In [2]:
```
#Reload changes -> always run this
%load_ext autoreload
%autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

HW 9.0: Short answer questions

What is PageRank and what is it used for in the context of web search?

PageRank is an algorithm developed to measure the importance of websites. The way this algorithm used is to count the number and quality of links to determine how important a website is. It is used to help determine what search results should be returned to answer a user query, although it is not the only determining factor.

What modifications have to be made to the webgraph in order to leverage the machinery of Markov Chains to compute the steady stade distibuton?

There are two modifications that we need to make. These are the following:

1. Stochasticity: which we use to resolve the dangling nodes. This means we will represent the web as a stochastic matrix, which means we will represent it as a square matrix of nonnegative real numbers, where each row sums to 1.
2. Primitivity: incoporates teleportation. This represents the random surfer "getting bored" and teleporting to a random page. A teleportation factor (1 - the dampening factor) is applied.

OPTIONAL: In topic-specific pagerank, how can we insure that the irreducible property is satified? (HINT: see HW9.4)

9.1: MRJob Implementation of Basic PageRank

Write a basic MRJob implementation of the iterative PageRank algorithm that takes sparse adjacency lists as input (as explored in HW 7). Make sure that you implementation utilizes teleportation (1-damping/the number of nodes in the network), and further, distributes the mass of dangling nodes with each iteration so that the output of each iteration is correctly normalized (sums to 1). [NOTE: The PageRank algorithm assumes that a random surfer (walker), starting from a random web page, chooses the next page to which it will move by clicking at random, with probability d, one of the hyperlinks in the current page. This probability is represented by a so-called 'damping factor' d, where d $\in$ (0, 1). Otherwise, with probability (1 − d), the surfer jumps to any web page in the network. If a page is a dangling end, meaning it has no outgoing hyperlinks, the random surfer selects an arbitrary web page from a uniform distribution and "teleports" to that page]

As you build your code, use the test data

s3://ucb-mids-mls-networks/PageRank-test.txt Or under the Data Subfolder for HW7 on Dropbox with the same file name. (On Dropbox https://www.dropbox.com/sh/2c0k5adwz36lkcw/AAAAKsjQfF9uHfv-X9mCqr9wa?dl=0 (https://www.dropbox.com/sh/2c0k5adwz36lkcw/AAAAKsjQfF9uHfv-X9mCqr9wa?dl=0))

with teleportation parameter set to 0.15 (1-d, where d, the damping factor is set to 0.85), and crosscheck your work with the true result, displayed in the first image in the Wikipedia article:

https://en.wikipedia.org/wiki/PageRank (https://en.wikipedia.org/wiki/PageRank)

and here for reference are the corresponding PageRank probabilities:

A,0.033

B,0.384

C,0.343

D,0.039

E,0.081

F,0.039

G,0.016

H,0.016

I,0.016

J,0.016

K,0.016

In [49]:
```
%%writefile numberNodesMR.py

from mrjob.job import MRJob
from mrjob.step import MRStep

#this job calculates the number of nodes in a network, including da
ngling nodes. Used as input to other jobs
class NumberNodes(MRJob):

    def mapper1(self, _, line):
        newLine = line.split('\t') #split input line

        node = newLine[0]
        neighbors = eval(newLine[1]) #use eval to make sure diction
ary is preserved
        yield node, 1 #yield original node
        for neighbor in neighbors.keys():
            yield neighbor, 1 #get all neighbors

    def reducer1(self, key, values):
        yield key, 1

    def mapper2(self, key, values):
        yield None, 1 #don't care about nodes or neighbors anymore

    def reducer2(self, key, values):
        total = sum(values) #sum results
        yield None, total


    def steps(self):
        return [MRStep(mapper = self.mapper1, reducer = self.reduce
r1),
                MRStep(mapper = self.mapper2, reducer = self.reducer
2)]

if __name__ == "__main__":
    NumberNodes.run()
```

```
Writing numberNodesMR.py
```

In [3]:
```
from numberNodesMR import NumberNodes

filename = 'PageRank-test.txt' #using for testing

mr_job = NumberNodes(args = [filename])

with mr_job.make_runner() as runner:
    runner.run()
    print "Number of nodes in " + filename
    for line in runner.stream_output():
        print mr_job.parse_output_line(line)[1]
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:

Number of nodes in PageRank-test.txt
11
```

In [17]:
```
%%writefile initPR.py

from mrjob.job import MRJob
from mrjob.step import MRStep

#this job will initialize a text file graph so it can be used for t
he next series of jobs
class InitPRJob(MRJob):

    def configure_options(self):
        super(InitPRJob, self).configure_options()
        #pass number of nodes as an argument
        self.add_passthrough_option('--numNodes', type = float, def
ault = 10, help = 'Number of Nodes in Graph')


    def mapper(self, _, line):

        line = line.split('\t') #split input
        node = line[0]
        adj = eval(line[1]) #eval here to get dictionary
        for neighbor in adj.keys():
            yield neighbor, {} #yield an empty dictionary for all n
eighbors
        yield node, adj #get original


    def reducer(self, key, values):
        nid = key
        adj = {}
        dangling = True
        newValues = [value for value in values] #unpack values to a
list
        for dictionary in newValues:
            if len(dictionary) != 0:
                adj = dictionary #set adjacency list if not a dangl
ing node

        PageRank = float(1)/self.options.numNodes #calculate starti
ng pagerank, 1/numberNodes

        yield nid, (PageRank, adj)

    def steps(self):
        return [
            MRStep(mapper = self.mapper, reducer = self.reducer)
        ]

if __name__ == "__main__":
    InitPRJob.run()
```

```
Overwriting initPR.py
```

In [5]:
```
from initPR import InitPRJob

mr_job = InitPRJob(args = ['PageRank-test.txt', '--numNodes', '1
1'])

with open('initData.txt', 'w+') as myfile: #this file will be used
as input for the next series of jobs
    with mr_job.make_runner() as runner:
        runner.run()
        for line in runner.stream_output():
            myfile.write(line)
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
```

In [11]:

```
%%writefile pageRank.py

from mrjob.job import MRJob
from mrjob.step import MRStep

#calculate one iteration of pageRank
class PageRankMR(MRJob):

    def configure_options(self):
        super(PageRankMR, self).configure_options()
        self.add_passthrough_option('--alpha', type = float, defaul
t = 0.15, help = "Alpha") #alpha
        self.add_passthrough_option('--numNodes', type = int, defau
lt = 10, help = 'Number Nodes') #number of Nodes

    def mapper1(self, _, line):
        data = line.split('\t')
        nid = eval(data[0]) #evaluate the node id which is a string
        data2 = eval(data[1]) #Structure is (PageRank, neighbors)
        curPr = float(data2[0]) #convert to float
        neighbors = data2[1]

        yield nid, neighbors #yield original node and neighbors to
preserve structure

        if len(neighbors) == 0:
            yield '*', curPr #we are at a dangling node
        else:
            newPR = curPr/len(neighbors) #new pagerank based on the
number of neighbors
            for nid in neighbors.keys():
                yield nid, newPR #yield neighbor and pagerank

    def reducer_init(self):
        self.mass = 0 #stores the dangling mass

    def reducer1(self, key, values):
        if key == '*':
            for value in values:
                curPr = value
                self.mass += curPr #increment dangling mass
        else:
            newPr = float(0) #new pageRank
            adj = {} #neighbors

            for value in values:
                if type(value) == dict:
                    adj = value #our node has neighbors so set the
neighbors dict
                else:
                    newPr += value #we're at a pagerank value so in
crement the new pagerank

            yield key, (newPr, adj) #yield results
```

```
    def reducer_final(self):
        yield '*', self.mass #pass dangling mass to be distributed
later

    def reducer2_init(self):
        self.mass = 0.0 #initialize, probably a more efficient way
to deal with this

    def reducer2(self, key, values):
        nid = key
        if nid == '*': #pass mass along here
            for value in values:
                self.mass += value #if we've gotten dangling mass i
ncrement its value here
        else:
            valList = [value for value in values][0] #unpack incomi
ng values, (pageRank, neighbors)
            curPr = float(valList[0]) #set variables
            neighbors = valList[1]

            newMass = self.mass/self.options.numNodes #new mass dis
tributed among all nodes
            alpha = self.options.alpha

            newPr = (alpha/float(self.options.numNodes)) + ((1-alph
a) * float(newMass + curPr)) #final pagerank

            yield nid, (newPr, neighbors)

    def steps(self):
        return [
            MRStep(mapper = self.mapper1,
                   reducer_init = self.reducer_init,
                   reducer = self.reducer1,
                   reducer_final = self.reducer_final),
            MRStep(reducer_init = self.reducer2_init, reducer = sel
f.reducer2)
        ]

if __name__ == "__main__":
    PageRankMR.run()
```

```
Overwriting pageRank.py
```

In [12]:
```
from pageRank import PageRankMR

#This just runs one iteration to check our output

mr_job = PageRankMR(args = ['initData.txt', '--alpha', '0.15', '--n
umNodes', '11'])

with mr_job.make_runner() as runner:
    runner.run()
    for line in runner.stream_output():
        print mr_job.parse_output_line(line)
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:

('A', [0.059297520661157024, {}])
('B', [0.3168732782369146, {'C': 1}])
('C', [0.09793388429752066, {'B': 1}])
('D', [0.046418732782369146, {'A': 1, 'B': 1}])
('E', [0.32975206611570246, {'B': 1, 'D': 1, 'F': 1}])
('F', [0.046418732782369146, {'B': 1, 'E': 1}])
('G', [0.02066115702479339, {'B': 1, 'E': 1}])
('H', [0.02066115702479339, {'B': 1, 'E': 1}])
('I', [0.02066115702479339, {'B': 1, 'E': 1}])
('J', [0.02066115702479339, {'E': 1}])
('K', [0.02066115702479339, {'E': 1}])
```

In [13]:
```
from initPR import InitPRJob
from pageRank import PageRankMR
import os

#Driver for controlling all phases from initiation to iteration

# mr_job_init = InitPRJob(args = ['PageRank-test.txt'])
# with open('initData.txt', 'w+') as myfile:
#     with mr_job_init.make_runner() as runner:
#         runner.run()
#         for line in runner.stream_output():
#             myfile.write(line)

iteration = 0

while(iteration < 40): #set iteration
    if iteration == 0: #if on 0 iteration take the input file and p
roduce the intermediate results
        with open('interResults.txt', 'w+') as myfile:
            mr_job = PageRankMR(args = ['initData.txt', '--alpha',
'0.15', '--numNodes', '11'])
            with mr_job.make_runner() as runner:
                runner.run()
                for line in runner.stream_output():
                    myfile.write(line)
        iteration += 1
    else: #otherwise we're on an iteration and run
        with open('newFile.txt', 'w+') as myfile:
            mr_job = PageRankMR(args = ['interResults.txt', '--alph
a', '0.15', '--numNodes', '11'])
            with mr_job.make_runner() as runner:
                runner.run()
                for line in runner.stream_output():
                    myfile.write(line)
        iteration += 1
        os.rename('newFile.txt', 'interResults.txt')

print "All done"

#Would probably want a more formal definition of convergence.
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
```

```
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
```

```
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
```

```
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
```

```
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:

