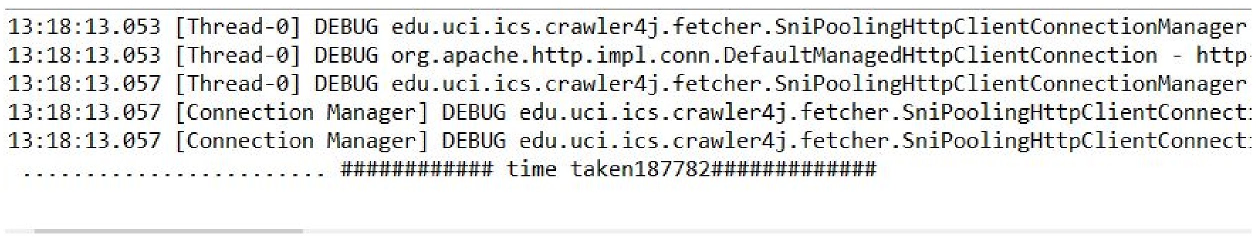
### [1] Time taken vs threads

| threads | time taken(s) |
|---|---|
| 1 | 187 |
| 2 | 143 |
| 3 | 130 |
| 5 | 110 |

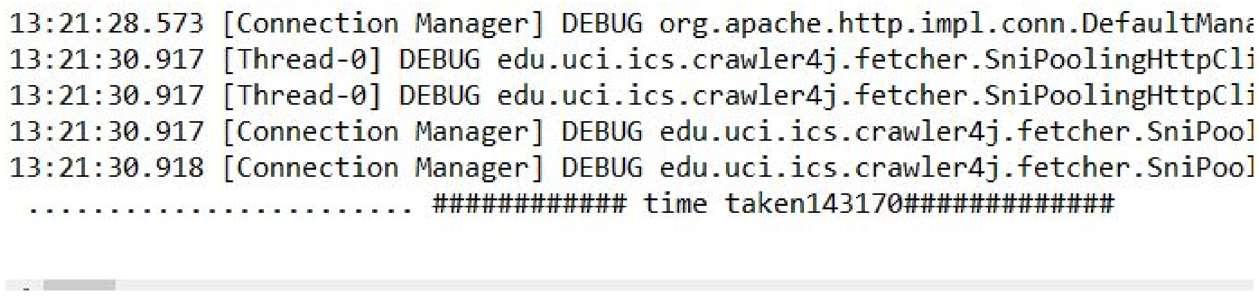
As the number of threads increases, we can see that the time taken is decreasing rapidly. The site visited is the main website of crawler4j itself.

The max number of pages to be fetched was set to 100.

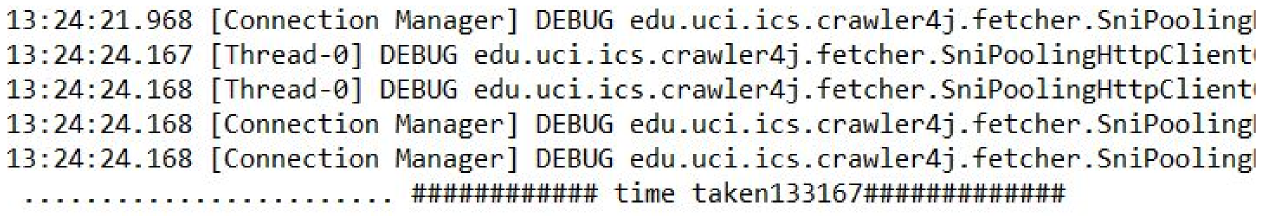
**Screenshots**:



