15-319 / 15-619 Cloud Computing

Recitation 12

April 05th, 2016

Overview

- **Administrative issues**
  - Tagging, 15619Project, project code
- **Last week's reflection**
  - Project 3.5
  - Unit 5 - Module 18
  - Quiz 10
- **This week's schedule**
  - Project 4.1, Batch Processing with MapReduce
  - Unit 5 - Module 19, 20
  - Quiz 11
- **Twitter Analytics: The 15619Project**

Reminders

- Monitor AWS expenses regularly and tag all resources
  - Check your bill both on AWS and TPZ
- Piazza Guidelines
  - Please tag your questions appropriately
  - Search for an existing answer first
- Provide clean, modular and well documented code
  - Large penalties for not doing so.
  - **Double check** that your code is submitted!! (verify by downloading it from TPZ from the submissions page)
- Utilize Office Hours
  - We are here to help (but not to give solutions)

Project 3.5 : FAQs

Problem 1: Out-of-memory issue during partitioning

- Should make sure the partition is really necessary
- Creating a large number of partitions on a big table may drain the datanode's memory

Modules to Read

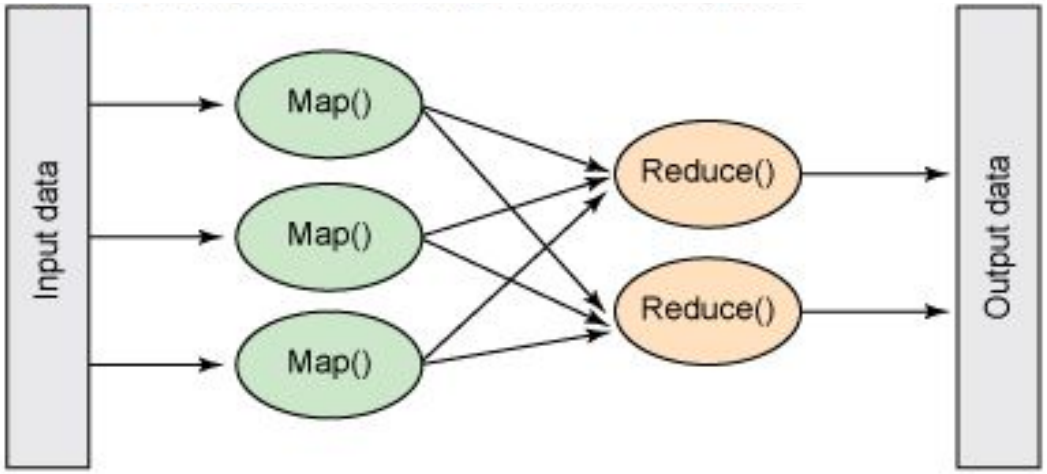
- UNIT 5: Distributed Programming and Analytics Engines for the Cloud
  - Module 18: Introduction to Distributed Programming for the Cloud
  - Module 19: Distributed Analytics Engines for the Cloud: MapReduce
    - Hadoop 1.0
    - Hadoop 2.0 - YARN
  - Module 20: Distributed Analytics Engines for the Cloud: Spark
  - Module 21: Distributed Analytics Engines for the Cloud: GraphLab

Project 4

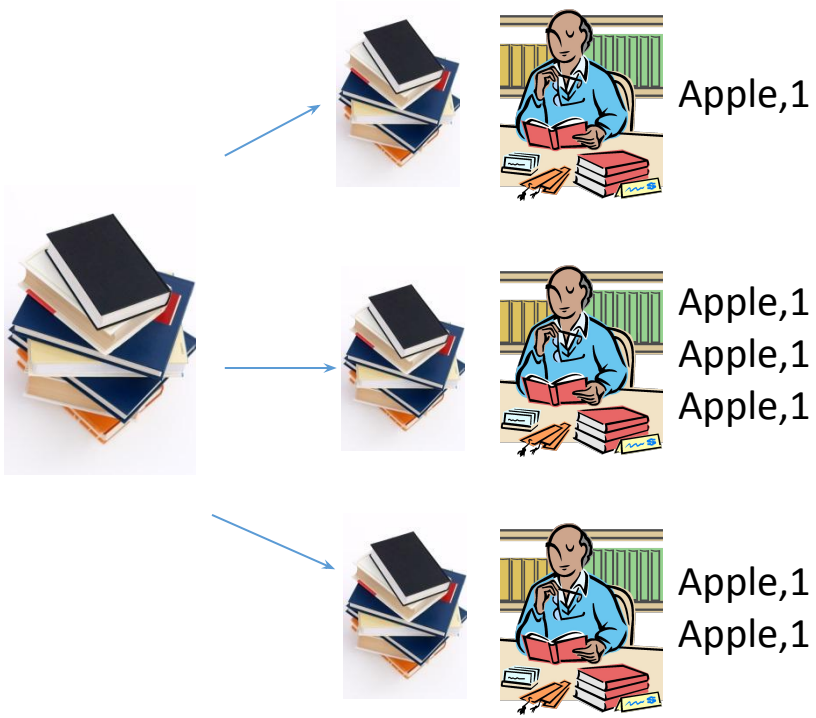
- Project 4.1, Batch Processing with MapReduce
  - MapReduce Programming Using YARN
- Project 4.2
  - Iterative Programming Using Apache Spark
- Project 4.3
  - Stream Processing using Kafka/Samza

Introduction to MapReduce

- **Definition**: Programming model for processing large data sets with a parallel, distributed algorithm on a cluster
- **Phases of MapReduce:**
  - **Map**
  - **Shuffle**
  - **Reduce**



