EE 239 AS – Project 1

Crawling and Data Collection on the Web

Winter 2014

Project Report

**Team:**

PRAPHULL KUMAR (204271732)

LAKSHMAN KRISHNAMOORTHY (604354810)

ATHARAV (504271368)

**Topic Chosen:**

BIOMEDICAL ENGINEERING (<http://www.quora.com/Biomedical-Engineering?share=1>)

Aim of the project is to crawl Quora and gather useful data from its pages. Following is the description of the script implemented to crawl and collect data from [www.quora.com](http://www.quora.com):

**Script 1: 1quoraSubTopic.py**

This script crawls the about page of the topic assigned to get the list of all sub-topics. Then, it performs a Depth-First-Search to crawl the about pages of all the topics under the main topic. While scanning the topics graph it prints the hierarchy of the Topic name and Topic urls in topic\_names.txt and topic\_urls.txt. It also generates the list of Distinct Topic URLs (list\_topic\_url.txt) that will be used by 2quoraTopicQuestionList.py script to get the entire question URLs.

To remove the duplicate Sub-Topics in the list of topic urls python dictionaries are used.

To improve the performance of the script, the chrome browser is launched at the start of the script and then closed at the end of the script. This reduced the running time of the script as Launching and Closing of the browsers takes a lot of time.

Click on "View More" link (if number of topics is more than 10 then page displays only first 10 topics and rest are hidden) to load the hidden topics.

Selenium is used for the following:

1. Launch the browser
2. Browse the URL
3. Click on "View More" link, only if it is present
4. Read the 'page source'
5. Browse the next URL
6. Close the browser once all the topics are crawled

Beautiful Soup:

1. It finds the class name "row related\_topics\_list p1" and reads all the child topics present under that class.

**Script 2: 2quoraTopicQuestionList.py**

This script crawls the entire topic URLs to get the questions listed on each topic. It saves the Distinct List of Questions for Each topic in a file named list\_topic\_question.txt which will be used by 3quoraAllAnswers.py to read all the answers for each questions.

To improve the performance of the script, the chrome browser is launched at the start of the script and then closed at the end of the script. This reduced the running time of the script as Launching and Closing of the browsers takes a lot of time.

Selenium is used for the following:

1. Launch the browser
2. Browse the URL
3. Browse the page till end
4. Read the 'page source'
5. Browse the next URL
6. Close the browser once all the topics are crawled

Beautiful Soup:

1. It finds all the question hrefs by splitting h3 tags in the page source.

**Script 3: 3quoraAllAnswers.py**

This script crawls all the questions and read all the answers and their details.

It saves the answers details in answers.csv in the format mentioned in the requirement doc.

Also, it creates the list of Distinct User URLs (list\_users.txt) that will be used by 4quoraAllUsers.py script to get all the details about the user

To remove the duplicate in the list of topic urls python dictionaries are used.

To improve the performance of the script, the chrome browser is launched at the start of the script and then closed at the end of the script. This reduced the running time of the script as Launching and Closing of the browsers takes a lot of time.

Click on "View More" link (if number of people up voted was more than 5 then all names are not displayed) to load the hidden topics.

Selenium is used for the following:

1. Launch the browser
2. Browse the URL
3. Click on "View More" link, only if it is present
4. Browse the page till end
5. Read the 'page source'
6. Browse the next URL
7. Close the browser once all the topics are crawled

Beautiful Soup:

1. It first finds the class name "pagedlist\_item". For each answer there is a different "pagedlist\_item". All the details about the answers are present in this class.

**Script 4: 4quoraAllUsers.py**

This script crawls all the user urls and gets the details.

To improve the performance of the script, the chrome browser is launched at the start of the script and then closed at the end of the script. This reduced the running time of the script as Launching and Closing of the browsers takes a lot of time.

Selenium is used for the following:

1. Launch the browser
2. Browse the URL
3. Read the page source
4. Browse the followers’ link
5. Browse the page till end
6. Read the 'page source'
7. Browse the following link
8. Browse the page till end
9. Browse the next URL
10. Close the browser once all the topics are crawled

Beautiful Soup:

1. It reads the page source of the user's page and also its followers and following pages to get all the required details

We than developed two parsers to read both of our final files namely answers.csv and users.csv. This we implemented by parsing both the files line by line and then separated the various fields in different lists. These parsers are written in Python and are named answers\_parser.py and users\_parser.py

**Script 5: answers\_parser.py**

Parser will read one line of answers.csv and return the fields in separate variables.

**Script 6: users\_parser.py**

Parser will read one line of users.csv and return the fields in separate variables.

**Order of executing the files:**

Execute the files in the following order. When all the four files are successfully completed the parser scripts can be executed to extract the data from csv files.

1. 1quoraSubTopic.py
2. 2quoraTopicQuestionList.py
3. 3quoraAllAnswers.py
4. 4quoraAllUsers.py

**Challenges faced during the project execution:**

1. Processor speed was low. So multithreading was not possible.
2. Intermittent Internet connection. So it was broken, and getting timed out.
3. Too much data to parse. Out of Memory Error was received during execution.
4. Quora blocked our IP address due to over crawling.

**Block Diagram to represent the execution flow of the script**

