Analysis of the PageRank Algorithm

CSC 466
Lab3
Jorge Guevara
Arkadiy Kraminsky

1. Overview

The PageRank algorithm assigns numerical weights to elements of a set and computes the prestige of the element. The elements are represented as nodes in the algorithm with edges to other nodes in the set. The edges represent a connection with a node, and in practical use are hyperlinks that go into a certain website and links that go out. The implementation of the PageRank algorithm (for the purposes of this lab) consists of two distinct parts: the parsing of csv files and graph construction, and the application of the PageRank algorithm until the data converges.

Parsing of the csv files and creating a graph representation of the elements was trivial. For each comma delimited line, we add the elements into the graph (represented by a python dictionary) along with a list of elements or nodes the current node is connected with. These connections are represented as edges, which are created as undirected edges or directed if one of the elements has a greater weight than the other. As the graph or adjacency matrix is being built, other data structures are being maintained as well. This implementation of PageRank tracks the nodes from which another node came from and also adds to the degree every time there is an outgoing edge.

After the graph is constructed, the PageRank algorithm is finally applied. The algorithm first applies an automatic rank of 1/N to each node in the graph. N represents the number of nodes in the set. We then loop through all of the nodes in the graph and compute the rank of the node. As the algorithm loops through every node, it keeps a running sum of the PageRank divided by the degree of each of its sources. A .8 value is used as the constant d to finish up the calculation of the PageRank. With each new PageRank computed for each node, the algorithm performs a check for whether the old PageRank and the new PageRanks converged using an epsilon value of .00001

2. Results

2.1 NCAA Football

Description
This was run with no additional flags

Results

```
Read time: 23.400ms
Processing time: 21.085 ms
Iterations until convergence: 33

pageRanks:
1  obj: Mississippi with PageRank: 0.025487
2  obj: Florida with PageRank: 0.020637
3  obj: Utah with PageRank: 0.014658
4  obj: Oklahoma with PageRank: 0.014400
```

```
5  obj: Texas Tech with PageRank: 0.014206
6  obj: James Madison with PageRank: 0.012600
7  obj: Wake Forest with PageRank: 0.012407
8  obj: Texas with PageRank: 0.011919
9  obj: Oregon State with PageRank: 0.011917
10  obj: Alabama with PageRank: 0.011643
11  obj: Virginia Tech with PageRank: 0.011588
12  obj: Richmond with PageRank: 0.011564
13  obj: Montana with PageRank: 0.011355
14  obj: Vanderbilt with PageRank: 0.010016
15  obj: USC with PageRank: 0.009599
16  obj: Georgia Tech with PageRank: 0.009542
17  obj: Boston College with PageRank: 0.009514
18  obj: Virginia with PageRank: 0.009274
19  obj: South Carolina with PageRank: 0.008762
20  obj: Duke with PageRank: 0.008755
21  obj: North Carolina with PageRank: 0.008427
22  obj: Weber State with PageRank: 0.008213
23  obj: Florida State with PageRank: 0.007933
24  obj: Villanova with PageRank: 0.007831
25  obj: Maryland with PageRank: 0.007637
26  obj: Miami (FL with PageRank: 0.007579
27  obj: North Carolina State with PageRank: 0.007525
28  obj: TCU with PageRank: 0.007416
29  obj: Clemson with PageRank: 0.007252
30  obj: West Virginia with PageRank: 0.007016
31  obj: East Carolina with PageRank: 0.006838
32  obj: Georgia with PageRank: 0.006813
33  obj: Penn State with PageRank: 0.006604
34  obj: Cincinnati with PageRank: 0.006452
35  obj: Pittsburgh with PageRank: 0.006417
36  obj: LSU with PageRank: 0.006190
37  obj: Iowa with PageRank: 0.006183
38  obj: Appalachian State with PageRank: 0.005657
39  obj: Tulsa with PageRank: 0.005596
40  obj: Oregon with PageRank: 0.005305
```

Observation

The PageRank algorithm in this data set did discover the proper ranking. For example Mississippi came out on top because they lost only 4 games and beat teams with lower pageranks.

