CS535 BIG DATA

PART 1. BATCH COMPUTING MODELS FOR BIG DATA ANALYTICS

1. DISTRIBUTED MODEL FOR SCALABLE BATCH COMPUTING - MAPREDUCE

Sangmi Lee Pallickara
Computer Science, Colorado State University
http://www.cs.colostate.edu/~cs535

FAQs

- Programming Assignment 1
  - Due Sept. 28
  - Submission via Canvas
  - Please check the course Web Page at least twice a week
- Use the configuration file with memory size 2GB per node
  - Current one on the web page

Today's topics

- MapReduce with examples
- Understanding PageRank algorithm and implementing with MapReduce

Matrix-Vector Multiplication Using MapReduce

Matrix-Vector multiplication using MapReduce (1/3)

- Suppose we have **an n x n matrix M**, whose element in **row i** and **column j** will be denoted M_{ij}
- v is a n x 1 column vector
- Then the matrix-vector product is the vector x of length n, whose i^{th} element x_i is given by:

$$x_i = \sum_{j=1}^{n} m_{ij} v_j$$

Matrix-Vector multiplication using MapReduce (2/3)

- If n = 100, we do not need MapReduce
- However, if this calculation is a part of ranking Web pages (n could be 10^7) performed by the search engine?
  - The vector v cannot fit in main memory

Matrix-Vector multiplication using MapReduce (3/3)

- The matrix M and the vector v will each be stored in a file of the DFS (e.g. HDFS)
- Assume that row-column coordinates of each matrix element will be discoverable
  - Either from its position in the file or explicit coordinates

The Map function

- The **Map function** is written to apply to **one element of M**
- Each **Map task** will operate **on a block of the matrix M**
- From each matrix element m_{ij}, it produces the key-value pair $(i, m_{ij}v_j)$
  - Why?
- All terms of the sum that make up the component x_i of the matrix-vector product will get the same key, i

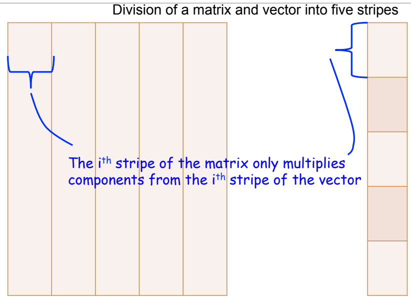
The Reduce function

- Sums all the values associated with a given key I
  - n number of pairs per i
- The result will be a pair (i, x_i)
  - Sort pairs in ascending order of i
    - $(x_0, x_1, \ldots, x_i)$

If the vector v cannot fit in main memory?

- It is possible that the vector v is so large that it will not fit in main memory entirely
- We can divide the matrix into vertical stripes of equal width and divide the vector into an equal number of horizontal stripes of the same height
  - The goal is to use enough stripes so that the portion of the vector in one stripe can fit into main memory

Division of a matrix and vector into five stripes



Matrix M Vector v

Programming Assignment 1:

Estimating PageRank Values of Wikipedia Articles using MapReduce

Objectives

- The goal of this programming assignment is to enable you to gain experience in:
  - Implementing iterative algorithms to estimate PageRank values of Wikipedia articles
  - Designing and implementing batch layer computations using Apache Spark and HDFS

Tasks

- Estimation of PageRank values under ideal conditions
- Estimation of the PageRank values while considering dead-end articles
- Analysis of the above results
  - Implement a Wiki bomb

Requirements

- A sorted list of Wikipedia pages based on their ideal PageRank value in descending order
- A sorted list (in descending order) of Wikipedia pages based on their PageRank value with taxation
- Average difference between the ideal PageRank (from A) and PageRank with taxation (from B) for each Wikipedia article

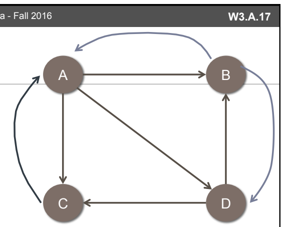
Requirements - Continued

- This computation should be performed in your own Hadoop cluster with **10 machines**
- You should present results from at least **25 iterations**
- You should use the algorithms included **in this description**
- **You are NOT allowed to use existing PageRank implementations**
- Demonstration of your software should be on machines in CSB-120
- Demonstration will include an interview discussing implementation and design details
- Your submission should include your **source codes**
  - Your submission should be via Canvas

Definition of PageRank

- A function that assigns a real number to each page in the Web
- The higher the PageRank of a page, the more **"important"** it is
- There are *many* algorithms for assignment of PageRank

