Maciej Medyk – COT6777 – Web Mining

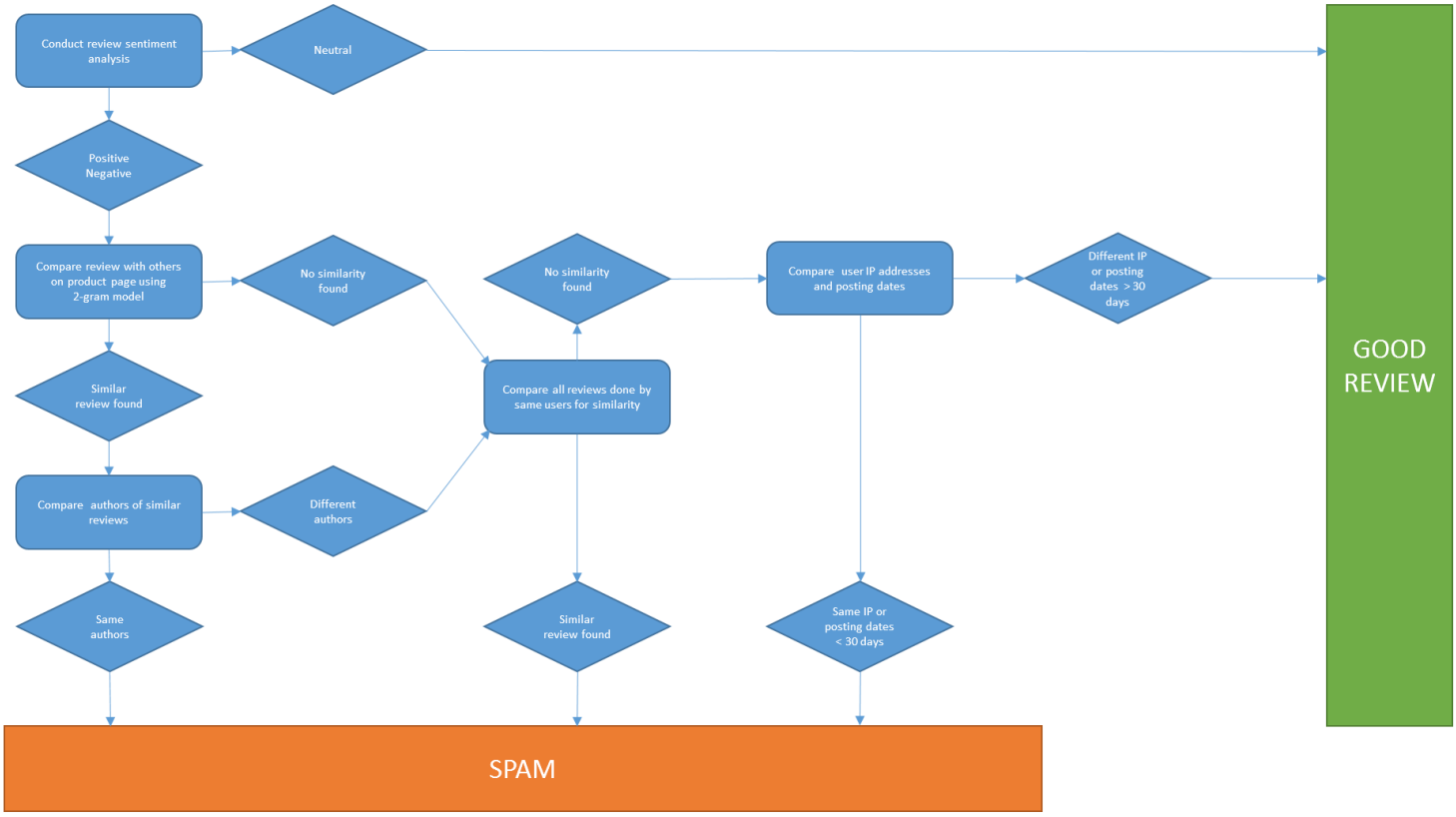
Question 1 – [6.00pt]

What is sentiment analysis for online review? What are the possible applications of sentiment analysis [0.75pt].

