MegaFon Course: Big Data

Hadoop MapReduce

Dral Alexey, aadral@bigdatateam.org
CEO at BigData Team, http://bigdatateam.org/
https://www.facebook.com/bigdatateam/

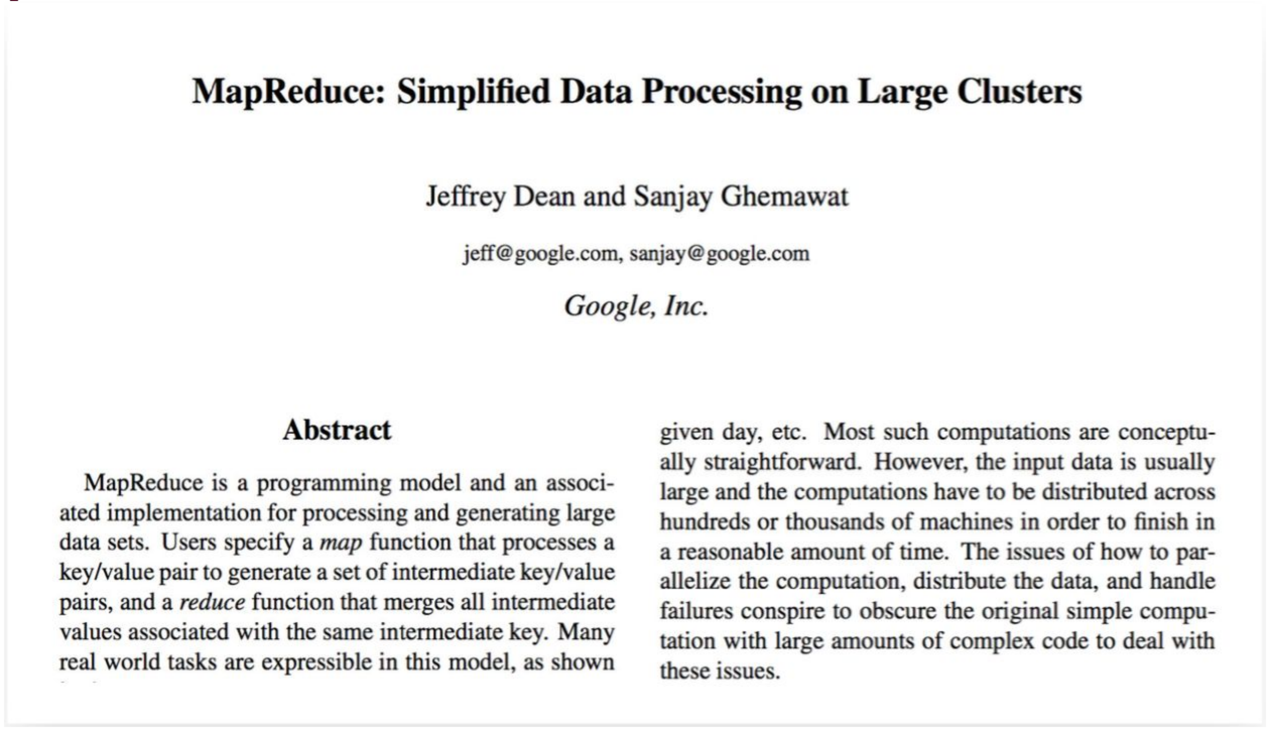
13.06.2019, Moscow, Russia

Outline

- MapReduce (MR)
- Распределенный shell и формальная модель MR
- Fault Tolerance
- MapReduce Streaming
- MapReduce Word Count

MapReduce (MR)

MapReduce



MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

Abstract

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a *map* function that processes a key/value pair to generate a set of intermediate key/value pairs, and a *reduce* function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown

given day, etc. Most such computations are conceptually straightforward. However, the input data is usually large and the computations have to be distributed across hundreds or thousands of machines in order to finish in a reasonable amount of time. The issues of how to parallelize the computation, distribute the data, and handle failures conspire to obscure the original simple computation with large amounts of complex code to deal with these issues.

MapReduce: Simpli ed Data Processing on Large Clusters, Symposium on Operating Systems Design and Implementation (OSDI, 2004)

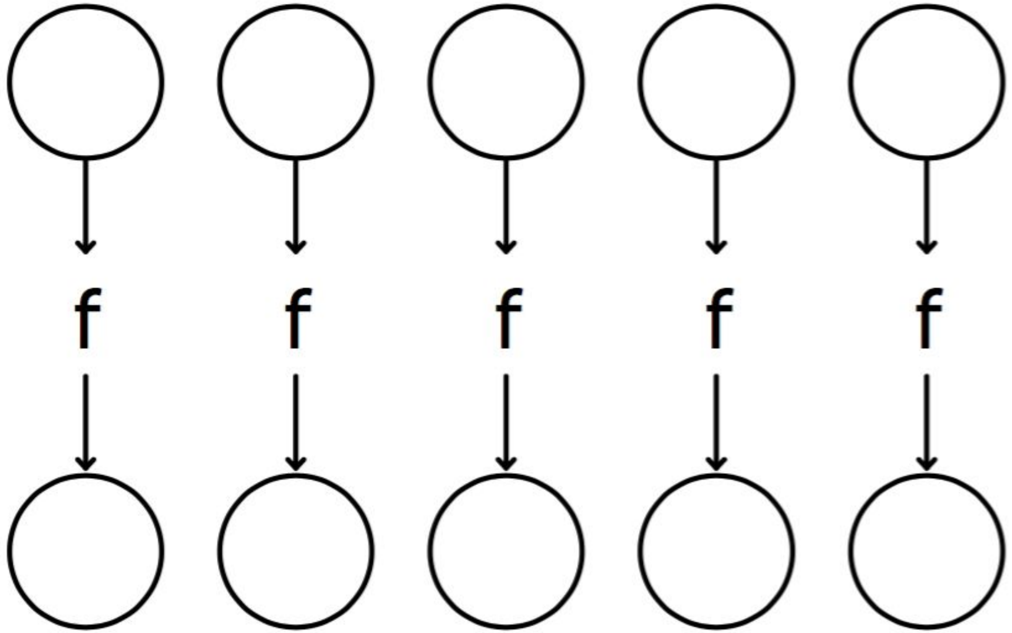
BIGDATA TEAM

Jeffrey Dean

- Когда Jeff Dean разрабатывает ПО, он сначала создает бинарник, а потом пишет исходный код как документацию.
- Однажды Jeff Dean не прошел тест Тьюринга, потому что корректно посчитал 203 число Фибоначчи менее чем за 1 секунду.
- Скорость, с которой Jeff Dean разрабатывает ПО выросла в 40 раз в конце 2000, когда он обновил свою клавиатуру до USB2.0.
- Вы используете только 10% мозга. Остальные 90% используются под запуск MapReduce задач Джефа.

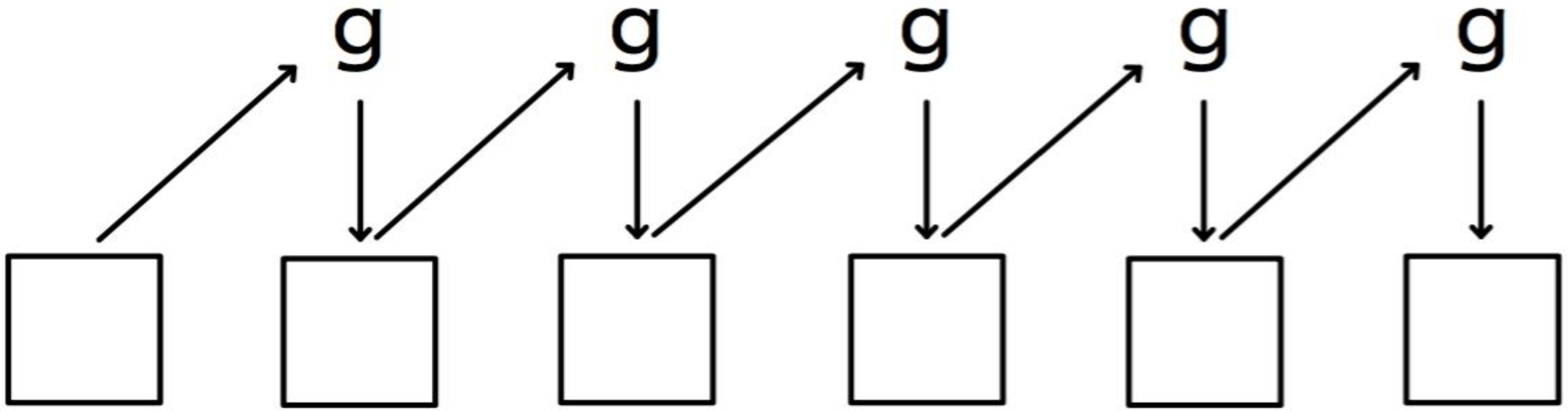
https://www.quora.com/What-are-all-the-Jeff-Dean-facts

Map



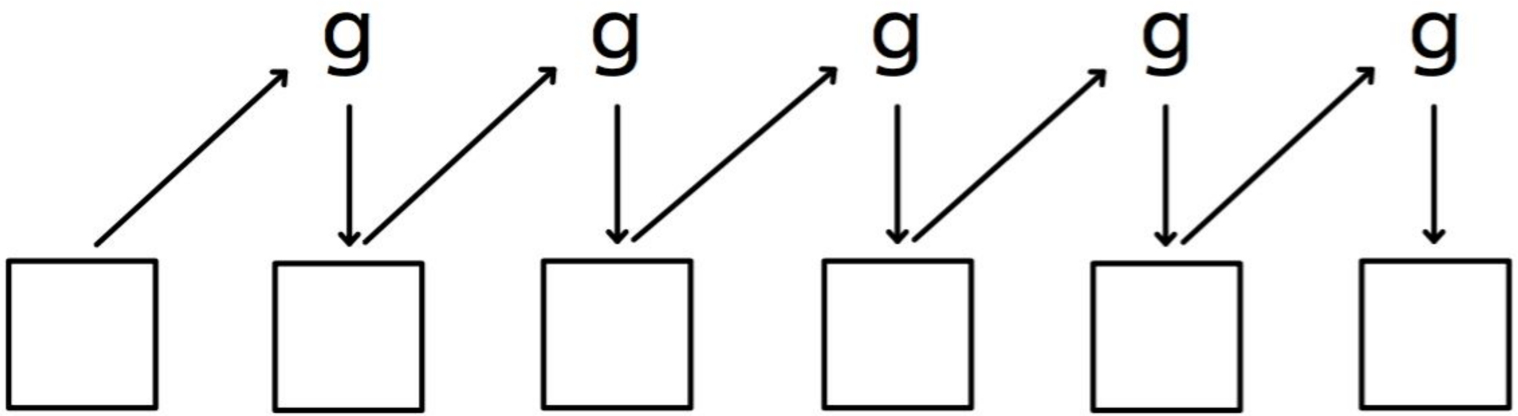
```
>>> map(lambda x: x*x, [1,2,3,4])
???
```