All done
```

Confirming the results from our text file here:

"A" [0.03278149316111876, {}]

"B" [0.3842426353874476, {"C": 1}]

"C" [0.3430685989242862, {"B": 1}]

"D" [0.03908709210193935, {"A": 1, "B": 1}]

"E" [0.08088569323767447, {"B": 1, "D": 1, "F": 1}]

"F" [0.03908709210193935, {"B": 1, "E": 1}]

"G" [0.016169479017118762, {"B": 1, "E": 1}]

"H" [0.016169479017118762, {"B": 1, "E": 1}]

"I" [0.016169479017118762, {"B": 1, "E": 1}]

"J" [0.016169479017118762, {"E": 1}]

"K" [0.016169479017118762, {"E": 1}]

HW 9.2: Exploring PageRank teleportation and network plots

In order to overcome problems such as disconnected components, the damping factor (a typical value for d is 0.85) can be varied. Using the graph in HW1, plot the test graph (using networkx, https://networkx.github.io/ (https://networkx.github.io/)) for several values of the damping parameter alpha, so that each nodes radius is proportional to its PageRank score. In particular you should do this for the following damping factors: [0,0.25,0.5,0.75, 0.85, 1]. Note your plots should look like the following:

Factors: 0

0.25

0.5

0.75

0.85

1

https://en.wikipedia.org/wiki/PageRank#/media/File:PageRanks-Example.svg (https://en.wikipedia.org/wiki/PageRank#/media/File:PageRanks-Example.svg)

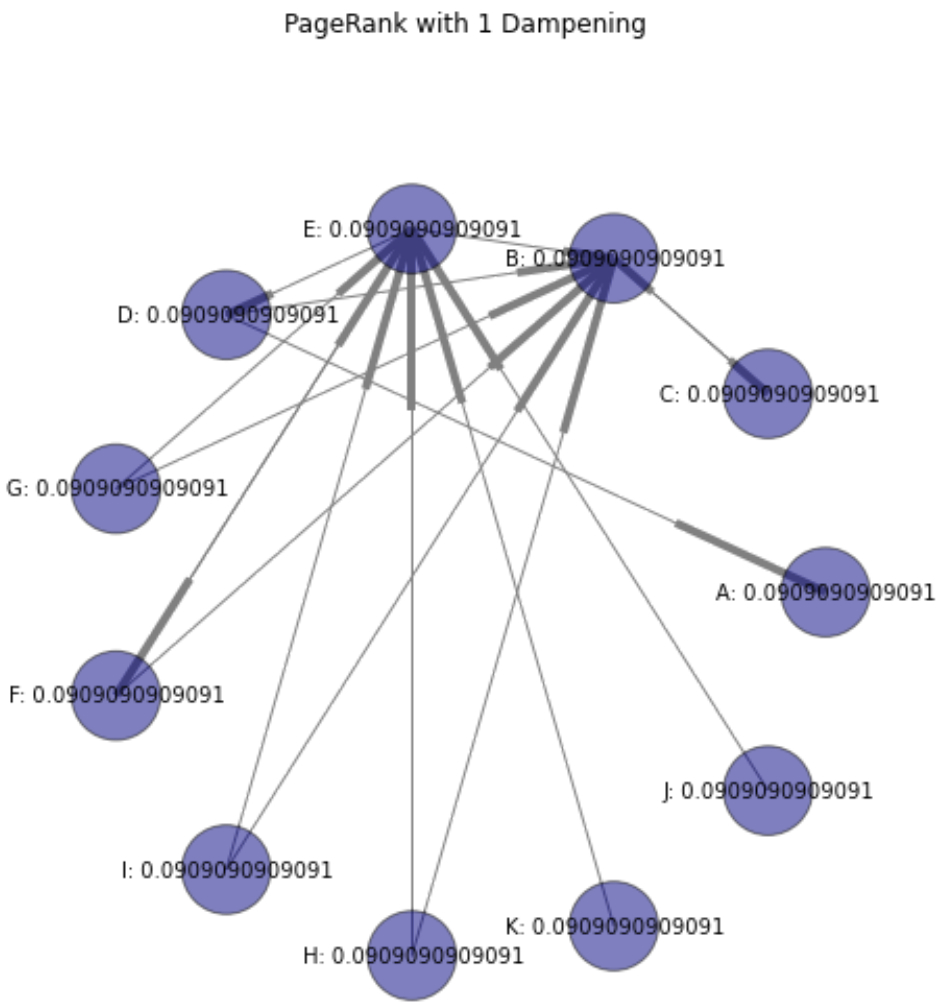
In [1]:
```
from pageRank import PageRankMR

#Create list of teleportation factors, which are 1-the dampening fa
ctor provided to work with our code
d = [1, 0.75, 0.5, 0.25, 0.15, 0]

for damp in d: #generate output files to use for input, we are only
using one iteration
    with open('output' + str(damp) + '.txt', 'w+') as myfile:
        mr_job = PageRankMR(args = ['initData.txt', '--alpha', str
(damp), '--numNodes', '11'])
        with mr_job.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                myfile.write(line)
```

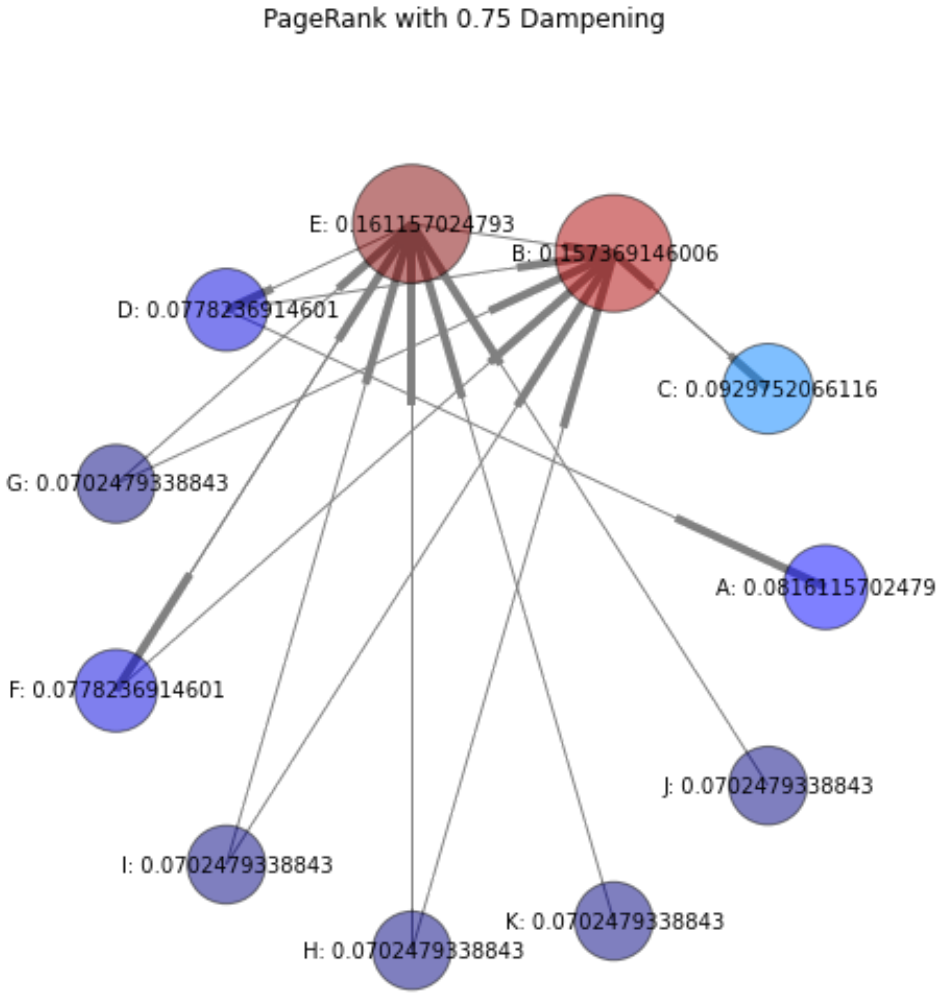
```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
```

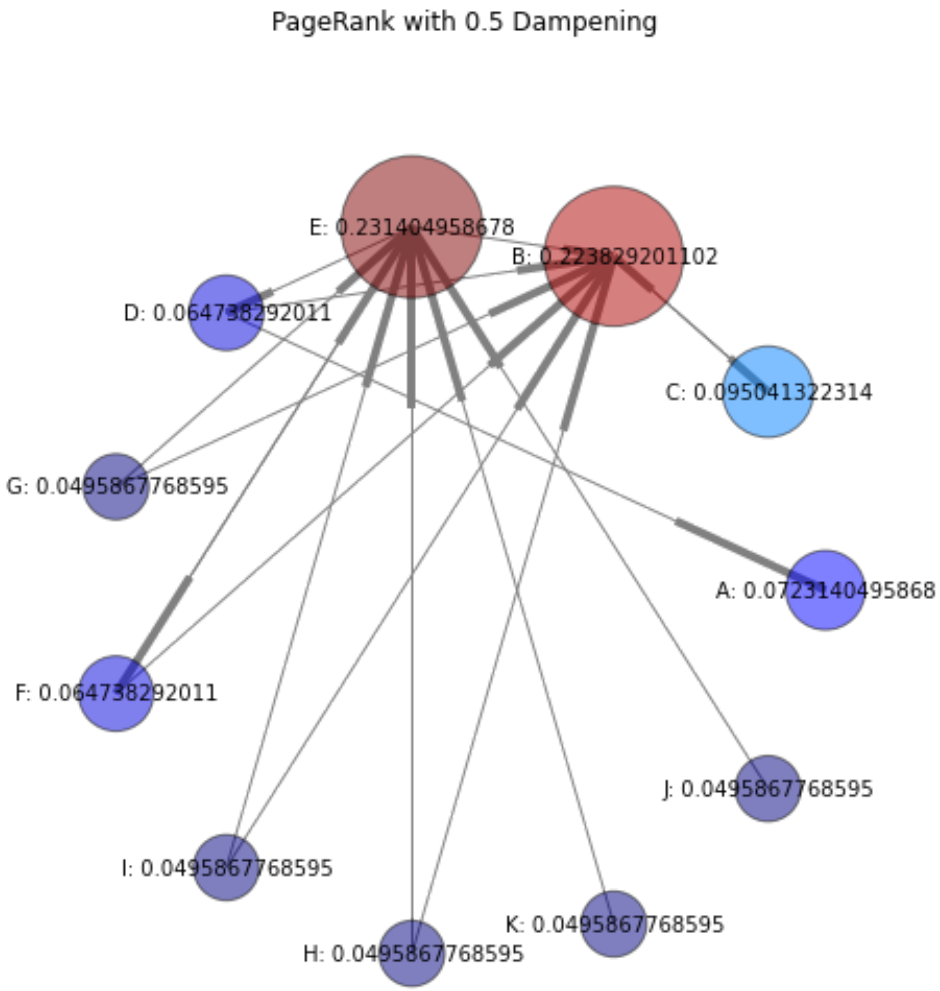
In [2]:

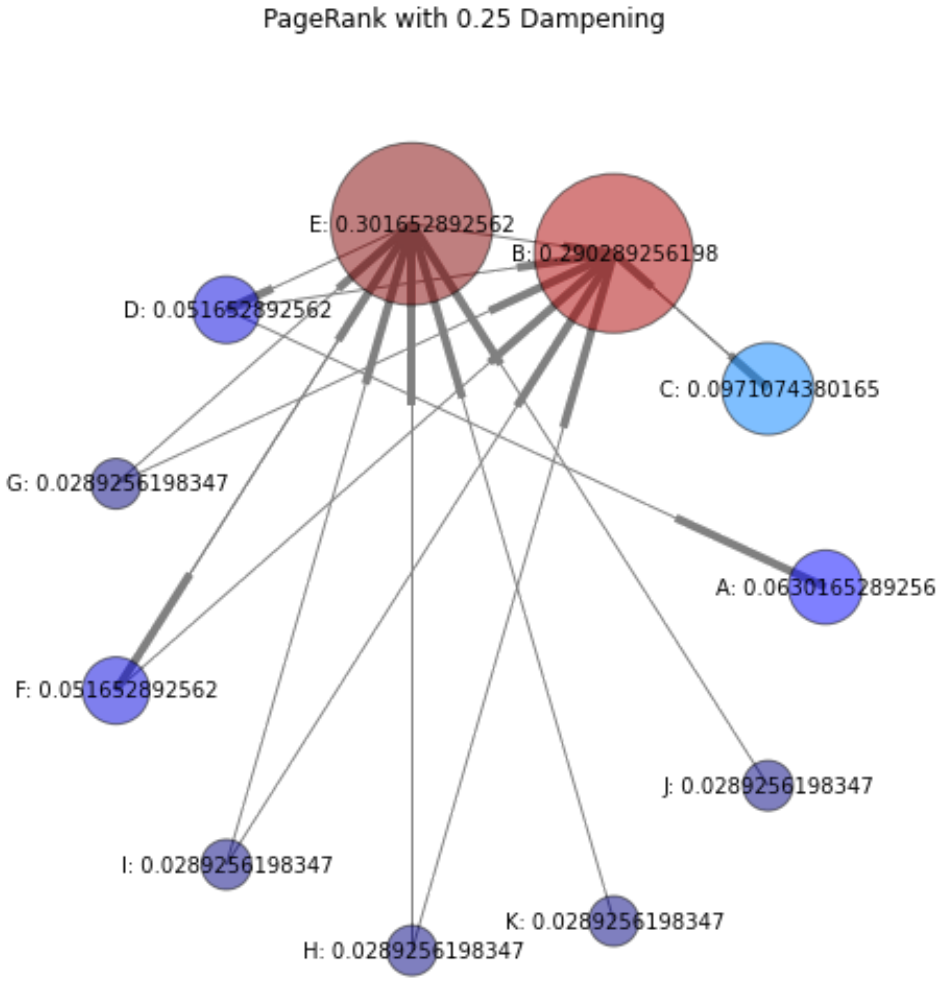
```
%matplotlib inline
import networkx as nx
from matplotlib import pyplot as plt