2.2 State Borders

Description

This was run with no additional flags

Results

```
Read time: 2.908ms
Processing time: 3.788ms
Iterations until convergence: 32

pageRanks:
1  obj: MA with PageRank: 0.028341
2  obj: TN with PageRank: 0.025236
3  obj: NY with PageRank: 0.025191
4  obj: ID with PageRank: 0.024185
5  obj: PA with PageRank: 0.023968
6  obj: MO with PageRank: 0.023254
7  obj: AR with PageRank: 0.023107
8  obj: KY with PageRank: 0.022680
9  obj: GA with PageRank: 0.022261
10  obj: OK with PageRank: 0.021829
11  obj: VA with PageRank: 0.021759
12  obj: NV with PageRank: 0.021440
13  obj: NH with PageRank: 0.020856
14  obj: TX with PageRank: 0.020762
15  obj: MD with PageRank: 0.019909
16  obj: UT with PageRank: 0.019145
17  obj: SD with PageRank: 0.018986
18  obj: WY with PageRank: 0.018882
19  obj: OR with PageRank: 0.018856
20  obj: CO with PageRank: 0.018780
21  obj: NB with PageRank: 0.018686
22  obj: OH with PageRank: 0.018465
23  obj: IA with PageRank: 0.018364
24  obj: VT with PageRank: 0.018049
25  obj: AL with PageRank: 0.017820
26  obj: CT with PageRank: 0.017764
27  obj: AZ with PageRank: 0.017734
28  obj: IL with PageRank: 0.017607
29  obj: NC with PageRank: 0.017284
30  obj: NM with PageRank: 0.017035
31  obj: MS with PageRank: 0.016970
32  obj: IN with PageRank: 0.015804
```

```
33  obj: WI with PageRank: 0.015793
34  obj: MT with PageRank: 0.015570
35  obj: MN with PageRank: 0.015435
36  obj: NJ with PageRank: 0.014959
37  obj: CA with PageRank: 0.014670
38  obj: LA with PageRank: 0.014549
39  obj: DE with PageRank: 0.014292
40  obj: MI with PageRank: 0.013195
```

Observations

The states that share the most borders came up on top, which makes a lot of sense for this dataset. States that share more borders, have more edges, which means more prestige runs through them.

2.3 Karate dataset

Description

This was run with no additional flags

Results

```
Read time: 9.985ms
Processing time: 2.347ms
Iterations until convergence: 21

pageRanks:
1  obj: 34 with PageRank: 0.098333
2  obj: 1 with PageRank: 0.094560
3  obj: 33 with PageRank: 0.070064
4  obj: 3 with PageRank: 0.055110
5  obj: 2 with PageRank: 0.051523
6  obj: 32 with PageRank: 0.036729
7  obj: 4 with PageRank: 0.035207
8  obj: 24 with PageRank: 0.031386
9  obj: 6 with PageRank: 0.029654
10  obj: 7 with PageRank: 0.029654
11  obj: 9 with PageRank: 0.029204
12  obj: 14 with PageRank: 0.028920
13  obj: 30 with PageRank: 0.026528
14  obj: 28 with PageRank: 0.025721
15  obj: 31 with PageRank: 0.024433
16  obj: 8 with PageRank: 0.024293
17  obj: 5 with PageRank: 0.022556
18  obj: 11 with PageRank: 0.022556
19  obj: 25 with PageRank: 0.021679
20  obj: 26 with PageRank: 0.021582
```

```
21  obj: 20 with PageRank: 0.019817
22  obj: 29 with PageRank: 0.019816
23  obj: 17 with PageRank: 0.017744
24  obj: 27 with PageRank: 0.015815
25  obj: 13 with PageRank: 0.015305
26  obj: 22 with PageRank: 0.015190
27  obj: 18 with PageRank: 0.015190
28  obj: 21 with PageRank: 0.015181
29  obj: 23 with PageRank: 0.015181
30  obj: 15 with PageRank: 0.015181
31  obj: 16 with PageRank: 0.015181
32  obj: 19 with PageRank: 0.015181
33  obj: 10 with PageRank: 0.014918
34  obj: 12 with PageRank: 0.010610
```

Observations

In this karate dataset, the order of ranks the algorithm computed is correct. Element 34 is the first in the list because it has the most incoming edges, and and other very high ranking nodes such as 32 and 33 are connected to 34. Likewise 1 is also ranked second because it has the second most incoming nodes and is also connected to very high ranking nodes.

