CS-6240 HW-3 Report

Ajay Baban Kauthale(ajayk@ccs.neu.edu**)**

Page Rank Source Code

Source code is in "2. Source Code" directory

Pseudo Codes

Following constants are declared in PageRank class which will be referred in pseudo codes
pageCnt – to store page count
MAX_ITR – to store iteration maximum (i.e. 10)
Input – input
danglingNodeScore – to store dangling node score
pageRankConf – to store page rank configuration
TOP_K – to store topK results max (i.e. 100)

1. Pre-processing Job

```
public static void preprocessBZ2() {
        // create job by setting parser input, output, mapper class and another configuration
        Job job;
        // set number of reducers to single
        job.setNumReduceTasks(1);
        // run the job and wait for completion
        boolean ok = job.waitForCompletion(true);
        Counters counters = job.getCounters();
        // update the page count with counter
        pageCnt = counters.findCounter(PageRankCounter.PAGE_COUNTER).getValue();
}
```

Following class is used for parser mapping

```
class ParserMapper {
        // This reads records from BZ2 input files and emits records in following format
        // Z#A~B~C#PR_VALUE
        // where A,B,C are outlinks(adjacency list) of NODE Z
        // and, PR_VALUE is page rank value of the NODE Z
}
```

2. PageRank Job

```
public static void getPageRank() {
        // set iteration, page count and dangling node score in page rank configuration
        // create job by setting input, output, mapper class, reducer class and another configuration
        Job job;
        // run the job and wait for completion
        boolean ok = job.waitForCompletion(true);
```

```
        Counters counters = job.getCounters();
        // update the page count with counter and dangling node score
        pageCnt = counters.findCounter(PageRankCounter.PAGE_COUNTER).getValue();
        danglingNodeScore =
        counters.findCounter(PageRankCounter.DANGLING_NODE_SCORE).getValue();
}
```

Following class is used for page rank mapping

```
class PageRankMapper {
        // constants for page count and iteration count
        Long cnt;
        Integer currentItr;

        // set the iteration and page count to -1 in setup() method

        // in map method following logic will occur
        If line is empty then
                Return
        Else
                // read the record which is in format as follows
                // Z#A~B~C#PR_VALUE, where A,B,C are outlinks(adjacency list) of NODE Z
                // PARSE records according to the format, to process
                String[] tokens = line.toString().split("#");
                String node = tokens[0];
                Double pageRankValue = 0.0;

                If current iteration is 0 then
                        pageRankValue = Double.valueOf(1.0/cnt);
                Else
                        pageRankValue = Double.parseDouble(tokens[2]);

                If current node is dangling node then
                        // add global counter and emit adjacency list
                Else
                        // distribute page rank score to outlinks
                        // emit adjacency list
}
```

Following class is used for reducing page rank

```
class PageRankReducer {
        // declare constants for node list, alpha (i.e. 0.15), dangling score and page count
        // set node list as empty, page count and dangling score to -1 in setup() method

        // in reduce method following logic will appear
        For each node in node list do
                If page rank is present
```

```
                    // Add page rank into total
            Else
                    // Update adjacency list

    // distribute the dangling node score to all nodes in current iteration
    // calculate the page rank score
    Double pagerRankScore = (ALPHA/pageCount)+((1-ALPHA) * (danglingScoreDistribution +
prSummation));

    // emit the record
}
```

3. Top-k Job

```
private static void topKResults() {

    // create job by setting input, output, mapper class, reducer class, partitioner and another
configuration
    Job job;
    // run the job and wait for completion
    boolean ok = job.waitForCompletion(true);
    // it will create page rank output for each iteration
}
```

Following mapper class is used for top k mapping

```
class TopKMapper {

    private Map<String, Double> resultMap;
    // set constants for top k results to -1 and result map to empty in setup() method
    // following will occur in map method
    If line is empty then
            Return
    Else
            // split the tokens and add page rank and node to map

    // clean up and emit top k records

}
```

Following reducer class is used for top k reducing

```
class TopKReduce {

            // declare constants for top k and list to store top records
            // set constants for top k results to -1 and top records to empty in setup() method
            // following will occur in reduce method
            For each node in node list do
                    If record size > top_k
```

// add record to top records

// emit the top k records

}

// PageRankPartitioner class is used to make sure all data goes to single reducer