def draw(inputFile, d):
    edges = [] #variables to hold our results
    pr = {}

    with open(inputFile, 'r') as myfile:
        lines = myfile.readlines()
        for line in lines:
            data = line.split('\t') #split and grab data
            nid = eval(data[0]) #evaluate string

            data2 = eval(data[1]) #evaluate (PageRank, neighbors)
            curPr = float(data2[0])
            neighbors = data2[1]

            for neighbor in neighbors.keys():
                edges.append((nid, neighbor)) #append edges between
nodes and neighbors

            pr[nid] = curPr #add pagerank

    #Have now gotten all the inputs needed for the graph

    plt.figure(figsize = (8,8))
    DG = nx.DiGraph()

    for edge in edges:
        DG.add_edge(edge[0], edge[1]) #draw an edge

    node_size = [pr[node] * 20000 for node in DG.nodes()] #set size
of nodes
    graph_pos = nx.circular_layout(DG) #use circular layout

    labels = {}
    for node in DG.nodes():
        labels[node] = node + ': ' + str(pr[node]) #add node ID and
pagerank as label


    nx.draw_networkx_nodes(DG, graph_pos, node_size = node_size, no
de_color = node_size, alpha = 0.5) #graph
    nx.draw_networkx_edges(DG, graph_pos, edge_color = 'grey', arro
ws = True)
    nx.draw_networkx_labels(DG, graph_pos, labels = labels, font_si
ze = 10)

    plt.title("PageRank with " + str(d) + " Dampening") #check
    plt.axis('off')
    plt.tight_layout()
    plt.show()
```

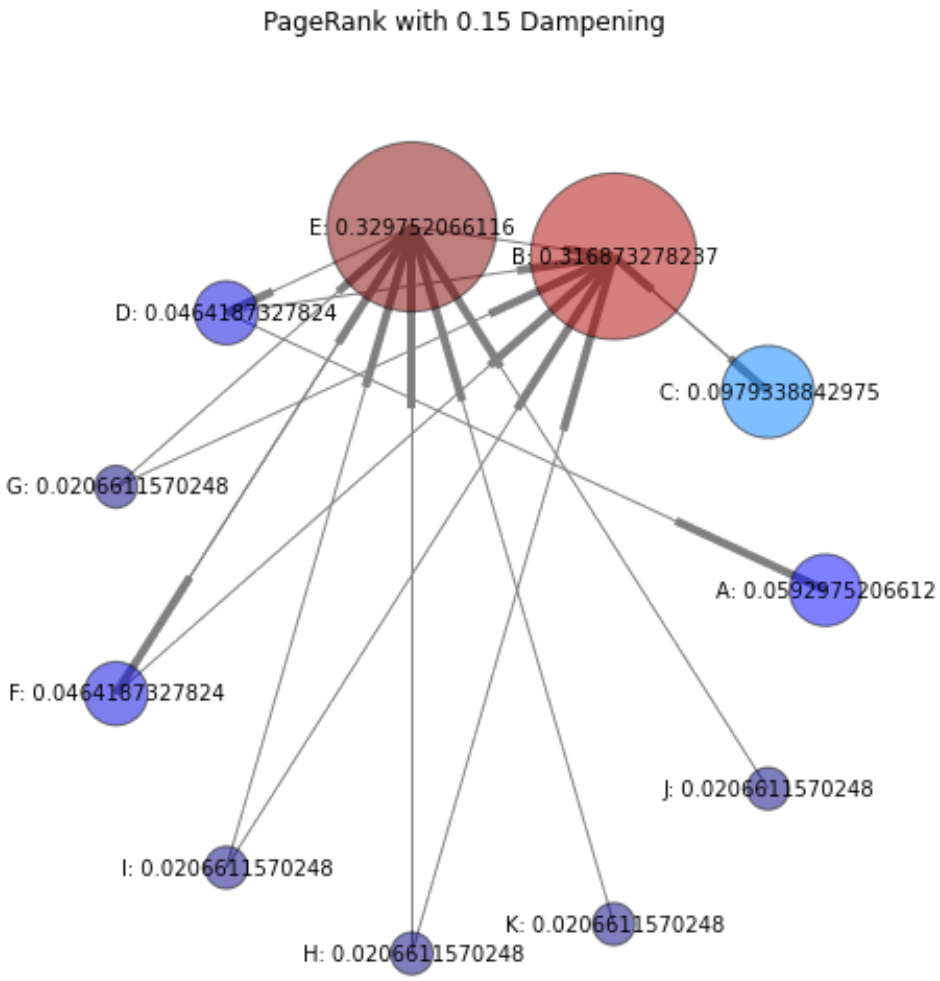
```
for damp in d:
    draw('output' + str(damp) + '.txt', damp)
```

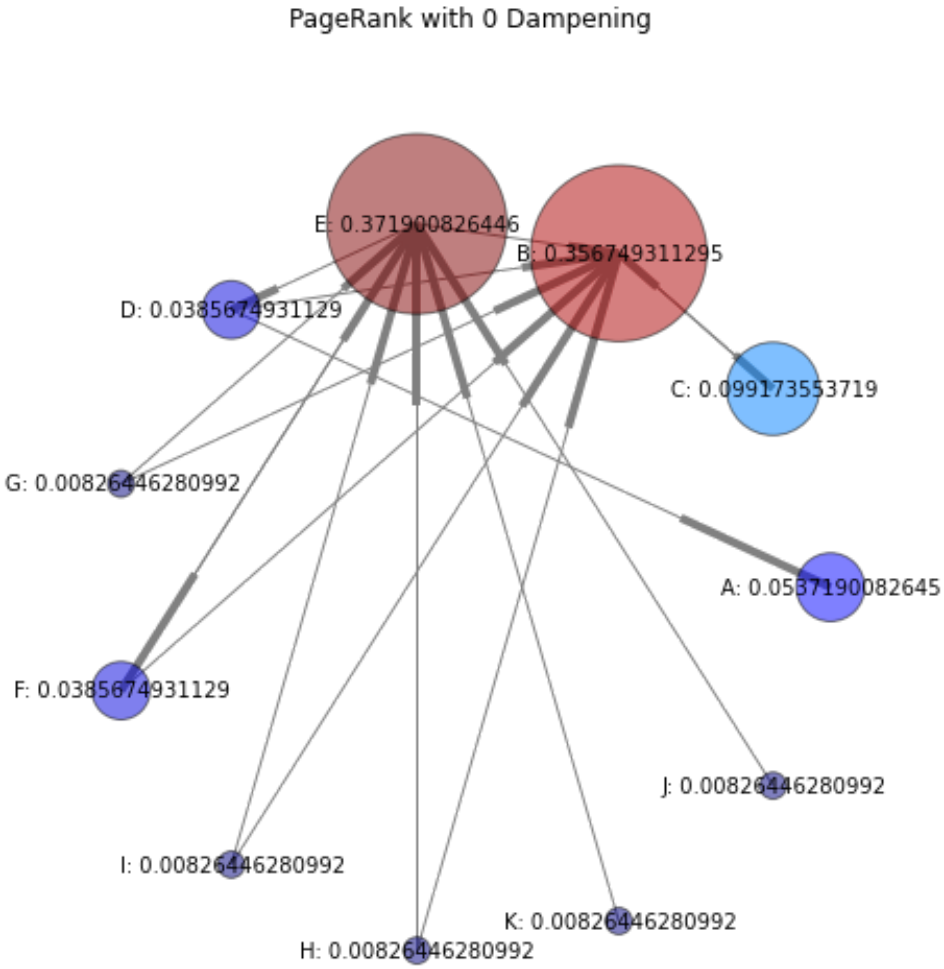

PageRank with 1 Dampening
E: 0.0909090909091
B: 0.0909090909091
D: 0.0909090909091
C: 0.0909090909091
G: 0.0909090909091
A: 0.0909090909091
F: 0.0909090909091
J: 0.0909090909091
I: 0.0909090909091
K: 0.0909090909091
H: 0.0909090909091


PageRank with 0.75 Dampening
E: 0.161157024793
B: 0.157369146006
D: 0.0778236914601
C: 0.0929752066116
G: 0.0702479338843
A: 0.0816115702479
F: 0.0778236914601
J: 0.0702479338843
I: 0.0702479338843
K: 0.0702479338843
H: 0.0702479338843


PageRank with 0.5 Dampening
E: 0.231404958678
B: 0.223829201102
D: 0.0647382920 11
C: 0.095041322314
G: 0.04958677685 95
A: 0.07231404958 68
F: 0.0647382920 11
J: 0.04958677685 95
I: 0.04958677685 95
K: 0.04958677685 95
H: 0.04958677685 95


PageRank with 0.25 Dampening
E: 0.301652892562
B: 0.290289256198
D: 0.051652892562
C: 0.0971074380165
G: 0.0289256198347
A: 0.0630165289256
F: 0.051652892562
J: 0.0289256198347
I: 0.0289256198347
K: 0.0289256198347
H: 0.0289256198347


PageRank with 0.15 Dampening
E: 0.329752066116
B: 0.316873278237
D: 0.0464187327824
C: 0.0979338842975
G: 0.0206611570248
A: 0.0592975206612
F: 0.0464187327824
J: 0.0206611570248
I: 0.0206611570248
K: 0.0206611570248
H: 0.0206611570248



HW 9.3: Applying PageRank to the Wikipedia hyperlinks network

Run your PageRank implementation on the Wikipedia dataset for 5 iterations, and display the top 100 ranked nodes (with alpha = 0.85).

Run your PageRank implementation on the Wikipedia dataset for 10 iterations, and display the top 100 ranked nodes (with teleportation factor of 0.15). Have the top 100 ranked pages changed? Comment on your findings. Plot the pagerank values for the top 100 pages resulting from the 5 iterations run. Then plot the pagerank values for the same 100 pages that resulted from the 10 iterations run.

In [3]:
```
%%writefile danglingpagerank.py

