MapReduce and PageRank

Question 1:

Suppose our input data to a map-reduce operation consists of integer values (the keys are not important). The map function takes an integer *i* and produces the list of pairs (*p*,*i*) such that *p* is a prime divisor of *i*. For example, map(12) = [(2,12),(3,12)].

The reduce function is addition. That is, reduce(p,[i_1,i_2,...,i_k]) is (p,i_1+i_2+...+i_k).

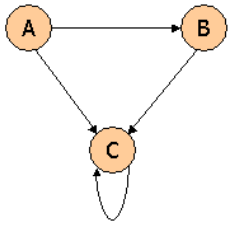
Compute the output, if the input is the set of integers 15, 21, 24, 30, 49.

Sol:

prime no:2,3,5,7,11,.....
15:[3,15],[5,15]
21:[3,21],[7,21]
24:[2,24],[3,24]
30:[2,30],[3,30],[5,30]
49:[7,49]
by combining all common elements part i.e compare left element and add rightmost element of that to get the solution.
[2,(24+30)],[3,(15+21+24+30)],[5,(15+30)],[7,(21+49)]

Question 2:

Consider three Web pages with the following links:



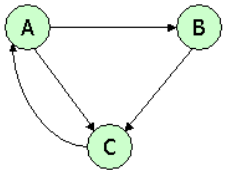
Suppose we compute PageRank with a β of 0.7, and we introduce the additional constraint that the sum of the PageRanks of the three pages must be 3, to handle the problem that otherwise any multiple of a solution will also be a solution. Compute the PageRanks *a*, *b*, and *c* of the three pages A, B, and C, respectively.

Solution:

```
import numpy as np

 Adjacency matrix
 m1 = [ 0,  0, 0]
      [0.5, 0, 0]
      [0.5, 1, 1]

m1 = np.matrix([[0, 0, 0],[0.5, 0, 0],[0.5, 1, 1]])
beta = 0.7

r = beta * m1 * r + ((1-beta)/N)
def r_p(r):
  return beta * m1 * r + np.matrix([0.1,0.1,0.1]).T

r = np.matrix([1.0/3,1.0/3,1.0/3]).T

for i in range(1000):
  r = r_p(r)
print "Final PageRank: \n" + str(r*3)
```

```
Final PageRank:

[[ 0.3  ]
 [ 0.405]
 [ 2.295]]
```

Question 3:



Suppose we compute PageRank with β=0.85. Write the equations for the PageRanks *a*, *b*, and *c* of the three pages A, B, and C, respectively.

Solution:

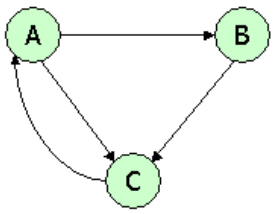
import numpy as np

```
# Adjacency matrix
# m2 = [ 0,  0, 1]
#      [0.5, 0, 0]
#      [0.5, 1, 0]

m2 = np.matrix([[0, 0, 1],[0.5, 0, 0],[0.5, 1, 0]])
beta =0.85

def r_p(r):
  return beta * m2 * r + np.matrix([0.05,0.05,0.05]).T
```

```
r = np.matrix([1.0/3,1.0/3,1.0/3]).T

for i in range(1000):
  r = r_p(r)
print "Final PageRank: \n" + str(r)
```

```
Final PageRank:

[[ 0.38778971]
 [ 0.21481063]
 [ 0.39739966]]
```

Question 4:



Assuming no "taxation," compute the PageRanks *a*, *b*, and *c* of the three pages A, B, and C,

using iteration, starting with the "0th" iteration where all three pages have rank $a = b = c = 1$. Compute as far as the 5th iteration, and also determine what the PageRanks are in the limit.

Solution:

```
import numpy as np


# Adjacency matrix
# m3 = [ 0,  0, 1]
#      [0.5, 0, 0]
#      [0.5, 1, 0]

m3 = np.matrix([[0, 0, 1],[0.5, 0, 0],[0.5, 1, 0]])
beta = 1
r = np.matrix([1,1,1]).T

for i in range(50):
  r = m3.dot(r)
  print i+1
  print "Final PageRank: \n" + str(r)
```

Final PageRank:

```
[[ 1.20000002]
 [ 0.59999999]
 [ 1.19999999]]
```