Example

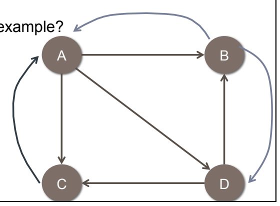


- Page A has links to B, C and D
- Page B has links to A and D
- Page C has a link to A
- Page D has links to B and C

- Suppose that a random surfer starts at page A
- Page B, C and D will be the next with probability 1/3
  - 0 probability of being at A

Example (2/3)

- Now the random surfer at B has probability of ½ of being at A, ½ of being at D and 0 of being at B or C
- Transition matrix M
  - What happens to random surfers after one step?
  - M has n rows and columns
  - What is the transition matrix for this example?



Example

$$
M = \begin{array}{c} \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \\ \begin{array}{cccc} \text{A} & \text{B} & & \end{array} \end{array}
$$

- The first column
  - a surfer at A has a 1/3 probability of being at each of the other pages next
- The second column
  - a surfer at B has a ½ probability of being at A next and the same for being at D

What does this matrix mean? (1/4)

- The probability distribution for the location of a random surfer
  - A column vector whose jth component is the probability that the surfer is at page j
- If we surf any of the n pages of the Web with equal probability
  - The initial vector v_o will have $1/n$ for each component
  - If M is the transition matrix of the Web
    - After one step, the distribution of the surfer will be Mv_o
    - After two steps, $M(Mv_o) = M^2 v_0$ and so on
- Multiplying the initial vector v_0 by M a total of i times
  - The distribution of the surfer after i steps

What does this matrix mean? (2/4)

- The probability x_i that a random surfer will be at node i at the next step

$$x_i = \sum_j m_{ij} v_j$$

- m_{ij} is the probability that a surfer at node j will move to node i at the next step
- v_j is the probability that the surfer was at node j at the previous step

What does this matrix mean? (3/4)

- The distribution of the surfer approaches a limiting distribution v that satisfies $v = Mv$ provided two conditions are met:

1. The graph is strongly connected
   - It is possible to get from any node to any other node
2. There are no dead ends
   - Dead ends==Nodes that have no arcs out

What does this matrix mean? (4/4)

- The limit is reached when multiplying the distribution by M another time does not change the distribution
  - The limiting v is an eigenvector of M
  - Since M is stochastic (its columns each add up to 1), v is the principle eigenvector
    - Its associated eigenvalue is the largest of all eigenvalues
- For the Web, 50-75 iterations are sufficient to converge to within the error limits of double-precision arithmetic

$$M=\begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

- Suppose we apply this process to the matrix M
- The initial vector v0 and v1 after multiplying M

$$v_1=Mv_0=\begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}=\begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}$$

- The sequence of approximations to the limit we get by multiplying at each step by M is:

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}\begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}\begin{bmatrix} 15/48 \\ 11/48 \\ 11/48 \\ 11/48 \end{bmatrix}\begin{bmatrix} 11/32 \\ 7/32 \\ 7/32 \\ 7/32 \end{bmatrix}\ \ldots\ \begin{bmatrix} 3/9 \\ 2/9 \\ 2/9 \\ 2/9 \end{bmatrix}$$

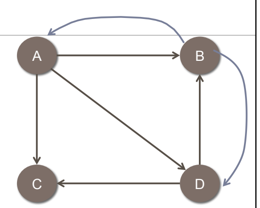
- This difference in probability is not noticeable
- In the real Web, there are billions of nodes of greatly varying importance
  - The probability of being at a node like www.amazon.com is orders of magnitude greater than others

Avoiding Dead Ends

- If we allow dead ends
  - The transition matrix of the Web is no longer stochastic
    - Some of the columns will sum to 0 rather than 1
- If we compute $M^i v$ for increasing powers of a substochastic matrix
  - Some of all of the components of the vector go to 0
  - substochastic matrix
    - A matrix whose column sums are **at most** 1
  - Importance "drains out" of the Web
    - No information about the relative importance of pages

Example: Dead end

- Remove the arc from C to A?
  - C becomes a dead end
  - If a random surfer reaches C, they disappear at the next round



$$M=\begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

Example: Dead end (contd.)

