PageRank:

Mapping and Ranking the ROBLOX Domain

Ryan LaRouche

**Introduction**

Our project implements the well-known PageRank algorithm used by Google Search to rank web-pages in accordance to their “importance.” Deviating from common implementations and projects that rely on pre-existing XML sitemaps (such as those found via Stanford’s SNAP project), we developed an Objective-C program running complement to the PageRank algorithm to create our own XML sitemaps. Our project focuses on a sitemap and PageRank of [Roblox.com](https://www.roblox.com), which hosts a popular online kid-friendly building and programming game.

Our java program is based on the opensource project on GitHub originally developed by Abhijeet Nayak in 2014 with the purpose of ranking Wikipedia pages. For more information about his initial project, follow the link: https://github.com/abnayak/WikipediaPagerank.

The framework of our project is implemented using Hadoop 2.7.3. And contains a package called ‘src’ that contains all the initial java classes necessary to run out algorithm. Our main class PageeRank.java handles all the MapReduce jobs to calculate the page rank of all the pages in the dataset. The main class implements the program by running various classes designed to handle different tasks. The divided classes include:

1. XmlInputFormat.java
2. OutlinkMapperStage1.java, OutlinkMapperStage2.java
3. OutlinkMapperStage2.java, OutlinkMapperStage2.java
4. LinkCountMapper.java, LinkCountReducer.java
5. RankCalculateMapperStage1.java, RankCalculateReducerStage1.java, RankCalculateMapper.java
6. SortMapper.java, SortReducer.java

After these classes are executed, we are left with an output file containing a ranked list of all the list in order from highest to lowest page rank. We dive into each component of our algorithm and implementation in the next section, methods.

**Methods**

The project is implemented using Hadoop 2.7.3, the latest build at the time. The project contains a folder ‘src’ that holds numerous class files implemented by a main class called PageRank.java. The main class handles all the divided MapReduce jobs to calculate the page rank of all the pages in the dataset. We approached the project with a methodology of divide and conquer across numerous java classes via MapReduce. First, we read and extract data from an inputted XML file. Then, we then break down the input into individual pages and process the page contents to create (key, value) pairs formatted as (title, index #) *[Our implementation substitutes URLs in place of titles, but we use the term ‘titles’ for consistency across different implementations]*. Upon making these pairs, we remove dead links and duplicated, nullifying our need for index numbers, which we consequently remove following the creation of an out-link agency graph. We count the number of unique pages to find N (used in our PageRank algorithm) and emit the result to a temporary file to be accessed later. We then emit the unique *unranked* titles to a temporary file. Finally, we use a PageRank algorithm (discussed in the next paragraph) over the course of 8 iterations to calculate precise ranks for all of the pages. These ranks are reduced and outputted as unordered (rank, title) key value pairs. A sorting class reads the file containing these unsorted, yet ranked, titles and emits the titles in the result file PageRank.n.out ordered in descending order from highest ranking to lowest ranking title.

The core method of these jobs, and the program in general is that of PageRank. Like we noted in the introduction, the PageRank algorithm is a commonly used method for ranking a given page’s relevance. This method was first invented by Google for use in its search engine. Its ranking algorithm is determined using the following formula:

PR(p1) = (1 - d)/N + (PR(p2)/L(p2) + PR(p3)/L(p3) + ... + (PR(pN)/L(pN))

where...

d = damping factor

PR(p1) = page rank of page p1

N = total number of pages

L(p2) = total number of out-links on page p2

Via this formula our PageRank method scans through a given input (e.g. the Roblox.com domain) and created a ranked list of pages according to their relevancy rank PR(pn).

