CS 6240 – Parallel Data Processing with Map Reduce
Section-01, HW-3, Biyanta Vipulbhai Shah

Design Discussion

Page-Rank algorithm pseudo code:

Using alpha = 0.15

```
map (key, value)
        delta = value retrieved from counter
        totalPages = value retrieved from counter

        for each line in values
                extract value of pageName, pageRank and out-links list using appropriate
                delimiters
                if pageRank = -1.0
                // -1.0 is default page rank for all page ranks set after pre processing. For 1st
                iteration
                pageRank = 1/totalPages

                if delta != -1.0
                //-1.0 is the default value of delta, which is when delta is not set
                // delta is set, meaning dangling nodes mass found.
                pageRank += (1-alpha) * (delta/totalPages)

                object = {out-links list, pageRank}
                emit (pageName, object)

                if out-links list = empty
                        emit ("dummy", pageRank)
                else
                        for each link in out-link list
                                p = pageRank / |out-link list|
                                emit (link, p)

reduce (key, <Iterable> value)
        totalPages = value retrieved from counter
        s, danglingMass = 0.0
        object = NULL;

        if key = "dummy"
                for each value in values
                        danglingMass += pageRank
                        set global counter, delta value, to danglingMass

        else
```

```
        for each value in values
        if !(isPage(value))
                s += value // adding the page rank to the running sum
        else
                // recovering the graph structure
                extract value of pageName, pageRank and out-links list using appropriate
                delimiters
                object = {out-link list, pageRank}


object.pageRank = (alpha/totalPages) + (1-alpha) * (s)

emit (key, object)
```

Pre-processing approach is similar to the one Professor is expecting. While adding a link to the out-links list, I have not added duplicate links and self referencing loops of the pages inside the out-links list.

For the page rank algorithm, there were three approaches mentioned for calculating the **δ**.

Solution 1: Asks to add a job which computes **δ** and then pass the newly computed **δ** as a parameter to the modified MapReduce program to update page ranks. Thus we will have an extra MR job for each iteration. At the end of 10 iterations, we will have 10 extra jobs run. This is a lot of overhead for huge data.

Solution 3: Uses order inversion. This sends all the old values of page ranks of all dangling nodes to each reducer. This increases the data transfer between map and reduce.

Solution 2: Merges computation of **δ** into the previous reduce phase. Here one reducer receives old page ranks of all the dangling nodes. It computes **δ** in i iteration which will be used i+1 iteration in the map, which updates the page ranks for the i th iteration. Thus we need an extra map task to get the updated correct value of page ranks for the last iteration. So **compared to Solution 1**, we use significantly less number of map reduce jobs. And also **compared to Solution 3**, instead of sending the page ranks of dangling nodes to all reducers, we send it to only one reducer thus significantly reducing the data transfer between map and reduce.

So **Solution 2**: Merging the computation of **δ** into the previous reduce phase and using it in the next Map phase, seems like the **most optimal solution,** out of the three**,** for calculating the dangling mass, and thus I have used that.

For RUN-1 (1 master 5 workers)

Iteration Number	Data from Mapper to Reducer in bytes	Data transferred from Reducer to HDFS(S3) in bytes
1	1531918140	1184546407
2	1999614658	1184549026
3	2001050731	1184539018
4	2000874020	1184530349
5	2000660267	1184537770
6	2001669253	1184527641
7	2001566300	1184529930
8	2001253870	1184528505
9	2001293465	1184530189
10	2001412978	1184531689
11	775075904	1186289573

For RUN-2 (1 master 10 workers)

Iteration Number	Data from Mapper to Reducer in bytes	Data transferred from Reducer to HDFS(S3) in bytes
1	1568417520	1184530251
2	2044719223	1184528769
3	2045556504	1184520205
4	2046116665	1184520999
5	2046373632	1184515655
6	2046060507	1184517602
7	2046304220	1184506684
8	2045893002	1184509141
9	2046301105	1184516206
10	2046349009	1184511689
11	788003532	1186268376

Does the amount of data transferred in each iteration change over time?