2.4 Dolphins dataset

Description

This was run with no additional flags

Results

```
Read time: 12.129ms
Processing time: 5.212ms
Iterations until convergence: 22

pageRanks:
1  obj: Jet with PageRank: 0.031694
2  obj: Trigger with PageRank: 0.031419
3  obj: Grin with PageRank: 0.030895
4  obj: Web with PageRank: 0.029709
5  obj: SN4 with PageRank: 0.028784
6  obj: Topless with PageRank: 0.028428
7  obj: Scabs with PageRank: 0.027808
8  obj: Patchback with PageRank: 0.026151
9  obj: Gallatin with PageRank: 0.025554
10  obj: Beescratch with PageRank: 0.024103
11  obj: Kringel with PageRank: 0.023888
```

```
12  obj: SN63 with PageRank: 0.023785
13  obj: Feather with PageRank: 0.023062
14  obj: Stripes with PageRank: 0.021637
15  obj: SN9 with PageRank: 0.021300
16  obj: Upbang with PageRank: 0.021099
17  obj: SN100 with PageRank: 0.020317
18  obj: DN21 with PageRank: 0.019798
19  obj: Haecksel with PageRank: 0.019570
20  obj: Jonah with PageRank: 0.018918
21  obj: TR99 with PageRank: 0.018709
22  obj: SN96 with PageRank: 0.017469
23  obj: Number1 with PageRank: 0.017190
24  obj: TR77 with PageRank: 0.017125
25  obj: Double with PageRank: 0.016864
26  obj: Beak with PageRank: 0.016684
27  obj: MN105 with PageRank: 0.016611
28  obj: MN83 with PageRank: 0.016590
29  obj: Hook with PageRank: 0.016243
30  obj: Shmuddel with PageRank: 0.016054
31  obj: SN90 with PageRank: 0.015879
32  obj: DN63 with PageRank: 0.015559
33  obj: PL with PageRank: 0.015282
34  obj: Fish with PageRank: 0.015120
35  obj: Zap with PageRank: 0.014738
36  obj: Oscar with PageRank: 0.014736
37  obj: DN16 with PageRank: 0.014570
38  obj: Ripplefluke with PageRank: 0.013980
39  obj: Bumper with PageRank: 0.013664
40  obj: Thumper with PageRank: 0.013029
```

Observation

All the dolphins with many connections have a higher pagerank than the dolphins with few connections. For example Grin which has 12 connections and SN4 which has 11 connections, both have a higher pagerank than dolphins like Thumper which has 4 connections.