from mrjob.job import MRJob
from mrjob.step import MRStep

class DanglingPageRankJob(MRJob): #calculates the dangling mass, wh
ich will be fed as input to the second job

    def mapper(self, _, line):
        data = line.split('\t') #evaluate data
        nid = eval(data[0])
        data2 = eval(data[1])
        curPr = float(data2[0])
        neighbors = data2[1]

        if len(neighbors) == 0:
            yield '*', curPr

    def reducer(self, key, values): #accumulate dangling mass
        total = sum(values)
        yield 'mass', total

if __name__ == '__main__':
    DanglingPageRankJob.run()
```

```
Writing danglingpagerank.py
```

In [5]:
```
from danglingpagerank import DanglingPageRankJob

mr_job = DanglingPageRankJob(args = ['initData.txt'])

with mr_job.make_runner() as runner:
    runner.run()
    for line in runner.stream_output():
        print line
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:

"mass"  0.09090909090909091
```

In [3]:

```
%%writefile pageRank2.py

from mrjob.job import MRJob
from mrjob.step import MRStep
from operator import itemgetter

class PageRankMR2(MRJob):

    def configure_options(self):
        super(PageRankMR2, self).configure_options() #input options
        self.add_passthrough_option('--alpha', type = float, defaul
t = 0.15, help = 'Alpha/Teleportation')
        self.add_passthrough_option('--numNodes', type = int, defau
lt = 10, help = 'Number of Nodes')
        self.add_passthrough_option('--mass', type = float, default
= 0.0, help = 'Mass from Dangling Nodes')

    def mapper1(self, _, line):
        data = line.split('\t')
        nid = eval(data[0]) #evaluate data

        data2 = eval(data[1])
        curPr = float(data2[0])
        neighbors = data2[1]

        yield nid, neighbors

        if len(neighbors) == 0: #skip dangling nodes
            pass
        else:
            newPr = curPr/len(neighbors)
            for nid in neighbors.keys():
                yield nid, newPr #output new pagerank

    def reducer1(self, key, values):
        newPr = float(0)
        adj = {}

        for value in values:
            if type(value) == dict:
                adj = value
            else:
                newPr += value

        newMass = self.options.mass/self.options.numNodes #using ma
ss as input
        alpha = self.options.alpha

        finalPr = (alpha/float(self.options.numNodes)) + ((1-alpha)
* float(newMass + newPr)) #final pagerank

        yield key, (finalPr, adj)
```

```
    def steps(self):

        return [MRStep(mapper = self.mapper1, reducer = self.reduce
r1)]



if __name__ == '__main__':
    PageRankMR2.run()
```

```
Overwriting pageRank2.py
```

In [3]:
```
%%writefile topRank.py

from mrjob.job import MRJob
from mrjob.step import MRStep
from operator import itemgetter

class TopRankJob(MRJob): #get the top results for an input N

    def configure_options(self):
        super(TopRankJob, self).configure_options()
        self.add_passthrough_option('--N', type = int, default = 1
0, help = 'Number of top ranked to keep')

    def mapper_sort(self, _, line):
        data = line.split('\t') #evaluate input
        node = eval(data[0])

        data2 = eval(data[1])
        pageRank = data2[0]

        if pageRank < 0.00001: #ignore results with tiny probabilit
y
            pass
        else:
            yield None, (node, pageRank) #send all to same reducer


    def reducer_sort(self, _, values):
        sortedList = [] #not efficient but stored in memory as list
        for node, score in values:
            sortedList.append((node, score))
            sortedList = sorted(sortedList, key = itemgetter(1), re
verse = True)

            if len(sortedList) > self.options.N: #pop off result th
at is smallest
                sortedList.pop()

        for node, score in sortedList:
               yield node, score



    def steps(self):
        return [MRStep(mapper = self.mapper_sort, reducer = self.re
ducer_sort)]

if __name__ == '__main__':
    TopRankJob.run()
```

```
Overwriting topRank.py
```

In [14]:

```
from danglingpagerank import DanglingPageRankJob
from pageRank2 import PageRankMR2
from topRank import TopRankJob
import os

iteration = 1

while iteration <= 10: #driver for running locally
    if iteration == 1: #on first iteration
        curMass = 0.0

        mr_job_dangling = DanglingPageRankJob(args = ['initData.tx
t']) #dangling job

        with mr_job_dangling.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                result = mr_job_dangling.parse_output_line(line)
                curMass += result[1]

        mr_job = PageRankMR2(args = ['initData.txt', '--numNodes',
'11', '--mass', str(curMass)]) #main job

        with open('interResults' + str(iteration) + '.txt', 'w+') a
s myfile:
            with mr_job.make_runner() as runner:
                runner.run()
                for line in runner.stream_output():
                    myfile.write(line)


        iteration +=1
    else:
        curMass = 0.0 #repeat for other iterations

        mr_job_dangling = DanglingPageRankJob(args = ['interResult
s' + str(iteration - 1) + '.txt'])
        with mr_job_dangling.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                result = mr_job_dangling.parse_output_line(line)
                curMass += result[1]

        mr_job = PageRankMR2(args = ['interResults' + str(iteration
-1) + '.txt',
                                     '--numNodes', '11', '--mass',
str(curMass)])

        with open('interResults' + str(iteration) + '.txt', 'w+') a
s myfile:
            with mr_job.make_runner() as runner:
                runner.run()
                for line in runner.stream_output():
                    myfile.write(line)
```

```
        iteration += 1

mr_job_rank = TopRankJob(args = ['interResults' + str(iteration-1)
+ '.txt', '--N', '3'])
with mr_job_rank.make_runner() as runner:
    runner.run()
    for line in runner.stream_output():
        print mr_job_rank.parse_output_line(line)


for i in range(1,11):
    os.remove('interResults' + str(i) + '.txt')

#           mr_job_rank = TopRankJob(args = ['interResults' + str(i
teration) + '.txt', '--N', '3'])
#         with mr_job_rank.make_runner() as runner:
#             runner.run()
#             for line in runner.stream_output():
#                 print mr_job_rank.parse_output_line(line)
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
```

```
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
```

```
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
```

```
('B', 0.36323594898893996)
('C', 0.36288372803873453)
('E', 0.08114525762550133)
```

In [3]:
```
!python -m mrjob.tools.emr.create_job_flow '--conf-path' 'mrjob.con
f'
```

```
creating new scratch bucket mrjob-f3522a9f2dabe25e
using s3://mrjob-f3522a9f2dabe25e/tmp/ as our scratch dir on S3
Creating persistent job flow to run several jobs in...
creating tmp directory /var/folders/tx/cr7tg62d7rdd750f_czjczfm000
0gn/T/no_script.dunmireg.20160318.190133.469512
writing master bootstrap script to /var/folders/tx/cr7tg62d7rdd750
f_czjczfm0000gn/T/no_script.dunmireg.20160318.190133.469512/b.py
creating S3 bucket 'mrjob-f3522a9f2dabe25e' to use as scratch spac
e
Copying non-input files into s3://mrjob-f3522a9f2dabe25e/tmp/no_sc
ript.dunmireg.20160318.190133.469512/files/
Waiting 5.0s for S3 eventual consistency
Creating Elastic MapReduce job flow
Can't access IAM API, trying default instance profile: EMR_EC2_Def
aultRole
Can't access IAM API, trying default service role: EMR_DefaultRole
Job flow created with ID: j-3BZDNDD1NP99F
j-3BZDNDD1NP99F
```

In [4]:

```
from danglingpagerank import DanglingPageRankJob
from pageRank2 import PageRankMR2
from topRank import TopRankJob
import os

iteration = 1
clusterID = 'j-3BZDNDD1NP99F' #do same thing as above but on emr

while iteration <= 5:
    if iteration == 1: #first iteration
        curMass = 0.0 #dangling job
        mr_job_dangling = DanglingPageRankJob(args = ['s3://dunmire
g/HW9/wikiInput/', '-r', 'emr',
                                                      '--no-strict-p
rotocols',
                                                      '--emr-job-flo
w-id', clusterID])

        with mr_job_dangling.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                result = mr_job_dangling.parse_output_line(line)
                curMass += float(result[1])

        mr_job = PageRankMR2(args = ['s3://dunmireg/HW9/wikiInpu
t/', '--numNodes', '15192277', '--mass', str(curMass),
                                     '--no-strict-protocols', '-r',
'emr',
                                     '--emr-job-flow-id', clusterID,
                                     '--output-dir', 's3://dunmireg/
HW9/output' + str(iteration)])

        with mr_job.make_runner() as runner:
            runner.run()

        print 'Done Iteration 1'
        iteration +=1
    else:
        curMass = 0.0 #further iterations

        mr_job_dangling = DanglingPageRankJob(args = ['s3://dunmire
g/HW9/output' + str(iteration-1) + '/',
                                                       '-r', 'emr',
                                                      '--no-strict-p
rotocols',
                                                      '--emr-job-flo
w-id', clusterID])
        with mr_job_dangling.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                result = mr_job_dangling.parse_output_line(line)
                curMass += float(result[1])

        mr_job = PageRankMR2(args = ['s3://dunmireg/HW9/output' + s
```

```
tr(iteration - 1) + '/',
                                        '--numNodes', '15192277', '--m
ass', str(curMass),
                                       '-r', 'emr',
                                       '--emr-job-flow-id', clusterID,
                                       '--output-dir', 's3://dunmireg/
HW9/output' + str(iteration)])

        with mr_job.make_runner() as runner:
            runner.run()