MapReduce - Introduced in Project 1

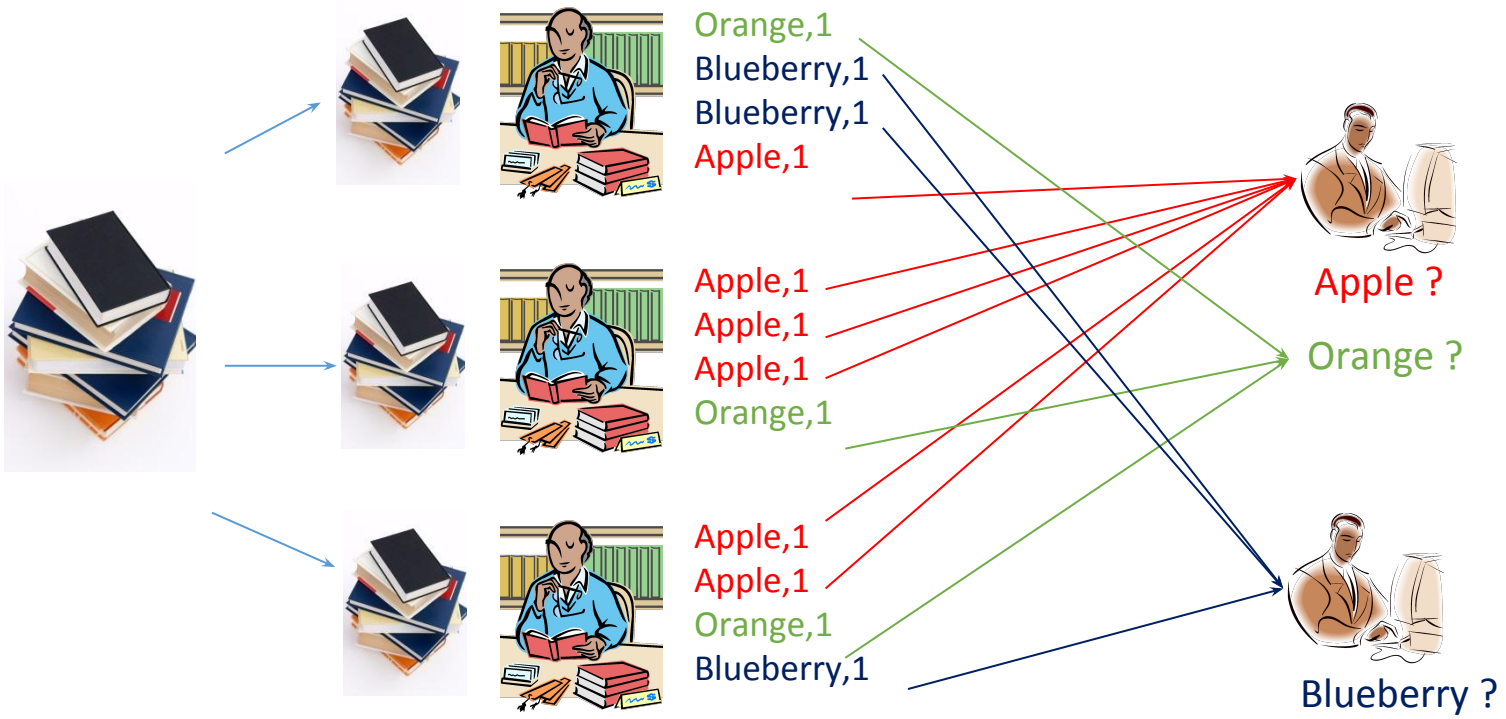


How many times does the word “apple” appear in all books in Hunt Library?

I heard 6 “Apple”s !

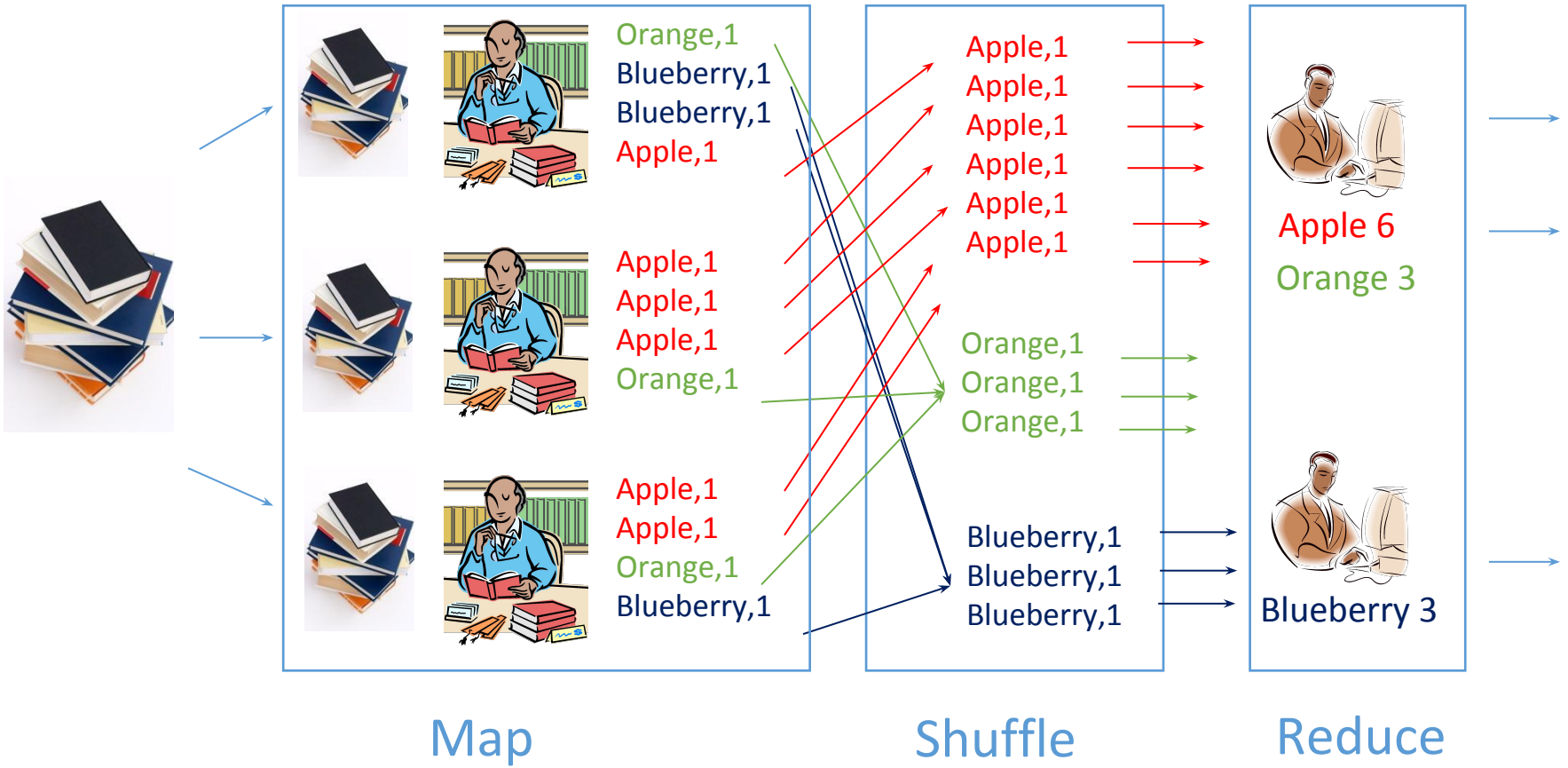
MapReduce Example

What if we want to count the number of times all fruits appeared in these books?