Fold / Reduce / Aggregate



```
>>> reduce(operator.sum, [1, 4, 9, 16])
???
```

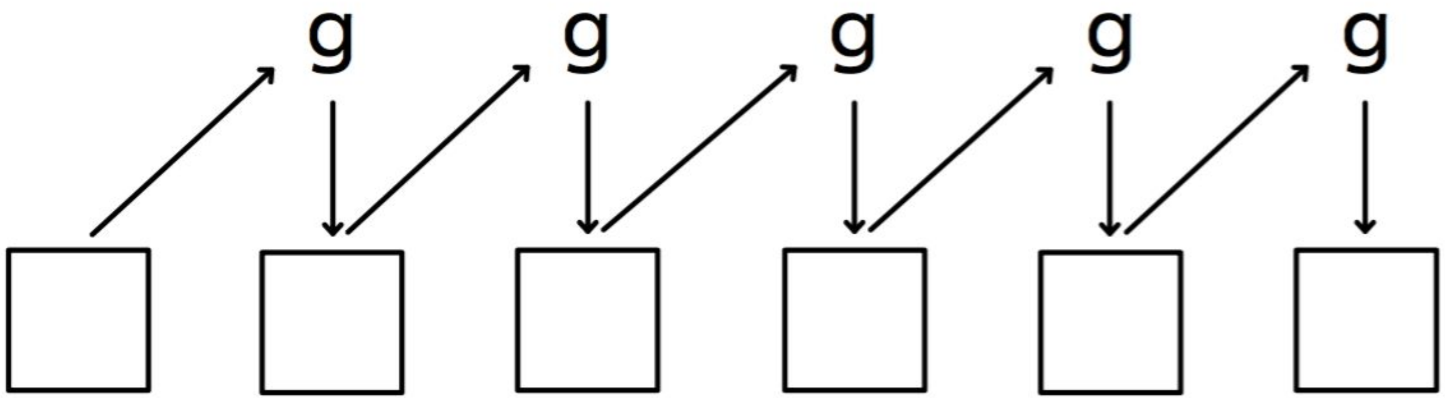
Fold / Reduce / Aggregate



```
>>> reduce(operator.sum, [1, 4, 9, 16])
>>> reduce(operator.sum, [5, 9, 16]) >>>
reduce(operator.sum, [14, 16])

30
```

Fold / Reduce / Aggregate

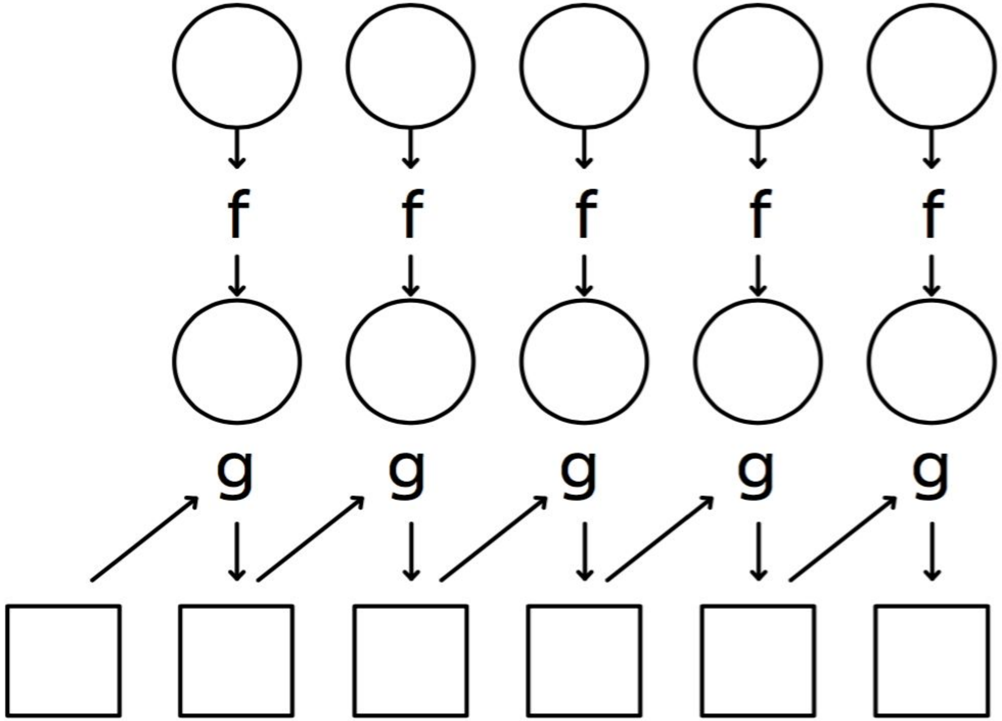


```
>>> average = lambda x, y: (x + y) / 2.
```

```
>>> reduce(average, [1, 2, 3])
2.25
```

```
>>> reduce(average, [3, 2, 1])
1.75
```

MapReduce

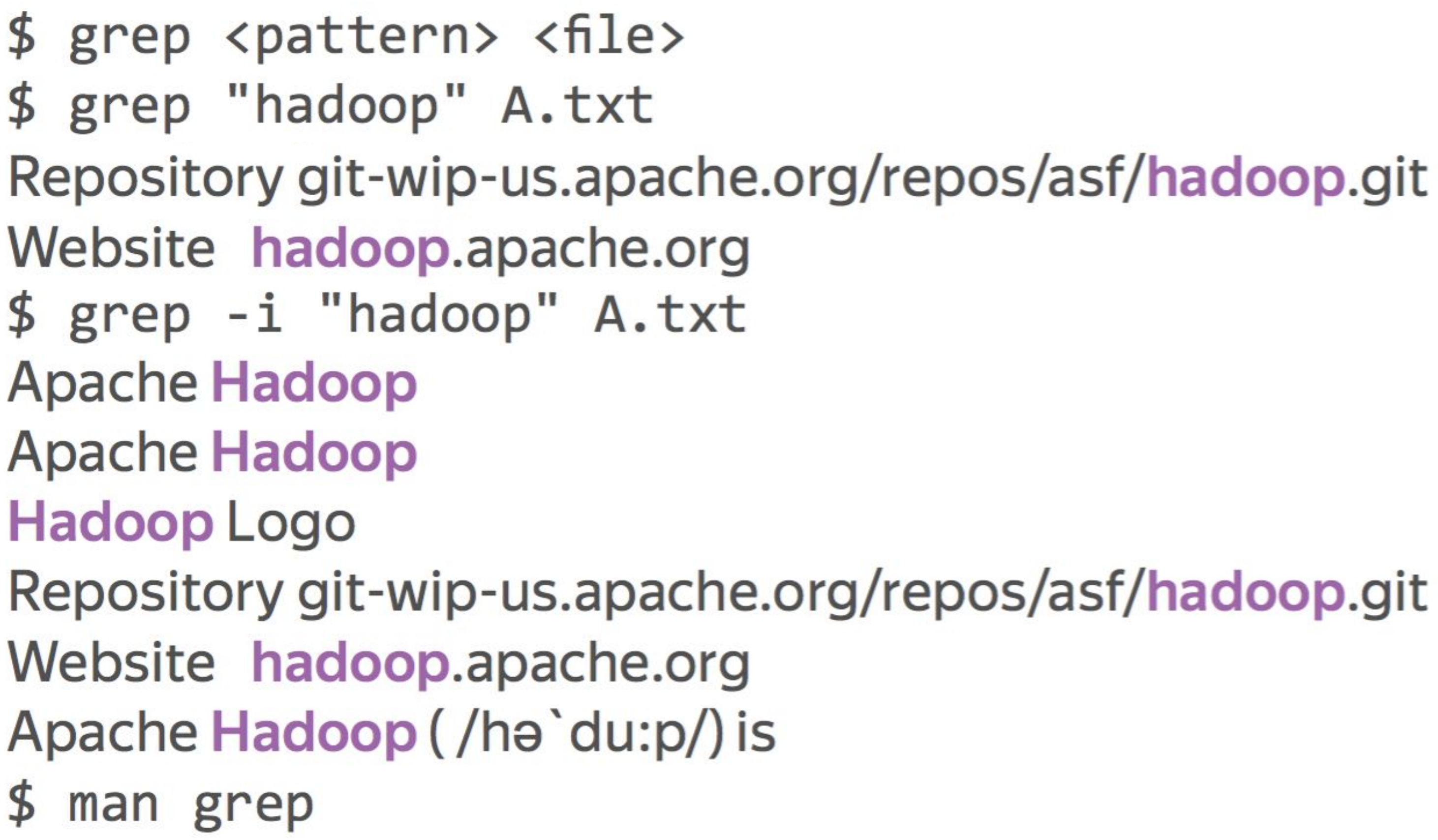


```
>>> reduce(operator.add, map(lambda x: x*x, [1, 2, 3, 4]))

30
```