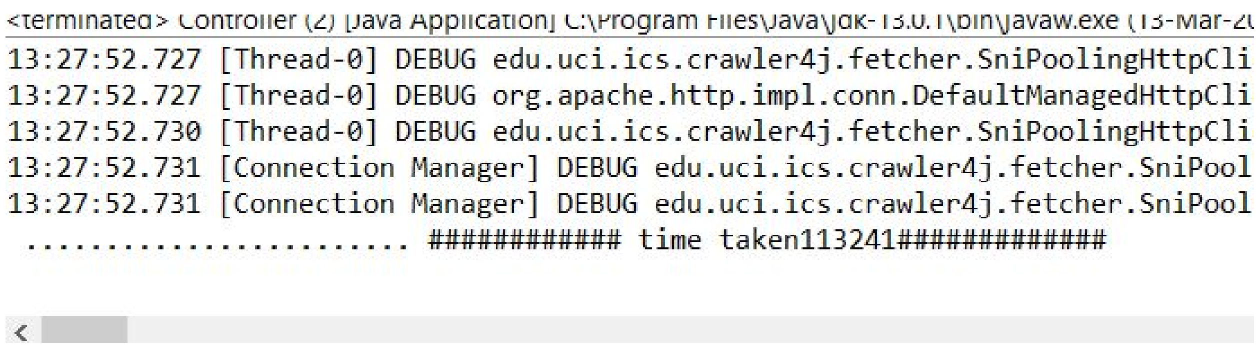
```
13:18:13.053 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClientConnectionManager
13:18:13.053 [Thread-0] DEBUG org.apache.http.impl.conn.DefaultManagedHttpClientConnection - http
13:18:13.057 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClientConnectionManager
13:18:13.057 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClientConnect
13:18:13.057 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClientConnect
 ........................ ############ time taken187782#############
```



```
13:21:28.573 [Connection Manager] DEBUG org.apache.http.impl.conn.DefaultMana
13:21:30.917 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpCli
13:21:30.917 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpCli
13:21:30.917 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPool
13:21:30.918 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPool
 ........................ ############ time taken143170#############
```



```
13:24:21.968 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPooling
13:24:24.167 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClient
13:24:24.168 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpClient
13:24:24.168 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPooling
13:24:24.168 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPooling
 ........................ ############ time taken133167#############
```



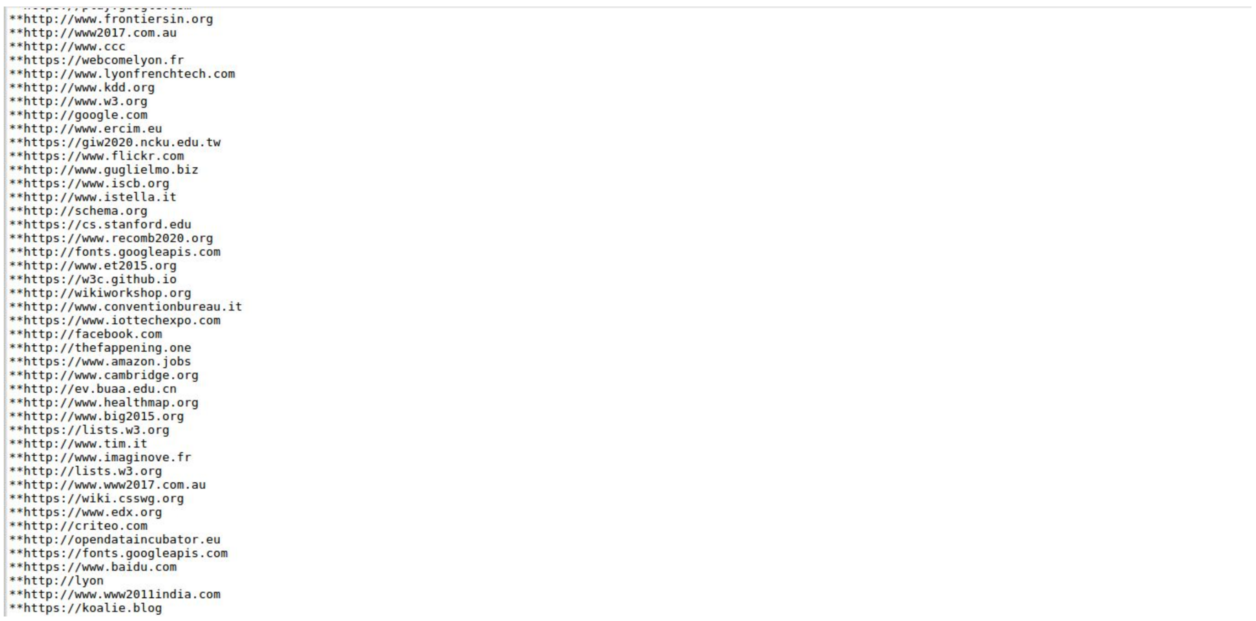
```
<terminated> Controller (2) [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (13-Mar-20
13:27:52.727 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpCli
13:27:52.727 [Thread-0] DEBUG org.apache.http.impl.conn.DefaultManagedHttpCli
13:27:52.730 [Thread-0] DEBUG edu.uci.ics.crawler4j.fetcher.SniPoolingHttpCli
13:27:52.731 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPool
13:27:52.731 [Connection Manager] DEBUG edu.uci.ics.crawler4j.fetcher.SniPool
 ........................ ############ time taken113241#############
```

From the screenshots above we can clearly see that the time taken is decreasing linearly as the number of threads increase.