Results

2.5 Les Miserables dataset

Description

This was run with no additional flags

Results

```
Read time: 5.026ms
Processing time: 4.900ms
Iterations until convergence: 28

pageRanks:
1  obj: Valjean with PageRank: 0.074421
2  obj: Myriel with PageRank: 0.044325
3  obj: Gavroche with PageRank: 0.034359
4  obj: Marius with PageRank: 0.029599
5  obj: Javert with PageRank: 0.029264
6  obj: Thenardier with PageRank: 0.027091
7  obj: Fantine with PageRank: 0.026317
8  obj: Enjolras with PageRank: 0.020620
9  obj: Cosette with PageRank: 0.020196
10  obj: MmeThenardier with PageRank: 0.018978
11  obj: Bossuet with PageRank: 0.017965
12  obj: Courfeyrac with PageRank: 0.017580
13  obj: Eponine with PageRank: 0.017119
14  obj: Mabeuf with PageRank: 0.017074
15  obj: MlleGillenormand with PageRank: 0.016744
16  obj: Joly with PageRank: 0.016329
17  obj: Bahorel with PageRank: 0.016329
18  obj: Babet with PageRank: 0.016038
19  obj: Gueulemer with PageRank: 0.016038
20  obj: Claquesous with PageRank: 0.015899
21  obj: Tholomyes with PageRank: 0.015293
22  obj: Bamatabois with PageRank: 0.015289
23  obj: Gillenormand with PageRank: 0.015189
24  obj: Feuilly with PageRank: 0.015127
25  obj: Combeferre with PageRank: 0.015127
26  obj: Montparnasse with PageRank: 0.014639
27  obj: Grantaire with PageRank: 0.013823
28  obj: Prouvaire with PageRank: 0.012617
29  obj: Favourite with PageRank: 0.012507
30  obj: Fameuil with PageRank: 0.012507
31  obj: Dahlia with PageRank: 0.012507
32  obj: Blacheville with PageRank: 0.012507
33  obj: Zephine with PageRank: 0.012507
34  obj: Listolier with PageRank: 0.012507
35  obj: Chenildieu with PageRank: 0.012386
36  obj: Judge with PageRank: 0.012386
37  obj: Champmathieu with PageRank: 0.012386
38  obj: Brevet with PageRank: 0.012386
```

```
39  obj: Cochepaille with PageRank: 0.012386
40  obj: Fauchelevent with PageRank: 0.012377
```

Observations

Valjean has the highest pagerank for a reason, he has 36 connections with other characters. The other character with the highest pagerank is Myriel, but this character only has 10 connections. The disparity of connections between Valjean and Myriel is apparent in the difference in pagerank.

2.6 Political Blogs dataset

Description

This was run with no additional flags

Results

```
Read time: 142.929ms
Processing time: 186.999ms
Iterations until convergence: 30

pageRanks:
1  obj: 155 with PageRank: 0.012522
2  obj: 55 with PageRank: 0.010292
3  obj: 855 with PageRank: 0.009051
4  obj: 1051 with PageRank: 0.008560
5  obj: 641 with PageRank: 0.008525
6  obj: 963 with PageRank: 0.008043
7  obj: 1153 with PageRank: 0.007484
8  obj: 729 with PageRank: 0.007069
9  obj: 1245 with PageRank: 0.006188
10  obj: 798 with PageRank: 0.005927
11  obj: 1112 with PageRank: 0.005838
12  obj: 323 with PageRank: 0.005785
13  obj: 1461 with PageRank: 0.004751
14  obj: 1306 with PageRank: 0.004746
15  obj: 1437 with PageRank: 0.004652
16  obj: 1041 with PageRank: 0.004605
17  obj: 1179 with PageRank: 0.004564
18  obj: 1463 with PageRank: 0.004493
19  obj: 990 with PageRank: 0.004195
20  obj: 535 with PageRank: 0.004159
21  obj: 642 with PageRank: 0.003752
22  obj: 180 with PageRank: 0.003720
23  obj: 301 with PageRank: 0.003667
24  obj: 1067 with PageRank: 0.003662
```

```
25  obj: 756 with PageRank: 0.003593
26  obj: 514 with PageRank: 0.003577
27  obj: 1086 with PageRank: 0.003513
28  obj: 297 with PageRank: 0.003369
29  obj: 1479 with PageRank: 0.003312
30  obj: 1270 with PageRank: 0.003238
31  obj: 741 with PageRank: 0.003148
32  obj: 878 with PageRank: 0.003129
33  obj: 1101 with PageRank: 0.003065
34  obj: 434 with PageRank: 0.002969
35  obj: 1317 with PageRank: 0.002930
36  obj: 170 with PageRank: 0.002918
37  obj: 493 with PageRank: 0.002887
38  obj: 1159 with PageRank: 0.002822
39  obj: 1293 with PageRank: 0.002739
40  obj: 979 with PageRank: 0.002722
```

Observations

It makes sense that 155 has the highest page rank since it is sited 338 times. The blogs with the most citations bubble up to the higher prestige ranking. Blog 798 is the first blog where the citation count goes below 200. So pagerank picks out the most cited blogs, but they are not necessarily ordered by their number of citations.