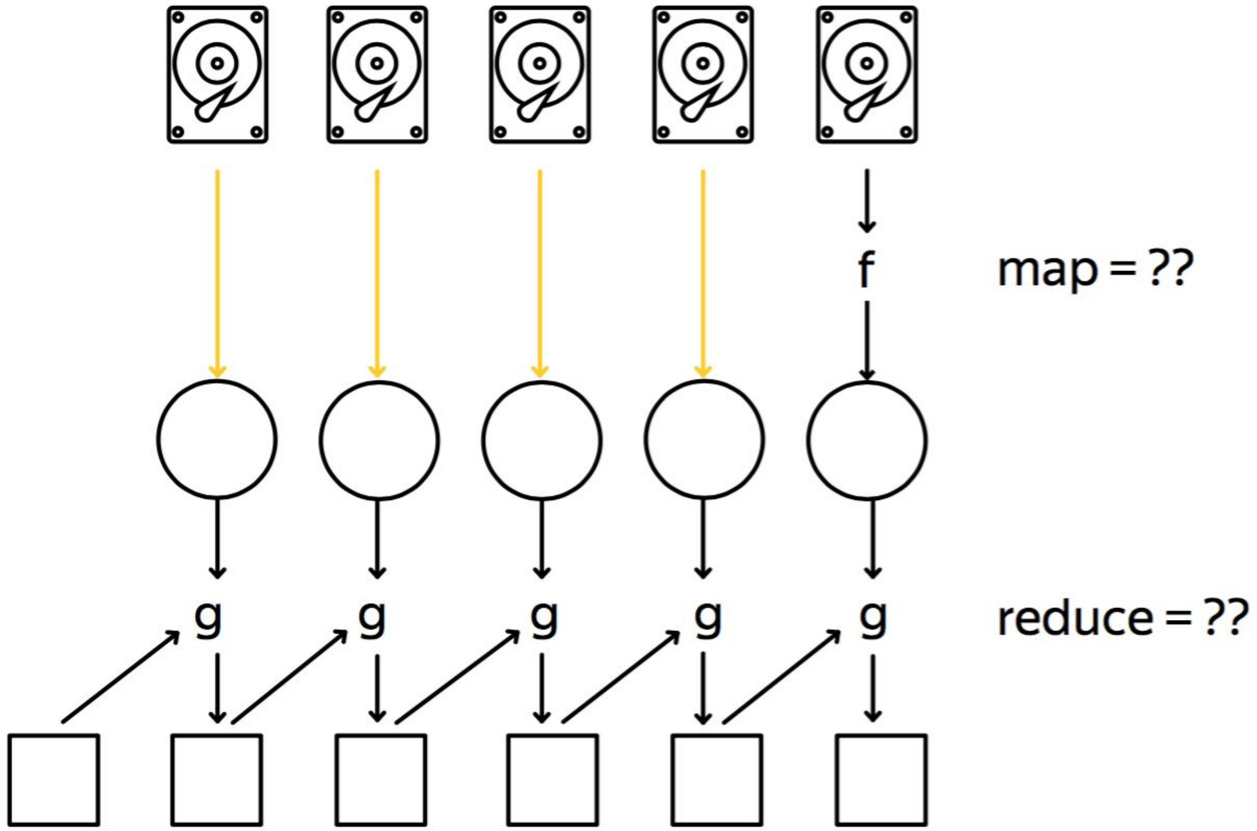
Распределенный shell и формальная модель MR

grep

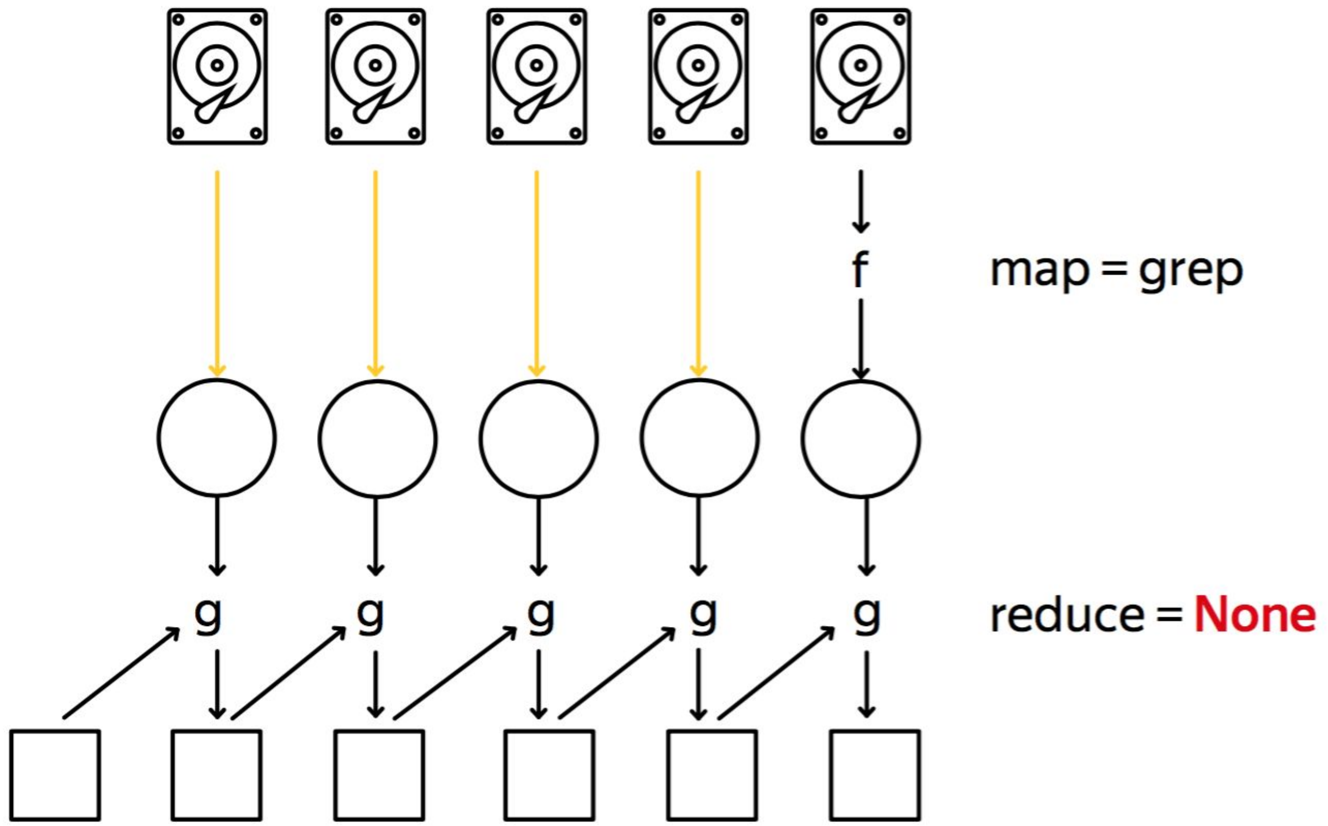


```
$ grep <pattern> <file>
$ grep "hadoop" A.txt
Repository git-wip-us.apache.org/repos/asf/hadoop.git
Website  hadoop.apache.org
$ grep -i "hadoop" A.txt
Apache Hadoop
Apache Hadoop
Hadoop Logo
Repository git-wip-us.apache.org/repos/asf/hadoop.git
Website  hadoop.apache.org
Apache Hadoop ( /həˋdu:p/) is
$ man grep
```

