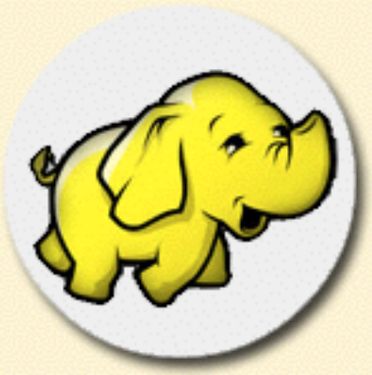


Welcome to

Big Data & Hadoop

Session

Session 3 - Adv. Map Reduce

+91-9538998962

sandeep@knowbigdata.com

WELCOME - KNOWBIGDATA

- Interact - Ask Questions
- Lifetime access of content
- Class Recording
- Cluster Access
- 24x7 support
- Real Life Project
- Quizzes & Certification Test
- 10 x (3hr class)
- Socio-Pro Visibility
- Mock Interviews

ABOUT ME

2014	**KnowBigData**	Founded
2014	**Amazon**	Built High Throughput Systems for Amazon.com site using in-house NoSql.
2012		
2012	**InMobi**	Built Recommender after churning 200 TB
2011		
	tBits Global	Founded tBits Global Built an enterprise grade Document Management System
2006		
	D.E.Shaw	Built the big data systems before the term was coined
2002		
2002	**IIT Roorkee**	Finished B.Tech somehow.

COURSE CONTENT

I	Understanding BigData, Hadoop Architecture
II	Environment Overview, MapReduce Basics
III	Adv MapReduce & Testing
IV	Analytics using Pig
V	Analytics using Hive
VI	NoSQL, HBASE
VII	Oozie, Mahout,
VIII	Zookeeper, Apache Storm
IX	Apache Flume, Apache Spark
X	YARN, Big Data Sets & Project Assignment

TODAY'S CLASS

- **Streaming Job**
  - **Description**
  - **Visualization**
  - **Hands ON**
- **Discusion on Problems**
  - **Limitations**
  - **Testing**
  - **Assignments**

