CS6240 Map Reduce – Akash Kadam (Section 2)

Data Transfer Information from Mapper to Reducer

Iterations	Map O/P Records	Map O/p bytes	Map O/P Materialized bytes	File Output Format Counters Bytes Written
1	74346327	3937633848	1418516300	1371479022
2	74617227	3947521858	1418363254	1756343119
3	74617227	3947521858	14183332913	1756893505
4	74617227	3947521858	1418328676	1756869390
5	74617227	3947521858	1418309162	1756823047
6	74617227	3947521858	1418294385	1756725261
7	74617227	3947521858	1418298786	1756766184
8	74617227	3947521858	1418281027	1756628967
9	74617227	3947521858	1418924177	1756707889
10	74617227	3947521858	1418282056	1756562670

If you observe the above values there is a slight decrease in the value of the map o/p records, map o/p bytes. This is because the dangling nodes are handled in the second iteration of the page rank computation.

Time comparison of AWS runs

Number of m4.large machine	Job1 - Preprocessing (min)	Job2 – 10 Iterations (min)	Job3 - Top-100 (min)
6	18	29	3
11	10	48	2

According to my findings from the data above it seems that the preprocessing time is reduced when we use more number of machines but the time required for 10 iterations when using 11 machine was surprisingly greater as compared with 6 machines. I assume this could be due to some issues at the amazon web services end.

Output Analysis of simple and full data sets

I observe that the top 100 values are very similar for the simple and full data sets. There are minor changes in the page rank values obtained but they are not that huge.

Pseudo code for Page Rank

```
//mapper of job 1
class InputParseMapper {
    map(key, value){
        parses the input files in the for of "pageName->(list of all outlinks)->pageRank"
}

// mapper of job2
class PageRankMapper(Object, Text, Text, GraphData){
    setup(){
        itr = get from Configuration;
        totalPages = get from Configuration;
    }

    map(key, value){
        line = value.toString();
        lineSplit[] = line.split("->");
        outlinks[] = lineSplit[1]; //geting the list of outlinks

        if(itr == 0){
            pageRval = 1 / totalPages;
        }
        else
            pageRval = lineSplit[2];
        if(outlinks == null){
            set the Global counter "danglingFactor"
        }

        else{
            write(pageName, new Graph(true, pageRval, outlinks.length))
        }
        write (lineSplit[0], new Graph(false, lineSplit[1]);
    }



// reducer of job2
class PageRankReducer(Text, GraphData, Text, NullWritable){

    setup(){
        danglingFactor = get from Config
        pageCOunt = get from Config
        pageList = new HashSet<String>
        }

    reduce(key, Iterable<GraphData>){
        for Graph d : values {
            if(d.isnotPRdata)
            outlinklist = d.getOutlinklist();

            else{
            summation += d.getProfm / d.getOutlinkCount;
            }
    randomjump = alpha / pageCount;
    followingLink = (1.0 - alpha) * ((danglingFactor / pageCount) + summation)

    pageRank = randomjump + followingLink
    String output = "pagename->(list)->pageRank"
    write(output, Nullwritable.get())

    cleanup()
    { update noOfPage counter based on pageList count}
}




// mapper of job 3
class OutputMapper(Object, Text, DoubleWritable, Text){

    Map<String, Double> topPages; // hashmap to store top pages

    setup(){
        topPages = HashMap
    }

    map(){
        parts = value.split("->")
        put in hashmap pageRank and pageName;
    }
```

```
    cleanup(){
        sort in descending order of values in hashmap
        write(pagerank, page)
    }

}



// reducer of job 3
class OutputReducer(DoubleWritable, Text, DoubleWritable, Text){
    private int count = 0 ;

    reduce(key, Iterable<value>){
        for each page in value
            coun++
            write(key, page) until count reaches 100
    }



class PageRankSort {
    override compare() and compare such that you get the pagerank list in
    descending order
}

class Enum CounterGraph{
     //the global counter
    danglingFactor,
    numberOfPages
};
```