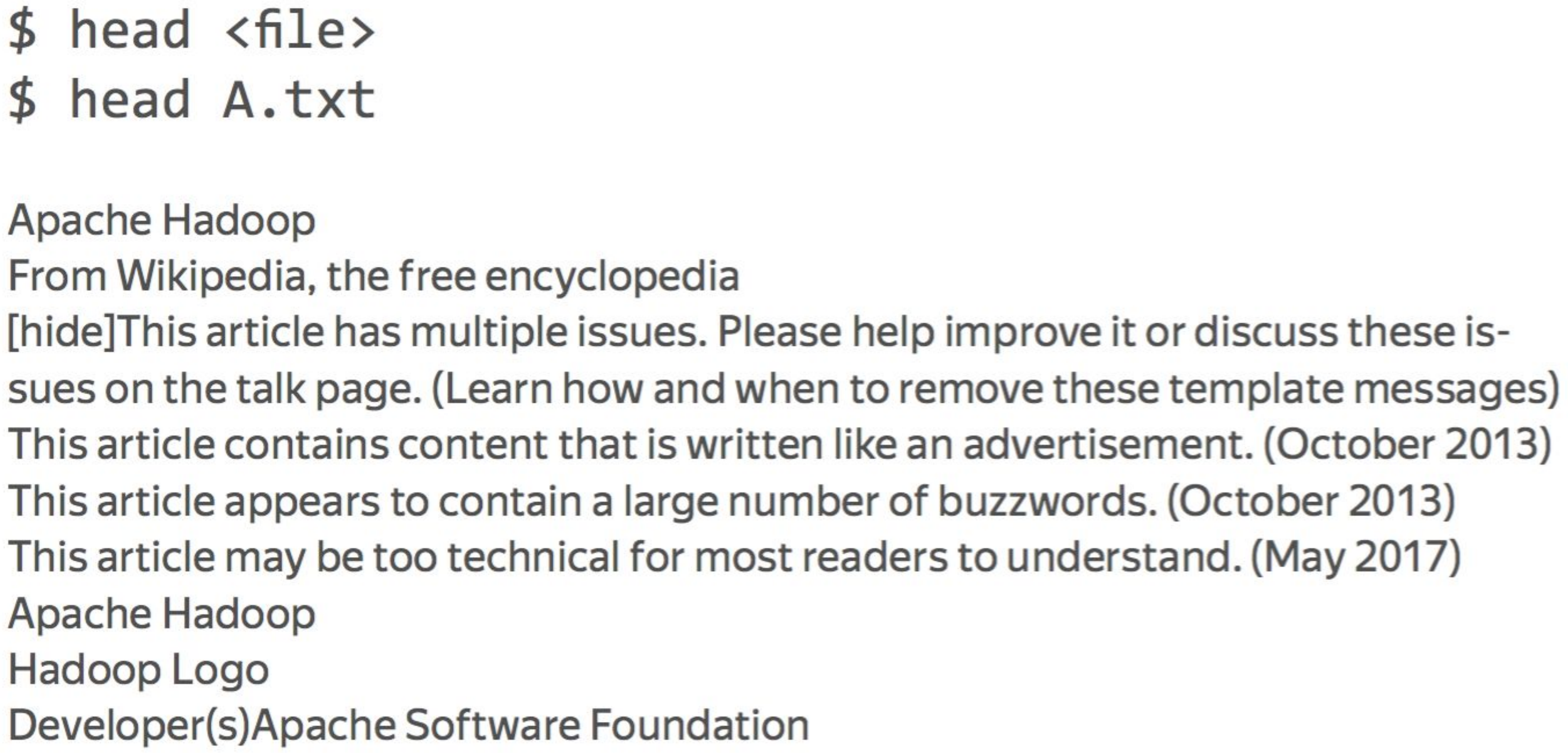
Distributed Shell: grep



Distributed Shell: grep



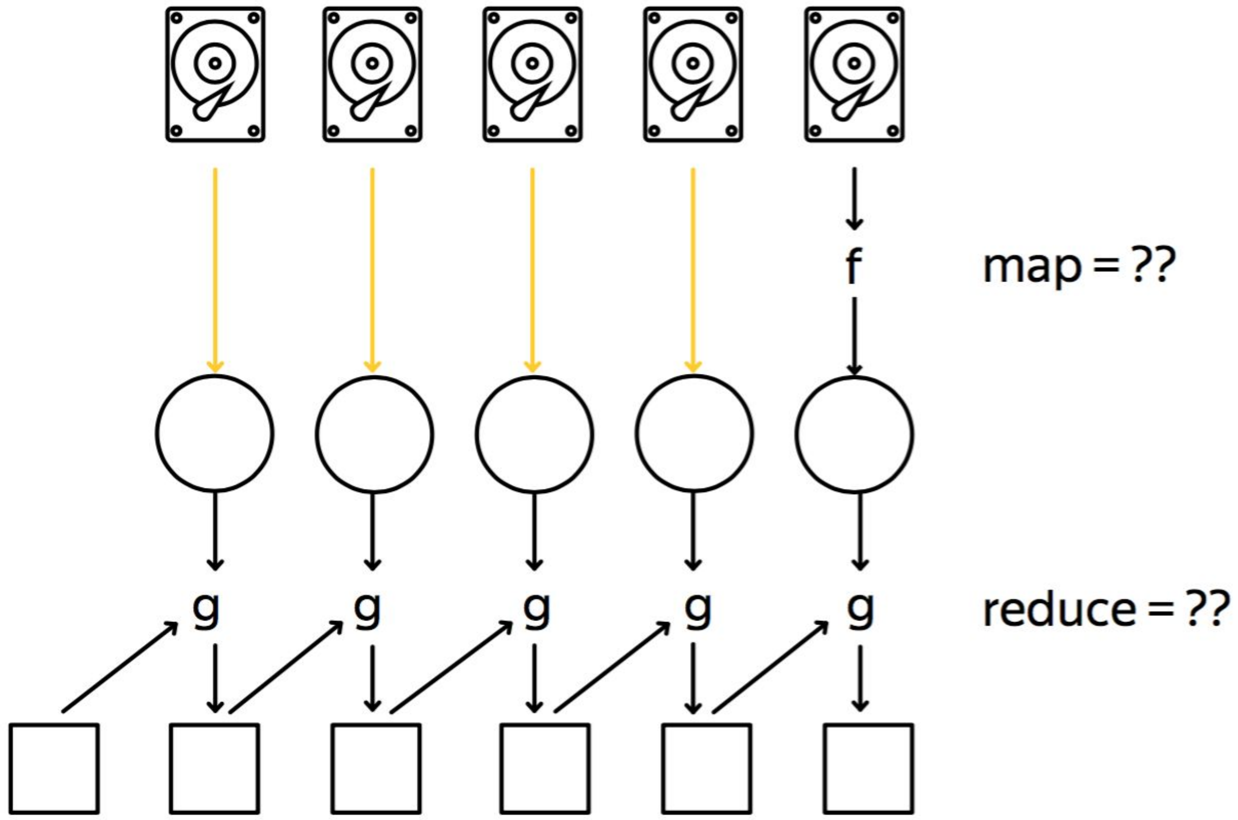
head



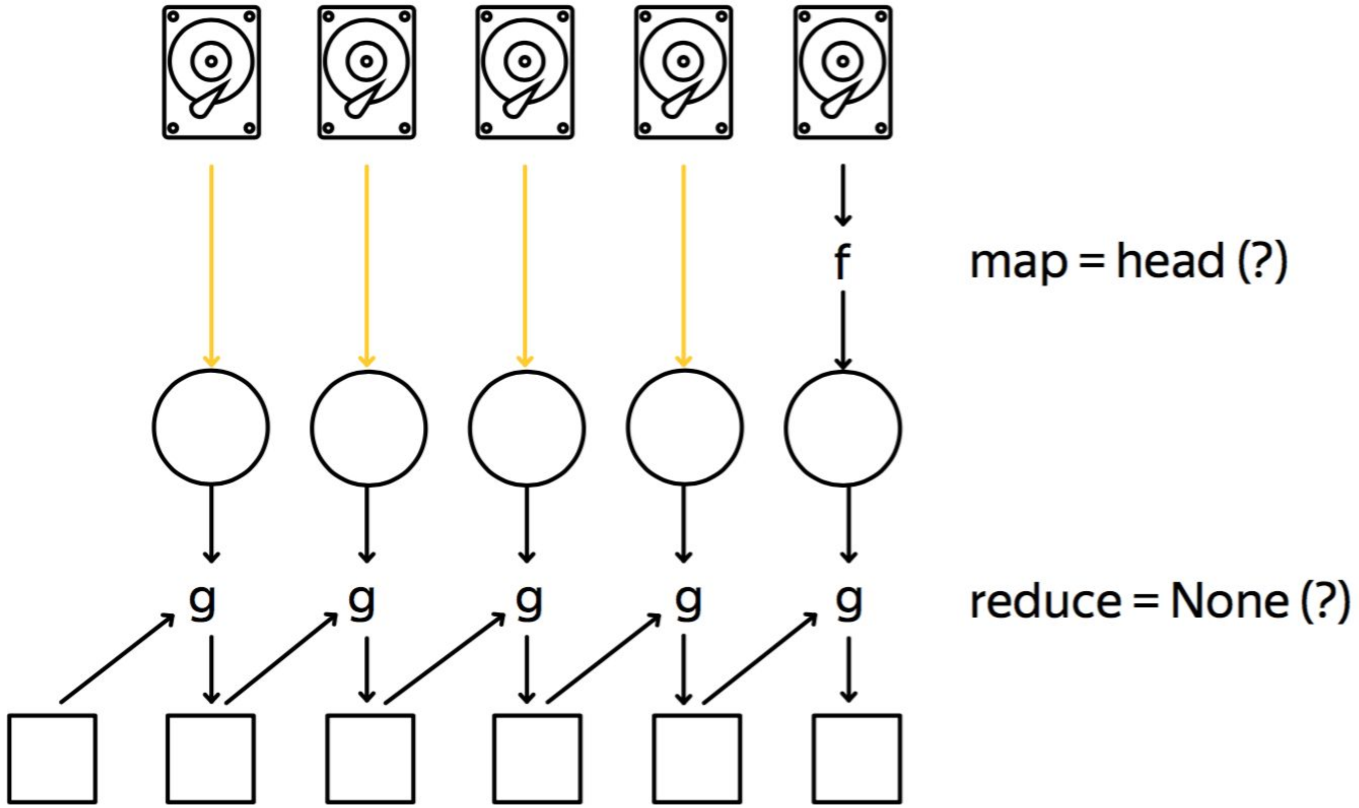
```
$ head <file>
$ head A.txt

Apache Hadoop
From Wikipedia, the free encyclopedia
[hide]This article has multiple issues. Please help improve it or discuss these is-
sues on the talk page. (Learn how and when to remove these template messages)
This article contains content that is written like an advertisement. (October 2013)
This article appears to contain a large number of buzzwords. (October 2013)
This article may be too technical for most readers to understand. (May 2017)
Apache Hadoop
Hadoop Logo
Developer(s)Apache Software Foundation
```