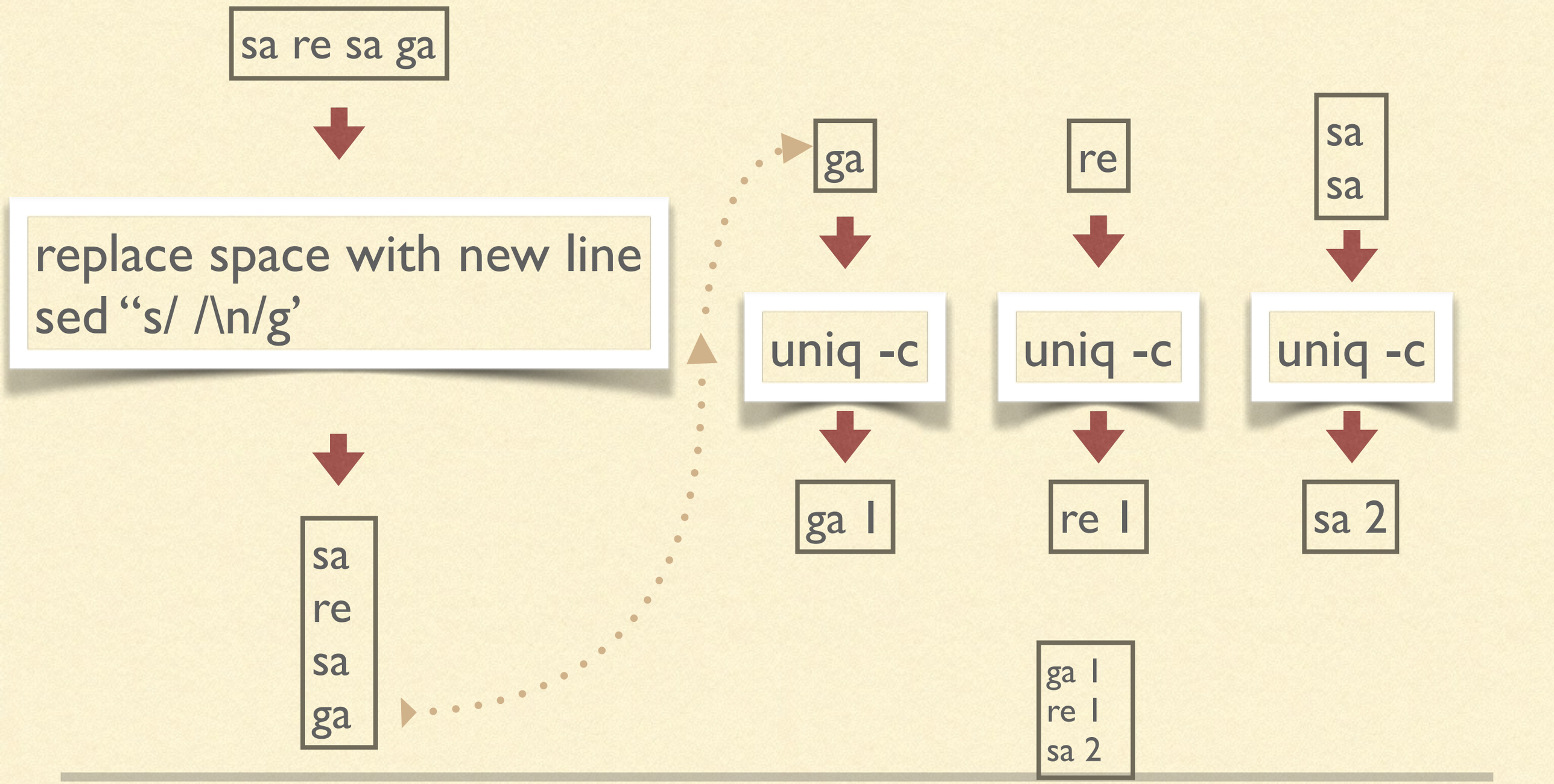
Streaming Job

A Hadoop Library which makes it possible to use *any binary* as mapper or reducer

Example

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar
  -input sgiri/wordcount/input/
  -output mylogin/output/
  -mapper 'sed "s/ /\n/g"'
  -reducer "/usr/bin/uniq -c"
```

Streaming Job



Ship a script

#mycmd.sh - clean up further

```
#!/bin/bash
sed "s/ /\n/g"|sed "s/[^a-zA-Z0-9]//g"|tr "A-Z" "a-z"
```

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar
-input sgiri/wordcount/input/
-output sgiri/wordcount/output7/
-mapper ./mycmd.sh
-reducer "/usr/bin/uniq -c"
-file mycmd.sh
```

STREAMING JOB - HANDS-ON

Problems Discussion

1. Frequencies of letters [a-z] - Do you need Map/Reduce?

Problems Discussion

1. Frequencies of letters [a-z] - Do you need Map/Reduce?

Without MR Approach:

- Create an integer array A of 26 size.
- Scan the text character by character
  - Increase A[0] for ‘a’ and A[25] for ‘z’
    - and others in between

Problems Discussion

1. Frequencies of letters [a-z] - Do you need Map/Reduce?

Without MR Approach:

- Create an integer array A of 26 size.
- Scan the text, character by character
  - Increase A[0] for ‘a’ and a[25] for ‘z’
    - and others in between

Problems?

1. Although memory/RAM will not be a limitation
2. The Network or Disk IO will be bottle neck
3. The CPU will be bottle

Problem 1: Discussion

1. Frequencies of letters [a-z] - Do you need Map/Reduce?

MR Approach:

Mapper

- Does the same thing as previous example
- Prints the array at the end

Reducer

- Sums up the values at the end

Mapper1

a 234
b 23028409
c 328782
d 37637
…26 line

Mapper2

a 12
b 122
c 90
d 22
…26 line

a [234, 12 ….]
b [23028409, 122 …]

Problem 2: Anagram

2. Find anagrams in a huge text. An anagram is basically a different arrangement of letters in a word.

Input:

"the cat act in tic tac toe."

Output:

cat, tac, act

Problem 2: Anagram

"the cat act in tic tac toe."

Mapper
Forms Key by Sorting Chars and value is actual word

eht	*the*
act	*cat*
act	*act*
in	*in*
cit	*tic*
act	*tac*
eot	*toe*

act	act
act	cat
act	tac
cit	tic
eht	the
eot	toe
in	in

act	act, cat, tac
cit	tic
eht	the
eot	toe
in	in

Reducer
Simply prints distinct values for a key

act, cat, tac
in
the
tic
toe

Problem 3a: DNA

3a. A file contains the DNA sequence of people. Find all the people who have same DNAs.

Input:

"User1 ACGT"
"User2 TGCA"
"User3 ACG"
"User4 ACGT"
"User5 ACG"
"User6 AGCT"

Output:

User1, User4
User2
User3, User 5
User6

select dna, concat(users) from mytable group by dna

Problem 3b: DNA

3b. A file contains the DNA sequence of people. Find all the people who have same or mirror image of DNAs.