Sentiment analysis is a process that evaluates the how positive, neutral, or negative is the review. The process pulls the opinion text and checks it for special words that indicate how positive or negative the comment is. Each of the special words have a weight attached that does indicate the significance and strength of the word. For example word 'loved' (+0.85) is positively stronger than 'liked' (+0.45) and would have higher positive weight. On the other hand word 'appalled' (-0.85) would be negatively stronger than 'disliked' (-0.45) and would have higher negative weight. Once all words in text are evaluated the polarity ratio is calculated to figure out if comment overall was positive or negative. The possible application of sentiment analysis can be in evaluating public opinion on services and products. For example if movie is about to release you can find out if people are excited about movie and can have expectations of box office numbers. Another example would be to run sentiment analysis on comments in amazon page for products reviews and find out if people expressed like or dislike in their comments and if those correlate to other metrics like number of stars selected.

Please outline major steps of building an automated online review spam detection system for a website [0.75pt].

When building spam detection system for a website we need to take into the sentiment of the message, similarity of the message to others on product page, similarity of the messages made by same author, posting frequency of similar messages, differences in IP addresses between different reviewers. Initially the spam detector would conduct sentiment analysis. Based on the results it would compare the review with others on the page using 2-gram method to find similarity in posting. If similarity in posting would be found we would compare both users when it comes to their IP addresses, their timing of posts, the similarity of all messages authored by both users, and based on all those factors would choose if user is considered as spam generator. Flowchart below illustrates basic steps in the procedure that automated online review spam detector would follow in order to detect and mark messages as spam or not.



What is a bag-of-word representation model? What is N-gram for representing a document? Explain the advantage and disadvantage of Bag-of-Word model vs. the N-gram model [0.75pt].

Bag-of-words model is where each word in the document is tokenized and counted separately from context. For example sentence 'walks like a duck and quacks like a duck' would be represented as {walks: 1, like: 2, a: 2, duck: 2, and: 1, quacks: 1}. Bag-of-words model does not take context into account and sentence like 'quacks like a duck and walks like a duck' would be represented exactly the same way. On the other hand n-gram model takes into account words nearby. Common model is 2-gram model where same sentence 'walks like a duck and quacks like a duck' would be represented as {walks like: 1, like a: 2, a duck: 2, duck and: 1, and quacks: 1, quacks like: 1} and this method takes context into account. Same sentence written other way would not yield the same representation. Bag-of-words is very common model as it is fast and simple to implement as many programming libraries include tokenizers, but it ignores context and ignores positioning of words. N-gram model is more precise; however, it requires far more computation and storage as N-complexity grows. Advantage of N-gram model is that it takes context and position of words into consideration.

What is a Boolean retrieval model? What is a vector space model? Please explain their difference [0.75pt].

Boolean retrieval model focuses on collecting presence of the word inside the document. During the retrieval a term is either marked as present or absent from the document. There is no accounting of frequency of that term. The queries for this model are written using connecting terms like AND OR and NOT. For example Boolean query for 'recipe AND duck AND NOT rabbit'. Queries do not accept partial matches. Boolean model does not have any way of ranking matched documents by relevance. Vector space model on the other hand looks at documents and queries as vectors in multi-dimensional space. Each dimension represents TF-IDF for one term and documents are ranked by closeness to the query which is determined by similarity score calculation. In this model frequency of words is being tracked and taken to consideration and queries accept partial matches.

What is TF-IDF? What is the advantage of using TF-IDF for information retrieval [0.75pt].

TF-IDF is term frequency combined with inverse document frequency model that is intended to reflect how important word is in collection of documents. TF-IDF counts occurrence of word in each document and modifies it so that words that occur often do not over-flood the more significant words. To diminish the effects of words that are used often the model uses logarithmic function to calculate the word count. IDF portion of the model also is a logarithmic function that takes into account number of documents the words occurs into consideration. The advantage of TF-IDF is that the score does increase with number of occurrences inside the document but the model is truly designed to increase with rarity of the term in the document. The concept and the advantage behind TF-IDF is that rare terms are often more informative than common filler word terms. TF-IDF is calculated by following formula.

$$\mathrm{tf.idf}_{t,d} = (1 + \log_{10} \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

Please explain the PageRank score formula. How does PageRank score assess the importance of a webpage [0.75pt].