2.7 Wiki Vote

Description

Ran with the -d/--directed flag as the seconds argument

Results

```
Read time: 395.172ms
Processing time: 790.062ms
Iterations until convergence: 19

pageRanks:
1  obj: 4037 with PageRank: 0.002298
2  obj: 15 with PageRank: 0.001803
3  obj: 6634 with PageRank: 0.001659
4  obj: 2625 with PageRank: 0.001584
5  obj: 2470 with PageRank: 0.001288
6  obj: 2237 with PageRank: 0.001260
7  obj: 2398 with PageRank: 0.001246
8  obj: 4191 with PageRank: 0.001103
9  obj: 5254 with PageRank: 0.001051
10  obj: 7553 with PageRank: 0.001044
```

```
11  obj: 1186 with PageRank: 0.001035
12  obj: 2328 with PageRank: 0.000993
13  obj: 7620 with PageRank: 0.000939
14  obj: 1297 with PageRank: 0.000935
15  obj: 4335 with PageRank: 0.000924
16  obj: 4875 with PageRank: 0.000915
17  obj: 7632 with PageRank: 0.000904
18  obj: 5412 with PageRank: 0.000902
19  obj: 2654 with PageRank: 0.000881
20  obj: 3352 with PageRank: 0.000864
21  obj: 8293 with PageRank: 0.000860
22  obj: 6832 with PageRank: 0.000845
23  obj: 28 with PageRank: 0.000842
24  obj: 762 with PageRank: 0.000842
25  obj: 665 with PageRank: 0.000838
26  obj: 6946 with PageRank: 0.000831
27  obj: 737 with PageRank: 0.000830
28  obj: 214 with PageRank: 0.000826
29  obj: 6774 with PageRank: 0.000817
30  obj: 2535 with PageRank: 0.000812
31  obj: 3089 with PageRank: 0.000811
32  obj: 2066 with PageRank: 0.000810
33  obj: 3334 with PageRank: 0.000802
34  obj: 4735 with PageRank: 0.000779
35  obj: 7092 with PageRank: 0.000770
36  obj: 2565 with PageRank: 0.000752
37  obj: 5484 with PageRank: 0.000748
38  obj: 4310 with PageRank: 0.000696
39  obj: 5423 with PageRank: 0.000685
40  obj: 1211 with PageRank: 0.000684
```

Observations

Object 4037 has the highest pagerank, because it has over 400 connections. Object 6634 has only 200 connections, but since they are good connections, it gets the 3rd highest pagerank.

2.8 p2p-Gnutella05

Description

Ran with the -d/--directed flag as the seconds argument

Results

```
Read time: 125.194ms
Processing time: 225.855ms
```

```
Iterations until convergence: 12

pageRanks:
1  obj: 1676 with PageRank: 0.000311
2  obj: 1020 with PageRank: 0.000305
3  obj: 386 with PageRank: 0.000291
4  obj: 222 with PageRank: 0.000288
5  obj: 227 with PageRank: 0.000280
6  obj: 389 with PageRank: 0.000276
7  obj: 388 with PageRank: 0.000274
8  obj: 688 with PageRank: 0.000265
9  obj: 226 with PageRank: 0.000263
10  obj: 842 with PageRank: 0.000261
11  obj: 876 with PageRank: 0.000260
12  obj: 223 with PageRank: 0.000242
13  obj: 31 with PageRank: 0.000240
14  obj: 391 with PageRank: 0.000238
15  obj: 271 with PageRank: 0.000233
16  obj: 279 with PageRank: 0.000232
17  obj: 225 with PageRank: 0.000230
18  obj: 277 with PageRank: 0.000230
19  obj: 274 with PageRank: 0.000226
20  obj: 272 with PageRank: 0.000224
21  obj: 887 with PageRank: 0.000224
22  obj: 278 with PageRank: 0.000222
23  obj: 229 with PageRank: 0.000222
24  obj: 47 with PageRank: 0.000211
25  obj: 541 with PageRank: 0.000209
26  obj: 221 with PageRank: 0.000209
27  obj: 230 with PageRank: 0.000208
28  obj: 679 with PageRank: 0.000206
29  obj: 385 with PageRank: 0.000202
30  obj: 276 with PageRank: 0.000200
31  obj: 821 with PageRank: 0.000199
32  obj: 999 with PageRank: 0.000195
33  obj: 275 with PageRank: 0.000194
34  obj: 48 with PageRank: 0.000182
35  obj: 387 with PageRank: 0.000181
36  obj: 693 with PageRank: 0.000179
37  obj: 392 with PageRank: 0.000170
38  obj: 224 with PageRank: 0.000170
39  obj: 1086 with PageRank: 0.000162
40  obj: 1297 with PageRank: 0.000162
```