By enlarge the amount of data transferred from Mappers to Reducers remains **almost** the same across several iterations.
The data transfer from Reducers to S3 (S3, in my case), shows **almost** the same number of bytes transferred in each iteration. This is because the total number of pages are remaining the same, so the data transfer also remains of the same amount. This situation is the same for both the runs.
The small number of bytes that are changing in each iteration are due to the fact that we are updating the page ranks in each iteration.

Top-k algorithm pseudo code:

For Top-k algorithm I have used the same approach as used by the professor in Module 5: Basic Algorithms.

Pseudo Code from the Module

```
Class Mapper {
 localTopK

 setup() {
  initialize localTopK
 }

 map(…, x) {
  if (x is in localTopK)
   // Adding x also evicts the now
   // (k+1)-st record from localTopK
   localTopK.add( x )
 }

 cleanup() {
  for each x in localTopK
   emit( dummy, x )
 }
}
```

```
reduce(dummy, [x1, x2,…]) {

 initialize globalTopK

 for each record x in input list
  if (x is in globalTopK)
   // Adding x also evicts the now
   // (k+1)-st record from globalTopK
   globalTopK.add(x)

 for each record x in globalTopK
  emit(NULL, x)
}
```

Performance Comparison

Run Number	Pre-Processing Time	Time to calculate Page Rank	Time to run Top-k
Run 1	2237 seconds	1579 seconds	52 seconds
Run 2	1291 seconds	956 seconds	40 seconds

Critically evaluate the runtime results by comparing them against what you had expected to see

I had expected to see a much less running time for Run-2 than for Run-1 and the above results confirm the same. The time is almost halved from Run-1 to Run-2, and this is because there are double the number of machines in Run-2, thus the work gets divided over more machines, making the overall running time less.

Which of the computation phases showed a good speedup?

From the above observation, there is a very good speedup from Run-1 to Run-2 for the pre-processing phase. This is because the pre processing task is distributed over 10 workers in Run-2, while we have 5 workers in Run-1, thus Run-1 takes more time.

However, we can see that during the calculation of page rank, the speed up is not that good. From the syslog we can see that,
For Run-1: Shuffled Maps =162
For Run-2: Shuffled Maps =361.
The number of shuffled maps is almost double in Run-2, thus during the page rank calculation, it loses on time to shuffle the maps almost double times. This double shuffled maps are due to the fact that there are double the number of machines in Run-2 from Run-1. Thus there is speed-up but not as huge as it is in the pre-processing phase.

For the last phase; Time to run Top-k there is very less speedup because the number of reducers is 1. Thus the only parallel processing that is done is in the map phase.

Output of the simple Wikipedia data set on local machine (standalone mode)

PAGE NAMES	PAGE RANK
United_States_09d4	0.005189009
Wikimedia_Commons_7b57	0.004806766
Country	0.003940285
England	0.002752481
Water	2.69E-03
Animal	2.55E-03
City	2.51E-03
United_Kingdom_5ad7	2.36E-03
Germany	2.35E-03
Earth	2.32E-03
France	2.32E-03
Europe	2.04E-03
Wiktionary	1.75E-03
English_language	1.75E-03
Government	1.73E-03
Computer	1.72E-03
India	1.71E-03
Money	1.67E-03
Japan	1.55E-03
Plant	1.52E-03
Italy	1.51E-03
Canada	1.48E-03
Spain	1.47E-03
Food	1.42E-03
Human	1.41E-03
China	1.40E-03
People	1.38E-03
Australia	1.33E-03
Asia	1.28E-03
Capital_(city)	1.27E-03
Television	1.26E-03
Sun	1.26E-03
Number	1.24E-03
State	1.24E-03
Sound	1.24E-03
Science	1.23E-03
Mathematics	1.23E-03
Metal	1.19E-03
Year	1.18E-03
2004	1.17E-03
Language	1.15E-03
Russia	1.15E-03
Wikipedia	1.12E-03