        print 'Done with Iteration ' + str(iteration)
        iteration += 1

mr_job_rank = TopRankJob(args = ['s3://dunmireg/HW9/output' + str(i
teration-1) + '/', '--N', '100',
                                 '-r', 'emr',
                                 '--emr-job-flow-id', clusterID,
                                 '--output-dir', 's3://dunmireg/HW9/
outputDone'])
with open('5iterations.txt', 'w+') as myfile:
    with mr_job_rank.make_runner() as runner:
        runner.run()
        for line in runner.stream_output():
            myfile.write(line)
```

In [10]:
```
%%writefile join.py

from mrjob.job import MRJob

class JoinJob(MRJob):

    def mapper_init(self): #print results
        self.webIDs = {}
        with open('5iterations.txt', 'r') as myfile:
            lines = myfile.readlines()
            for line in lines:
                line = line.split('\t')
                node = eval(line[0])
                self.webIDs[node] = line[1].strip()
            print self.webIDs

    def mapper(self, _, line):
        line = line.strip().split('\t')
        if str(line[1]) in self.webIDs.keys():
            yield self.webIDs[str(line[1])], line[0]

if __name__ == "__main__":
    JoinJob.run()
```

```
Overwriting join.py
```

In [5]:
```
# from join import JoinJob

# mr_job = JoinJob(args = ['indices.txt', '--file', '5iterations.tx
t'])

# with mr_job.make_runner() as runner:
#     runner.run()
#     for line in runner.stream_output():
#         print mr_job.parse_output_line(line)
```

5 iterations

0.001461 United States
0.000683 Animal
0.000643 France
0.000576 Germany
0.000460 Arthropod
0.000458 List of sovereign states 0.000455 Insect
0.000446 Canada
0.000430 India
0.000429 United Kingdom
0.000422 England
0.000409 Iran
0.000382 World War II
0.000369 Poland
0.000350 village
0.000345 Countries of the world
0.000332 List of countries
0.000331 Japan
0.000328 Italy
0.000325 Australia
0.000321 Voivodeships of Poland
0.000314 Lepidoptera
0.000313 National Register of Historic Places 0.000311 Powiat
0.000305 Gmina
0.000280 London
0.000277 The New York Times
0.000268 English language
0.000263 China
0.000261 Russia
0.000260 Departments of France
0.000254 Communes of France
0.000253 New York City
0.000251 Spain
0.000250 moth
0.000248 Brazil
0.000241 Association football
0.000235 association football
0.000221 Counties of Iran
0.000220 Provinces of Iran
0.000220 California
0.000215 Romania
0.000215 Central European Time
0.000212 Bakhsh
0.000207 Rural Districts of Iran

0.000204 Sweden
0.000196 Netherlands
0.000193 Private Use Areas
0.000192 Iran Standard Time
0.000191 Central European Summer Time 0.000190 AllMusic
0.000189 World War I
0.000189 Mexico
0.000186 New York
0.000183 Iran Daylight Time
0.000180 Hangul
0.000173 gene
0.000172 Scotland
0.000169 Norway
0.000168 Allmusic
0.000166 Soviet Union
0.000163 Plant
0.000161 New Zealand
0.000160 Turkey
0.000159 Paris
0.000158 Geographic Names Information System 0.000155 Switzerland
0.000152 Los Angeles
0.000151 Romanize
0.000150 United States Census Bureau 0.000146 Europe
0.000145 Angiosperms
0.000144 Flowering plant
0.000142 South Africa
0.000141 census
0.000138 protein
0.000136 Austria
0.000135 U.S. state
0.000133 Chordate
0.000132 Political divisions of the United States 0.000132 Argentina
0.000131 population density
0.000126 Belgium
0.000126 Catholic Church
0.000125 BBC
0.000122 Chicago
0.000121 Pakistan
0.000117 Washington, D.C.
0.000116 Finland
0.000116 genus
0.000115 Czech Republic
0.000115 species
0.000114 Eastern European Time
0.000114 Ontario
0.000114 football (soccer)

0.000113 Philippines
0.000113 Denmark
0.000113 Eudicots
0.000113 Hungary
0.000113 Greece

10 Results

0.001461 United States
0.000666 Animal
0.000640 France
0.000575 Germany
0.000450 Arthropod
0.000447 Canada
0.000445 Insect
0.000444 List of sovereign states 0.000433 United Kingdom
0.000428 India
0.000423 England
0.000398 Iran
0.000385 World War II
0.000363 Poland
0.000344 village
0.000338 Countries of the world
0.000329 Japan
0.000329 Italy
0.000326 List of countries
0.000325 Australia
0.000313 Voivodeships of Poland
0.000310 National Register of Historic Places 0.000308 Lepidoptera
0.000304 Powiat
0.000298 Gmina
0.000286 The New York Times
0.000283 London
0.000269 English language
0.000264 China
0.000261 Russia
0.000258 New York City
0.000255 Departments of France
0.000251 Spain
0.000249 Communes of France
0.000245 moth
0.000245 Brazil
0.000239 Association football
0.000233 association football
0.000221 California
0.000215 Counties of Iran
0.000215 Provinces of Iran
0.000211 Central European Time
0.000211 Romania
0.000207 Bakhsh
0.000203 Sweden

0.000203 Rural Districts of Iran
0.000197 Netherlands
0.000191 Private Use Areas
0.000191 World War I
0.000188 Central European Summer Time 0.000188 New York
0.000187 Mexico
0.000187 Iran Standard Time
0.000185 AllMusic
0.000179 Iran Daylight Time
0.000178 Hangul
0.000173 Scotland
0.000170 gene
0.000168 Soviet Union
0.000167 Norway
0.000165 Allmusic
0.000161 Paris
0.000161 New Zealand
0.000159 Turkey
0.000158 Plant
0.000155 Geographic Names Information System 0.000155 Switzerland
0.000153 Los Angeles
0.000149 Romanize
0.000148 United States Census Bureau 0.000147 Europe
0.000142 Angiosperms
0.000141 South Africa
0.000139 census
0.000138 Flowering plant
0.000136 Austria
0.000135 protein
0.000135 U.S. state
0.000131 Argentina
0.000130 Political divisions of the United States 0.000130 population density
0.000128 Catholic Church
0.000128 Chordate
0.000127 BBC
0.000127 Belgium
0.000124 Chicago
0.000121 Washington, D.C.
0.000120 Pakistan
0.000116 Finland
0.000114 The Guardian
0.000114 Latin
0.000114 Ontario
0.000114 Czech Republic
0.000113 Philippines
0.000113 Denmark

0.000113 Greece
0.000113 genus
0.000112 football (soccer)
0.000112 Hungary
0.000112 Eastern European Time

Generally it seems the results stayed the same. Some scores did shift slightly

In [ ]:
```
#To run on AWS use the following command:
#change the .py file and input and output directories to match the
job
# python numberNodesMR.py -r emr \
# s3://ucb-mids-mls-networks/wikipedia/all-pages-indexed-out.txt \
# --conf-path mrjob.conf \
# --output-dir=s3://dunmireg/HW9/wiki \
# --no-output \
# --no-strict-protocol
```

15,192,277 Nodes

In [ ]:
```

```

In [ ]:
```
# python initPr.py -r emr \
# s3://ucb-mids-mls-networks/wikipedia/all-pages-indexed-out.txt \
# --numNodes 15192277 \
# --conf-path mrjob.conf \
# --output-dir=s3://dunmireg/HW9/wikiInput \
# --no-output
# --no-strict-protocol
```

HW 9.4: Topic-specific PageRank implementation using MRJob

Modify your PageRank implementation to produce a topic specific PageRank implementation, as described in:

http://www-cs-students.stanford.edu/~taherh/papers/topic-sensitive-pagerank.pdf (http://www-cs-students.stanford.edu/~taherh/papers/topic-sensitive-pagerank.pdf)

Note in this article that there is a special caveat to ensure that the transition matrix is irreducible. This caveat lies in footnote 3 on page 3:

```
A minor caveat: to ensure that M is irreducible when p
contains any 0 entries, nodes not reachable from nonzero
nodes in p should be removed. In practice this is not problematic.
```

and must be adhered to for convergence to be guaranteed.

Run topic specific PageRank on the following randomly generated network of 100 nodes:

s3://ucb-mids-mls-networks/randNet.txt (also available on Dropbox)

which are organized into ten topics, as described in the file:

s3://ucb-mids-mls-networks/randNet_topics.txt (also available on Dropbox)

Since there are 10 topics, your result should be 11 PageRank vectors (one for the vanilla PageRank implementation in 9.1, and one for each topic with the topic specific implementation). Print out the top ten ranking nodes and their topics for each of the 11 versions, and comment on your result. Assume a teleportation factor of 0.15 in all your analyses.

One final and important comment here: please consider the requirements for irreducibility with topic-specific PageRank. In particular, the literature ensures irreducibility by requiring that nodes not reachable from in-topic nodes be removed from the network.

This is not a small task, especially as it it must be performed separately for each of the (10) topics.

So, instead of using this method for irreducibility, please comment on why the literature's method is difficult to implement, and what what extra computation it will require. Then for your code, please use the alternative, non-uniform damping vector:

vji = beta*(1/|Tj|); if node i lies in topic Tj

vji = (1-beta)*(1/(N - |Tj|)); if node i lies outside of topic Tj

for beta in (0,1) close to 1.

With this approach, you will not have to delete any nodes. If beta > 0.5, PageRank is topic-sensitive, and if beta < 0.5, the PageRank is anti-topic-sensitive. For any value of beta irreducibility should hold, so please try beta=0.99, and perhaps some other values locally, on the smaller networks.

In [3]:
```
#Grab number of nodes
from numberNodesMR import NumberNodes

filename = 'randNet.txt'

mr_job = NumberNodes(args = [filename])

with mr_job.make_runner() as runner:
    runner.run()
    print "Number of nodes in " + filename
    for line in runner.stream_output():
        print mr_job.parse_output_line(line)[1]
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:

Number of nodes in randNet.txt
100
```

if the node is in the topic:

$\beta * \frac{1}{|Tj|}$

else

$(1-\beta) * (\frac{1}{N-|Tj|})$

In [6]:

```
%%writefile initTopic.py
from mrjob.job import MRJob
from mrjob.step import MRStep

class initTopicJob(MRJob): #initialize a file with neighbors dict a
nd weights and pagerank

    def configure_options(self):
        super(initTopicJob, self).configure_options() #add config o
ptions
        self.add_passthrough_option('--numNodes', type = float, def
ault = 100.0, help = 'Number of Nodes')
        self.add_passthrough_option('--topic', type = int, default
= 1, help = 'Topic')
        self.add_passthrough_option('--beta', type = float, default
= 0.99, help = 'Beta')



    def mapper(self, _, line):
        line = line.split('\t') #evaluate input
        node = line[0]
        adj = eval(line[1])
        for neighbor in adj.keys():
            yield neighbor, {}
        yield node, adj




    def reducer_init(self):
        self.nodeList = []
        self.topicNum = 0
        with open('randNet_topics.txt', 'r') as myfile: #use topic
as input file
            for line in myfile.readlines():
                data = line.split('\t')
                node = str(data[0])
                curTopic = int(data[1])
                if curTopic == self.options.topic:
                    self.nodeList.append(node)
                    self.topicNum += 1



    def reducer(self, key, values):
        nid = key
        adj = {}
        newValues = [value for value in values]
        for dictionary in newValues:
            if len(dictionary) != 0:
                adj = dictionary #neighbors