Observations

Object 1676 has 76 incoming edges and is ranked first, whereas object 386 has 77 and is ranked third. The reason for this is that 386 source nodes collectively have lower rank than those of 1676 source nodes.

2.9 SlashdotZoo

Description

Ran with the -d/--directed flag as the seconds argument

Results

```
Read time: 2.132883s
Processing time: 9.657419s
Iterations until convergence: 28

pageRanks:
1  obj: 75 with PageRank: 0.002090
2  obj: 43 with PageRank: 0.002010
3  obj: 749 with PageRank: 0.001915
4  obj: 184 with PageRank: 0.001284
5  obj: 38 with PageRank: 0.001265
6  obj: 625 with PageRank: 0.000930
7  obj: 163 with PageRank: 0.000728
8  obj: 1810 with PageRank: 0.000630
9  obj: 57 with PageRank: 0.000582
10  obj: 651 with PageRank: 0.000556
11  obj: 34 with PageRank: 0.000549
12  obj: 85 with PageRank: 0.000548
13  obj: 74 with PageRank: 0.000541
14  obj: 15 with PageRank: 0.000527
15  obj: 1808 with PageRank: 0.000521
16  obj: 53 with PageRank: 0.000519
17  obj: 50 with PageRank: 0.000504
18  obj: 1832 with PageRank: 0.000498
19  obj: 877 with PageRank: 0.000434
20  obj: 3335 with PageRank: 0.000432
21  obj: 1116 with PageRank: 0.000412
22  obj: 1240 with PageRank: 0.000402
23  obj: 1397 with PageRank: 0.000365
24  obj: 28 with PageRank: 0.000349
25  obj: 13382 with PageRank: 0.000336
26  obj: 945 with PageRank: 0.000332
27  obj: 47 with PageRank: 0.000322
```

```
28  obj: 3537 with PageRank: 0.000315
29  obj: 1491 with PageRank: 0.000312
30  obj: 46 with PageRank: 0.000312
31  obj: 1981 with PageRank: 0.000277
32  obj: 17 with PageRank: 0.000275
33  obj: 523 with PageRank: 0.000273
34  obj: 670 with PageRank: 0.000270
35  obj: 165 with PageRank: 0.000270
36  obj: 1803 with PageRank: 0.000268
37  obj: 1850 with PageRank: 0.000267
38  obj: 2113 with PageRank: 0.000254
39  obj: 1300 with PageRank: 0.000248
40  obj: 885 with PageRank: 0.000247
```

Observations

Object 75 has the most incoming edges with 2532 and is ranked 1st. Object 43 has 2323 incoming edges and is ranked second. PageRank picked an accurate PageRank for the top results.