**Implementation**

1. **Sitemap Generation:**

Our project begins by implementing an Objective-C program (created specifically for our project) that inputs a .txt file containing a list of links to be web-crawled, and exports an XML sitemap in the correct format to be read by our PageRank .java program. We can also use the trial-version of a software called ‘[Web Content Extractor](http://www.newprosoft.com/web-content-extractor.htm)’ to perform the similar web-crawling task. The program will export a maximum of 150 pages as a free-trial. The Page Crawl settings must be changed to output the HTML contents of the pages and the crawl should start at the home address of your domain (e.g. <https://www.roblox.com>). All web-crawling software will work if they export an XML file with the following format:

<page> <title> [A] </title> <text> *HTML BODY* </text> </page>

<page> <title> [B] >/title> <text> *HTML BODY* </text> </page>

…

<page> <title> [N] >/title> <text> *HTML BODY* </text> </page>

We also noted that not all of the webpages we crawled had a unique title. Many had generic titles such as “Play Roblox.” In order to prevent later confusion when reading the sorted list (which prints the contents of <title></title>, we replaced the title of all webpages with the page’s URL.

Example:

<page> <title> Roblox – Home </title> <text> *HTML BODY* </text> </page>

Is now:

<page> <title> [https://www.roblox.com/home](https://www.roblox.com/home%20) </title> <text> *HTML BODY* </text> </page>

This is important in the next step of our implementation.

1. **OutlinkMapperStage1.java, OutlinkReducerStage1.java** *via* **XMLInputFormat.java:**

The first task within the java program itself is to extract the data we need from input XML file and remove the red links from the large dataset. We implement the XMLInputFormat class to extract the valid data. Using XMLInputFormat we determine where a webpage begins and ends via the contents of every <page> ... </page>.

Once we implement our desired input format, the first out-like MapReduce task, OutlinkMapperStage1, extracts the title of the page which is present in the <title> ... </title> and all the contents of the page within the <text> ... </text> to find all the out-links from current title. Remember that the content of <title> … </title> has been modified within our input to refer the page’s URL rather than the page’s title.

*(See the example in implementation stage 1)*

We use a regular expression to extract all the valid links from the text tag. In our case, there are two types of valid links, [[A]], and [[A|B]]. From both links A is extracted and all the spaces are replaced with the underscore. All the titles are emitted as a (key, value) pair in the format (title, #), where # is the index, so that for every title one pair will be created by the combiner. Instead of emitting (title, link) to reducer all the in-links of the page are emitted to reducer i.e. (link, title).

Example:

(key, value) -> (title, #) -> (<https://www.roblox.com/home>, 1)

(key, value) -> (title, #) -> (<https://blog.roblox.com/>, 2)

…

(key, value) -> (title, #) -> (URL, N)

Next, the reducer (OutlinkReducerStage1) puts all the contents of the bucket in the Set to keep only the unique links. As we emitted the index (#) in every title pair, if # is not present in the bucket then it is not a valid page. Now when the set has index (#) sent as a (title, #) which will go to its specific pair and all the other links are sent as single out-links.

1. **OutlinkMapperStage2.java, OutlinkReducerStage2.java:**

The second out-link MapReduce stage implements the job of generating an out-link adjacency graph. Mapper (OutlinkMapperStage2) splits all the value entries of the (key, value) pair and emits it to the reducer. Since we removed duplicate pages when reducing the (key, value) pairs in the last stage, we no longer need the indices (#’s) of the links. Thus, we remove the index value from all the links.

All the values are then combined into StringBuilder and emitted as string output to generate the adjacency graph. This output will be stored in PageRank.outlink.out (which is an unranked version of our final ranked output file).

1. **LinkCountMapper.java, LinkCountReducer.java:**

This part contains the MapReduce job which computes the total number of pages denoted as N in the equation (Displayed in *Table 2* of our results section). Upon receiving the output from OutlinkReducerStage2 stored in PageRank.outlink.out, there are two wats to calculate N:

1. Look for all the page and title tags in the large dataset

If we were to do this, we would have to sort through the initial input file and count the number of occurrences of <page> … </page>. Unfortunately, thing becomes increasingly taxing and timely of a method as the number of pages being ranked increases because of the additional content within each page adding to the file size and content that would have to be sorted.

***This method is not optimized.***

1. Instead of counting all the page tags in the large dataset count number of lines in the out-link graph which contains all the unique titles:

By doing this, we would only have to count the number of unique titles. Since there are no duplicates, we do not have to sort through any unnecessary information. As the number of pages increase, the increase in size is minimal per the size of the additional pages’ titles. LinkCountReducer will then emit N=Number of pages in the dataset and write it to the PageRank.n.out

***This method is optimized.***

We chose to implement the second method since it is optimized and therefore more efficient and faster, particularly as the data set increases in size.