        PageRank = float(1)/self.options.numNodes
```

```
        weight = None

        if nid in self.nodeList:
            weight = self.options.beta/self.topicNum
        else:
            weight = (1 - self.options.beta) * (float(1)/(self.opti
ons.numNodes - self.topicNum)) #yield results

        yield nid, (PageRank, weight, adj)



if __name__ == "__main__":
    initTopicJob.run()
```

```
Overwriting initTopic.py
```

In [4]:
```
# from initTopic import initTopicJob

# mr_job = initTopicJob(args = ['randNet.txt', '--file', 'randNet_t
opics.txt',
#                               '--numNodes', '100', '--topic',
'1']) #use default beta

# with open('initTopicData.txt', 'w+') as myfile:
#     with mr_job.make_runner() as runner:
#         runner.run()
#         for line in runner.stream_output():
#             myfile.write(line)
```

In [34]:

```
%%writefile topicPageRank.py

from mrjob.job import MRJob
from mrjob.step import MRStep

class TopicPageRankMR(MRJob): #one iteration of pagerank using the
new weights

    #follows the same structure as above job
    def configure_options(self):
        super(TopicPageRankMR, self).configure_options()
        self.add_passthrough_option('--alpha', type = float, defaul
t = 0.15, help = "Alpha")
        self.add_passthrough_option('--numNodes', type = int, defau
lt = 10, help = 'Number Nodes')

    def mapper1(self, _, line):
        data = line.split('\t') #evaluate input
        nid = eval(data[0])
        data2 = eval(data[1])
        curPr = float(data2[0])
        weight = float(data2[1])
        neighbors = data2[2]

        yield nid, (weight, neighbors)

        if len(neighbors) == 0:
            yield '*', curPr #dangling node
        else:
            newPR = curPr/len(neighbors)
            for nid in neighbors.keys():
                yield nid, newPR

    def reducer_init(self):
        self.mass = 0

    def reducer1(self, key, values):
        if key == '*':
            for value in values:
                curPr = value
                self.mass += curPr #accumulate dangling mass
        else:
            newPr = float(0)
            weight = 0
            adj = {}

            for value in values:
                if type(value) == list:
                    weight = value[0]
                    adj = value[1]
                else:
                    newPr += value

            yield key, (newPr, weight, adj)
```

```
    def reducer_final(self):
        yield '*', self.mass

    def reducer2_init(self):
        self.mass = 0.0

    def reducer2(self, key, values):
        nid = key
        if nid == '*': #pass mass along here
            for value in values:
                self.mass += value
        else:
            valList = [value for value in values][0] #unpack values
            curPr = float(valList[0])
            weight = float(valList[1])
            neighbors = valList[2]

            newMass = self.mass/self.options.numNodes
            alpha = self.options.alpha

            newPr = (alpha*weight) + ((1-alpha) * float(newMass + c
urPr)) #new pagerank

            yield nid, (newPr, weight, neighbors)

    def steps(self):
        return [
            MRStep(mapper = self.mapper1,
                   reducer_init = self.reducer_init,
                   reducer = self.reducer1,
                   reducer_final = self.reducer_final),
            MRStep(reducer_init = self.reducer2_init, reducer = sel
f.reducer2)
        ]

if __name__ == "__main__":
    TopicPageRankMR.run()
```

```
Overwriting topicPageRank.py
```

In [ ]:

```
#driver for all 10
from initTopic import initTopicJob
from topicPageRank import TopicPageRankMR
import os

topics = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']

for topic in topics:
    mr_job_init = initTopicJob(args = ['randNet.txt', '--file', 'ra
ndNet_topics.txt',
                               '--numNodes', '100', '--topic', topi
c]) #use default beta

    with open('initTopicData.txt', 'w+') as myfile:
        with mr_job_init.make_runner() as runner:
            runner.run()
            for line in runner.stream_output():
                myfile.write(line)

    iteration = 0
    while iteration < 10:
        if iteration == 0:
            mr_job = TopicPageRankMR(args = ['initTopicData.txt',
'--numNodes', '100'])
            with open('result.txt', 'w+') as myfile:
                with mr_job.make_runner() as runner:
                    runner.run()
                    for line in runner.stream_output():
                        myfile.write(line)
        else:
            mr_job = TopicPageRankMR(args = ['result.txt', '--numNo
des', '100'])
            with open('interResults.txt', 'w+') as myfile:
                with mr_job.make_runner() as runner:
                    runner.run()
                    for line in runner.stream_output():
                        myfile.write(line)
            os.rename('interResults.txt', 'result.txt')

        iteration += 1

    print "For Topic " + str(topic)
    print '======================'
    #Format is node + [pagerank, weight, nodes]

    results = {}
    with open('result.txt', 'r') as myfile:
        for line in myfile.readlines():
            data = line.split('\t')
            node = eval(data[0])
            data2 = eval(data[1])
            pageRank = data2[0]
            results[node] = pageRank
    sortedDict = sorted(results.items(), key = lambda x:x[1], rever
```

```
se = True)
    topicsFile = {}
    with open('randNet_topics.txt', 'r') as myfile:
        for line in myfile.readlines():
            data = line.split('\t')
            topicsFile[data[0]] = data[1]


    for i in range(10):
        data = sortedDict[i]
        node = data[0]
        pageRank = data[1]
        curTopic = topicsFile[node]
        print 'Node ' + str(node) + ' with PageRank: ' + str(pageRa
nk) + ' is in topic ' + str(curTopic)

    print '\n'
```

Results reproduced below

For Topic 1

Node 32 with PageRank: 0.0206458983252 is in topic 1

Node 77 with PageRank: 0.0205475696268 is in topic 1

Node 52 with PageRank: 0.0197543131007 is in topic 1

Node 92 with PageRank: 0.0195292382463 is in topic 1

Node 10 with PageRank: 0.0185655254483 is in topic 1

Node 27 with PageRank: 0.0185225398271 is in topic 1

Node 85 with PageRank: 0.0178405105718 is in topic 7

Node 98 with PageRank: 0.0176923895084 is in topic 1

Node 46 with PageRank: 0.017514128675 is in topic 1

Node 74 with PageRank: 0.0160281213179 is in topic 10

For Topic 2

========== Node 58 with PageRank: 0.0308474600264 is in topic 2

Node 71 with PageRank: 0.029665243325 is in topic 2

Node 9 with PageRank: 0.0292968468928 is in topic 2

Node 73 with PageRank: 0.0289148054187 is in topic 2

Node 12 with PageRank: 0.026888935387 is in topic 2

Node 59 with PageRank: 0.0257996818877 is in topic 2

Node 75 with PageRank: 0.0248496005312 is in topic 2

Node 82 with PageRank: 0.0228582118913 is in topic 2

Node 52 with PageRank: 0.0163220985388 is in topic 1

Node 17 with PageRank: 0.0151586721985 is in topic 10

For Topic 3

========== Node 15 with PageRank: 0.0315290683973 is in topic 3

Node 70 with PageRank: 0.0270765976939 is in topic 3

Node 86 with PageRank: 0.0265279640443 is in topic 3

Node 91 with PageRank: 0.0244633164168 is in topic 3

Node 66 with PageRank: 0.0241485278847 is in topic 3

Node 2 with PageRank: 0.0237050872437 is in topic 3

Node 31 with PageRank: 0.0227671114115 is in topic 3

Node 40 with PageRank: 0.0221785011566 is in topic 3

Node 20 with PageRank: 0.0197450514244 is in topic 3

Node 74 with PageRank: 0.0158999729289 is in topic 10

For Topic 4

========== Node 63 with PageRank: 0.0262020432074 is in topic 4

Node 83 with PageRank: 0.0217600796334 is in topic 4

Node 65 with PageRank: 0.0206237797771 is in topic 4

Node 78 with PageRank: 0.0202101011522 is in topic 4

Node 41 with PageRank: 0.0199084597505 is in topic 4

Node 84 with PageRank: 0.0195198800713 is in topic 4

Node 79 with PageRank: 0.0184286505659 is in topic 4

Node 38 with PageRank: 0.0175154293668 is in topic 4

Node 15 with PageRank: 0.0167521635394 is in topic 3

Node 72 with PageRank: 0.0166947294992 is in topic 4

For Topic 5

========== Node 99 with PageRank: 0.0289632665003 is in topic 5

Node 90 with PageRank: 0.0283449846715 is in topic 5

Node 88 with PageRank: 0.0271687032551 is in topic 5

Node 51 with PageRank: 0.0268307953328 is in topic 5

Node 45 with PageRank: 0.0255533084779 is in topic 5

Node 5 with PageRank: 0.0239199550282 is in topic 5

Node 34 with PageRank: 0.023909744585 is in topic 5

Node 4 with PageRank: 0.0233632743825 is in topic 5

Node 80 with PageRank: 0.0228396208641 is in topic 5

Node 100 with PageRank: 0.0167414852697 is in topic 8

For Topic 6

========== Node 13 with PageRank: 0.0345715708775 is in topic 6

Node 56 with PageRank: 0.0328539067145 is in topic 6

Node 37 with PageRank: 0.0317771558764 is in topic 6

Node 11 with PageRank: 0.0313396481985 is in topic 6

Node 69 with PageRank: 0.0301207963449 is in topic 6

Node 23 with PageRank: 0.028348902674 is in topic 6

Node 15 with PageRank: 0.017242580581 is in topic 3

Node 85 with PageRank: 0.0169894383441 is in topic 7

Node 52 with PageRank: 0.0166021696311 is in topic 1

Node 74 with PageRank: 0.0154610589784 is in topic 10

For Topic 7

========== Node 85 with PageRank: 0.0267942274953 is in topic 7

Node 25 with PageRank: 0.0266087786135 is in topic 7

Node 28 with PageRank: 0.0248272454578 is in topic 7

Node 53 with PageRank: 0.0247676541944 is in topic 7

Node 35 with PageRank: 0.0242008286788 is in topic 7

Node 97 with PageRank: 0.0233950139793 is in topic 7

Node 47 with PageRank: 0.0228650136657 is in topic 7

Node 55 with PageRank: 0.022560854064 is in topic 7

Node 30 with PageRank: 0.022134839131 is in topic 7

Node 50 with PageRank: 0.0200890801962 is in topic 7

For Topic 8

========== Node 100 with PageRank: 0.0328660304609 is in topic 8

Node 61 with PageRank: 0.0278585708224 is in topic 8

Node 39 with PageRank: 0.0271955215054 is in topic 8

Node 8 with PageRank: 0.0271530997436 is in topic 8

Node 62 with PageRank: 0.0253467935637 is in topic 8

Node 87 with PageRank: 0.0252984160248 is in topic 8

Node 6 with PageRank: 0.0235068526543 is in topic 8

Node 54 with PageRank: 0.022891738249 is in topic 8

Node 18 with PageRank: 0.020622660246 is in topic 8

Node 9 with PageRank: 0.0153797996337 is in topic 2

For Topic 9

========== Node 94 with PageRank: 0.0301988899341 is in topic 9

Node 14 with PageRank: 0.0294951976842 is in topic 9

Node 42 with PageRank: 0.0291978626818 is in topic 9

Node 21 with PageRank: 0.0283995691114 is in topic 9

Node 57 with PageRank: 0.0274617939811 is in topic 9

Node 96 with PageRank: 0.0262609496793 is in topic 9

Node 24 with PageRank: 0.0257730743595 is in topic 9

Node 63 with PageRank: 0.0171614920669 is in topic 4

Node 61 with PageRank: 0.0163773288809 is in topic 8

Node 74 with PageRank: 0.0142777129756 is in topic 10

For Topic 10

Node 74 with PageRank: 0.0263318320147 is in topic 10

Node 17 with PageRank: 0.0235905361122 is in topic 10

Node 49 with PageRank: 0.0235742677288 is in topic 10

Node 95 with PageRank: 0.0206293404961 is in topic 10

Node 7 with PageRank: 0.0199138051716 is in topic 10

Node 43 with PageRank: 0.0193641749279 is in topic 10

Node 68 with PageRank: 0.0190382718656 is in topic 10

Node 48 with PageRank: 0.0190050456807 is in topic 10

Node 1 with PageRank: 0.0189980838546 is in topic 10

Node 3 with PageRank: 0.0186395560408 is in topic 10

In [3]:
```
# from topicPageRank import TopicPageRankMR