PageRank score is simply a score of how important and how well connected the page is. The importance of the page is derived by number of pages that point to it otherwise known as in-links and the probability that the random surfer visits that page. To achieve this score we need to check all the direct neighbors of the current page that have and count the amount of in-links that lead to the current page and compare that to number of all out-links from those neighboring pages. PageRank is calculated by following formula.

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v}$$

In this formula 'c' is normalization factor followed by summation symbol of all in-link vertices. R(v) page rank of the neighboring page and it is divided by number of all out-links from that neighboring page.

Please explain the disadvantage of using TF-IDF and PageRank, alone, for information retrieval. Please propose one solution to combine TF-IDF and PageRank to retrieve information online [0.75pt].

It seems like TF-IDF and PageRank were built for different purposes and have different focus. TF-IDF looks at context of the page and judges the importance of words that are infrequent and carry a lot of meaning. It is a great tool to extract message from the pages and to query pages based on keywords, but it assumes all pages are trusted and relevant in some sort of way. TF-IDF is a great tool that tries to figure out if content matches the scope of search. On the other hand PageRank focuses on the page social status among other pages and determines importance of the page based the connections leading to the page and does not analyze page for its content. This is also important way of judging a document as page that is not relevant is not highly connected and not widely used as reference in other pages. Combination of both of those systems would be very useful for search engine as it would rank pages based on significance but also context. PageRank by itself would return pages that are highly connected but would not have any relevance to the topic searched, while TF-IDF would return relevant topic but often from unverified sites. Combination of both PageRank and TF-IDF would return a list pages with matching context that was verified and endorsed by other pages. TF-IDF portion would ensure the page is relevant as context and PageRank portion would ensure that the page is relevant as page is often used as reference and is widely agreed verified source of information.

What is inverted index? When using inverted index for retrieval, what is the difference between document-at-a-time vs. query-at-a-time models [0.75pt].

Normally in something like array we are looking for value by known index. So if we know index of 0 in array[0] will contain value that we want to extract we will use that index to obtain the value. Inversed index is exact opposite of that where we search for index position of the value that we already know. Many times when scanning the string of text we may receive multiple index values if text contains many of same words. So in document D1 which would contain sentence like 'walks like a duck and quacks like a duck' when parsing this sentence for a word duck we would obtain result {D1, 2, {4, 9}} meaning document D1 has 2 words at position 4 and 9. The query-at-a-time model uses this inverted index in order to find similar documents. It goes through the search terms accumulating the documents which contain those terms. The advantage of query-at-a-time model is early termination heuristics; however, the process has large memory footprint. The document-at-a-time model does the opposite and instead searches through each document one by one evaluating it against the search terms. This method has the huge disadvantage of having to actually read every document for every query and it doesn't scale for large libraries. The advantage of document-at-a-time model is that it has small memory footprint.

Question 2 – [3.00pt]

Given following five documents, and a query Q, please use TF-IDF to calculate the matching score between query Q and each document (Please only consider following terms in your calculation ∑ = {dish, duck, rabbit, Beijing, recipe, roast}.). Please show TF table and IDF table [2.00pt] and show the TF-IDF scores between query Q and each document [1.00pt]

Q: "Beijing duck recipe"

D1: "walks like a duck and quacks like a duck, must be a duck"

D2: "Beijing Duck is prized for the thin, crispy duck skin with authentic versions of the dish serving"

D3: "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."

D4: "I found this great recipe for rabbit Braised in Wine on cookingforengineers.com"

D5: "Last week Li has shown you how to make the Sechuan duck. Today we'll show you a recipe for making Chinese dumplings, a popular dish that I had a chance to try last summer in Beijing."

First we need to count number of occurrences of each word in each document. For example 'DUCK' appears 3 times in document 1.