1. **RankCalculateMapperStage1.java, RankCalculateReducerStage1.java, RankCalculateMapper.java:**

This MapReduce job is required to calculate the PageRank for 8 iterations. To initialize the page ranks of all the pages, we introduce initial rank for all the pages to 1/N where N equals the number of unique pages (previously stored in PageRank.n.out). Upon initialization, the Reducer outputs the initial rank values to tmp/PageRank.iter0.out in the format <title> <initialized rank> <out-links>. We then proceed to implement page rank iterations for more precise rankings.

During each of the 8 iterations we implement the following ranking process:

1. Split the (key, value) pairs into title, rank, and out-links.
2. Count all the out-links for each page title and calculate the rankVote of all each page’s out-links. We calculate rankVote using the equation:

rankVote = rank / outlinkCount

1. Emit this vote to all the out-links of that page.
2. In the Reducer, add all the rank votes from all the links corresponding to each page to count the page rank for each page using.

During our ranking process, our algorithm uses the following formula:

PR(p1) = (1 - d)/N + (PR(p2)/L(p2) + PR(p3)/L(p3) + ... + (PR(pN)/L(pN))

where...

d = damping factor

PR(p1) = page rank of page p1

N = total number of pages

L(p2) = total number of out-links on page p2

Once we are done with the page rank calculation, emit the newly calculated page rank values in the format <title> <new rank> <out-links> to PageRank.n.out. Note that while the pages have ranked, the output does not emit in a sorted order.

1. **SortMapper.java, SortReducer.java:**

After 8 iterations sorting is performed on iter1 and iter8 in this MapReduce job. Mapper of this part emits page rank as (key, value) pair (rank, page) in the sorted order. To emit the page ranks in the descending order we have overridden the compare() method in KeyComparator class.

The Reducer (SortReducer) now receives all the page ranks in descending order and compares if the page rank is greater than 5/N. If the rank is greater than the 5/N value then reducer emits the page rank as (page title, rank). If not, the page is ignored.

As the MapReduce splits all the output in parts we merge the corresponding outputs in single files. We have created new file using FSDataOutputStream to store all the data from output parts to single file. By opening a handle for every file some chunk of bytes are transferred to output file using read() method of FSDataInputStream class. **All the output files PageRank.outlink, PageRank.n.out, PageRank.iter1.out, PageRank.iter8.out are stored in the results directory.**

**Results**

Our program successfully ranks the inputted pages from the domain Roblox.com. The most important output is a file that contains a ranked list of web pages from our crawled domain. The list is ranked in descending order with the highest ranked pages at the top and lowest ranked at the bottom.

The final output of our program occurs within two files. We display the following data in *Table 1* and Table 2:

1. [PageRank.n.out]: A single line written as ‘N = #’ where # equals the number of pages being ranked. We printed the contents of this file for our results in Table 1 below.
2. [PageRank.outlink.out] :The results of the algorithm as a ranked list of 151 pages on the Roblox.com domain. Printed in Table 2 below.