Religion	1.10E-03
19th_century	1.10E-03
Music	1.09E-03
Scotland	1.05E-03
20th_century	1.05E-03
Greece	1.05E-03
Latin	1.03E-03
London	1.03E-03
Greek_language	1.00E-03
Energy	9.99E-04
World	9.86E-04
Centuries	9.76E-04
Culture	9.45E-04
History	9.36E-04
Liquid	9.15E-04
Netherlands	9.06E-04
Planet	9.05E-04
Light	9.02E-04
Society	9.01E-04
Atom	8.90E-04
Wikimedia_Foundation_83d9	8.88E-04
Scientist	8.88E-04
Image	8.88E-04
Law	8.86E-04
Geography	8.79E-04
List_of_decades	8.79E-04
Uniform_Resource_Locator_1b4e	8.62E-04
Africa	8.61E-04
Turkey	8.45E-04
Inhabitant	8.30E-04
Capital_city	8.23E-04
Plural	8.22E-04
Electricity	8.14E-04
Poland	7.97E-04
Building	7.97E-04
Car	7.95E-04
Sweden	7.92E-04
Book	7.91E-04
Biology	7.87E-04
War	7.71E-04
Chemical_element	7.68E-04
God	7.61E-04
North_America_e7c4	7.56E-04
September_7	7.55E-04
Website	7.46E-04
Nation	7.43E-04

Politics	7.40E-04
2006	7.33E-04
Fish	7.32E-04
Species	7.31E-04
Mammal	7.22E-04
Island	7.18E-04
Portugal	7.17E-04
Gas	7.16E-04
River	7.12E-04
Switzerland	7.06E-04
World_War_II_d045	7.02E-04

Output of the full Wikipedia data set for Run-1

PAGE NAMES	PAGE RANKS
United_States_09d4	0.002622934
2006	0.001228507
United_Kingdom_5ad7	0.001203149
Biography	9.82E-04
2005	9.17E-04
England	8.80E-04
Canada	8.56E-04
Geographic_coordinate_system	7.72E-04
France	7.25E-04
2004	7.20E-04
Australia	6.80E-04
Germany	6.54E-04
2003	5.87E-04
India	5.83E-04
Japan	5.83E-04
Internet_Movie_Database_7ea7	5.34E-04
Europe	5.09E-04
Record_label	4.91E-04
2001	4.87E-04
2002	4.83E-04
World_War_II_d045	4.78E-04
Population_density	4.70E-04
Music_genre	4.67E-04
2000	4.65E-04
Italy	4.46E-04
Wiktionary	4.36E-04
Wikimedia_Commons_7b57	4.35E-04
London	4.35E-04
English_language	4.18E-04
1999	4.06E-04

Spain	3.63E-04
1998	3.56E-04
Russia	3.44E-04
1997	3.37E-04
Television	3.36E-04
New_York_City_1428	3.35E-04
Football_(soccer)	3.26E-04
1996	3.24E-04
Census	3.24E-04
Scotland	3.22E-04
1995	3.10E-04
China	3.09E-04
Population	3.04E-04
Square_mile	3.04E-04
Scientific_classification	3.04E-04
California	3.02E-04
1994	2.91E-04
Sweden	2.88E-04
Public_domain	2.87E-04
Film	2.86E-04
Record_producer	2.84E-04
New_Zealand_2311	2.83E-04
New_York_3da4	2.79E-04
Netherlands	2.77E-04
Marriage	2.76E-04
1993	2.75E-04
United_States_Census_Bureau_2c85	2.75E-04
1991	2.72E-04
1990	2.68E-04
1992	2.66E-04
Politician	2.65E-04
Album	2.61E-04
Latin	2.60E-04
Actor	2.58E-04
Ireland	2.58E-04
Per_capita_income	2.56E-04
Studio_album	2.52E-04
Poverty_line	2.51E-04
Km^2	2.50E-04
1989	2.47E-04
Norway	2.41E-04
Website	2.39E-04
1980	2.35E-04
Animal	2.29E-04
Area	2.29E-04
1986	2.27E-04