- Repeatedly multiplying the vector by M:

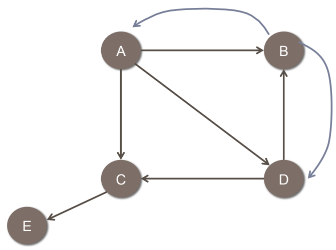
$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}\begin{bmatrix} 3/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}\begin{bmatrix} 5/48 \\ 7/48 \\ 7/48 \\ 7/48 \end{bmatrix}\begin{bmatrix} 21/288 \\ 31/288 \\ 31/288 \\ 31/288 \end{bmatrix}\ \ldots\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- The probability of a surfer being anywhere goes to 0 as the number of steps increase

Approaches to dealing with dead ends

- Recursive deletion
  - Drop dead ends from the graph
    - Drop their incoming arcs as well
    - Doing so may create more dead ends
      - Drop those new dead ends
- Taxation
  - Modify the process by which random surfers are assumed to move about the Web

Example of recursive deletion (1/4)



Example of recursive deletion (2/4)

- The final matrix for the graph is

$$M = \begin{bmatrix} 0 & ½ & 0 \\ ½ & 0 & 1 \\ ½ & ½ & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} \begin{bmatrix} 1/6 \\ 3/6 \\ 2/6 \end{bmatrix} \begin{bmatrix} 3/12 \\ 5/12 \\ 4/12 \end{bmatrix} \begin{bmatrix} 5/24 \\ 11/24 \\ 8/24 \end{bmatrix} \quad \ldots \quad \begin{bmatrix} 2/9 \\ 4/9 \\ 3/9 \end{bmatrix}$$

Example of recursive deletion (3/4)

- We still need to compute PageRank for deleted nodes (C and E)

- C was the last to be deleted
  - We know all its predecessors have PageRanks (A and D)
  - Therefore,
    - PageRank of C = 1/3 x 2/9 + ½ x 3/9 = 13/54

Example of recursive deletion (4/4)

- Now, we can compute the PageRank for E
  - Only one predecessor, C
    - The PageRank of E is the same as that of C (13/54)

- The sums of the PageRanks exceeds 1
  - It cannot represent the distribution of a random surfer
  - But it provides a good estimate

PageRank using Taxation (1/5)

- To avoid dead ends, we modify the calculation of PageRank
  - Allow each random surfer a small probability of ***teleporting*** to a random page
  - Rather than following an out-link from their current page

PageRank using Taxation (2/5)

- The iterative step, where we compute a new vector estimate of PageRanks v' from the current PageRank estimate v and the transition matrix M is

$$v' = \beta M v + (1-\beta)e/n$$

- Where β is a chosen constant
  - Usually in the range 0.8 to 0.9
- e is a vector with 1's for the appropriate number of components
- n is the number of nodes in the Web graph

PageRank using Taxation (3/5)

- The term βMv represents the case where:
  - With probability β, the random surfer decides to follow an out-link from their present page
- The term $(1-\beta)e/n$ is a vector
  - Each of whose components has value $(1-\beta)/n$
  - Represents the introduction, with probability $1-\beta$, of a new random surfer at a random page

$$v' = \beta Mv + (1-\beta)e/n$$

PageRank using Taxation (4/5)

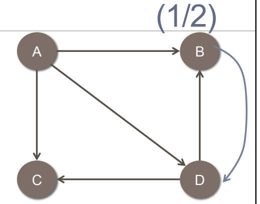
- If the graph has no dead ends
  - The probability of introducing a new random surfer is exactly equal to the probability that the random surfer will decide not to follow a link from their current page
  - Surfer decides either to follow a link or teleport to a random page

PageRank using Taxation (5/5)

- If the graph has dead ends
  - The surfer goes nowhere
  - The term $(1-\beta)e/n$ does not depend on the sum of the components of the vector v, there will be some fraction of a surfer operating on Web
  - When there are dead ends, the sum of the components of v may be less than 1
    - But it will never reach 0

Example of Taxation (1/2)

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$



$$v' = \beta Mv + (1-\beta)e/n$$

$\beta = 0.8$

$$v' = \begin{bmatrix} 0 & 2/5 & 0 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix} v + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \\ 1/20 \end{bmatrix}$$

Example of Taxation (2/2)

- For the first few iterations:

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} \begin{bmatrix} 9/60 \\ 13/60 \\ 13/60 \\ 13/60 \end{bmatrix} \begin{bmatrix} 41/300 \\ 53/300 \\ 53/300 \\ 53/300 \end{bmatrix} \begin{bmatrix} 543/4500 \\ 707/4500 \\ 707/4500 \\ 707/4500 \end{bmatrix} \cdots \begin{bmatrix} 15/148 \\ 19/148 \\ 19/148 \\ 19/148 \end{bmatrix}$$

Questions?