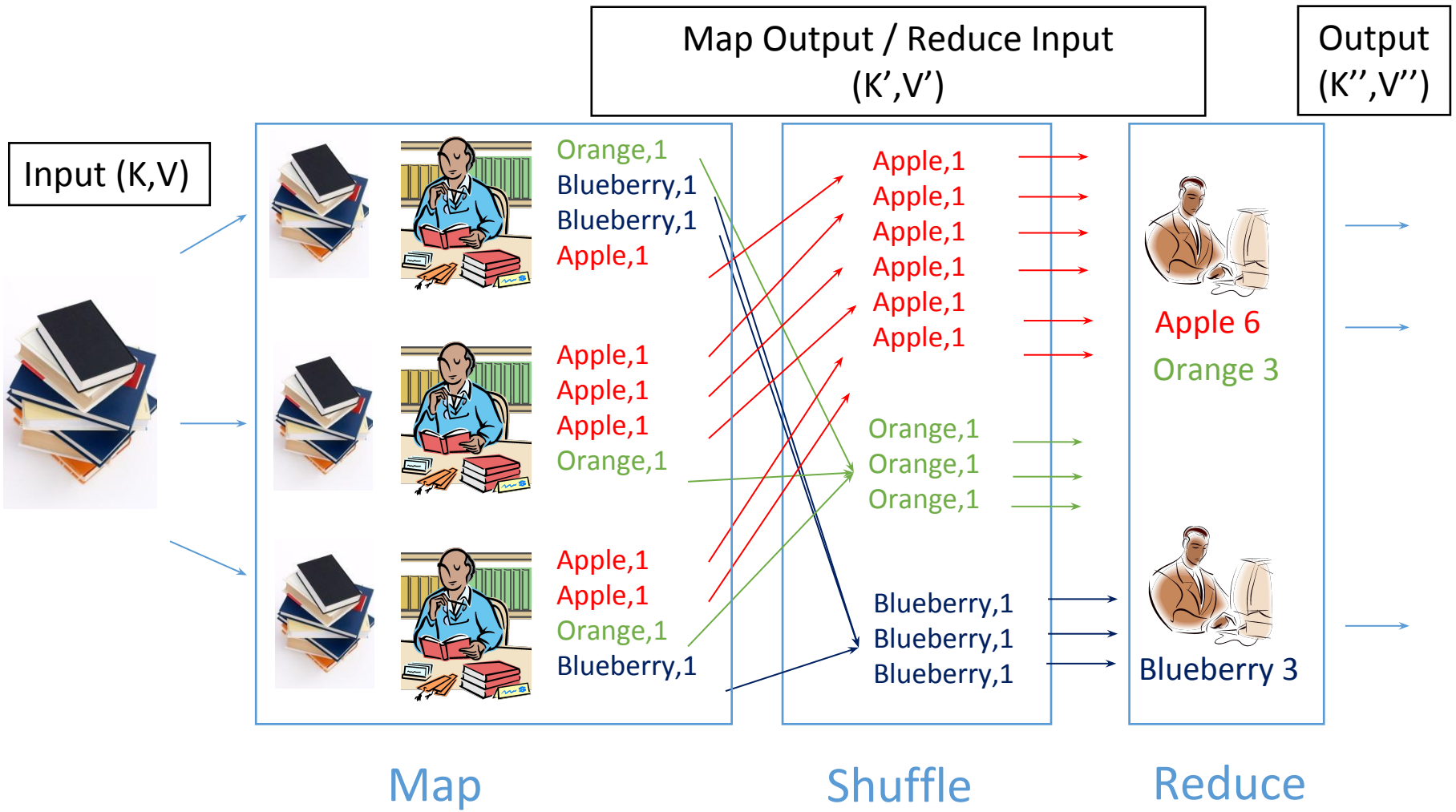


You can have multiple aggregators, each one working on a distinct set of “fruits”.

MapReduce Example



MapReduce Example



Steps of a MapReduce job

- Map
- Shuffle
- Reduce
- Produce final output

Steps of MapReduce - 1

- Map
  - Prepare input for mappers
    - Split input into parts and assign them to mappers
  - Map Tasks
    - Each mapper will work on its portion of the data
    - Output: **key-value pairs**
      - Keys are used in Shuffling and Merge to find the Reducer that handles it
      - Values are messages sent from mapper to reducer
      - e.g. (Apple, 1)

Steps of MapReduce - 2

- Shuffle
  - Sort and group by key:
    - Split keys and assign them to reducers (based on hashing)
    - Each key will be assigned to exactly one reducer
- Reduce
  - Input: mapper's output (key-value pairs)
  - Each reducer will work on one or more keys
  - Output: the result needed
- Produce final output
  - Collect all output from reducers
  - Sort them by key

MapReduce Data Types - 1

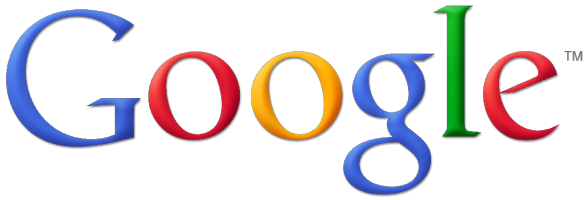
- Mapper (default)
  - Input: **key-value pairs**
    - **Key:** byte offset of the line
    - **Value:** the text content of the line
  - Output: **key-value pairs**
    - **Key:** specified by your program
    - **Value:** specified by your program based on what content you expect the reducer to receive as a list

(k1,v1) -> Mapper -> (k2,v2)

MapReduce Data Types - 2

- Reducer
  - Input: **key-value pairs**
    - A list of values for each key output from the mapper
  - Output: **key-value pairs**
    - The desired result from your aggregation

(k2,list(v2)) -> Reducer -> (k3,v3)