2.10 Amazon Product

Description

Ran with the -d/--directed flag as the seconds argument

Results

```
Read time: 14.620988s
Processing time: 86.616600s
Iterations until convergence: 32

pageRanks:
1  obj: 593 with PageRank: 0.001376
2  obj: 89 with PageRank: 0.001069
3  obj: 595 with PageRank: 0.001066
4  obj: 591 with PageRank: 0.001049
5  obj: 590 with PageRank: 0.000756
6  obj: 972 with PageRank: 0.000719
7  obj: 977 with PageRank: 0.000638
8  obj: 2612 with PageRank: 0.000616
9  obj: 976 with PageRank: 0.000605
10  obj: 974 with PageRank: 0.000593
11  obj: 975 with PageRank: 0.000575
12  obj: 120 with PageRank: 0.000570
13  obj: 634 with PageRank: 0.000560
14  obj: 978 with PageRank: 0.000528
```

```
15  obj: 598 with PageRank: 0.000462
16  obj: 585 with PageRank: 0.000410
17  obj: 4455 with PageRank: 0.000402
18  obj: 162 with PageRank: 0.000392
19  obj: 597 with PageRank: 0.000388
20  obj: 44 with PageRank: 0.000379
21  obj: 4458 with PageRank: 0.000370
22  obj: 88 with PageRank: 0.000363
23  obj: 596 with PageRank: 0.000350
24  obj: 39 with PageRank: 0.000345
25  obj: 1196 with PageRank: 0.000340
26  obj: 4460 with PageRank: 0.000331
27  obj: 594 with PageRank: 0.000328
28  obj: 605 with PageRank: 0.000300
29  obj: 2611 with PageRank: 0.000297
30  obj: 587 with PageRank: 0.000295
31  obj: 10999 with PageRank: 0.000292
32  obj: 4461 with PageRank: 0.000290
33  obj: 157 with PageRank: 0.000287
34  obj: 4459 with PageRank: 0.000281
35  obj: 4454 with PageRank: 0.000270
36  obj: 7241 with PageRank: 0.000261
37  obj: 2264 with PageRank: 0.000255
38  obj: 578 with PageRank: 0.000249
39  obj: 158 with PageRank: 0.000246
40  obj: 37 with PageRank: 0.000246
```

Observations

Object 593 has the most incoming nodes with the highest PageRank, 89 is ranked second. All top results have high incoming edges, but don’t have too many outgoing which results in other nodes getting less rank.

2.11 Live Journal

Description

Ran with the -d/--directed flag as the seconds argument

Results

```
N/A not enough memory
```

Observations

Took an extremely long time and failed due to not enough memory

3. Summary

PageRank attempts to assign a prestige rank to elements of a set which are linked by incoming and outgoing edges. Overall, of all the datasets ran against this implementation PageRank algorithm, most if not all results were accurate or appropriate. By eyeball observation, the top ranked objects were always had the most incoming nodes as well as total nodes. The basic idea behind PageRank is that it compares the current prestige each object has with its previous iterations prestige. The prestige of a node is calculated based on the amount of outgoing edges to the rank of all of the objects source nodes. In effect, a low amount of outgoing edges and more source nodes with high ranks result in a higher prestige for the object. The PageRank algorithm worked best with the larger SNAP data sets. The algorithm was fairly quick in computing results and the top ranked objects were always justified by the results.