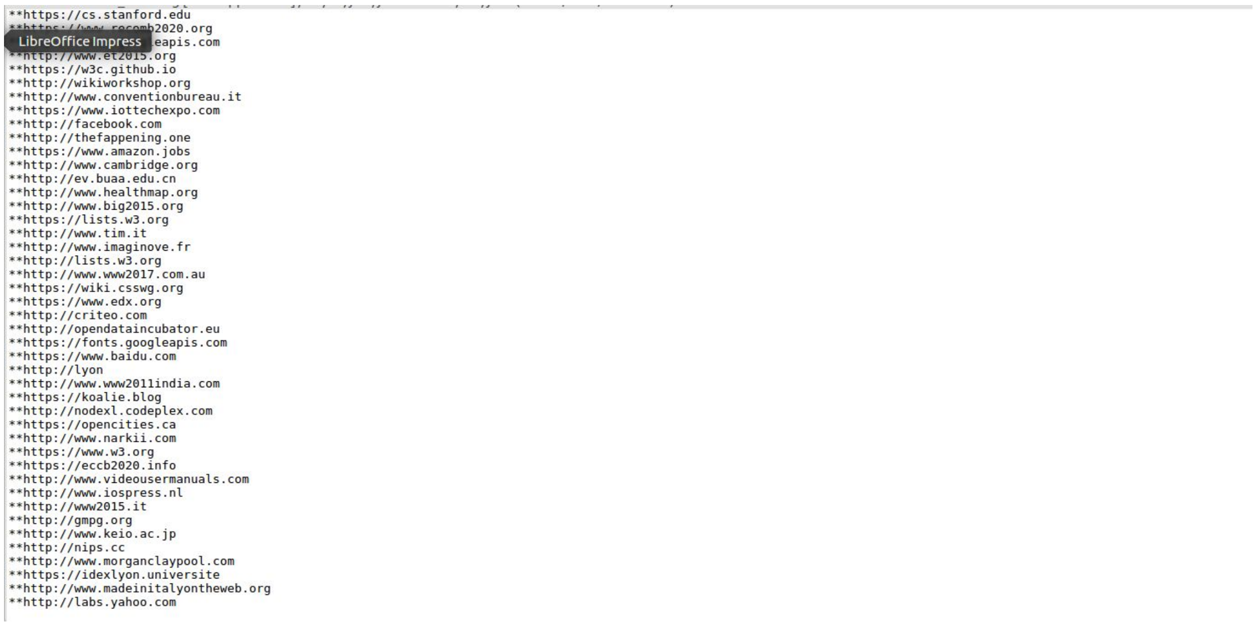
## [2] **BFS crawling**

**BFS** is a traversing algorithm where you should start traversing from a selected node (source or starting node) and traverse the graph layerwise thus exploring the neighbour nodes (nodes which are directly connected to source node).

It is more useful when we want to surf for similar topics



```
**http://www.frontiersin.org
**http://www2017.com.au
**http://www.ccc
**https://webcomelyon.fr
**http://www.lyonfrenchtech.com
**http://www.kdd.org
**http://www.w3.org
**http://google.com
**http://www.ercim.eu
**https://giw2020.ncku.edu.tw
**https://www.flickr.com
**http://www.guglielmo.biz
**https://www.iscb.org
**http://www.istella.it
**http://schema.org
**https://cs.stanford.edu
**https://www.recomb2020.org
**http://fonts.googleapis.com
**http://www.et2015.org
**https://w3c.github.io
**http://wikiworkshop.org
**http://www.conventionbureau.it
**https://www.iottechexpo.com
**http://facebook.com
**http://thefappening.one
**https://www.amazon.jobs
**http://www.cambridge.org
**http://ev.buaa.edu.cn
**http://www.healthmap.org
**http://www.big2015.org
**https://lists.w3.org
**http://www.tim.it
**http://www.imaginove.fr
**http://lists.w3.org
**http://www.www2017.com.au
**https://wiki.csswg.org
**https://www.edx.org
**http://criteo.com
**http://opendataincubator.eu
**https://fonts.googleapis.com
**https://www.baidu.com
**http://lyon
**http://www.www2011india.com
**https://koalie.blog
```