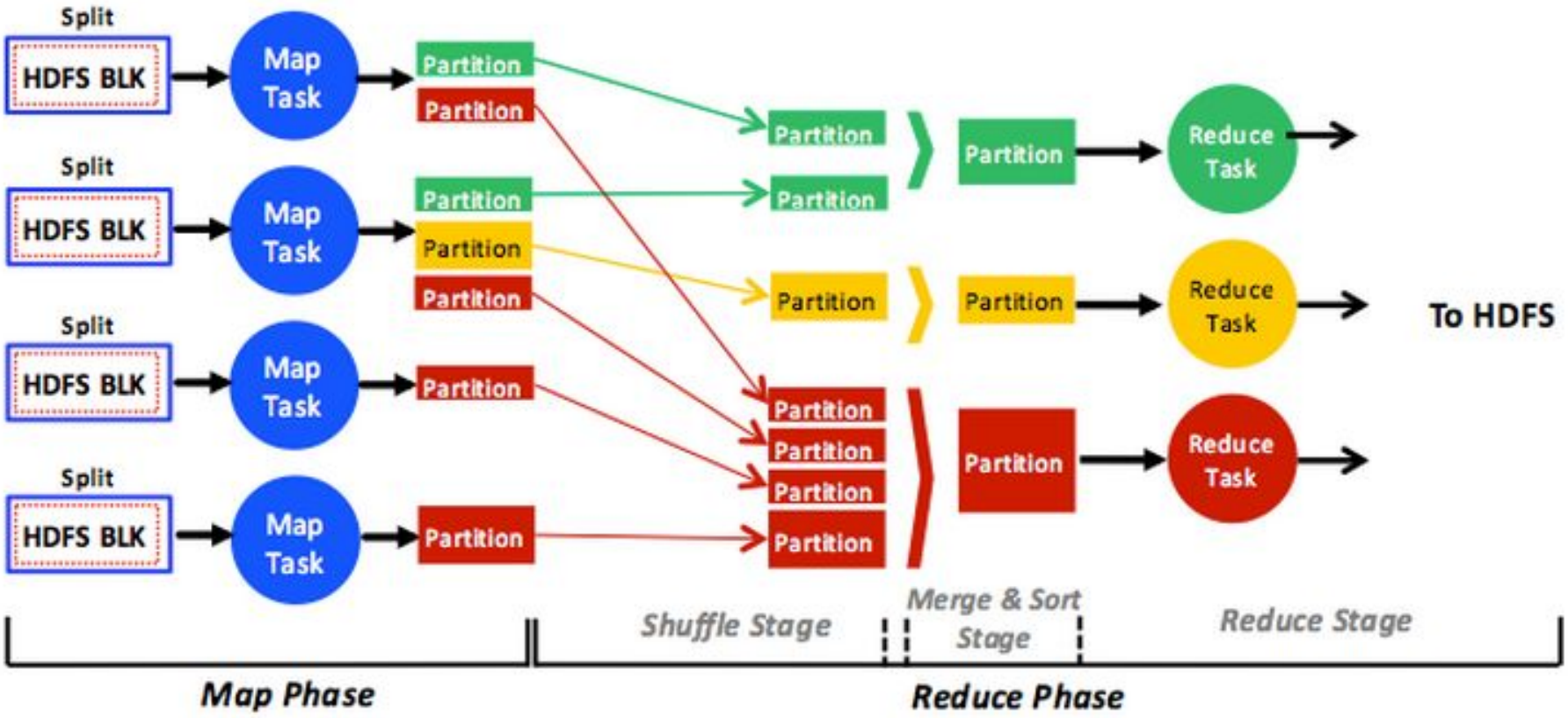
Proprietary		Open Source
GFS	→	HDFS
MapReduce	→	MapReduce
BigTable	→	HBase

MapReduce and Hadoop

- MapReduce
  - A programming model for processing large data sets using a parallel distributed algorithm
- Apache Hadoop
  - A framework for running MapReduce applications on a large cluster of commodity hardware
  - Implements the MapReduce computational paradigm
  - Uses HDFS for data storage
  - Engineers with little knowledge of distributed computing can write the code in a short period

MapReduce and HDFS

- Detailed workflow

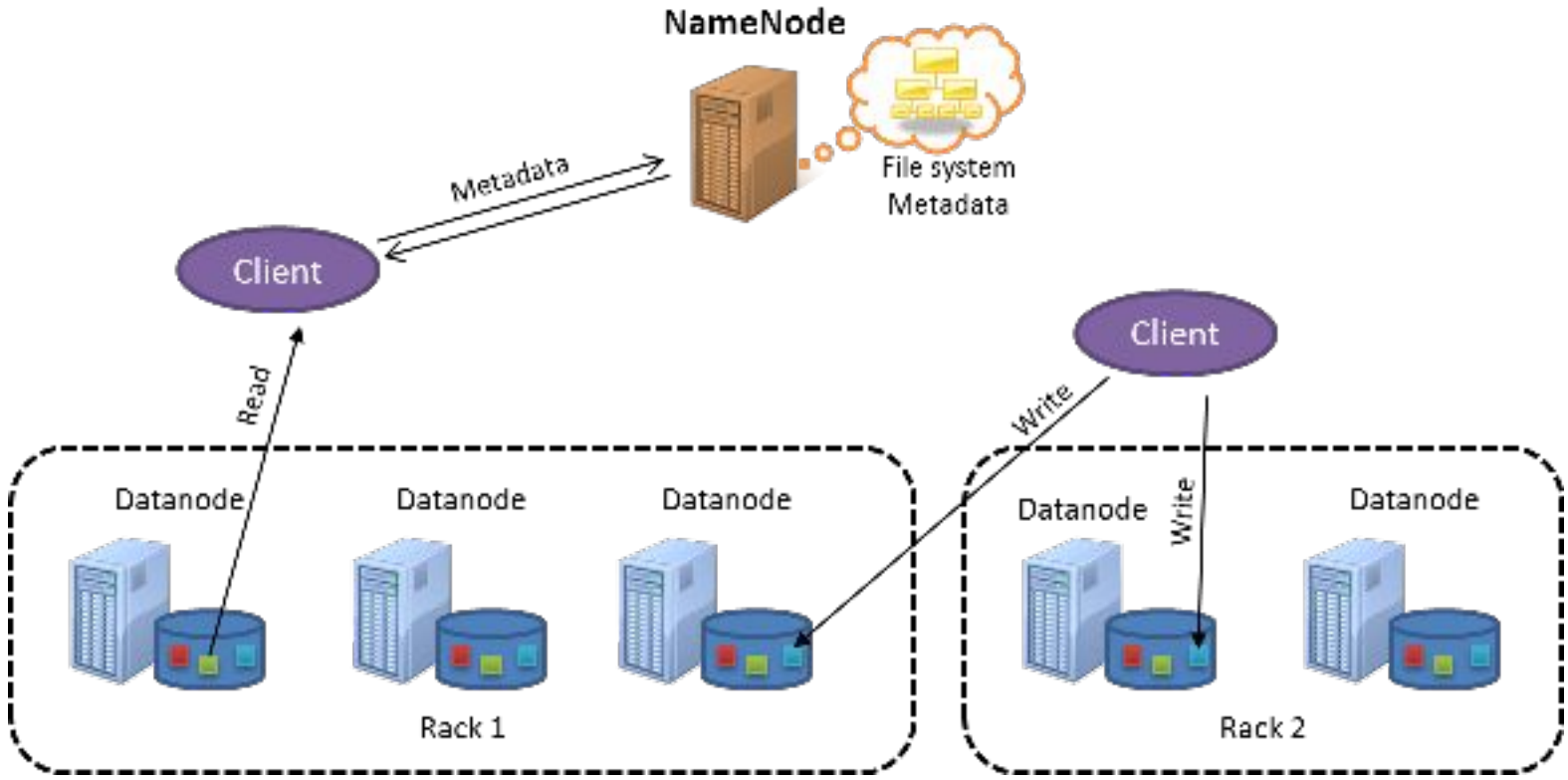


HDFS - Distributed File System

- Paper
  - The Hadoop Distributed File System, Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler, Yahoo!, 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)
- Purpose
  - Serve as the distributed storage to run Hadoop's MapReduce applications
  - An open-source framework which can be used by different clients with different needs