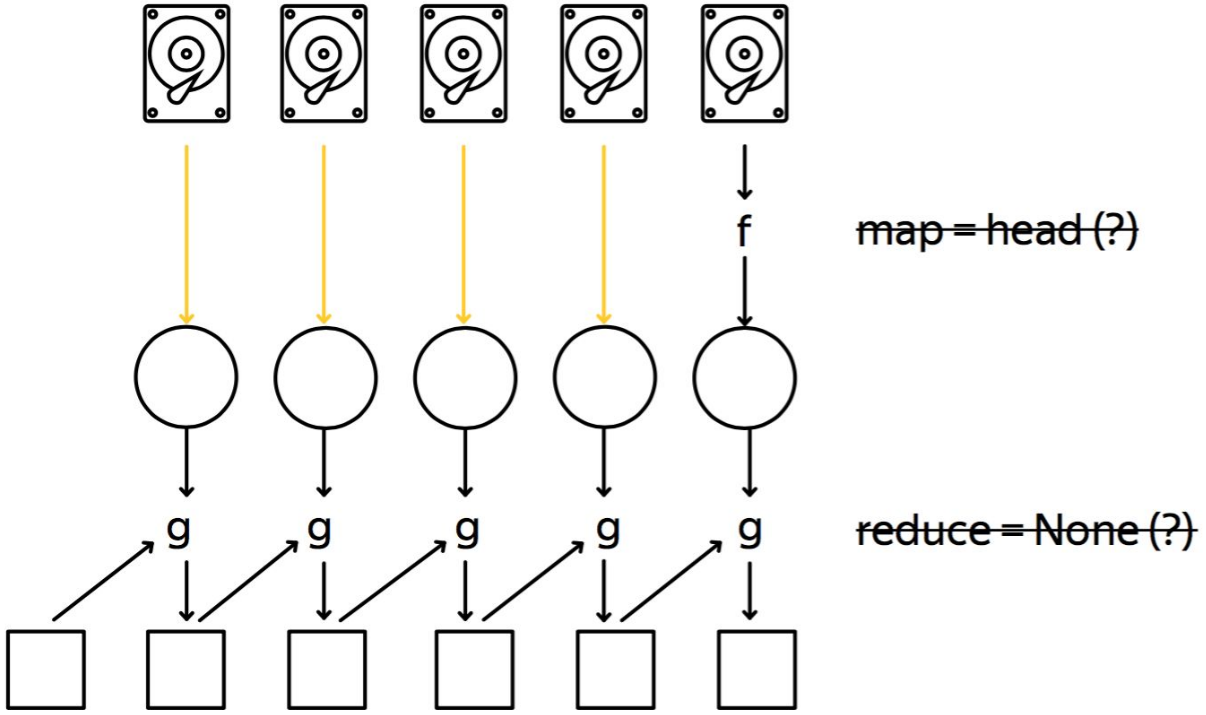
Distributed Shell: head



Distributed Shell: head

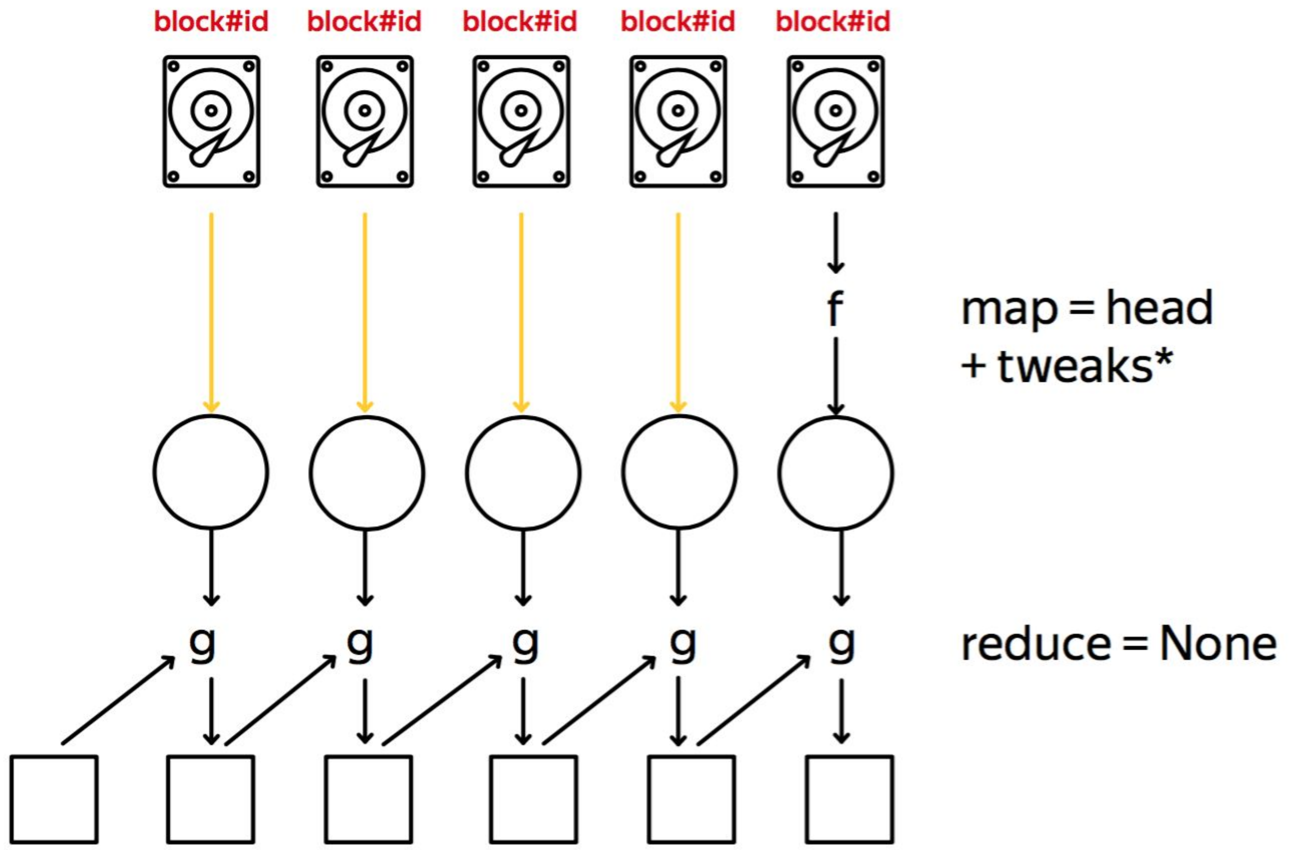


Distributed Shell: head

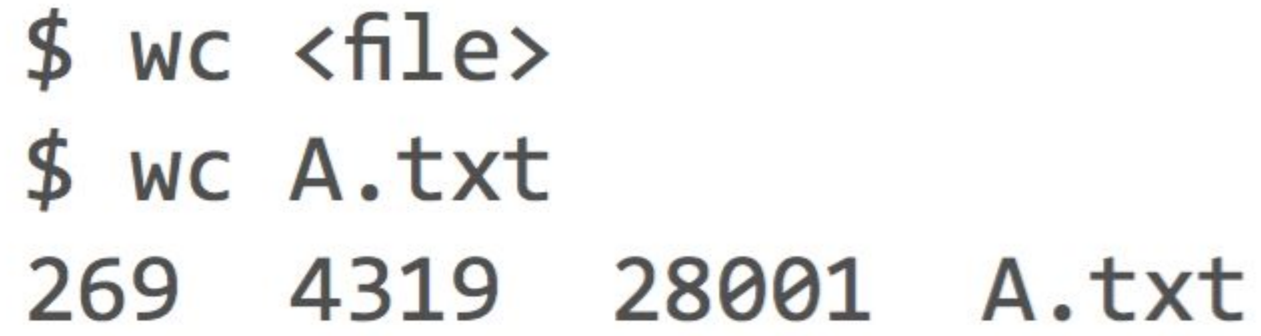


```
hdfs dfs -text distributed_A.txt | head
```

Distributed Shell: head

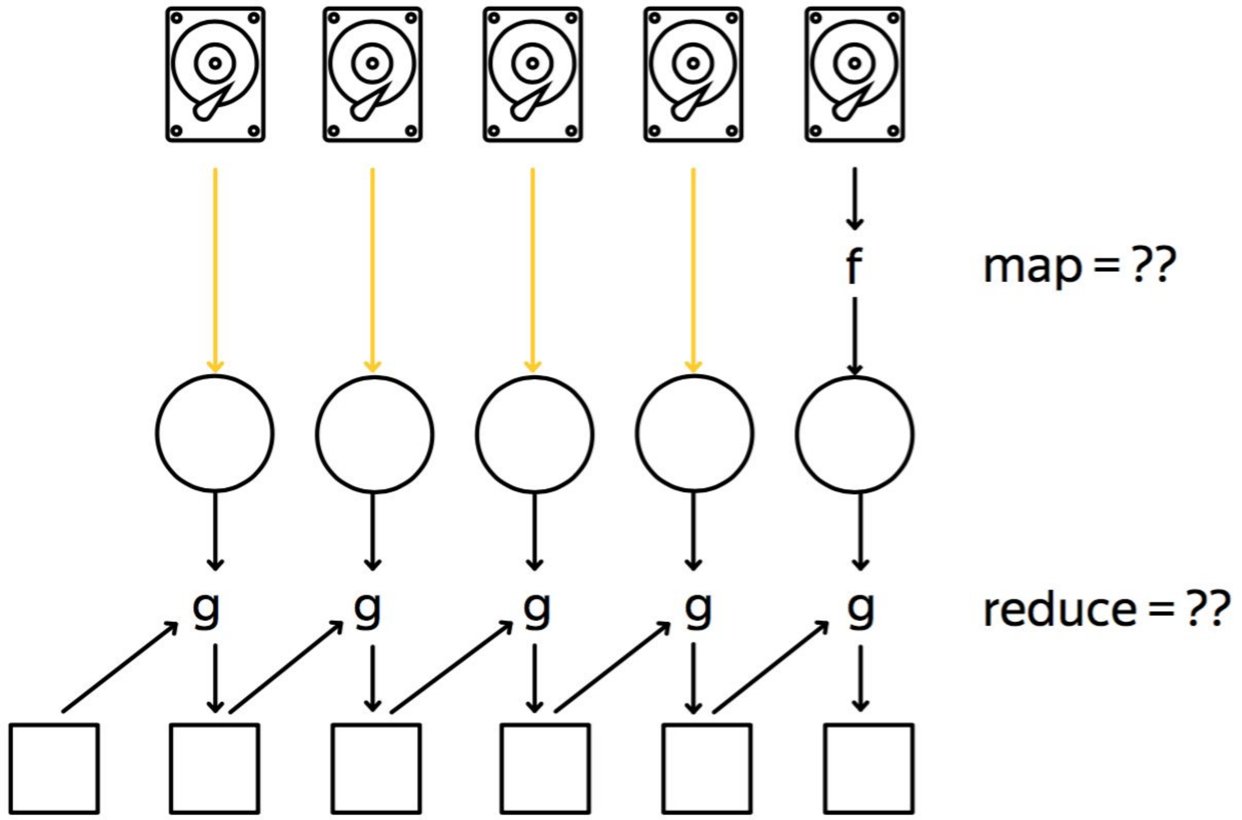


wc

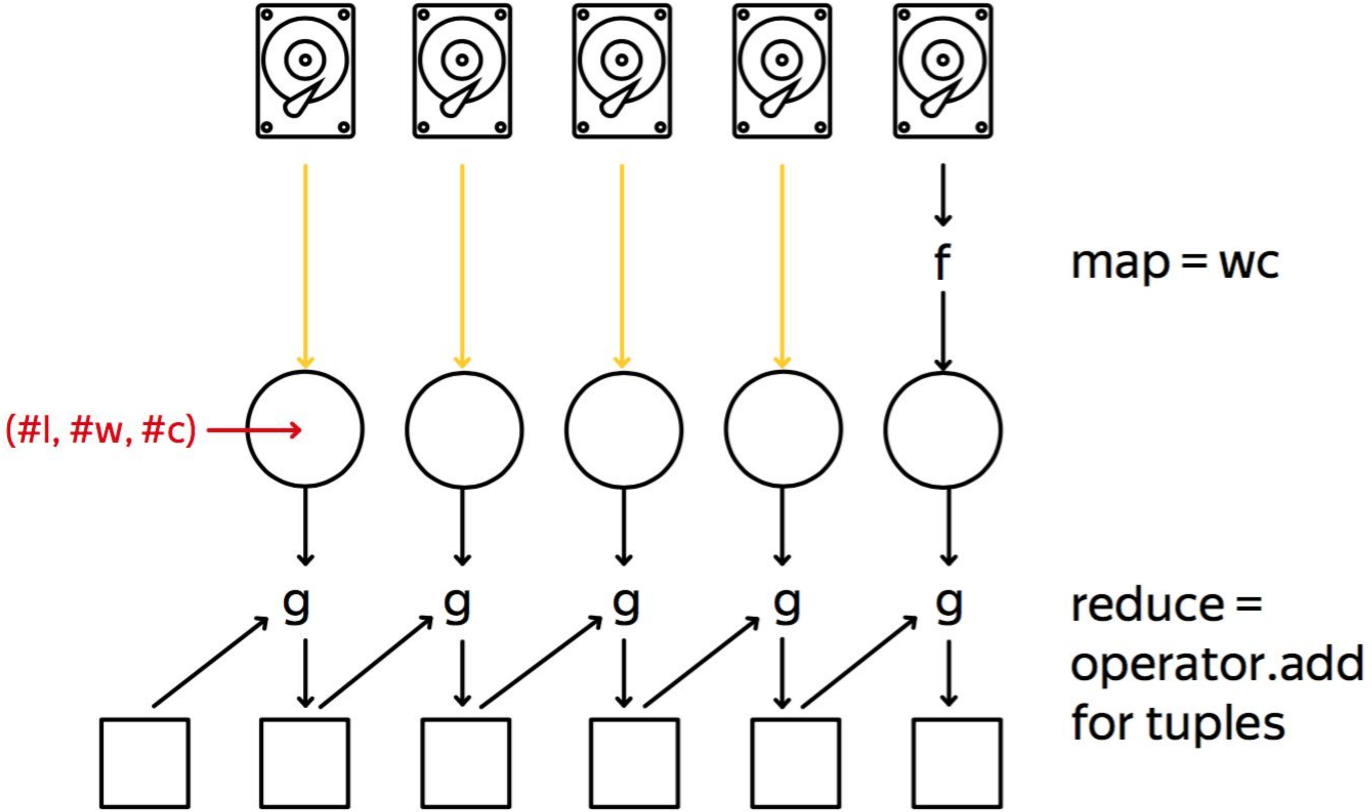


```
$ wc <file>
$ wc A.txt
269  4319  28001  A.txt
```