*[End of page, see tables on following pages]*

**TABLE 1:**

|  |
| --- |
| N=151 |

***Table 1*:** *[PageRank.n.out]* N equals 151 because our input contains 151 pages to be ranked by the algorithm.

**TABLE 2:**

|  |
| --- |
| http://wiki.roblox.com/index.php?title=Special:RobloxLandingPage |
| https://blog.roblox.com/ |
| https://blog.roblox.com/2017/03/keeping-community-safe/ |
| https://corp.roblox.com/ |
| https://corp.roblox.com/advertise/ |
| https://corp.roblox.com/careers/ |
| https://corp.roblox.com/contact/ |
| https://corp.roblox.com/news/ |
| https://corp.roblox.com/parents/ |
| https://corp.roblox.com/people/ |
| https://corp.roblox.com/press-kit/ |
| https://en.help.roblox.com/hc/en-us |
| https://en.help.roblox.com/hc/en-us/articles/203312870-General-Roblox-Issues |
| https://en.help.roblox.com/hc/en-us/articles/203312910-How-Do-I-Reinstall-the-Roblox-Software- |
| https://en.help.roblox.com/hc/en-us/articles/203312980-How-to-Uninstall-ROBLOX |
| https://en.help.roblox.com/hc/en-us/articles/203313020 |
| https://en.help.roblox.com/hc/en-us/articles/203313120-Safety-Features-Chat-Privacy-Filtering |
| https://en.help.roblox.com/hc/en-us/articles/203313200-How-to-Get-Robux |
| https://en.help.roblox.com/hc/en-us/articles/203313390-My-Account-was-Hacked-What-do-I-do- |
| https://en.help.roblox.com/hc/en-us/articles/203313410-Roblox-Rules-of-Conduct |
| https://en.help.roblox.com/hc/en-us/articles/203313840-How-to-Make-Ads |
| https://en.help.roblox.com/hc/en-us/articles/203314070-Audio-Files |
| https://en.help.roblox.com/hc/en-us/articles/203314270-Games-and-Gameplay-Reporting-Abuse-In-Game |
| https://en.help.roblox.com/hc/en-us/articles/204473560 |
| https://en.help.roblox.com/hc/en-us/articles/204473560-How-to-Install-and-Play-Roblox-Using-Google-Chrome-Microsoft-Edge-and-Mozilla-Firefox |
| https://en.help.roblox.com/hc/en-us/articles/206455923-Sponsored-Games |
| https://en.help.roblox.com/hc/en-us/articles/208448546 |
| https://en.help.roblox.com/hc/en-us/categories/200213820-Billing |
| https://en.help.roblox.com/hc/en-us/categories/200213830-Community |
| https://en.help.roblox.com/hc/en-us/categories/200217934-Account-and-Security |
| https://en.help.roblox.com/hc/en-us/categories/200217944 |
| https://en.help.roblox.com/hc/en-us/categories/200217944-Frequently-Asked-Questions |
| https://en.help.roblox.com/hc/en-us/categories/200217954-Gameplay-and-Building |
| https://en.help.roblox.com/hc/en-us/categories/200217964-Mobile-Xbox-and-Virtual-Reality |
| https://en.help.roblox.com/hc/en-us/sections/200866010-Roblox-Rules-and-Guidelines |
| https://en.help.roblox.com/hc/en-us/sections/200866090 |
| https://en.help.roblox.com/hc/en-us/sections/200866090-Common-Technical-Processes-and-Questions |
| https://itunes.apple.com/us/app/roblox-mobile/id431946152?mt=8 |
| https://www.roblox.com/ |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2FNewLogin%3FReturnUrl%3D%252fdevelop%252flibrary%253fCatalogContext%253d2%2526Category%253d6%2526Keyword%253d |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F%3FCatalogContext%3D1%26Keyword%3D |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F709229848%2FROBLOX-Egg-Launcher-2017 |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fdevelop |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fgames |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fgames%2F%3FKeyword%3D |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Flogin%2FforgotPasswordOrUsername%2F |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fnewlogin%3FreturnUrl%3Dhttps%253A%252F%252Fwww.roblox.com%252Fgames |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fnewlogin%3FreturnUrl%3Dhttps%253A%252F%252Fwww.roblox.com%252Flogin%252FforgotPasswordOrUsername%252F |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fpremium%2Fmembership%3Fctx%3Dpreroll |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fsearch%2Fusers%3Fkeyword%3D |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dcatalog |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dnav |
| https://www.roblox.com/?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dnav%2F |
| https://www.roblox.com/Badges.aspx |
| https://www.roblox.com/Info/privacy |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d10%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d19%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d21%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d42%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d45%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fUpgrades%2fPaymentMethods%3fap%3d46%26page%3dgrid |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fabusereport%2fUserProfile%3fid%3d1%26redirectUrl%3dhttps%253A%252F%252Fwww.roblox.com%252Fusers%252F1%252Fprofile |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fdevelop%2flibrary%3fCatalogContext%3d2%26Category%3d6%26Keyword%3d |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fgamecard%2fDefault.aspx |
| https://www.roblox.com/NewLogin?ReturnUrl=%2fmy%2faccount |
| https://www.roblox.com/NewLogin?ReturnUrl=https%3a%2f%2fwww.roblox.com%2frixtypin%2fDefault.aspx |
| https://www.roblox.com/account/signupredir |