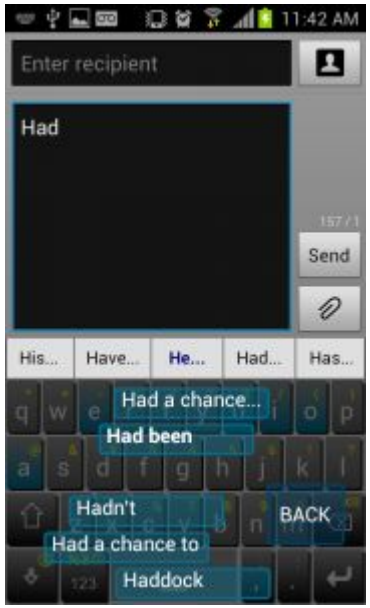
HDFS - Distributed File System

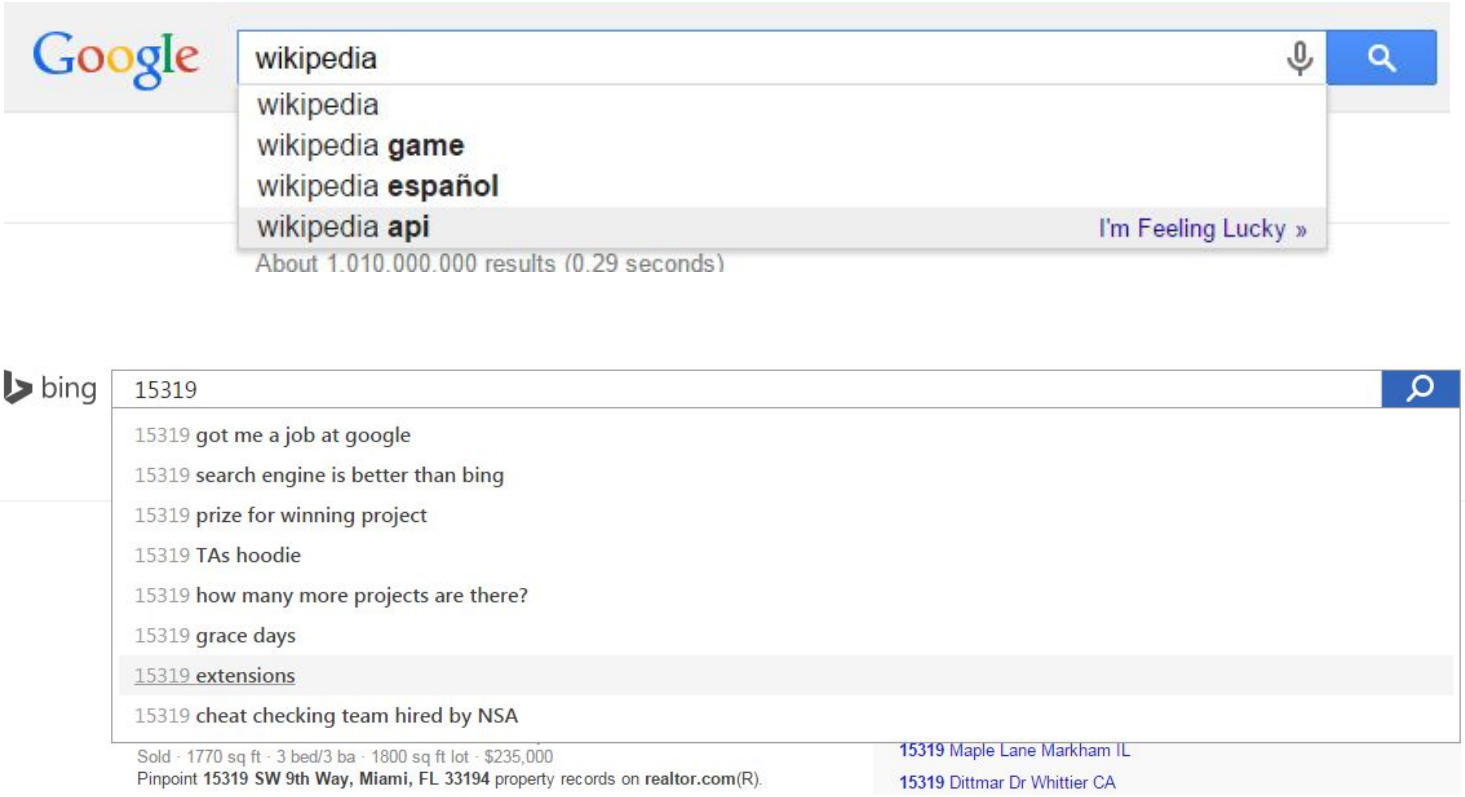
- Hadoop Distributed File System
- Open source version of Google File System



Project 4.1 - Input Text Predictor

- Suggest words based on phrases already typed





Project 4.1

- Steps for Input Text Predictor
  - Clean the input data
  - Perform the N-Gram count
  - Build the Statistical Language Model
  - Predict the next word given a phrase
- Have to use a Custom JAR in EMR
  - **CANNOT use EMR Streaming**

Construct an Input Text Predictor - 1

1. Given a language corpus
   - Wikipedia dataset (~8.6 GB)
2. Construct an n-gram model of the corpus
   - An n-gram is a phrase with n contiguous words
   - For example a set of 1,2,3,4,5-grams with counts:
     - this 1000
     - this is 500
     - this is a 125
     - this is a cloud 60
     - this is a cloud computing 20

Construct an Input Text Predictor - 2

3. Build a Statistical Language Model to calculate the probability of a word appearing after a phrase

$$\Pr(\text{word} \mid \text{phrase}) = \frac{\text{Count}(\text{phrase} + \text{word})}{\text{Count}(\text{phrase})}$$

$$\Pr(\text{is} \mid \text{this}) = \frac{\text{Count}(\text{this is})}{\text{Count}(\text{this})} = \frac{500}{1000} = 0.5$$

$$\Pr(a \mid \text{this is}) = \frac{\text{Count}(\text{this is } a)}{\text{Count}(\text{this is})} = \frac{125}{500} = 0.25$$