Personal_name	2.26E-04
Poland	2.26E-04
Brazil	2.26E-04
1985	2.24E-04
1987	2.23E-04
1983	2.22E-04
1982	2.21E-04
1981	2.19E-04
French_language	2.19E-04
1979	2.19E-04
1984	2.19E-04
World_War_I_9429	2.19E-04
1988	2.19E-04
Paris	2.18E-04
1974	2.18E-04
Mexico	2.16E-04
19th_century	2.12E-04
1970	2.11E-04
January_1	2.11E-04
USA_f75d	2.11E-04
1975	2.09E-04
1976	2.08E-04
Africa	2.08E-04
South_Africa_1287	2.07E-04

Output of the full Wikipedia data set for Run-2

PAGE NAMES	PAGE RANKS
United_States_09d4	0.002622937
2006	0.001228494
United_Kingdom_5ad7	0.001203143
Biography	9.82E-04
2005	9.17E-04
England	8.80E-04
Canada	8.56E-04
Geographic_coordinate_system	7.72E-04
France	7.25E-04
2004	7.20E-04
Australia	6.80E-04
Germany	6.54E-04
2003	5.87E-04
India	5.83E-04
Japan	5.83E-04
Internet_Movie_Database_7ea7	5.34E-04
Europe	5.09E-04
Record_label	4.91E-04
2001	4.87E-04
2002	4.83E-04
World_War_II_d045	4.78E-04
Population_density	4.70E-04
Music_genre	4.67E-04
2000	4.65E-04
Italy	4.46E-04
Wiktionary	4.36E-04
Wikimedia_Commons_7b57	4.35E-04
London	4.35E-04
English_language	4.18E-04
1999	4.06E-04
Spain	3.63E-04
1998	3.56E-04
Russia	3.44E-04
1997	3.37E-04
Television	3.36E-04
New_York_City_1428	3.35E-04
Football_(soccer)	3.26E-04
1996	3.24E-04
Census	3.24E-04
Scotland	3.22E-04
1995	3.10E-04

China	3.09E-04
Population	3.04E-04
Square_mile	3.04E-04
Scientific_classification	3.04E-04
California	3.02E-04
1994	2.91E-04
Sweden	2.88E-04
Public_domain	2.87E-04
Film	2.86E-04
Record_producer	2.84E-04
New_Zealand_2311	2.83E-04
New_York_3da4	2.79E-04
Netherlands	2.77E-04
Marriage	2.76E-04
1993	2.75E-04
United_States_Census_Bureau_2c85	2.75E-04
1991	2.72E-04
1990	2.68E-04
1992	2.66E-04
Politician	2.65E-04
Album	2.61E-04
Latin	2.60E-04
Actor	2.58E-04
Ireland	2.58E-04
Per_capita_income	2.56E-04
Studio_album	2.52E-04
Poverty_line	2.51E-04
Km^2	2.50E-04
1989	2.47E-04
Norway	2.41E-04
Website	2.39E-04
1980	2.35E-04
Animal	2.29E-04
Area	2.29E-04
1986	2.27E-04
Personal_name	2.26E-04
Poland	2.26E-04
Brazil	2.26E-04
1985	2.24E-04
1987	2.23E-04
1983	2.22E-04
1982	2.21E-04
French_language	2.19E-04
1981	2.19E-04
1979	2.19E-04
1984	2.19E-04

World_War_I_9429	2.19E-04
1988	2.19E-04
Paris	2.18E-04
1974	2.18E-04
Mexico	2.16E-04
19th_century	2.12E-04
1970	2.11E-04
January_1	2.11E-04
USA_f75d	2.11E-04
1975	2.09E-04
1976	2.08E-04
Africa	2.08E-04
South_Africa_1287	2.07E-04

Do they seem reasonable based on your intuition about important information on Wikipedia?

The top 100 page ranks have Wikimedia and Wiktionary. These are 2 non profit organizations which have links to one another, which is technically a self link. This is not a good representation about page rank. Since in page rank is the contribution to the amount of in-links and out-links.