Input:

"User1 ACGT"
"User2 TGCA"
"User3 ACG"
"User4 ACGT"
"User5 ACG"
"User6 AGCT"

Output:

User1, User2, User4
User3, User 5
User6

Problem 3: DNA

User1 ACGT
User2 TGCA
User3 ACG
User4 ACGT
User5 ACG
User6 AGCT
User7 TCGA

Mapper
Key:
Smaller (DNA , Reverse)
Value: User Id

min(ACGT,TCGA) =ACGT

ACGT	USER1
ACGT	*User2*
ACG	*User3*
ACGT	*User4*
ACG	*User5*
AGCT	*User6*
AGCT	*User7*

ACG	User3
ACG	User5
ACGT	USER1
ACGT	User2
ACGT	User4
AGCT	User6
AGCT	User7

ACG	User3,User5
ACGT	USER1,User2, User 4
AGCT	User6,User7

Reducer
Simply prints Users

User3,User5
USER1,User2, User 4
User6,User7

Problem 4: Voting

4. In an unusual democracy, everyone is not equal. The vote count is a function of worth of the voter. Though everyone is voting for each other.
As example, if A with a worth of 5 and B with a worth of 1 are voting for C, the vote count of C would be 6.
You are given a list of people with their value of vote. You are also given another list describing who voted for who all.

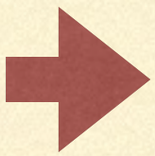
Find out what is the vote count of everyone?

List1

Voter	*Votee*
A	C
B	C
C	F
A	B

List2

Person	*Worth*
A	5
B	1
C	11



Result

Person	*VoteCount*
A	0
B	5
C	6
F	11

MAP / REDUCE

Problem 4: Voting

List1

Voter	*Votee*
A	C
A	B
B	C
C	F

List2

Person	*Worth*
A	5
B	1
C	11
D	12

Mapper

Merged
The two files

A	C
A	B
B	C
C	F
A	5
B	1
C	11
D	12

Hadoop

A	[5,B,C]
B	[1,C]
C	[11,F]
D	[12]

Map 2

B	5
C	5
A	0
C	1
B	0
F	11
C	0
D	0

hdp

A	[0]
B	[0,5]
C	[5,1,0]
D	[0]
F	[11]

Reducer

A	0
B	5
C	6
D	0
F	11

Limitation / When Not to Use?

- If the job can be done by a single machine in reasonable time
- Computation depends on previously computed values.
  - e.g. Fibonacci Series
- Full-text indexing or ad hoc searching
- Algorithms depend on shared global state

Testing

1. First test on very small data
2. Separately Test Mapper and Reducer
3. Steaming Job's Mapper could be tested with simple unix command:
    1. cat **inputfile** | ***mymapper*** | sort | my***reducer*** >> ***outputfile***
4. Use MRUnit (Java MR Session, Next class)
5. To test predictions, you may want to test the part

MAP / REDUCE - JAVA

```java
public class StubMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

  @Override
  public void map(LongWritable key, Text value, Context context)
      throws IOException, InterruptedException {

    /*
     * TODO implement
     */

  }
}
```

```java
public class StubReducer extends Reducer<Text, IntWritable, Text, DoubleWritable> {

  @Override
  public void reduce(Text key, Iterable<IntWritable> values, Context context)
      throws IOException, InterruptedException {

    /*
     * TODO implement
     */

  }
}
```

```java
public class StubDriver {
    public static void main(String[] args) throws Exception {
        Job job = new Job();
        job.setMapperClass(StubMapper.class);
        job.setReducerClass(StubReducer.class);

        /*
```

Assignment - Problem 1

Code M/R for all the problems in your favourite language.

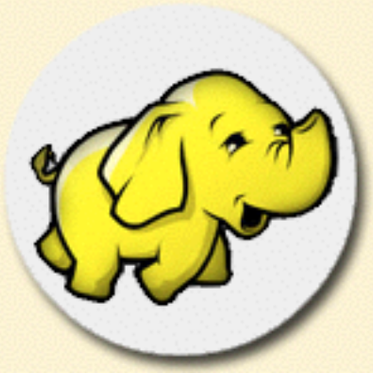
Assignment - Problem 2

Based on the content from a very large text archive, formulate the next words recommendation.

For each word, prepare a top 5 recommendations of the word that would go next.

happy birthday, newyear, marriage
how are, do, did
…

Download the content from wikipedia using this:
http://www.evanjones.ca/software/wikipedia2text.html



Big Data & Hadoop

Thank you.

+91-9538998962

sandeep@knowbigdata.com