4. Load the probability data to HBase and predict the next word based on the probabilities

P4.1 Bonus

- MapReduce for word auto-completion
  - Given prefix, suggest the most possible words
  - Example: given “car”,
    - Possible words are: card, cart, Carnegie…
    - Suggest the top five words with highest probability
  - Store probability data to HBase and connect our front-end to submit

- Worth 10%

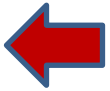
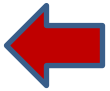
Recommendations

- Test for correctness with a small dataset first
- **Don't** start a new cluster for every job
  - EMR will charge you one hour of usage for instances even though your EMR job failed to start
- Version of Hadoop
  - It should match the version shown in the EMR AMI
- Start early and try the bonus

Using a Custom Jar in P4.1

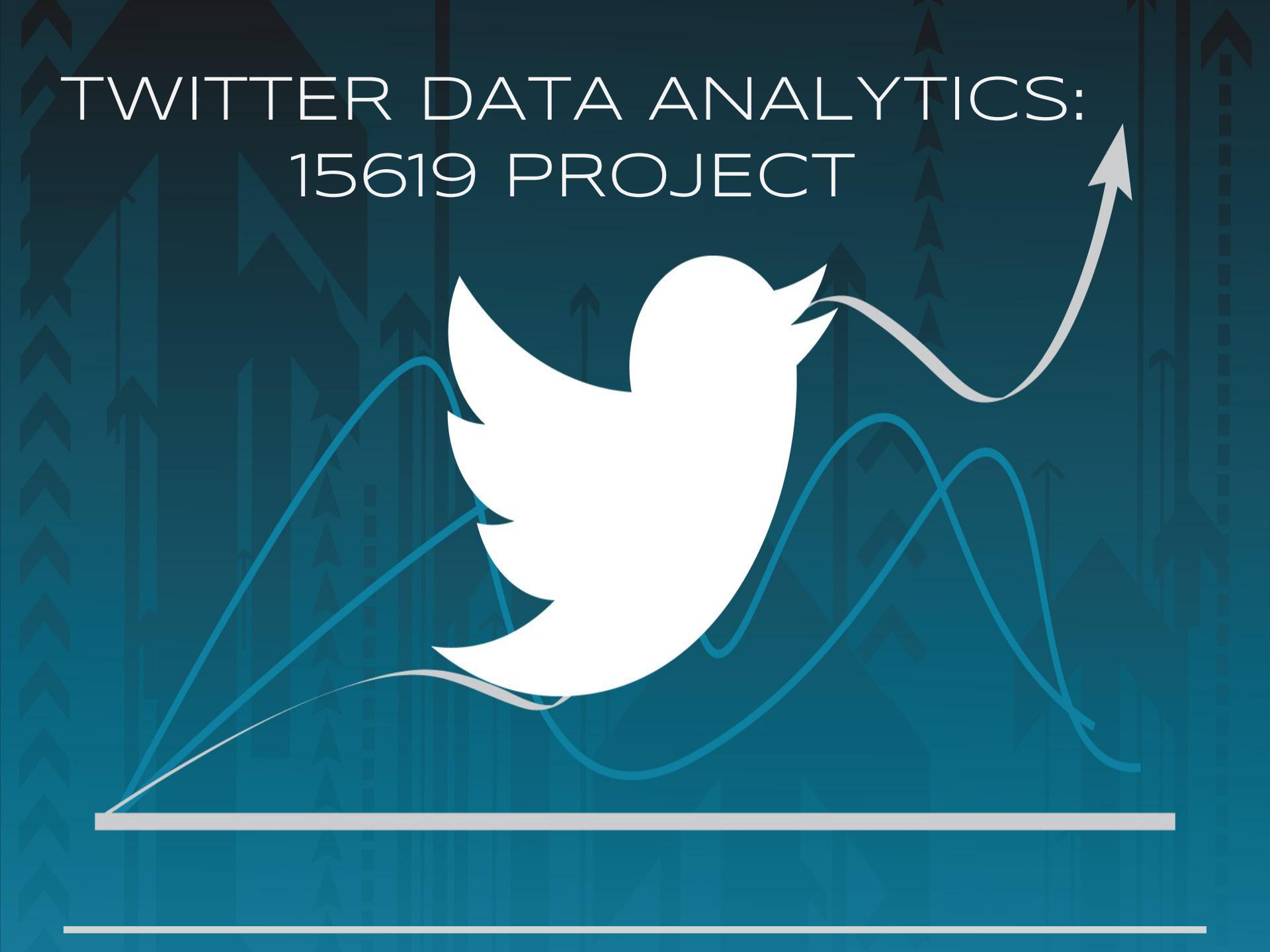
- What is a custom JAR
  - Customize your java MapReduce program
  - Run the MapReduce JAR in EMR
- Why custom JAR
  - More resources: HDFS/HBASE/S3
  - More job configuration flexibility
  - More control of how the resources are utilized