Amount of data transferred in each iteration

FOR 6 MACHINE CLUSTER

Iteration	Map-Reduce	Reduce-HDFS
1	Physical memory (bytes) snapshot=79847440384 Virtual memory (bytes) snapshot=355155296256 Total committed heap usage (bytes)=70757384192	HDFS: Number of bytes read=11130 HDFS: Number of bytes written=0
2	Physical memory (bytes) snapshot=25889951744 Virtual memory (bytes) snapshot=107990814720 Total committed heap usage (bytes)=24924127232	HDFS: Number of bytes read=2200 HDFS: Number of bytes written=0
3	Physical memory (bytes) snapshot=29552386048 Virtual memory (bytes) snapshot=131154235392 Total committed heap usage (bytes)=28151644160	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
4	Physical memory (bytes) snapshot=29483466752 Virtual memory (bytes) snapshot=131105878016 Total committed heap usage (bytes)=28138536960	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
5	Physical memory (bytes) snapshot=30099316736 Virtual memory (bytes) snapshot=131098529792 Total committed heap usage (bytes)=28352970752	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
6	Physical memory (bytes) snapshot=29410942976	HDFS: Number of bytes read=3186

	Virtual memory (bytes) snapshot=131112423424 Total committed heap usage (bytes)=27615821824	HDFS: Number of bytes written=0
7	Physical memory (bytes) snapshot=29125591040 Virtual memory (bytes) snapshot=131137265664 Total committed heap usage (bytes)=26850885632	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
8	Physical memory (bytes) snapshot=29344120832 Virtual memory (bytes) snapshot=131158695936 Total committed heap usage (bytes)=27843887104	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
9	Physical memory (bytes) snapshot=29074612224 Virtual memory (bytes) snapshot=131124305920 Total committed heap usage (bytes)=27523022848	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0
10	Physical memory (bytes) snapshot=29202677760 Virtual memory (bytes) snapshot=131148656640 Total committed heap usage (bytes)=27968667648	HDFS: Number of bytes read=3186 HDFS: Number of bytes written=0

FOR 11 MACHINE CLUSTER

Iteration	Map-Reduce	Reduce-HDFS
1	Physical memory (bytes) snapshot=78523531264 Virtual memory (bytes) snapshot=355146932224 Total committed heap usage (bytes)=69931106304	HDFS: Number of bytes read=11130 HDFS: Number of bytes written=0
2	Physical memory (bytes) snapshot=28170113024 Virtual memory (bytes) snapshot=154586431488 Total committed heap usage (bytes)=25731006464	HDFS: Number of bytes read=2200 HDFS: Number of bytes written=0
3	Physical memory (bytes) snapshot=35062804480 Virtual memory (bytes) snapshot=187625955328	HDFS: Number of bytes read=3540 HDFS: Number of bytes written=0

	Total committed heap usage (bytes)=32781107200	
4	Physical memory (bytes) snapshot=35359961088 Virtual memory (bytes) snapshot=190946529280 Total committed heap usage (bytes)=32493797376	HDFS: Number of bytes read=3658 HDFS: Number of bytes written=0
5	Physical memory (bytes) snapshot=34897113088 Virtual memory (bytes) snapshot=190910894080 Total committed heap usage (bytes)=32555663360	HDFS: Number of bytes read=3658 HDFS: Number of bytes written=0
6	Physical memory (bytes) snapshot=34931912704 Virtual memory (bytes) snapshot=194222399488 Total committed heap usage (bytes)=32392085504	HDFS: Number of bytes read=3776 HDFS: Number of bytes written=0
7	Physical memory (bytes) snapshot=35872489472 Virtual memory (bytes) snapshot=194281512960 Total committed heap usage (bytes)=33107738624	HDFS: Number of bytes read=3776 HDFS: Number of bytes written=0
8	Physical memory (bytes) snapshot=34630238208 Virtual memory (bytes) snapshot=187585650688 Total committed heap usage (bytes)=32252100608	HDFS: Number of bytes read=3540 HDFS: Number of bytes written=0
9	Physical memory (bytes) snapshot=36213600256 Virtual memory (bytes) snapshot=194184953856 Total committed heap usage (bytes)=33873199104	HDFS: Number of bytes read=3776 HDFS: Number of bytes written=0
10	Physical memory (bytes) snapshot=36126380032 Virtual memory (bytes) snapshot=194288349184 Total committed heap usage (bytes)=33356775424	HDFS: Number of bytes read=3776 HDFS: Number of bytes written=0

Performance Comparison

Cluster	pre-processing time	time to run ten iterations of PageRank	time to find the top-100 pages
6 m4.large machines	7 min	38 min	6 min
11 m4.large machines	7 min	22 min 4 seconds	4 min

I am expecting same processing time since because it must be independent on number if machines since we do not use reducers which is right.