# mr_job = TopicPageRankMR(args = ['initTopicData.txt', '--numNode
s', '100'])

# with mr_job.make_runner() as runner:
#     runner.run()
#     for line in runner.stream_output():
#         print mr_job.parse_output_line(line)
```

In [16]:
```
!python driver.py --emr --iterations=10
```

```
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
```

```
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
```

```
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
WARNING:mrjob.runner:
WARNING:mrjob.runner:PLEASE NOTE: Starting in mrjob v0.5.0, protoc
ols will be strict by default. It's recommended you run your job w
ith --strict-protocols or set up mrjob.conf as described at http
s://pythonhosted.org/mrjob/whats-new.html#ready-for-strict-protoco
ls
WARNING:mrjob.runner:
```

```
For Topic 1
======================
Node 32 with PageRank: 0.0206458983252 is in topic 1

Node 77 with PageRank: 0.0205475696268 is in topic 1

Node 52 with PageRank: 0.0197543131007 is in topic 1

Node 92 with PageRank: 0.0195292382463 is in topic 1

Node 10 with PageRank: 0.0185655254483 is in topic 1

Node 27 with PageRank: 0.0185225398271 is in topic 1

Node 85 with PageRank: 0.0178405105718 is in topic 7

Node 98 with PageRank: 0.0176923895084 is in topic 1

Node 46 with PageRank: 0.017514128675 is in topic 1

Node 74 with PageRank: 0.0160281213179 is in topic 10


For Topic 2
======================
Node 58 with PageRank: 0.0308474600264 is in topic 2

Node 71 with PageRank: 0.029665243325 is in topic 2

Node 9 with PageRank: 0.0292968468928 is in topic 2

Node 73 with PageRank: 0.0289148054187 is in topic 2

Node 12 with PageRank: 0.026888935387 is in topic 2

Node 59 with PageRank: 0.0257996818877 is in topic 2

Node 75 with PageRank: 0.0248496005312 is in topic 2

Node 82 with PageRank: 0.0228582118913 is in topic 2

Node 52 with PageRank: 0.0163220985388 is in topic 1

Node 17 with PageRank: 0.0151586721985 is in topic 10
```

Question 9.4 - Version 2

Modify your PageRank implementation to produce a topic specific PageRank implementation, as described in:

http://www-cs-students.stanford.edu/~taherh/papers/topic-sensitive-pagerank.pdf (http://www-cs-students.stanford.edu/~taherh/papers/topic-sensitive-pagerank.pdf)

Note in this article that there is a special caveat to ensure that the transition matrix is irreducible. This caveat lies in footnote 3 on page 3:

> A minor caveat: to ensure that M is irreducible when p contains any 0 entries, nodes not reachable from nonzero nodes in p should be removed. In practice this is not problematic.

and must be adhered to for convergence to be guaranteed. Run topic specific PageRank on the following randomly generated network of 100 nodes:

s3://ucb-mids-mls-networks/randNet.txt (also available on Dropbox)

which are organized into ten topics, as described in the file:

s3://ucb-mids-mls-networks/randNet_topics.txt (also available on Dropbox)

*Since there are 10 topics, your result should be 11 PageRank vectors (one for the vanilla PageRank implementation in 9.1, and one for each topic with the topic specific implementation). Print out the top ten ranking nodes and their topics for each of the 11 versions, and comment on your result. Assume a teleportation factor of 0.15 in all your analyses.

Solution:

To accomplish this, we need to modify our driver and PageRank algorithm. The key changes are the addition of passthrough options to control the topic check and topic iteration. Most of the changes occurr in the `PageRank` class, with minor changes in the driver.

Below we show both files modified.

Driver

Here we insert arguments to handle the topic-file used to assign topics to each node, and iteration control for each topic.

In [2]:

```
%%writefile topic_driver.py
from __future__ import division

from mrjob.job import MRJob
from mrjob.step import MRStep
from mrjob.emr import EMRJobRunner

from topic_sensitive_pr import pageRank
from number_of_nodes import numNodes

import cPickle as pickle
from collections import defaultdict
from operator import itemgetter
import argparse

# Storage files
s3Bucket = 's3://ucb-mids-mls-networks/wikipedia/all-pages-indexed-
out.txt'


def getName(obj, namespace):
        return [name for name in namespace if namespace[name] is ob
j]


def extractValues(job, runner):
        output = defaultdict(int)
        for line in runner.stream_output():
                key, value = job.parse_output_line(line)
                output[key] = value

        return output


def dumpToFile(variable, filename):
        with open(filename, 'w') as f:
                pickle.dump(variable, f)


def runJob(method, args, emr=False):
        job = method(args=args)

        methodName = getName(method, globals())[0]
        print '\n\t' + 'Running ' + methodName + '...'

        with job.make_runner() as runner:

                runner.run()
                result = extractValues(job, runner)

                print '\t' + 'Complete: ' + methodName

                return result
```

```
if __name__ == '__main__':

        # Arguments
        parser = argparse.ArgumentParser(description='Driver for Pa
geRank in MRJob.')

        parser.add_argument('--emr', default=None, action='store_tr
ue',
                                                                help='Flag
for using the EMR (and S3 bucket).')

        parser.add_argument('--iterations', default=10,
                                                                help='Numbe
r of iterations to use for PageRank.')

        parser.add_argument('--file', default='PageRank-test.txt',
                                                                help='File
to be passed to the PageRank class.')

        parser.add_argument('--topicFile', default='randNet_topics.
txt',
                                                                help='File
containing topic-labels for each node.')

        args = parser.parse_args()

        # Pass
        if args.emr:
                argInput = [s3Bucket, '-r', 'emr']

        else:
                argInput = [args.file]


        # Get node counts
        totalNodeTuple = runJob(numNodes, args=argInput)
        totalNodes = totalNodeTuple.values()[0]

        print 'NUMNODES: %s' % (totalNodes)

        # Execute

        for i in range(10):

                topNodes = runJob(pageRank, args=argInput + ['--num
berOfNodes=%s' % (totalNodes)] + \

                                                        ['--iterations=%s'
% (args.iterations)] + \

                                                        ['--topicFile=%s' %
(args.topicFile)] + \
```

```
                                                    ['--currentTopic=%
s' % (i + 1)] \

                                                    )

                nodeTuples = [(key, round(value, 5)) for (key, valu
e) in topNodes.iteritems()]
                sortedNodes = sorted(nodeTuples, key=itemgetter(1),
reverse=True)

                # Emit

                print 'TOPIC:%s' % (i + 1)

                for k, v in sortedNodes:
                        print 'ID: %s \t PR: %s' % (k, v)
```

```
Writing topic_driver.py
```

Topic-Sensitive PageRank

The main addition here is the conditional weight calculation that is used to distinguish nodes from different topics. The weight calculation is the same one as described in the problem statement of Question 9.4.

In [1]:

```
%%writefile topic_sensitive_pr.py
from __future__ import division

from mrjob.job import MRJob
from mrjob.step import MRStep

from collections import defaultdict
from operator import itemgetter

class pageRank(MRJob):

        """ This class implements the page-rank calculation. """

        def configure_options(self):

                """ Load options for the class. """

                super(pageRank, self).configure_options()

                self.add_passthrough_option('--alpha',
                        default=0.85, type=float, help='alpha: Damp
ening factor for teleportation in PageRank')

                self.add_passthrough_option('--iterations',
                        default=10, type=int, help='iterations: num
ber of iterations for PageRank')

                self.add_passthrough_option('--manualPower',
                        default=7, type=int, help='manualPower: ord
er of magnitude for number of nodes.')

                self.add_passthrough_option('--numberOfNodes',
                        default=None, type=int, help='numberOfNode
s: The number of nodes in your graph. Used for teleporation.')

                self.add_file_option('--topicFile',
                        default=None, type=str, help='topicFile: Fi
le containing the topic information for each node.')

                self.add_passthrough_option('--currentTopic',
                        default=None, type=str, help='currentTopic:
The current topic for the given PageRank iteration.')

                self.add_passthrough_option('--emittedNumber',
                        default=10, type=int, help='emittedNumber:
Top N nodes to emit sorted from the PageRank job.')


        def load_options(self, args):

                """ Initializes the arguments for each class. """

                super(pageRank, self).load_options(args)
                self.alpha = self.options.alpha
```

```
                self.iterations = self.options.iterations

                # Check number of nodes
                if self.options.numberOfNodes:
                        self.numberOfNodes = self.options.numberOfN
odes
                else:
                        self.numberOfNodes = pow(10, self.options.m
anualPower)

                # Check topic file
                if self.options.topicFile:
                        self.topicFile = self.options.topicFile
                else:
                        self.option_parser.error('Please supply a t
opic file containing node labels.')

                # Check current topic
                if self.options.currentTopic:
                        self.currentTopic = self.options.currentTop
ic

                else:
                        self.option_parser.error('Please supply a c
urrent topic of focus for PageRank.')