Upcoming Deadlines

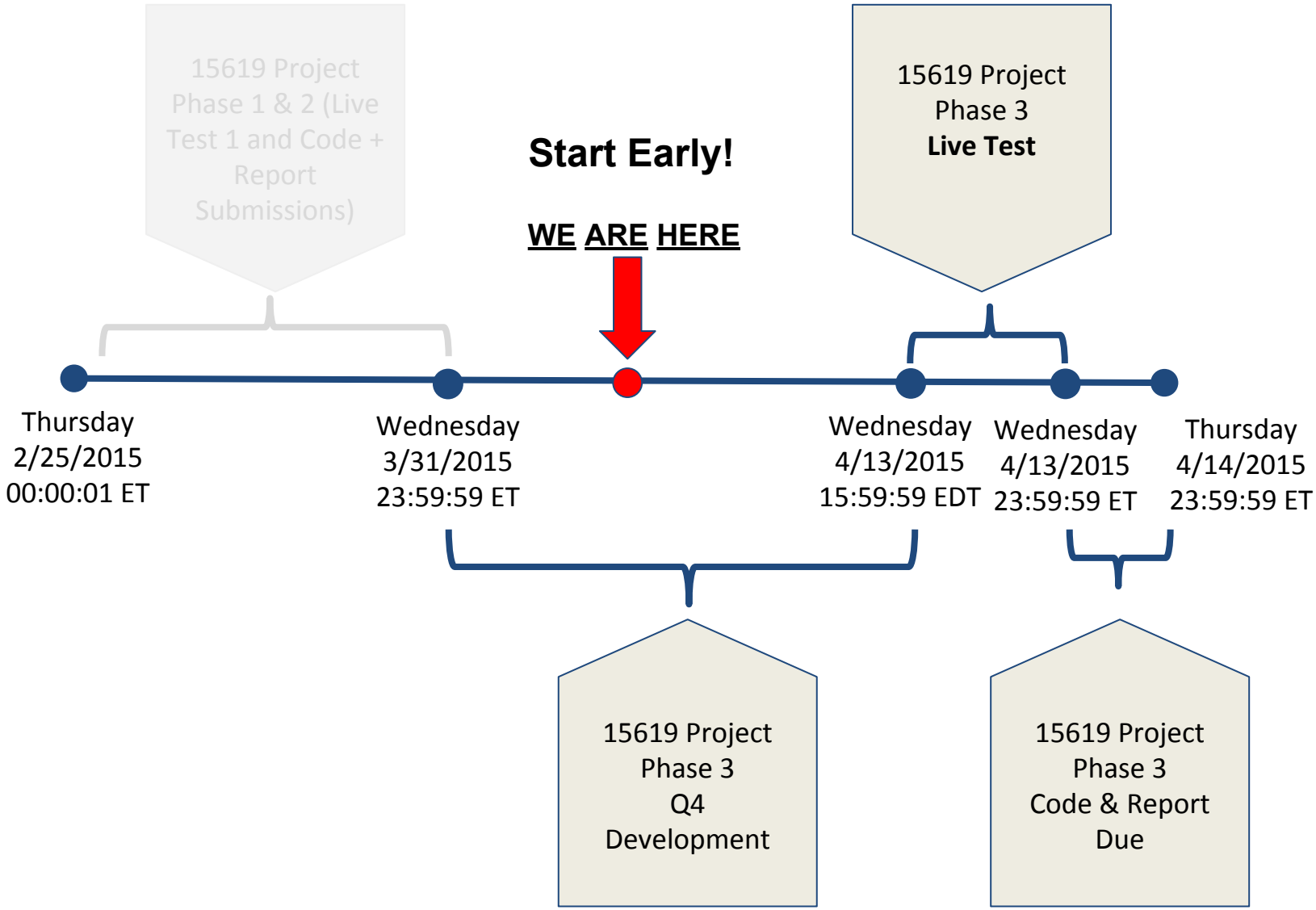
- Quiz 11: Unit 5 - Module 19, 20 
  - Due: 04/08/2016 11:59 PM Pittsburgh
- Project 4.1: Batch Processing with MapReduce
  - Due: 04/10/2016 11:59 PM Pittsburgh 
- 15619Project: Phase 3
  - Live-test DNS due: 04/13/2016 3:59 PM Pittsburgh
  - Code and report due: 04/14/2016 11:59 PM Pittsburgh

Questions?

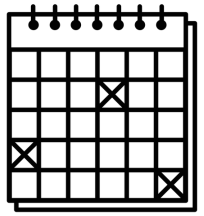
TWITTER DATA ANALYTICS: 15619 PROJECT



15619 Project Phase 3 Deadlines



15619Project Time Table



Phase (and query due)	Start	Deadline	Code and Report Due
Phase 1 Part 1 • Q1, Q2	Thursday 02/25/2016 00:00:01 EST	Wednesday 03/16/2016 23:59:59 EDT	Thursday 03/17/2016 23:59:59 EDT
Phase 2 • Q1, Q2, Q3	Thursday 03/17/2016 00:00:01 EDT	Wednesday 03/30/2016 15:59:59 EDT	
Phase 2 Live Test (Hbase/MySQL) • Q1, Q2, Q3	Wednesday 03/30/2016 18:00:01 EDT	Wednesday 03/30/2016 23:59:59 EDT	Thursday 03/31/2016 23:59:59 EDT
Phase 3 • Q1, Q2, Q3, Q4	Thursday 03/31/2016 00:00:01 EDT	Wednesday 04/13/2016 **15:59:59 EDT**	
Phase 3 Live Test • Q1, Q2, Q3, Q4	Wednesday 04/13/2016 **18:00:01 EDT**	Wednesday 04/13/2016 23:59:59 EDT	Thursday 04/13/2016 23:59:59 EDT

Results of Phase 2 Live Test

Congratulations to the teams on the leaderboard!

MySQL

Apollo	50
elder	50
MyHeartIsInTheWork	50
Sugoyi	50
OnePiece	50
JeanCloudVanDamme	50
SilverLining	50
ThreeKings	50
DaXiuZuiNiuBi	49.66
SteinsGate	49.51

HBase:

MyLittlePony	49.94
ccfighter	43.69
Hardship	42.41
MIB	41.14
RenRenYouOffer	36.13
SilverLining	35.44
YouKnowNothingJonSnow	35.39
YaoBuNengTing	35.35
elder	34.94
GiveSomeColorToCC	32.07

Common Issues

- Unexpected input or strange characters in the parameter fields?

  Remember that the live-test is a simulation of real world traffic. Try to make your front-end more robust so that it can handle any unexpected input without failure.

- AWS outage during live-test

  AWS Virginia data center encountered an outage in EC2 and ELB. Spot instances were terminated, API calls were throttled, could not start new instances.

  - 14 teams were allowed to participate in a make-up.

Phase 3

- One last query (Q4)
  - No ETL!
  - Serving write requests
  - Front end caching will not work during the live test
  - Two types of requests, set & get
- **Live Test!**
  - **Warmup, Q1, Q2, Q3, Q4, Mixed Q1-Q4**
    - **Each for 30 min**
  - **Choose HBase or MySQL**
    - **Submit One DNS**

Query 4: Tweet Server

There are five different parameters in the request URL for a request to /q4.

- tweetid (tweet ID)
- op (operation type)
- seq (sequence number)
- fields (comma separated fields involved in the request)
- payload (comma separated payload encoded in Base64)

Execute the requests of a tweetid by the seq (sequence number)

Query 4: Tweet Server

field	type	example
tweetid	long int	15213
userid	long int	156190000001
username	string	CloudComputing
timestamp	string	Mon Feb 15 19:19:57 2016
text	string	Welcome to P4!#CC15619#P3
hashtag	comma separated string	CC15619,P3
ip	string	128.2.217.13
coordinates	string	-75.14310264,40.05701649
repliedby	comma separated userid	156190000001,156190000002,156190000003
reply_count	long int	3
mentioned	comma separated userid	156190000004,156190000005,156190000006
mentioned_count	long int	3
favoritedby	comma separated userid	156190000007,156190000008,156190000009
favorite_count	long int	3
useragent	string	Mozilla/5.0 (iPhone; CPU iPhone OS ...)
filter_level	string	PG-13
lang	string	American

Query 4: Tweet Server

- SET Request `/q4?tweetid=15213&op=set&seq=1&fields=repliedby,reply_count&payload=MzM2NDE5MzE2NjUsMTc0Mjg5OTA1OTksOTQ5MDczNzc5NjQsMzkzMjIxMzU4NjQsMTg0NDA4MDg5NTUsNTE2MjU1MzMxOTgsOTI4MzA3NTgwNzQ=,Nw==`

- Response

  `TEAMID,TEAM_AWS_ACCOUNT_ID\n`

  `success\n`

Query 4: Tweet Server

- GET Request `/q4?tweetid=15213&op=get&seq=2&fields=repliedby,reply_count&payload=`

- Response

```
TEAMID,TEAM_AWS_ACCOUNT_ID\n
MzM2NDE5MzE2NjUsMTc0Mjg5OTA1OTksOTQ5MDczNzc5NjQsMzkzMjIxMzU4NjQsMTg0NDA4MDg5NTUsNTE2MjU1MzMxOTgsOTI4MzA3NTgwNzQ=\n
Nw==\n
```

Please ensure that you maintain strong consistency for Q4.