Distributed Shell: wc



Distributed Shell: wc



Word Count

Apache Hadoop (/həˋdu:p/) is an open-source software framework used for distributed storage and processing of dataset of big data using the MapReduce programming model. It consists of computer clusters built from commodity hardware.

All the modules in Hadoop are designed with a fundamental assumption that hardware failures are common occurrences and should be automatically handled by the framework...

$$\downarrow$$

```
'the': 3, 'of': 3, 'hadoop': 2, …
```

Word Count

```
one computer: cat * | tr ' ' '\n' | sort | uniq -c
```

Word Count

```
distributed: cat * | tr ' ' '\n' | sort | uniq -c
```

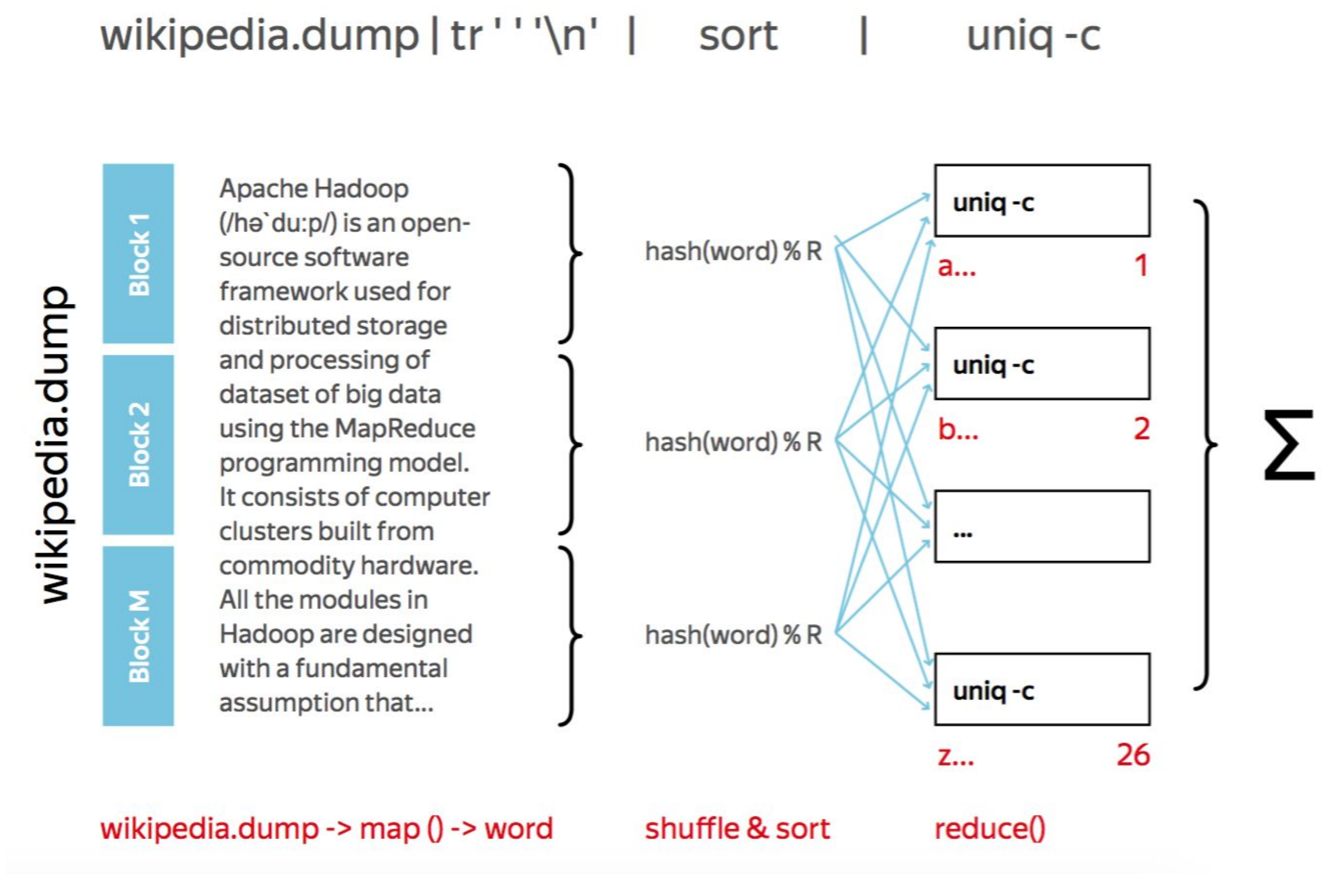
~~map=sort~~

~~reduce=sort~~ (не поместится в RAM / на диске)

BIGDATA TEAM

Map → Shuffle & Sort → Reduce

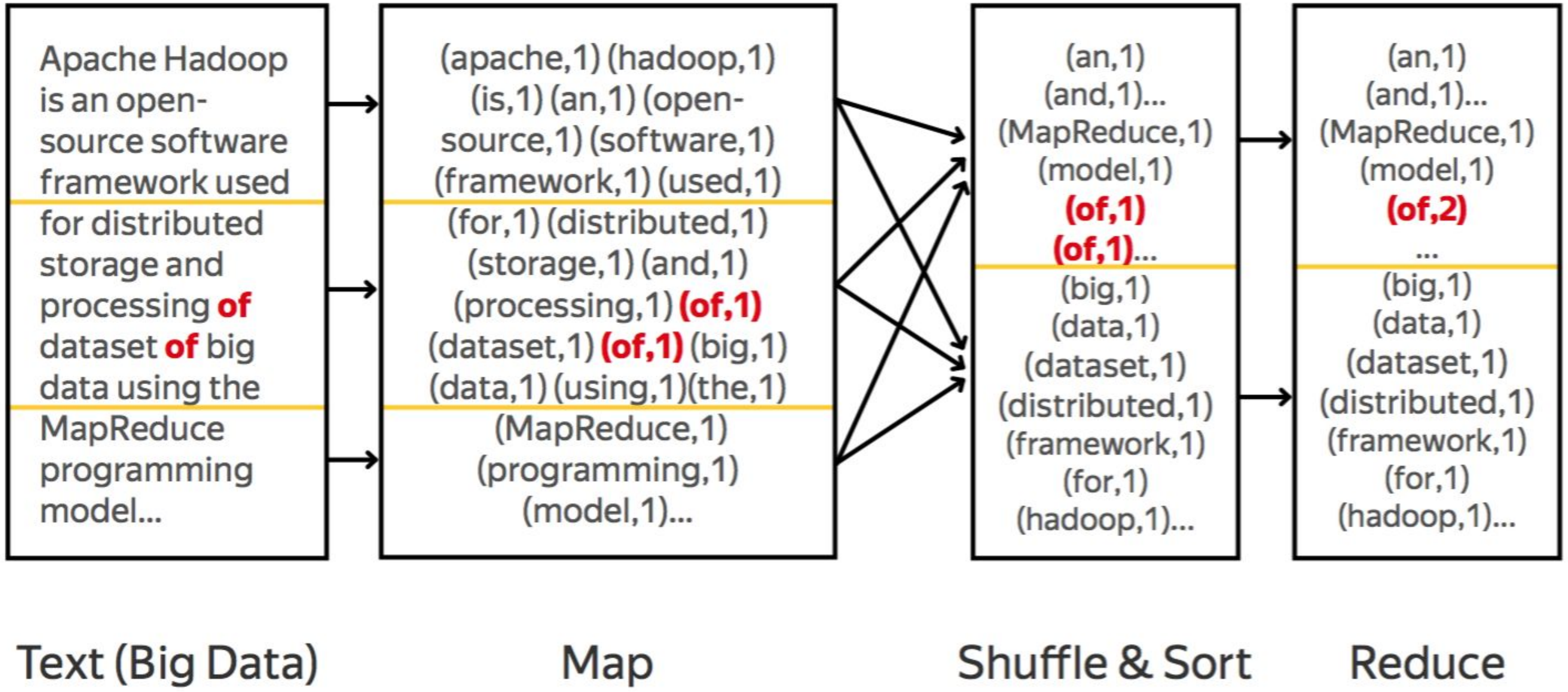
MapReduce (example)


wikipedia.dump | tr ' ' '\n' | sort | uniq -c
wikipedia.dump
Block 1
Block 2
Block M
Apache Hadoop (/hə`du:p/) is an open-source software framework used for distributed storage and processing of dataset of big data using the MapReduce programming model. It consists of computer clusters built from commodity hardware. All the modules in Hadoop are designed with a fundamental assumption that...
hash(word) % R
hash(word) % R
hash(word) % R
uniq -c
a... 1
uniq -c
b... 2
...
uniq -c
z... 26
Σ
wikipedia.dump -> map () -> word
shuffle & sort
reduce()

MapReduce Formal Model

- map: (key, value) → (key, value)
- reduce: (key, value) → (key, value)