| https://www.roblox.com/catalog |
| https://www.roblox.com/catalog/ |
| https://www.roblox.com/catalog/100427922/Starry-Rune-Antlers |
| https://www.roblox.com/catalog/106690045/Classic-Swordpack-Throwback |
| https://www.roblox.com/catalog/108147364/Blue-Spring-Topper |
| https://www.roblox.com/catalog/121389389/Business-Cat |
| https://www.roblox.com/catalog/139574419/Frost-Guard-General |
| https://www.roblox.com/catalog/139610216/Korblox-Deathspeaker |
| https://www.roblox.com/catalog/16630147/Beautiful-Hair-for-Beautiful-People |
| https://www.roblox.com/catalog/212641536/Golden-Super-Fly-Boombox |
| https://www.roblox.com/catalog/215719598/Black-Wings |
| https://www.roblox.com/catalog/358252359/Jeans-with-Red-Kicks |
| https://www.roblox.com/catalog/368243527/Carrot-King |
| https://www.roblox.com/catalog/375817661/Pink-Spring-Bonnet |
| https://www.roblox.com/catalog/387256603/Silver-Punk-Face |
| https://www.roblox.com/catalog/583146148/Star-Lord-s-Headphones |
| https://www.roblox.com/catalog/600979849/Star-Lord-s-Facemask |
| https://www.roblox.com/catalog/607702162/Roblox-Baseball-Cap |
| https://www.roblox.com/catalog/607785314/ROBLOX-Jacket |
| https://www.roblox.com/catalog/619547874/Levitation-Animation-Pack |
| https://www.roblox.com/catalog/636786424/Yondu-s-Fin |
| https://www.roblox.com/catalog/658839027/Ninja-Animation-Package |
| https://www.roblox.com/catalog/709229469/Bunny-Ears-2017 |
| https://www.roblox.com/catalog/709229848/ROBLOX-Egg-Launcher-2017 |
| https://www.roblox.com/catalog/727320877/Long-Pastel-Hair |
| https://www.roblox.com/catalog/738679517/BunnyTail |
| https://www.roblox.com/catalog/738925737/Horde-of-Attack-Chirps |
| https://www.roblox.com/catalog/77518833/Korblox-Mage |
| https://www.roblox.com/catalog/86500008/Man-Torso |
| https://www.roblox.com/catalog/86500036/Man-Right-Arm |
| https://www.roblox.com/catalog/86500054/Man-Left-Arm |
| https://www.roblox.com/catalog/86500064/Man-Left-Leg |
| https://www.roblox.com/catalog/86500078/Man-Right-Leg |
| https://www.roblox.com/catalog/?CatalogContext=1 |
| https://www.roblox.com/catalog/?CatalogContext=1&amp;Keyword= |
| https://www.roblox.com/develop |
| https://www.roblox.com/developer-exchange/help |
| https://www.roblox.com/developer-exchange/tos |
| https://www.roblox.com/gamecards |
| https://www.roblox.com/games |
| https://www.roblox.com/games/ |
| https://www.roblox.com/games/41324860/Welcome-to-ROBLOX-Building |
| https://www.roblox.com/games/65033/Classic-Happy-Home-in-Robloxia |
| https://www.roblox.com/games/?Keyword= |
| https://www.roblox.com/games/?SortFilter=default&amp;TimeFilter=0&amp;GenreFilter=1 |
| https://www.roblox.com/groups/group.aspx?gid=127081 |
| https://www.roblox.com/groups/group.aspx?gid=7 |
| https://www.roblox.com/info/terms-of-service |
| https://www.roblox.com/login/forgotPasswordOrUsername/ |
| https://www.roblox.com/newlogin?ReturnUrl=https%3A%2F%2Fwww.roblox.com%2Fdevelop%3Fclose%3D1 |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2FNewLogin%3FReturnUrl%3D%252fdevelop%252flibrary%253fCatalogContext%253d2%2526Category%253d6%2526Keyword%253d |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F%3FCatalogContext%3D1%26Keyword%3D |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fcatalog%2F709229848%2FROBLOX-Egg-Launcher-2017 |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fdevelop |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fgames |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fgames%2F%3FKeyword%3D |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Flogin%2FforgotPasswordOrUsername%2F |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fnewlogin%3FreturnUrl%3Dhttps%253A%252F%252Fwww.roblox.com%252Fgames |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fnewlogin%3FreturnUrl%3Dhttps%253A%252F%252Fwww.roblox.com%252Flogin%252FforgotPasswordOrUsername%252F |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fpremium%2Fmembership%3Fctx%3Dpreroll |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fsearch%2Fusers%3Fkeyword%3D |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dcatalog |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dnav |
| https://www.roblox.com/newlogin?returnUrl=https%3A%2F%2Fwww.roblox.com%2Fupgrades%2Frobux%3Fctx%3Dnav%2F |
| https://www.roblox.com/premium/membership?ap=10&amp;page=grid |
| https://www.roblox.com/premium/membership?ap=19&amp;page=grid |
| https://www.roblox.com/premium/membership?ap=21&amp;page=grid |
| https://www.roblox.com/premium/membership?ap=42&amp;page=grid |
| https://www.roblox.com/premium/membership?ap=45&amp;page=grid |
| https://www.roblox.com/premium/membership?ap=46&amp;page=grid |
| https://www.roblox.com/premium/membership?cashout=obc |
| https://www.roblox.com/premium/membership?ctx=preroll |
| https://www.roblox.com/search/users?keyword= |
| https://www.roblox.com/upgrades/robux?ctx=catalog |
| https://www.roblox.com/upgrades/robux?ctx=nav |
| https://www.roblox.com/upgrades/robux?ctx=nav/ |
| https://www.roblox.com/upgrades/robux?ctx=upgrade |
| https://www.roblox.com/users/1/friends |
| https://www.roblox.com/users/1/profile |
| https://www.roblox.com/users/1/profile/ |