Terms	D1	D2	D3	D4	D5
dish	0	**1**	0	0	**1**
duck	**3**	**2**	**2**	0	**1**
rabbit	0	0	**1**	**1**	0
Beijing	0	**1**	0	0	**1**
recipe	0	0	**1**	**1**	**1**
roast	0	0	0	0	0

To calculate TF we need to calculate it using formula TF = 1 + LOG10(value). For example 'DUCK' in D1 appears 3 times therefore TF value of DUCK(D1) = 1 + LOG10(3) = 1.47712. Afterwards we have to calculate IDF value which for 'DUCK' would be LOG10(5/4) = 0.09691 as word duck appears 4 times in 5 documents. Then we multiply TF with IDF to get TF-IDF table so in our example DUCK(D1) = 1.47712 x 0.09691 = 0.14315

TF					
Terms	D1	D2	D3	D4	D5
dish	0	1.0000	0	0	1.0000
duck	1.47712	1.30103	1.30103	0	1.0000
rabbit	0	0	1.0000	1.0000	0
Beijing	0	1.0000	0	0	1.0000
recipe	0	0	1.0000	1.0000	1.0000
roast	0	0	0	0	0

IDF
0.39794
0.09691
0.39794
0.39794
0.22185
0.00000

TF-IDF					
Terms	D1	D2	D3	D4	D5
dish	0	0.39794	0	0	0.39794
duck	0.14315	0.12608	0.12608	0	0
rabbit	0	0	0.39794	0.39794	0
Beijing	0	0.39794	0	0	0.39794
recipe	0	0	0.22185	0.22185	0.22185
roast	0	0	0	0	0

Here we will start evaluating query compared to TF-IDF by taking cosine similarity where we compare TD-IDF with Q-IDF and as a final results we see that closest similarity to Q is D5. We get those results by using formula =SUMPRODUCT(Q-IDF,D#)/(SQRT(SUMPRODUCT(Q-IDF,Q-IDF))*SQRT(SUMPRODUCT(D#,D#)))

Terms	Q	Q-IDF
dish	0	0
duck	1	0.09691
rabbit	0	0
Beijing	1	0.39794
recipe	1	0.221849
roast	0	0

Similarity using Cosine Similarity					
Terms	D1	D2	D3	D4	D5
similarity	**0.20805**	**0.63497**	**0.27901**	**0.23192**	**0.76031**

Similarity using Score Formula					
Terms	D1	D2	D3	D4	D5
similarity	**0.14315**	**0.52402**	**0.34793**	**0.22185**	**0.71670**

Question 3 – [4.00pt]

Please show the adjacency matrix and normalize the adjacency matrix [1.00pt].

Matrix A							
	A	B	C	D	E	F	G
A	0	1	1	1	1	1	1
B	1	0	0	1	1	0	0
C	1	0	0	0	0	1	1
D	1	1	0	0	0	0	0
E	1	1	0	0	0	0	0
F	1	0	1	0	0	0	0
G	1	0	1	0	0	0	0

Normalized Matrix A							
	A	B	C	D	E	F	G
A	0	0.333	0.333	0.500	0.500	0.500	0.500
B	0.167	0	0	0.500	0.500	0	0
C	0.167	0	0	0	0	0.500	0.500
D	0.167	0.333	0	0	0	0	0
E	0.167	0.333	0	0	0	0	0
F	0.167	0	0.333	0	0	0	0
G	0.167	0	0.333	0	0	0	0

Please use "Simple Iteration" to calculate the importance score for each website, after the first and the second iterations [The initial importance scores can be set as 1/n = 0.15]. Please also report the difference of the Page Rank score (using L1 norm) after the first and the second iterations, respectively. [2.00pts].

	A	B	C	D	E	F	G	X0	X1	X2	X3	X4	X5
A	0	0.333	0.333	0.500	0.500	0.500	0.500	0.1500	0.4000	0.2667	0.3444	0.2963	0.3272
B	0.167	0	0	0.500	0.500	0	0	0.1500	0.1750	0.1417	0.1694	0.1491	0.1633
C	0.167	0	0	0	0	0.500	0.500	0.1500	0.1750	0.1417	0.1694	0.1491	0.1633
D	0.167	0.333	0	0	0	0	0	0.1500	0.0750	0.1250	0.0917	0.1139	0.0991
E	0.167	0.333	0	0	0	0	0	0.1500	0.0750	0.1250	0.0917	0.1139	0.0991
F	0.167	0	0.333	0	0	0	0	0.1500	0.0750	0.1250	0.0917	0.1139	0.0991
G	0.167	0	0.333	0	0	0	0	0.1500	0.0750	0.1250	0.0917	0.1139	0.0991
							Difference in Page Rank	1.0500	0.6000	0.4000	0.2667	0.1778	0.1185

Please also use eigenvector based PageRank algorithm to calculate the PageRank scores for each web page [1.00pt].

