PART 1. LARGE SCALE DATA ANALYTICS
WEB-SCALE LINK AND SOCIAL NETWORK ANALYSIS
Sangmi Lee Pallickara
Computer Science, Colorado State University
http://www.cs.colostate.edu/~cs435

Topics
• How MapReduce works
• Large-scale Analytics 1. Web-Scale Link and Social Network Analysis

FAQs
• PA1 has been posted
  • Feb. 21, 5:00PM via Canvas
  • Individual submission (No team submission)
• Team assignment is available in Canvas
  • Any questions? Please contact me!

Shuffle and Sort
• Sort
  • MapReduce makes the guarantee that the input to every reducer is sorted by key
• Shuffle
  • MapReduce transfers the map outputs to the reducers as inputs
Shuffle and sort

In a map task

- Each map task has a circular memory buffer
  - For output
    - 100MB by default
    - io.sort.mb

- When the contents of the buffer reaches the threshold, a background thread starts spill the contents to disk
  - Default 0.80
  - Spills are written in round-robin-fashion to the local directory
    - mapred.local.dir

Partitioning

- Before the data is written to the local disk, data is divided into partitions corresponding to the reducers

  - The background thread performs an in-memory sort by key
    - Within each partition

  - Each time the memory buffer reaches the spill threshold, a new spill file is created
    - There can be several spill files after the last output record is written
    - The spill files are merged into a single partitioned and sorted output file

Output file

- Combiner is run
- Output files from map can be compressed
- The output file's partitions are made available to the reducers over HTTP

Execution of Combiner

- Hadoop does not guarantee on how many times a combiner will be called for each output key

  Properties of a Combiner
  
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Combiner's functionality must be</td>
<td>Commutative</td>
</tr>
<tr>
<td></td>
<td>Associative</td>
</tr>
<tr>
<td>Input types (key/value pair) and the output types must be the same</td>
<td>Same as output of mapper as well</td>
</tr>
</tbody>
</table>
Shuffle and sort: In a reduce task

Copy phase
- The reduce task needs the map output from several map tasks across the cluster
  - The reduce task starts copying their outputs as soon as each completes
    - Merges them into larger and sorted files
    - Decompresses the compressed files

Sort phase
- Sort phase
  - All of the map outputs should be moved and copied to the reduce task
  - Merging and sorting the map outputs
  - Sorting is done in rounds
    - If there are 50 map outputs and the merge factor was 10
      - 5 intermediate files
      - Merging intermediate files: additional round
      - N rounds of merging will be required
    - Final round
      - A mixture of in-memory and on-disk segments
      - Simply feeds the reduce function
      - Without writing a single sorted file to disk

Failures in Classic MapReduce
- The child task fails
  - Runtime exception from the user code
  - The child JVM reports the error back to its parent before it exits
  - Written in the user logs
  - Tasktracker marks the task attempt as failed
  - Forces a slot to another task
  - Sudden exit of the child JVM
    - Tasktracker notion that the process has exited and marks the attempt as failed
- Hanging tasks
  - If there is no progress update for a while
    - Mark the task as failed
    - Timed out periodically normally 10 minutes
      -iegel, Cascio, Trenchard

Tasktracker failure in Classic MR
- Tasktracker stops sending heartbeats
  - Jobtracker will notice if it hasn’t received one for 10 minutes (configurable)
  - Remove it from the pool of tasktrackers
- Jobtracker arranges tasks including the completed jobs
  - Because the output may not be accessible
- Tasktracker can also be blacklisted if more than four tasks from the same job fail (set by mapred.max.tracker.failures)
  - Blocklisted tasktrackers are not assigned tasks
  - Until faults expire
Jobtracker failure in Classic MR

- The most serious failure mode
- Hadoop has no mechanism for dealing with jobtracker failure
  - Single point of failure
- All running jobs fail
- After restarting a jobtracker
  - Job should be resubmitted
- This is improved with YARN

Task failure in YARN

- Failure of the running task is similar to the classic case
  - Runtime exception and sudden exit of the JVM are propagated back to the application master
  - The task attempt is marked as failed
  - Hanging tasks are noticed by the application manager by the absence of a ping over the umbilical channel

Application master failure in YARN

- No heartbeats to the resource manager from the application master
  - The resource manager will detect the failure and start a new instance of the master running in a new container
  - All tasks will be rerun (default)
    - Recovery can be enabled
- Client will access resource manager to get the new address of the application master

Node manager failure in YARN

- Resource manager will stop getting heartbeats
  - Remove the failed node manager from the pool of available nodes
- Any task or application master running on the failed node manager will be recovered

Resource manager failure

- After a crash and a new resource manager instance is brought up (by administrator)
  - It recovers from the saved state
    - Check points saved in the persistent storage
    - Non-completed jobs are included

Part 1. Large Scale Data Analytics

1. Web-Scale Link Analysis and Social Network Analysis
   - Web-Scale Link Analysis
This material is built based on,

  - Chapter 5

- http://infolab.stanford.edu/~ullman/mmds.html

What are these?