4. Performance Analysis

4.1 Analysis

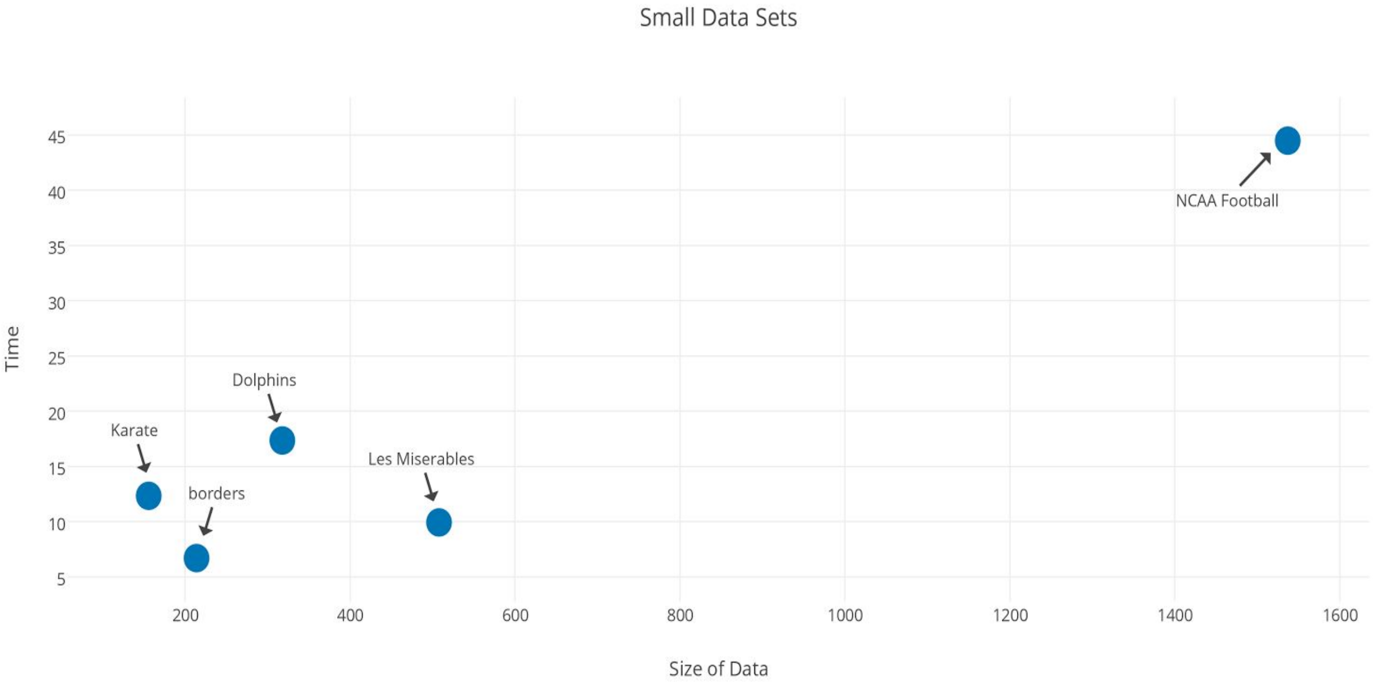
To conclude whether the PageRank algorithm is efficient and consistent in finding the prestige of elements in a set, the performance of the algorithm must be analyzed. On smaller datasets, such as the basic non SNAP datasets, the algorithm's read time and processing time is very quick. Because each element in the smaller datasets can only have so many edges from where it came from and edges to where it's going, the amount of time required to calculate new PageRanks is small. Likewise, the algorithm performs quickly on the large SNAP datasets, with the caveat being that each element doesn't have an absurd amount of incoming and outgoing edges to other elements. For example, the SNAP dataset for Live Journal has an extremely large amount of incoming and outgoing edges for many elements in the set and therefore requires a significant amount of computing power and time. In real world application such as on the world wide web, this implementation of PageRank would be inefficient. Like the SNAP dataset for Live Journal, the representation of all of the webpages on the internet would be robust and would take hours if not days to completely compute the PageRanks.

This implementation of the PageRank algorithm successfully worked on all but the large SNAP dataset for Live Journal, and the failure was due to insufficient computing power and resources. The wiki-vote SNAP dataset read and processed the data in 1.185 seconds, p2p-Gnutella ran in 351.05 milliseconds, Slashdotzoo ran in 11.79 seconds, and the Amazon SNAP took 101.237 seconds to complete.

Most of the performance comes from the data structures that we used. We thought representing the graph as a matrix, but we quickly realized that such approach would waste a lot of space. We assumed that we would be working with mostly sparse data, so we opted to represent our graph as an adjacency matrix. Our graph is represented by a dictionary, where the node is the key and the edges are represented by a list that are mapped to the node. So if node 'x' has an edge to 'y' and 'z', we can represent this in python as: destinations['x'] = ['y', 'z']

The above implementation works, but to speed things up, we created an inverted index where all targets point to their sources. Using the above example, we will have sources['y'] = ['x'] and sources['z'] = ['x']. This structure allows us to do very quick lookups when we are computing the pagerank of a node and we need to look at all the other nodes that are pointing at it. Essentially, our algorithm is made up of dictionary access operations and basic math operations. We are very happy with our implementation of PageRank.

4.2 Graphs


Small Data Sets
Time
45
40
35
30
25
20
15
10
5
NCAA Football
Dolphins
Karate
Les Miserables
borders
200
400
600
800
1000
1200
1400
1600
Size of Data

5. Appendix

To run the program just type python pagerank.py filename, where filename is the name of the dataset. Note that SNAP datasets require a special flag to run. To run SNAP datasets, please use python pagerank.py -d.