***Table 2*:** *[PageRank.outlink.out]* The output of the PageRank algorithm as a ranked list of 151 pages within the roblox.com domain.

**Conclusions**

Our program successfully implements a modified version of Abhijeet Nayak’s open source Wikipedia PageRank algorithm, which we tailored to the Roblox.com domain. Implementation of the program results in the creation and output of a list ranked in descending order with the highest ranked pages at the top and lowest ranked at the bottom. The most critical contribution made by our project is the creation of a program that crawls the web and formats XML sitemaps specifically to match the required input format of our PageRank algorithm. Although we make note of an alternative program, ‘[Web Content Extractor](http://www.newprosoft.com/web-content-extractor.htm)’, our program is much less complex.

The most important component of our results stems from how the results are used and analyzed. Much of the time, the results are used by Google to calculate where pages rank in their search algorithm, or by sites and domains themselves to determine where their page(s) rank and if they need to improve the rank to appear higher on Google’s search pages.

For instance, from our results, we can extract numerous bits of information. The highest-ranking page is the domain’s wiki landing page (<http://wiki.roblox.com/index.php?title=Special:RobloxLandingPage>) and the lowest the first user’s profile page (<https://www.roblox.com/users/1/profile/>). While these results are of little use to us, we could use this information to analyze what pages underperform in regard to how they appear in Google searches. Roblox likely wants their Wiki landing page to rank the highest because it is the hub of the help section of their domain, which most users want to find when searching for information about the company.

Finally, through the implementation of an open source algorithm, we developed a more complete understanding of the PageRank algorithm, how Google’s search engine operates, the Hadoop filesystem, and MapReduce.

Works Cited

Nayak, Abhijeet, Wikipedia PageRank, (2014), GitHub Repository,

https://github.com/abnayak/WikipediaPagerank

Rogers, Ian. "Pagerank Explained Correctly with Examples." *Princeton University*. The

Trustees of Princeton University, n.d. Web. 18 Apr. 2017. <http://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm>.