Input matrix:

```
0.000 0.330 0.333 0.500 0.500 0.500 0.500
0.167 0.000 0.000 0.500 0.500 0.000 0.000
0.167 0.000 0.000 0.000 0.000 0.500 0.500
0.167 0.333 0.000 0.000 0.000 0.000 0.000
0.167 0.333 0.000 0.000 0.000 0.000 0.000
0.167 0.000 0.333 0.000 0.000 0.000 0.000
0.167 0.000 0.333 0.000 0.000 0.000 0.000
```

Eigenvalues Eigenvectors:

Eigenvalues:

```
( 1.000, 0.000i)
(-0.333, 0.000i)
(-0.667, 0.000i)
(-0.577, 0.000i)
( 0.577, 0.000i)
( 0.000, 0.000i)
( 0.000, 0.000i)
```

Eigenvectors:

```
(-0.717, 0.000i) (-0.816, 0.000i) (-0.634, 0.000i) ( 0.000, 0.000i) ( 0.000, 0.000i) ( 0.000, 0.000i) ( 0.000, 0.000i)
(-0.359, 0.000i) ( 0.409, 0.000i) (-0.315, 0.000i) ( 0.544, 0.000i) ( 0.549, 0.000i) ( 0.000, 0.000i) ( 0.000, 0.000i)
(-0.359, 0.000i) ( 0.409, 0.000i) (-0.315, 0.000i) (-0.551, 0.000i) (-0.547, 0.000i) ( 0.000, 0.000i) ( 0.000, 0.000i)
(-0.239, 0.000i) ( 0.000, 0.000i) ( 0.316, 0.000i) (-0.314, 0.000i) ( 0.317, 0.000i) (-0.707, 0.000i) ( 0.000, 0.000i)
(-0.239, 0.000i) ( 0.000, 0.000i) ( 0.316, 0.000i) (-0.314, 0.000i) ( 0.317, 0.000i) ( 0.707, 0.000i) ( 0.000, 0.000i)
(-0.239, 0.000i) ( 0.000, 0.000i) ( 0.316, 0.000i) ( 0.318, 0.000i) (-0.316, 0.000i) ( 0.000, 0.000i) (-0.707, 0.000i)
(-0.239, 0.000i) ( 0.000, 0.000i) ( 0.316, 0.000i) ( 0.318, 0.000i) (-0.316, 0.000i) ( 0.000, 0.000i) ( 0.707, 0.000i)
```

	A	B	C	D	E	F	G	X0	X1	X2	X3	X4	X5
A	0	0.333	0.333	0.500	0.500	0.500	0.500	-0.7170	-0.7173	-0.7173	-0.7173	-0.7173	-0.7173
B	0.167	0	0	0.500	0.500	0	0	-0.3590	-0.3585	-0.3587	-0.3586	-0.3587	-0.3586
C	0.167	0	0	0	0	0.500	0.500	-0.3590	-0.3585	-0.3587	-0.3586	-0.3587	-0.3586
D	0.167	0.333	0	0	0	0	0	-0.2390	-0.2392	-0.2391	-0.2391	-0.2391	-0.2391
E	0.167	0.333	0	0	0	0	0	-0.2390	-0.2392	-0.2391	-0.2391	-0.2391	-0.2391
F	0.167	0	0.333	0	0	0	0	-0.2390	-0.2392	-0.2391	-0.2391	-0.2391	-0.2391
G	0.167	0	0.333	0	0	0	0	-0.2390	-0.2392	-0.2391	-0.2391	-0.2391	-0.2391
							Difference in Page Rank	2.3910	0.0020	0.0009	0.0006	0.0004	0.0003

Question 4 – [2.00pt]

Because PageRank scores play important role for search engines, some disreputable sites try to obtain a high PageRank score by playing different types of tricks. One example is to have many spam pages to point to a Page A, and further point Page A to Page B, as shown in the following figure. Please calculate PageRank scores for all pages (using either simple iteration or eigenvector based methods) [1.00pt].