Word Count



Apache Hadoop is an open-source software framework used for distributed storage and processing of dataset of big data using the MapReduce programming model...
(apache,1) (hadoop,1) (is,1) (an,1) (open-source,1) (software,1) (framework,1) (used,1) (for,1) (distributed,1) (storage,1) (and,1) (processing,1) (of,1) (dataset,1) (of,1) (big,1) (data,1) (using,1)(the,1) (MapReduce,1) (programming,1) (model,1)...
(an,1) (and,1)... (MapReduce,1) (model,1) (of,1) (of,1)... (big,1) (data,1) (dataset,1) (distributed,1) (framework,1) (for,1) (hadoop,1)...
(an,1) (and,1)... (MapReduce,1) (model,1) (of,2) ... (big,1) (data,1) (dataset,1) (distributed,1) (framework,1) (for,1) (hadoop,1)...
Text (Big Data)
Map
Shuffle & Sort
Reduce

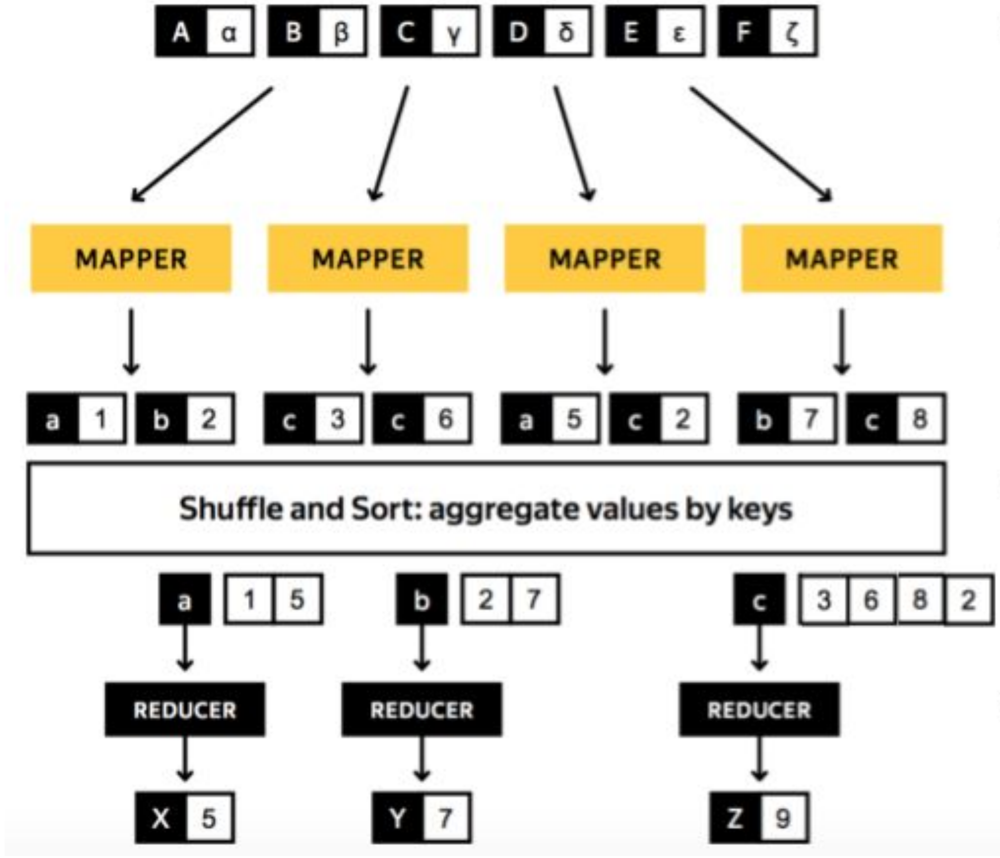
Word Count (example)

```
$ cat -n wikipedia.dump | tr ' ' '\n'|
sort | uniq -c
```

- cat **-n** wikipedia.dump: [(line_no, line), ...]
- **read: [(k_in, v_in), ...]**
- tr ‘ ’ ‘\n’: (-, line) —> [(word, 1), ...]
- **map: (k_in, v_in) —> [(k_interm, v_interm), ...]**
- **Shuffle & Sort: sort and group by k_interm**
- uniq -c: (word, [1, ...]) —> (word, count)
- **reduce: (k_interm, [(v_interm, ...)]) —> [(k_out, v_out), ...]**

BIGDATA TEAM

MapReduce

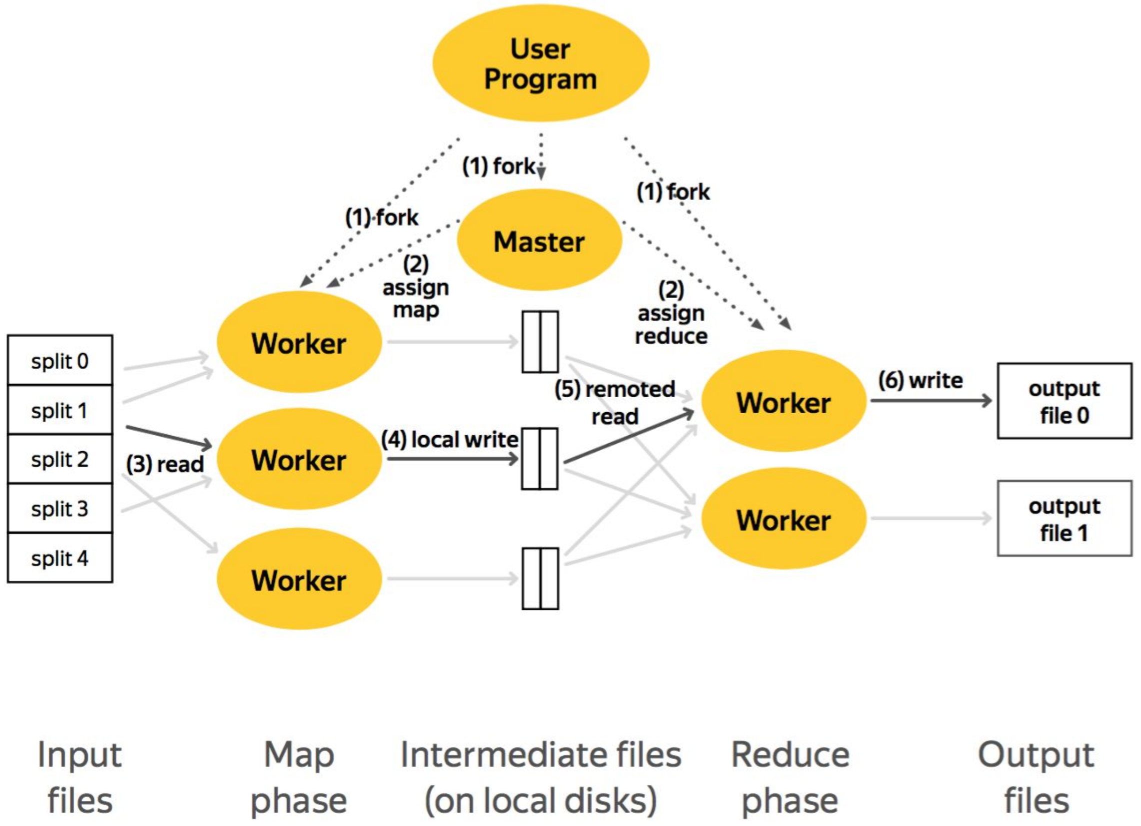


- read: [(k_in, v_in), ...]
- map: (k_in, v_in) —> [(k_interm, v_interm), ...]
- Shuffle & Sort: sort and group by k_interm
- reduce: (k_interm, [(v_interm, ...)]) —> [(k_out, v_out), ...]