- Archie
- Veronica
- Infoseek
- Snap
- Direct Hit
- Lycos
- AltaVista
- Excite
- Yahoo
- Google

Early Search Engines

- They worked by crawling the Web and listing the terms
- Words or other strings of characters other than white space
- In an inverted index

- An **inverted index** is a data structure that makes it easy, given a term, to find (pointer to) all the places where that term occurs

Inverted index (1/2)

- For given texts,
  
  - We have the following inverted file index

  - “Colorado State University”
  - “Colorado water source”
  - “University of Colorado”

- A term search for the terms, “Colorado”, “State”, and “University” would give the set
  
  \[ \{0, 1, 2\} \cap \{0\} \cap \{0, 2\} = \{0\} \]

Inverted index (2/2)

Term spam

- If you were selling shirts on the Web
  - All you care about was that people would see your page

- You could add a term like “movie” to your page
  - Add thousands of times
  - It does not even need to show
  - Give the same color as background to the letters
  - A search engine would think this page is very important one about “movie”

- You could go to the search engine and search “movie” and see the first listed page
  - Copy that page with the same color as background
Part 1. Large Scale Data Analytics

1. Web-Scale Link Analysis and Social Network Analysis

Web-Scale Link Analysis: PageRank Algorithm

Current total number of Web pages: More than 1.4 B indexed pages

PageRank

• Goals
  - Providing effective summaries for the search results
  - Ordering/Ranking results

• Simulate random Web surfers
  - Pages that would have a large number of surfers were considered more "important" than pages that would rarely be visited

• The content of a page was judged not only by the terms appearing on that page
  - But by the terms used in or near the links to that page

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 is NOT one fixed algorithm for assignment of PageRank
Example

• 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

• Now suppose the random surfer at B
• B has probability of 1/3 of being at A, 1/3 of being at D and 0 of being at B or C

Example

• Transition matrix $M$
  • What happens to random surfers after one step
  • $M$ has $n$ rows and columns ($n$ pages)
  • What is the transition matrix for this example?

Example

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

• The first column
  • A surfer at A has a 1/3 probability of next being at each of the other pages

• The second column
  • A surfer at B has a 1/3 probability of being next at A and the same for being at D
What does this matrix mean? [1/6]

- The probability distribution for the location of a random surfer
  - A column vector whose \( j \)th component is the probability that the surfer is at page \( j \)

What does this matrix mean? [2/6]

- If we surf at any of the \( n \) pages of the Web with equal probability
  - The initial vector \( v_0 \) will have \( 1/n \) for each component
  - If \( M \) is the transition matrix of the Web
    - After the first one step, the distribution of the surfer will be \( Mv_0 \)
    - After two steps, \( M^2v_0 \) and so on

What does this matrix mean? [3/6]

- 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? [4/6]

- The probability \( x_i \) that a random surfer will be at node \( i \) at the next step
  \[ x_i = \sum_j m_{ij}x_j \]
  - \( m_{ij} \) is the probability that a surfer at node \( j \) will move to node \( i \) at the next step
  - \( x_j \) is the probability that the surfer was at node \( j \) at the previous step

What does this matrix mean? [5/6]

- The distribution of the surfer approaches a limiting distribution \( \pi \) that satisfies \( \pi = M\pi \) 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
     - Nodes that have no arcs out

What does the matrix mean? [6/6]

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

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

• Suppose we apply this process to the matrix \( M \)
• The initial vector \( v_0 \) and \( v_1 \) after multiplying \( M \)

\[
v_{1/n} = M v_{1/(n-1)}
\]

What is the \( v_2 \)?

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

• Suppose we apply this process to the matrix \( M \)
• The initial vector \( v_0 \) and \( v_1 \) after multiplying \( M \)

\[
v_{1/n} = M v_{1/(n-1)}
\]

Example continued

• The sequence of approximations to the limit
  • We get by multiplying at each step by \( M \)

\[
\begin{array}{c|c|c|c|c}
\text{Step} & 1/24 & 35/48 & 11/32 & 2/9 \\
\text{v0} & 1/24 & 1/48 & 7/32 & 2/9 \\
\text{v1} & 1/24 & 1/48 & 7/32 & 2/9 \\
\end{array}
\]

• 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

Example

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

• Suppose we apply this process to the matrix \( M \)
• The initial vector \( v_0 \) and \( v_1 \) after multiplying \( M \)

\[
v_{1/n} = M v_{1/(n-1)}
\]

Matrix-vector Multiplication by MapReduce  [1/3]

• Suppose we have an \( n \times n \) matrix \( M \), whose element in row \( i \) and column \( j \) will be denoted \( M_{ij} \)
• Then the matrix-vector product is the vector \( x \) of length \( n \), whose \( i^{th} \) element \( x_i \) is given by,

\[
x_i = \sum_{j=1}^{n} M_{ij} x_j
\]
Matrix-vector Multiplication by MapReduce [2/3]

- If \( n = 100 \), we do NOT need DFS or MapReduce

- However, if this calculation is a part of ranking Web pages (\( n \) is 10M) that goes on at search engine? The vector \( v \) cannot fit in main memory
  - More than 1.4B pages

The Map function

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

Matrix-vector Multiplication by MapReduce [3/3]

- The matrix \( M \) and the vector \( v \) each will be stored in a file of the DFS(HDFS)

- Assume that row-column coordinates of each matrix element will be discoverable
  - Either from its position in the file or explicit coordinates

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 its 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

The Reduce function

- Sums all the values associated with a given key \( i \)
- The result will be a pair \((i, x_i)\)
The $i$th stripe of the matrix multiplies only components from the $i$th stripe of the initial vector.

\[
\begin{pmatrix}
0.002 & 0.007 & 0.003 & 0.000 \\
0.000 & 0.000 & 0.001 & 0.000 \\
0.002 & 0.000 & 0.001 & 0.000 \\
0.000 & 0.001 & 0.120 & 0.000 \\
\end{pmatrix}
\]

Mapper 1
Mapper 2
Mapper 3
Mapper 4
Mapper 5