                # Load topic file
                self.topicListing = defaultdict(str)
                with open(self.topicFile, 'r') as f:
                        for line in f.readlines():

                                # Insert
                                node, topic = line.split()
                                self.topicListing[node] = topic

                # Topic sizes
                self.topicAmounts = defaultdict(int)
                for topic in self.topicListing.values():
                        self.topicAmounts[topic] += 1


                # Misc
                self.emittedNumber = self.options.emittedNumber


        def mapper_init_pr(self, _, line):

                """ This initializes the PageRank algorithm by asse
mbling the node list
                for the initial PageRank values. """

                # Parse
                line = line.split('\t')
```

```
                node = line[0]
                adjacencyList = eval(line[1])

                # Track
                for neighbor in adjacencyList.keys():

                        # Emit raw nodes
                        yield neighbor, None


                # Pass values
                yield node, adjacencyList


        def reducer_init_pr(self, node, initTuple):

                """ This attaches initial PageRanks for the algorit
hm. """

                adjacencyList = dict()

                # Re-discover
                for element in initTuple:
                        if isinstance(element, dict):
                                adjacencyList = element

                # Initialize PR
                PageRank = float(1) / float(self.numberOfNodes)

                # Emit
                yield node, (adjacencyList, PageRank)


        def mapper_iterate_pr(self, node, nodeTuple):

                """ This projects all of the PageRank weights for e
ach node's neighbor. """

                adjacencyList, PageRank = nodeTuple

                if not adjacencyList:
                        pass

                else:

                        # Emit PR
                        for neighbor in adjacencyList.keys():
                                yield neighbor, PageRank / len(adja
cencyList)

                # Emit structure
                yield node, adjacencyList
```

```
        def reducer_iterate_pr(self, node, PRNodeObject):

                """ This reconstructs the graph structure form the
updated PageRanks. """

                updatedPR = 0

                # Combine PR
                for value in PRNodeObject:
                        if isinstance(value, dict):
                                adjacencyList = value

                        else:
                                updatedPR += value

                # Custom weights
                nodeTopic = self.topicListing[node]
                currentTopicQuantity = self.topicAmounts[self.curre
ntTopic]

                if nodeTopic == self.currentTopic:
                        weight = ((1 - self.alpha) / currentTopicQu
antity)

                else:
                        weight = self.alpha / (self.numberOfNodes -
currentTopicQuantity)


                # Update
                updatedPR = (1 - self.alpha) * weight + self.alpha
* updatedPR

                # Emit
                yield node, (adjacencyList, updatedPR)


        def mapper_sort(self, node, nodeTuple):

                """ Emits the page rank for each node. """

                adjacencyList, PageRank = nodeTuple

                yield None, (node, PageRank)


        def reducer_sort(self, _, PageRankPair):

                """ Keeps the top N PageRank values. """

                sortedList = []

                # Iterate and remove
                for node, score in PageRankPair:
```

```
                        sortedList.append((node, score))
                        sortedList = sorted(sortedList, key=itemget
ter(1), reverse=True)

                        if len(sortedList) > self.emittedNumber:
                                sortedList.pop()

                # Emit
                for node, score in sortedList:
                        yield node, score


        def steps(self):

                """ Determines the steps for the job. Has two phase
s- initiate PR and iterate. """

                initializeStep = [

                        MRStep(mapper=self.mapper_init_pr,
                                        reducer=self.reducer_init_p
r)

                ]

                iterateStep = [

                        MRStep(mapper=self.mapper_iterate_pr,
                                        reducer=self.reducer_iterat
e_pr)

                ]

                sortStep = [

                        MRStep(mapper=self.mapper_sort,
                                        reducer=self.reducer_sort)

                ]

                return initializeStep + iterateStep * self.iteratio
ns + sortStep


if __name__ == '__main__':
                        pageRank().run()
```

```
Overwriting topic_sensitive_pr.py
```

Execution

Here we run our topic-sensitive PageRank with additional arguments passed to the driver for topic loading and management.

In [3]:
```
!python topic_driver.py --file=randNet.txt --topicFile=randNet_topi
cs.txt
```

```
        Running numNodes...
        Complete: numNodes
NUMNODES: 100

        Running pageRank...
        Complete: pageRank
TOPIC:1
ID: 15   PR: 0.01642
ID: 74   PR: 0.01597
ID: 63   PR: 0.01586
ID: 100          PR: 0.01543
ID: 85   PR: 0.01511
ID: 9    PR: 0.01503
ID: 58   PR: 0.01483
ID: 71   PR: 0.01449
ID: 61   PR: 0.01446
ID: 52   PR: 0.01418

        Running pageRank...
        Complete: pageRank
TOPIC:2
ID: 15   PR: 0.01615
ID: 9    PR: 0.01613
ID: 58   PR: 0.01606
ID: 74   PR: 0.01585
ID: 63   PR: 0.01569
ID: 71   PR: 0.01566
ID: 100          PR: 0.01533
ID: 85   PR: 0.01495
ID: 52   PR: 0.01447
ID: 61   PR: 0.01416

        Running pageRank...
        Complete: pageRank
TOPIC:3
ID: 15   PR: 0.01737
ID: 74   PR: 0.01596
ID: 63   PR: 0.01563
ID: 100          PR: 0.01534
ID: 9    PR: 0.01508
ID: 85   PR: 0.01507
ID: 58   PR: 0.01472
ID: 61   PR: 0.01432
ID: 71   PR: 0.01431
ID: 52   PR: 0.01429

        Running pageRank...
        Complete: pageRank
TOPIC:4
ID: 15   PR: 0.01637
ID: 63   PR: 0.01601
ID: 74   PR: 0.0159
ID: 100          PR: 0.01533
ID: 85   PR: 0.01516
```

```
ID: 9    PR: 0.01499
ID: 58   PR: 0.01477
ID: 71   PR: 0.01453
ID: 61   PR: 0.01432
ID: 52   PR: 0.01424

        Running pageRank...
        Complete: pageRank
TOPIC:5
ID: 15   PR: 0.01636
ID: 74   PR: 0.01586
ID: 63   PR: 0.01568
ID: 100          PR: 0.01547
ID: 58   PR: 0.01495
ID: 85   PR: 0.01492
ID: 9    PR: 0.01476
ID: 71   PR: 0.01446
ID: 61   PR: 0.01436
ID: 52   PR: 0.01416

        Running pageRank...
        Complete: pageRank
TOPIC:6
ID: 15   PR: 0.01644
ID: 74   PR: 0.01592
ID: 63   PR: 0.01558
ID: 85   PR: 0.01535
ID: 13   PR: 0.01525
ID: 100          PR: 0.01502
ID: 9    PR: 0.01486
ID: 58   PR: 0.01467
ID: 52   PR: 0.01453
ID: 71   PR: 0.01423

        Running pageRank...
        Complete: pageRank
TOPIC:7
ID: 15   PR: 0.01633
ID: 74   PR: 0.01592
ID: 85   PR: 0.01583
ID: 63   PR: 0.01566
ID: 100          PR: 0.01518
ID: 9    PR: 0.01494
ID: 58   PR: 0.01484
ID: 71   PR: 0.01438
ID: 61   PR: 0.01436
ID: 52   PR: 0.01421

        Running pageRank...
        Complete: pageRank
TOPIC:8
ID: 100          PR: 0.01654
ID: 15   PR: 0.01606
ID: 74   PR: 0.01571
```

```
ID: 63   PR: 0.01561
ID: 61   PR: 0.0153
ID: 9    PR: 0.01506
ID: 85   PR: 0.01493
ID: 58   PR: 0.01473
ID: 52   PR: 0.01428
ID: 71   PR: 0.01424

        Running pageRank...
        Complete: pageRank
TOPIC:9
ID: 15   PR: 0.01612
ID: 63   PR: 0.01589
ID: 74   PR: 0.01582
ID: 100          PR: 0.0152
ID: 9    PR: 0.01487
ID: 85   PR: 0.01483
ID: 61   PR: 0.01458
ID: 58   PR: 0.01447
ID: 71   PR: 0.01422
ID: 52   PR: 0.01421

        Running pageRank...
        Complete: pageRank
TOPIC:10
ID: 74   PR: 0.01633
ID: 15   PR: 0.01631
ID: 63   PR: 0.01581
ID: 100          PR: 0.01531
ID: 85   PR: 0.0151
ID: 9    PR: 0.01495
ID: 58   PR: 0.01471
ID: 61   PR: 0.01445
ID: 71   PR: 0.01443
ID: 52   PR: 0.01422

No handlers could be found for logger "mrjob.runner"
```

Vanilla Implementation

Below is the implementation for normal PageRank on `randNet.txt`.

In [4]:
```
!python driver.py --file=randNet.txt
```

```
        Running numNodes...
        Complete: numNodes
NUMNODES: 100

        Running pageRank...
        Complete: pageRank
ID: 15   PR: 0.01636
ID: 74   PR: 0.01597
ID: 63   PR: 0.01577
ID: 100          PR: 0.01538
ID: 85   PR: 0.01518
ID: 9    PR: 0.01503
ID: 58   PR: 0.01483
ID: 71   PR: 0.01449
ID: 61   PR: 0.01441
ID: 52   PR: 0.01431
ID: 77   PR: 0.01366
ID: 92   PR: 0.01365
ID: 32   PR: 0.01331
ID: 13   PR: 0.01318
ID: 88   PR: 0.01314
ID: 17   PR: 0.01307
ID: 70   PR: 0.01307
ID: 25   PR: 0.01296
ID: 90   PR: 0.01286
ID: 49   PR: 0.01255
ID: 53   PR: 0.01221
ID: 39   PR: 0.01208
ID: 51   PR: 0.01179
ID: 73   PR: 0.01164
ID: 45   PR: 0.0116
ID: 99   PR: 0.01154
ID: 28   PR: 0.01151
ID: 35   PR: 0.0115
ID: 56   PR: 0.0114
ID: 55   PR: 0.01113
ID: 27   PR: 0.01112
ID: 10   PR: 0.01112
ID: 94   PR: 0.01111
ID: 41   PR: 0.01109
ID: 95   PR: 0.01108
ID: 91   PR: 0.01103
ID: 65   PR: 0.01085
ID: 86   PR: 0.0107
ID: 84   PR: 0.01059
ID: 62   PR: 0.01056
ID: 46   PR: 0.01054
ID: 2    PR: 0.01033
ID: 78   PR: 0.01027
ID: 97   PR: 0.01019
ID: 83   PR: 0.01017
ID: 8    PR: 0.01007
ID: 43   PR: 0.01005
ID: 14   PR: 0.00986
```

```
ID: 21   PR: 0.00973
ID: 12   PR: 0.00968
ID: 6    PR: 0.00966
ID: 98   PR: 0.00952
ID: 42   PR: 0.00939
ID: 11   PR: 0.00934
ID: 37   PR: 0.00921
ID: 68   PR: 0.00921
ID: 57   PR: 0.00917
ID: 54   PR: 0.00915
ID: 80   PR: 0.00913
ID: 31   PR: 0.0091
ID: 67   PR: 0.00906
ID: 34   PR: 0.00903
ID: 4    PR: 0.00902
ID: 30   PR: 0.00895
ID: 7    PR: 0.00892
ID: 44   PR: 0.00886
ID: 66   PR: 0.0087
ID: 75   PR: 0.00868
ID: 87   PR: 0.00862
ID: 29   PR: 0.00839
ID: 40   PR: 0.00837
ID: 3    PR: 0.00829
ID: 72   PR: 0.00819
ID: 47   PR: 0.00803
ID: 59   PR: 0.00802
ID: 48   PR: 0.00795
ID: 79   PR: 0.00792
ID: 26   PR: 0.00788
ID: 1    PR: 0.00787
ID: 69   PR: 0.00783
ID: 81   PR: 0.00777
ID: 16   PR: 0.00768
ID: 33   PR: 0.00766
ID: 38   PR: 0.00743
ID: 24   PR: 0.00736
ID: 89   PR: 0.00722
ID: 64   PR: 0.00707
ID: 50   PR: 0.00689
ID: 5    PR: 0.00682
ID: 93   PR: 0.00674
ID: 18   PR: 0.00643
ID: 36   PR: 0.00603
ID: 23   PR: 0.00597
ID: 96   PR: 0.00597
ID: 76   PR: 0.00578
ID: 60   PR: 0.00567
ID: 19   PR: 0.00553
ID: 22   PR: 0.00521
ID: 20   PR: 0.00506
ID: 82   PR: 0.00456

No handlers could be found for logger "mrjob.runner"
```

In [ ]: