MS&E 317/CS 263: Algorithms for Modern Data Models, Spring 2014
`http://msande317.stanford.edu`.
Instructors: Ashish Goel and Reza Zadeh, Stanford University.

Lecture 4, 4/9/2014. Scribed by Burak Yavuz.

4.1 Outline

1. Matrix Vector Multiply (Av)
2. PageRank
   - on MapReduce
   - on RDD's / Spark

4.2 Matrix Vector Multiplication on MapReduce

We have a sparse matrix A stored in the form $< i, j, a_{ij} >$, where i, j are the row and column indices and a vector v stored as $< j, v_j >$. We wish to compute Av.

For the following algorithm, we assume v is small enough to fit into the memory of the mapper.

Algorithm 1 Matrix Vector Multiplication on MapReduce

```
function MAP(< i, j, a_ij >)
    Emit(i, a_ij v[j])
end function
function REDUCE(key,values)
    ret ← 0
    for val ∈ values do
        ret ← ret + val
    end for
    Emit(key, ret)
end function
```

4.3 PageRank

For a graph G with n nodes, we define the transition matrix $Q = D^{-1}A$, where $A \in \mathbb{R}^{n \times n}$ is the adjacency matrix and $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix composed of the outgoing edges from each node.

We use Power Iteration to estimate importance values for webpages as $v^{(k+1)} = v^{(k)}Q$, where $v \in \mathbb{R}^n$ is a row vector, and k is the number of iterations. We set $v^{(0)} = \mathbf{1}$, a vector with each element equaling one.

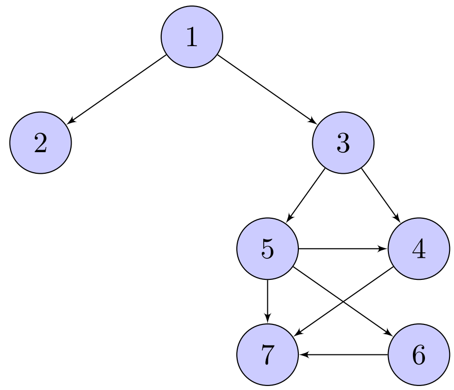

Figure 1: Graph G

Using Q as the probability distribution for random walks is a problem when G contains dead-ends, i.e. "sink" nodes (nodes 2 and 7 in Figure 1). We introduce the idea of random teleports. With probability α, the random walker can teleport to a random webpage or continue walking with probability $1-\alpha$ where $0 < \alpha < 1$. Then we have a new matrix:

$$P = (1-\alpha)Q + \alpha\Lambda$$

where

$$\Lambda = \begin{bmatrix} --- & \lambda & --- \\ --- & \lambda & --- \\ & . & \\ & . & \\ & . & \\ --- & \lambda & --- \end{bmatrix}_{n\times n}$$

and $\alpha \in \mathbb{R}^n$ is composed of the probability distribution of teleporting to a webpage.

The Power Iteration applies again: $\pi^{(k+1)} = \pi^{(k)}Q$.

Theorem 4.1

$$\|\pi - v^{(k)}\|_2 \leq e^{-ak}$$

for some constant $a > 0$.

According to 4.1, for $n = 10^9$, around 9 iterations are enough to get correct ranking.

4.3.1 PageRank on MapReduce

P is stored as $< i, \{(j, P_{ij})\} >$, where $\sum_j P_{ij} = 1, \forall i \in [1, n]$.

v is stored as $< i, v_i^{(k)} >$.

We use a two-step algorithm:

Step 1:

Annotate P_i with v_i, i.e. Emit $< i, v_i, \{(j, P_{ij})\} >$.

Step 2:

Algorithm 2 PageRank Computation on MapReduce, Step 2

```
function MAP(< i, v_i, {(j, P_ij)} >)
    for (j, P_ij) ∈ links do
        Emit(j, P_ij v_i^(k))
    end for
end function
function REDUCE(key,values)
    v_i^(k+1) = Σ_{v∈values} v
    Emit (i, v_i^(k+1))
end function
```