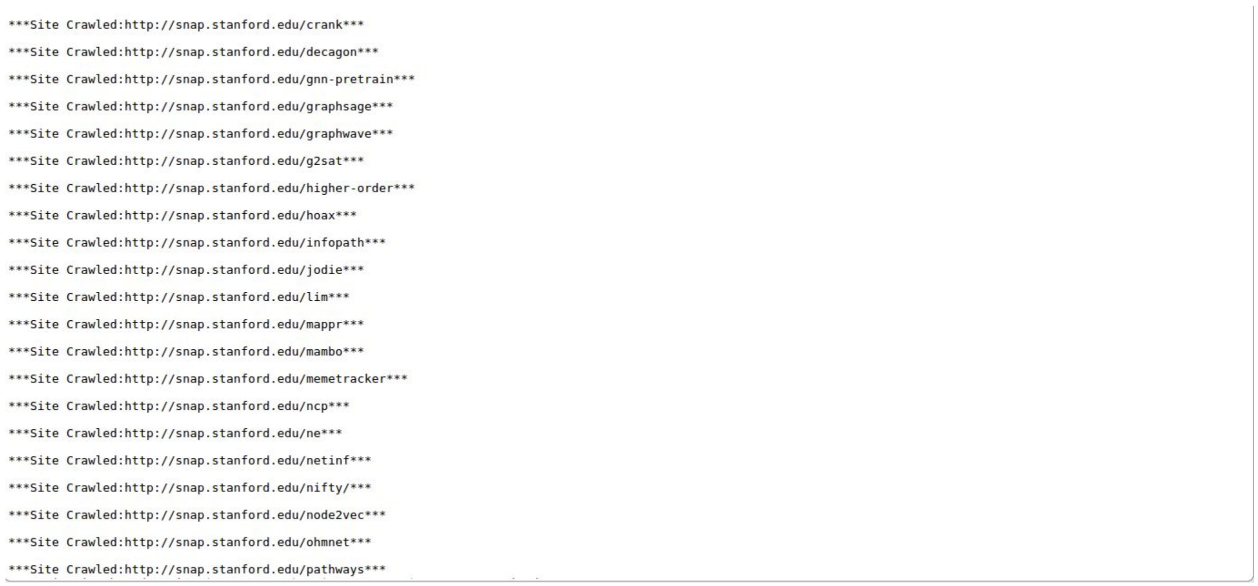


```
**https://cs.stanford.edu
**https://www.recomb2020.org
eapis.com
**http://www.et2015.org
**https://w3c.github.io
**http://wikiworkshop.org
**http://www.conventionbureau.it
**https://www.iottechexpo.com
**http://facebook.com
**http://thefappening.one
**https://www.amazon.jobs
**http://www.cambridge.org
**http://ev.buaa.edu.cn
**http://www.healthmap.org
**http://www.big2015.org
**https://lists.w3.org
**http://www.tim.it
**http://www.imaginove.fr
**http://lists.w3.org
**http://www.www2017.com.au
**https://wiki.csswg.org
**https://www.edx.org
**http://criteo.com
**http://opendataincubator.eu
**https://fonts.googleapis.com
**https://www.baidu.com
**http://lyon
**http://www.www2011india.com
**https://koalie.blog
**http://nodexl.codeplex.com
**https://opencities.ca
**http://www.narkii.com
**https://www.w3.org
**https://eccb2020.info
**http://www.videousermanuals.com
**http://www.iospress.nl
**http://www2015.it
**http://gmpg.org
**http://www.keio.ac.jp
**http://nips.cc
**http://www.morganclaypool.com
**https://idexlyon.universite
**http://www.madeinitalyontheweb.org
**http://labs.yahoo.com
```