To calculate this problem I started small with 10 spam pages and notice that the PageRank value vectors do not converge but they alternate between Page A having score of 10 then Page B having score of 10 and then all pages being back to 1. The tables below will illustrate that the numbers never converge and the maximum PageRank values where highlighted in yellow.

Matrix A

	A	B	C	D	E	F	G	H	I	J	K	L
A	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
B	**1**	0	0	0	0	0	0	0	0	0	0	0
C	0	**1**	0	0	0	0	0	0	0	0	0	0
D	0	**1**	0	0	0	0	0	0	0	0	0	0
E	0	**1**	0	0	0	0	0	0	0	0	0	0
F	0	**1**	0	0	0	0	0	0	0	0	0	0
G	0	**1**	0	0	0	0	0	0	0	0	0	0
H	0	**1**	0	0	0	0	0	0	0	0	0	0
I	0	**1**	0	0	0	0	0	0	0	0	0	0
J	0	**1**	0	0	0	0	0	0	0	0	0	0
K	0	**1**	0	0	0	0	0	0	0	0	0	0
L	0	**1**	0	0	0	0	0	0	0	0	0	0

Normalized Matrix A

	A	B	C	D	E	F	G	H	I	J	K	L
A	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
B	**1**	0	0	0	0	0	0	0	0	0	0	0
C	0	**0.1**	0	0	0	0	0	0	0	0	0	0
D	0	**0.1**	0	0	0	0	0	0	0	0	0	0
E	0	**0.1**	0	0	0	0	0	0	0	0	0	0
F	0	**0.1**	0	0	0	0	0	0	0	0	0	0
G	0	**0.1**	0	0	0	0	0	0	0	0	0	0
H	0	**0.1**	0	0	0	0	0	0	0	0	0	0
I	0	**0.1**	0	0	0	0	0	0	0	0	0	0
J	0	**0.1**	0	0	0	0	0	0	0	0	0	0
K	0	**0.1**	0	0	0	0	0	0	0	0	0	0
L	0	**0.1**	0	0	0	0	0	0	0	0	0	0

	A	B	C	D	E	F	G	H	I	J	K	L	X0	X1	X2	X3	X4	X5	X6
A	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0.0833	0.8333	0.0833	0.0833	0.8333	0.0833	0.0833
B	**1**	0	0	0	0	0	0	0	0	0	0	0	0.0833	0.0833	0.8333	0.0833	0.0833	0.8333	0.0833
C	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
D	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
E	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
F	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
G	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
H	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
I	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
J	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
K	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
L	0	**0.1**	0	0	0	0	0	0	0	0	0	0	0.0833	0.0083	0.0083	0.0833	0.0083	0.0083	0.0833
													1.000	1.500	1.500	1.500	1.500	1.500	1.500
													1.000	1.000	1.000	1.000	1.000	1.000	1.000

As we see from small example that pages do not converge to any defined page value I decided to write a program that would take 1002 pages, page A, page B, and 1000 spam pages, and based on that calculate the output into an excel spreadsheet. After obtaining the results from the program we noticed same behavior as we did with 10 spam page simulation where PageRank values did not converge. The program and the spreadsheet holding results can be found under those links. Spreadsheet contains red matrix and green PageRank vectors.
Results : https://www.dropbox.com/s/zzd2z70z1ytu61m/Homework%2004%20-%20Excel.xlsx?dl=0
Program: https://www.dropbox.com/s/m42rffn0v36wg7n/Homework%2004%20-%20Code.ZIP?dl=0

Please explain how this approach can be used to mislead the search engine [0.50pt] and explain how search engine can detect this type of misuse [0.50pt].

This form of spamming can make page A and page B rank very high depending on which iteration values are used to calculate PageRank of both pages. The observation that I have from this example is that regardless of how many spam pages you have the PR average will always be equal to 1. To detect this the engine bot should do several PageRank calculations to find out if PageRank values are converging or they are stuck in the loop. If they would not be converging that would indicate the issue.