General Hints

- Don’t blindly optimize for every component, identify the bottlenecks using fine-grained profiling
- Use caches wisely: cache in HBase and MySQL is obviously important, storing everything in the frontend cache will lead to failure during the live test
- Review what we have learned in previous project modules
  - Scale out
  - Load balancing
  - Replication and sharding
  - Strong consistency (correctness is very important in Q4)
- Look at the feedback of your Phase 1 report!

Q4 Hints

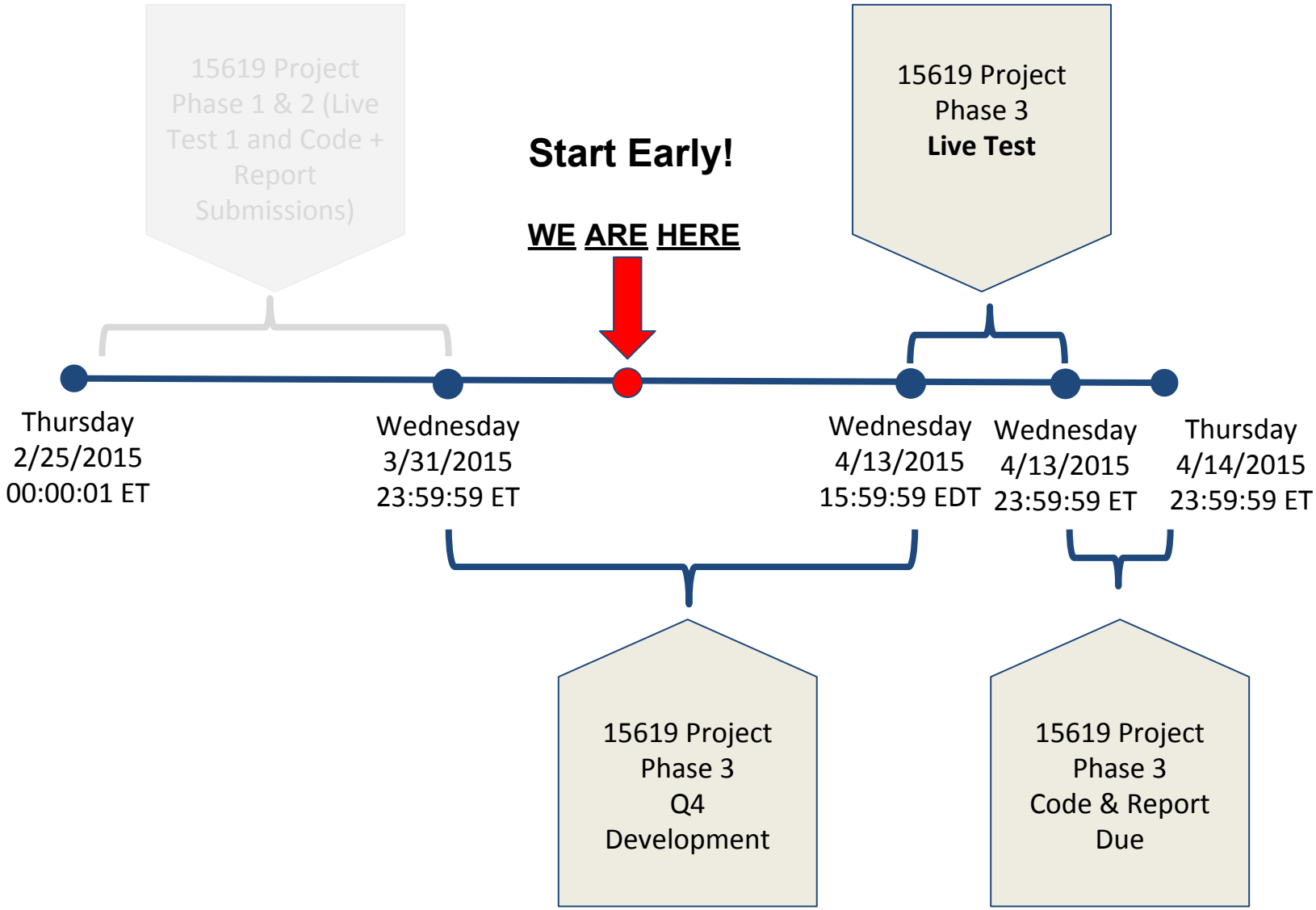
- MySQL DBs behind an ELB may require a forwarding mechanism.
- Consider forwarding the requests but pay attention to latency.
- Consider batch writes.
- Think about effective distributed caching techniques.
- Don’t block your frontend server.

Phase 3 Live Test

Time	Value	Target	Weight
6:00 pm - 6:30 pm	Warm-up (Q1 only)	-	0%
6:30 pm - 7:00 pm	Q1	27000	5%
7:00 pm - 7:30 pm	Q2	10000	15%
7:30 pm - 8:00 pm	Q3	6000	15%
8:00 pm - 8:30 pm	Q4	10000	15%
8:30 pm - 9:00 pm	Mixed Reads(Q1,Q2,Q3, Q4)	TBD	5+5+5+5 = 20%

Phase 3 report is worth 30% of the Phase 3 grade.

15619 Project Phase 3 Deadlines



What's due soon?

- Phase 3 Development
  - **Submission by 15:59 ET (Pittsburgh) Wed 04/13**
    - **Live Test from 6 PM to 10 PM EDT**
  - Fix Q1 - Q3 if you did not go well
  - New query Q4
  - Phase 3 counts for **60%** of the 15619Project grade
- Phase 3 Report
  - **Submission 23:59:59 ET (Pittsburgh) Thur 04/14**
  - Explain in detail the strategies you used
  - Difficulties you encountered even if you didn't get a good score