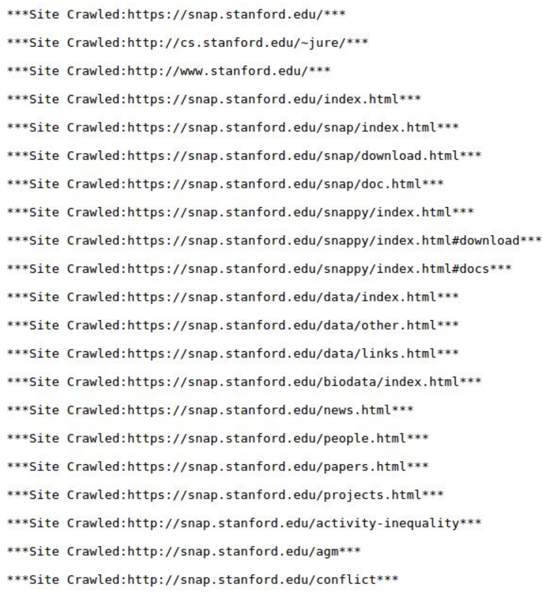
## DFS Crawling

Depth-first search is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node and explores as far as possible along each branch before backtracking.

DFS crawling is suitable to see all the reachable pages from a given website



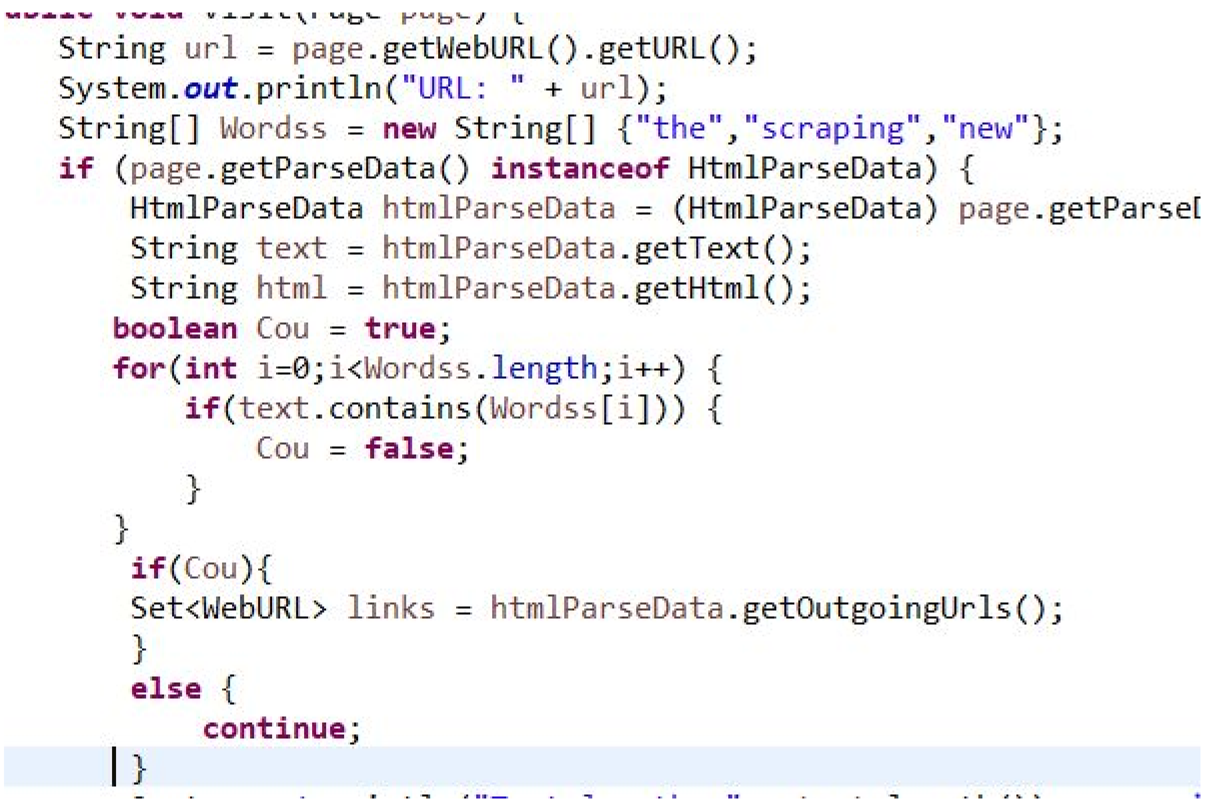
```
***Site Crawled:http://snap.stanford.edu/crank***
***Site Crawled:http://snap.stanford.edu/decagon***
***Site Crawled:http://snap.stanford.edu/gnn-pretrain***
***Site Crawled:http://snap.stanford.edu/graphsage***
***Site Crawled:http://snap.stanford.edu/graphwave***
***Site Crawled:http://snap.stanford.edu/g2sat***
***Site Crawled:http://snap.stanford.edu/higher-order***
***Site Crawled:http://snap.stanford.edu/hoax***
***Site Crawled:http://snap.stanford.edu/infopath***
***Site Crawled:http://snap.stanford.edu/jodie***
***Site Crawled:http://snap.stanford.edu/lim***
***Site Crawled:http://snap.stanford.edu/mappr***
***Site Crawled:http://snap.stanford.edu/mambo***
***Site Crawled:http://snap.stanford.edu/memetracker***
***Site Crawled:http://snap.stanford.edu/ncp***
***Site Crawled:http://snap.stanford.edu/ne***
***Site Crawled:http://snap.stanford.edu/netinf***
***Site Crawled:http://snap.stanford.edu/nifty/***
***Site Crawled:http://snap.stanford.edu/node2vec***
***Site Crawled:http://snap.stanford.edu/ohmnet***
***Site Crawled:http://snap.stanford.edu/pathways***
```



```
***Site Crawled:https://snap.stanford.edu/***
***Site Crawled:http://cs.stanford.edu/~jure/***
***Site Crawled:http://www.stanford.edu/***
***Site Crawled:https://snap.stanford.edu/index.html***
***Site Crawled:https://snap.stanford.edu/snap/index.html***
***Site Crawled:https://snap.stanford.edu/snap/download.html***
***Site Crawled:https://snap.stanford.edu/snap/doc.html***
***Site Crawled:https://snap.stanford.edu/snappy/index.html***
***Site Crawled:https://snap.stanford.edu/snappy/index.html#download***
***Site Crawled:https://snap.stanford.edu/snappy/index.html#docs***
***Site Crawled:https://snap.stanford.edu/data/index.html***
***Site Crawled:https://snap.stanford.edu/data/other.html***
***Site Crawled:https://snap.stanford.edu/data/links.html***
***Site Crawled:https://snap.stanford.edu/biodata/index.html***
***Site Crawled:https://snap.stanford.edu/news.html***
***Site Crawled:https://snap.stanford.edu/people.html***
***Site Crawled:https://snap.stanford.edu/papers.html***
***Site Crawled:https://snap.stanford.edu/projects.html***
***Site Crawled:http://snap.stanford.edu/activity-inequality***
***Site Crawled:http://snap.stanford.edu/agm***
***Site Crawled:http://snap.stanford.edu/conflict***
```