Also, time to run 10 iterations also makes sense since 11 machine cluster time will be always less than 6 machine cluster, but I was expecting time difference around 50% less for 11 machine cluster since cluster machines are doubled which is not the case.

Finally, for getting top 100 pages time for 11 machine cluster should be less than 6 machine cluster which is correct. Here the difference is almost 50% less for 11 machine cluster since cluster machines are doubled.

TOP-100 results

Text	Page Rank
United_States_09d4	0.002725073055432421
2006	0.002432593815902133
United_Kingdom_5ad7	0.001295372744688698
2005	0.0011215494484298663
Biography	9.006728485005461E-4
Canada	8.482882915933882E-4
England	8.420820677562086E-4
France	8.294077921272027E-4
2004	7.813425933921996E-4
Germany	7.140083532555729E-4
Australia	6.939463879269735E-4
Geographic_coordinate_system	6.8281278386201E-4
2003	6.291001688126065E-4
India	6.111654158356218E-4
Japan	6.053072469068819E-4
Italy	5.058401890589968E-4
2001	5.057526534193177E-4
2002	4.994863624653983E-4
Internet_Movie_Database_7ea7	4.979774245551695E-4
Europe	4.8090590644820605E-4
2000	4.7331983743565797E-4
World_War_II_d045	4.5502283344317376E-4
London	4.3993925806785455E-4
Population_density	4.2606997039945E-4

Record_label	4.2350311760535974E-4
1999	4.1800655636824787E-4
Spain	4.137678437249756E-4
English_language	4.1275059187188573E-4
Russia	3.902095471666167E-4
Race_(United_States_Census)_a07d	3.891881053812668E-4
Wiktionary	3.8105105579855196E-4
Wikimedia_Commons_7b57	3.622857660118733E-4
1998	3.614495433467641E-4
Music_genre	3.563166171058277E-4
1997	3.446638072426295E-4
Scotland	3.3938408522067267E-4
New_York_City_1428	3.3864976229991346E-4
Football_(soccer)	3.330810392335861E-4
1996	3.233988408245644E-4
Television	3.196644378091154E-4
Sweden	3.1910775278292747E-4
Census	3.0852889408941093E-4
Square_mile	3.083252289179598E-4
1995	3.04560303594175E-4
California	3.028756510434319E-4
China	2.9739901860715385E-4
New_Zealand_2311	2.931710382783961E-4
Netherlands	2.931129410479939E-4
1994	2.906589827004597E-4
1991	2.7699831465680976E-4
1993	2.748298052717016E-4
1990	2.729399206307405E-4
New_York_3da4	2.7197233254795276E-4
Public_domain	2.715217477820071E-4
1992	2.633009205764747E-4
United_States_Census_Bureau_2c85	2.626662878633281E-4
Film	2.626536923179618E-4
Actor	2.610887349527015E-4
Scientific_classification	2.609331384423768E-4
Norway	2.5721240695977734E-4
Ireland	2.555200765950505E-4
Population	2.544739065364726E-4
Poland	2.5318999642128117E-4
1989	2.467214736721913E-4
Marriage	2.4137327569661205E-4
1980	2.4076014975157458E-4
Brazil	2.395133949117868E-4
January_1	2.3936704024449438E-4
Mexico	2.3796577967292432E-4
Politician	2.379619311001671E-4

Latin	2.3593743939022424E-4
1986	2.3432254811386348E-4
1985	2.2869239195500414E-4
1979	2.2807678362136267E-4
Per_capita_income	2.2796458536376564E-4
1982	2.2771274908216772E-4
Album	2.2767532954732548E-4
1981	2.2740596120582977E-4
French_language	2.262839287392105E-4
1974	2.251159571606207E-4
Switzerland	2.2376580961382448E-4
Record_producer	2.2370177227434157E-4
1984	2.234269821034726E-4
1987	2.2337798415320422E-4
South_Africa_1287	2.233230736137018E-4
1983	2.2311339454318148E-4
1970	2.190503141533552E-4
1988	2.1821606566035728E-4
1976	2.1661266739194125E-4
Km²	2.1660091456871894E-4
1975	2.140788304289986E-4
Paris	2.1142068894674143E-4
Personal_name	2.1124827734192238E-4
1969	2.1102910371337318E-4
Greece	2.1100766643904133E-4
1972	2.0973882864217373E-4
1945	2.0952025907032844E-4
Poverty_line	2.0824998875631012E-4
1977	2.0817950201098813E-4
1978	2.0726661679128689E-4

The results seems reasonable enough to believe since the terms are the most frequently used one for wiki search.