[3] **Crawling only those pages with certain text**

We will add links to frontier only if certain texts are found in the text of a page. This will limit the URL’s to be visited based on  certain keywords



```
   String url = page.getWebURL().getURL();
   System.out.println("URL: " + url);
   String[] Wordss = new String[] {"the","scraping","new"};
   if (page.getParseData() instanceof HtmlParseData) {
       HtmlParseData htmlParseData = (HtmlParseData) page.getParseI
       String text = htmlParseData.getText();
       String html = htmlParseData.getHtml();
      boolean Cou = true;
      for(int i=0;i<Wordss.length;i++) {
          if(text.contains(Wordss[i])) {
              Cou = false;
          }
      }
       if(Cou){
       Set<WebURL> links = htmlParseData.getOutgoingUrls();
       }
       else {
           continue;
      |}
```

## [4] **Data Structures**

### Frontier

A crawl frontier is one of the components that make up the architecture of a web crawler. The crawl frontier contains the logic and policies that a crawler follows when visiting websites

The policies can include such things as what pages should be visited next, the priorities for each page to be searched, and how often the page is to be visited.

The initial list of URLs contained in the crawler frontier are known as seeds. The web crawler will constantly ask the frontier what pages to visit. As the crawler visits each of those pages, it will inform the frontier with the response of each page. The crawler will also update the crawler frontier with any new hyperlinks contained in those pages it has visited. These hyperlinks are added to the frontier and will visit those new web pages based on the policies of the crawler frontier.

### Tree

A tree structure can be used to store links and then traverse based on that in bfs or dfs