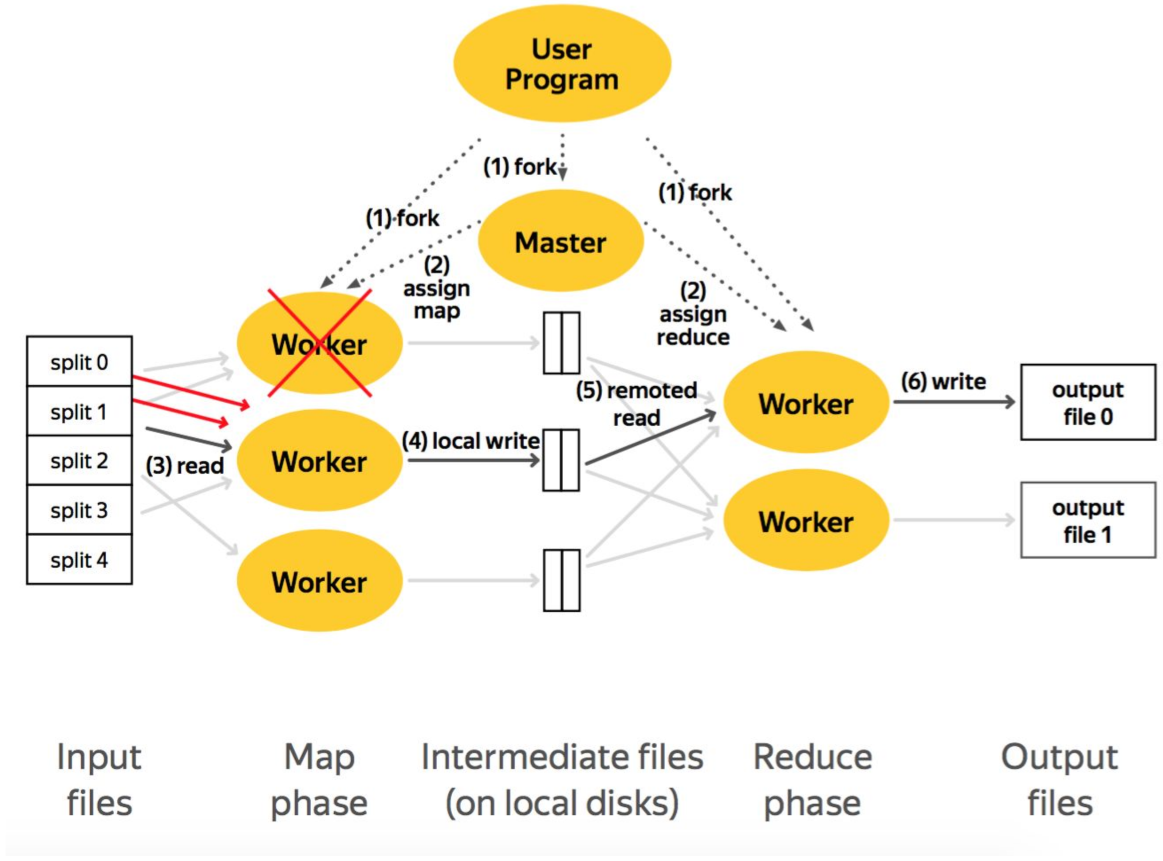
Fault Tolerance

BIGDATA TEAM

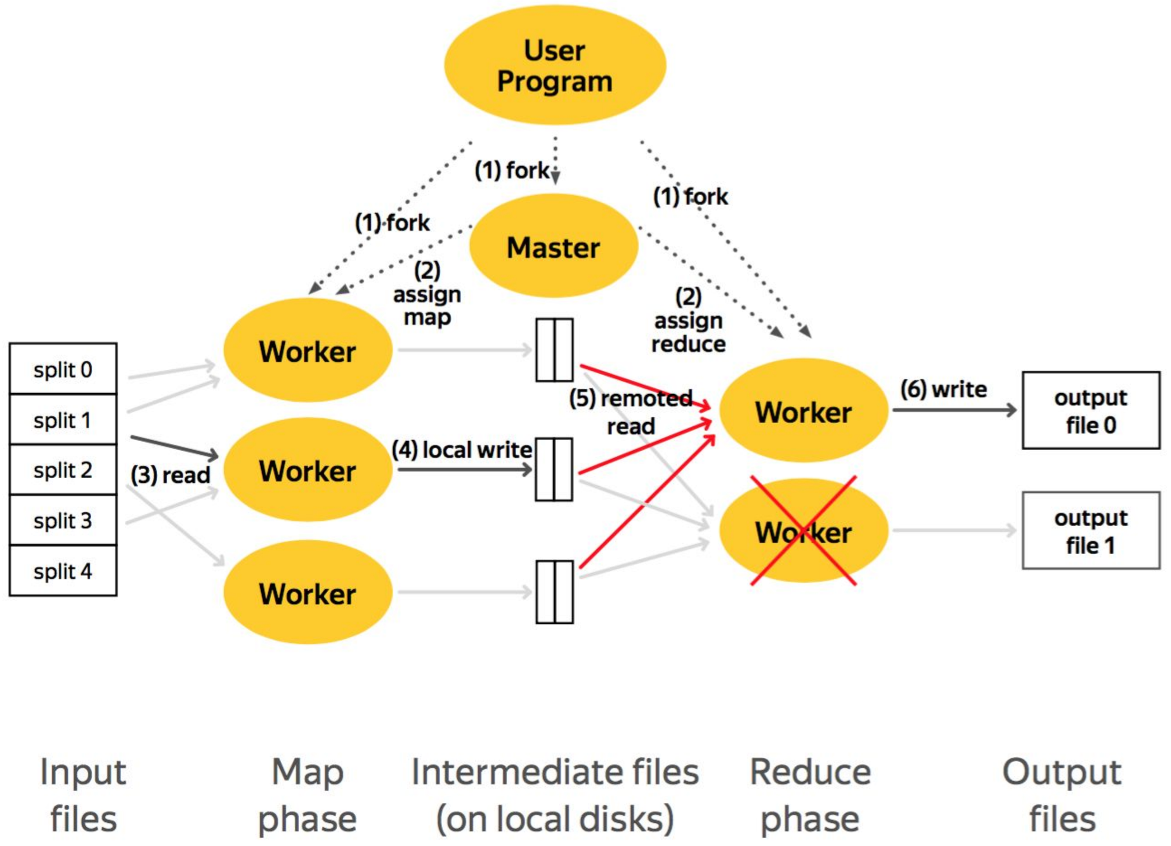
Fault Tolerance



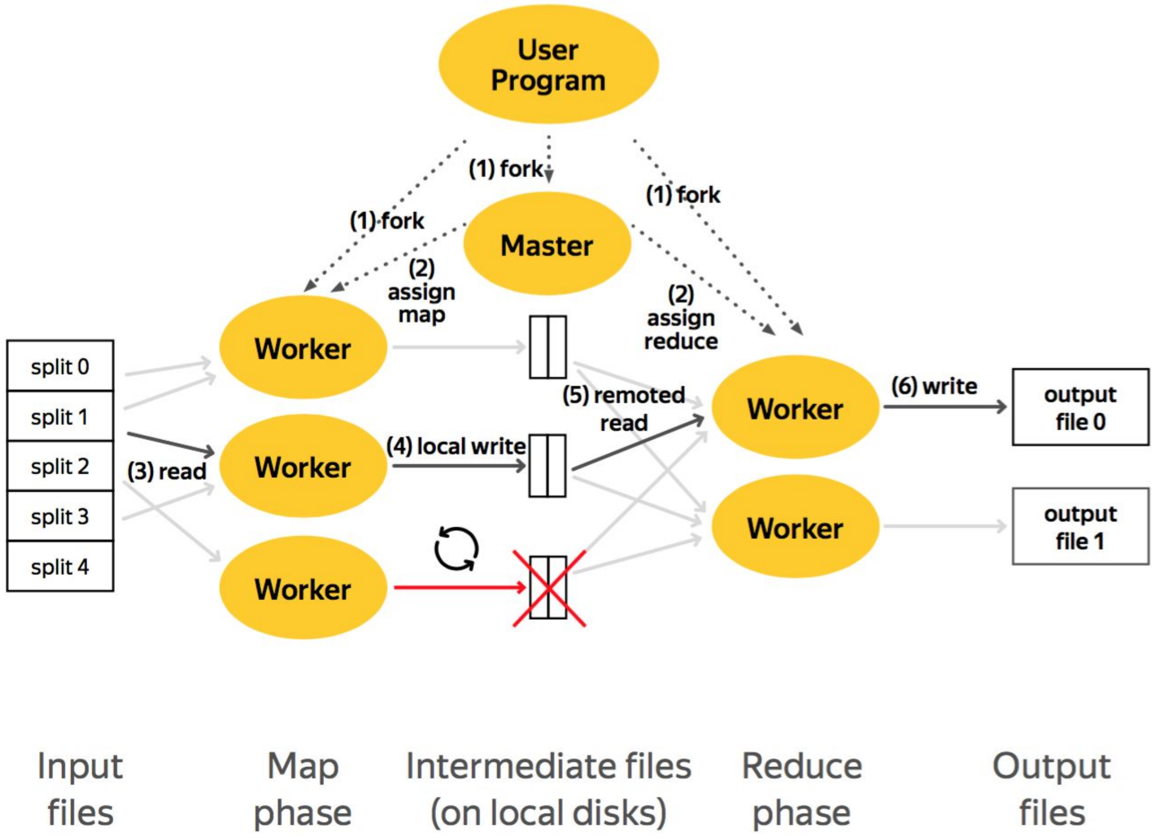
Fault Tolerance



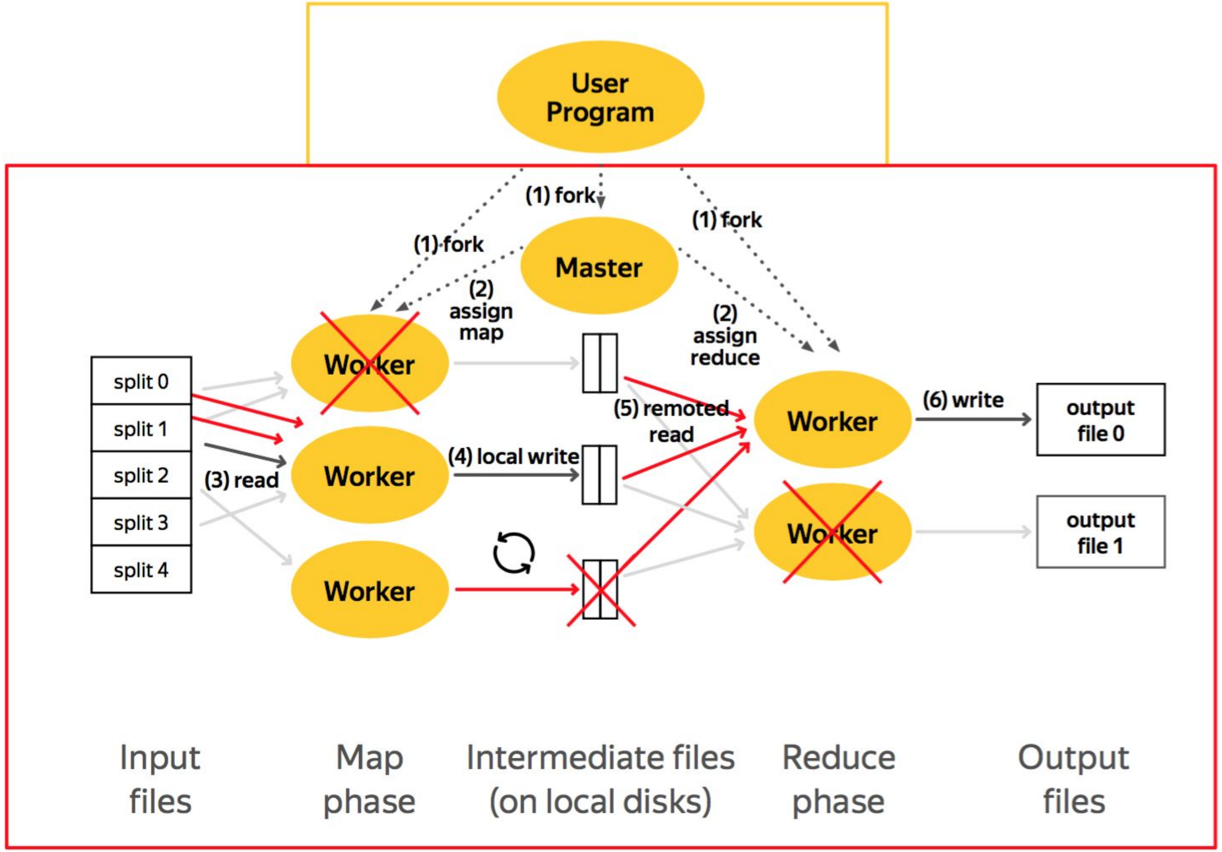
Fault Tolerance



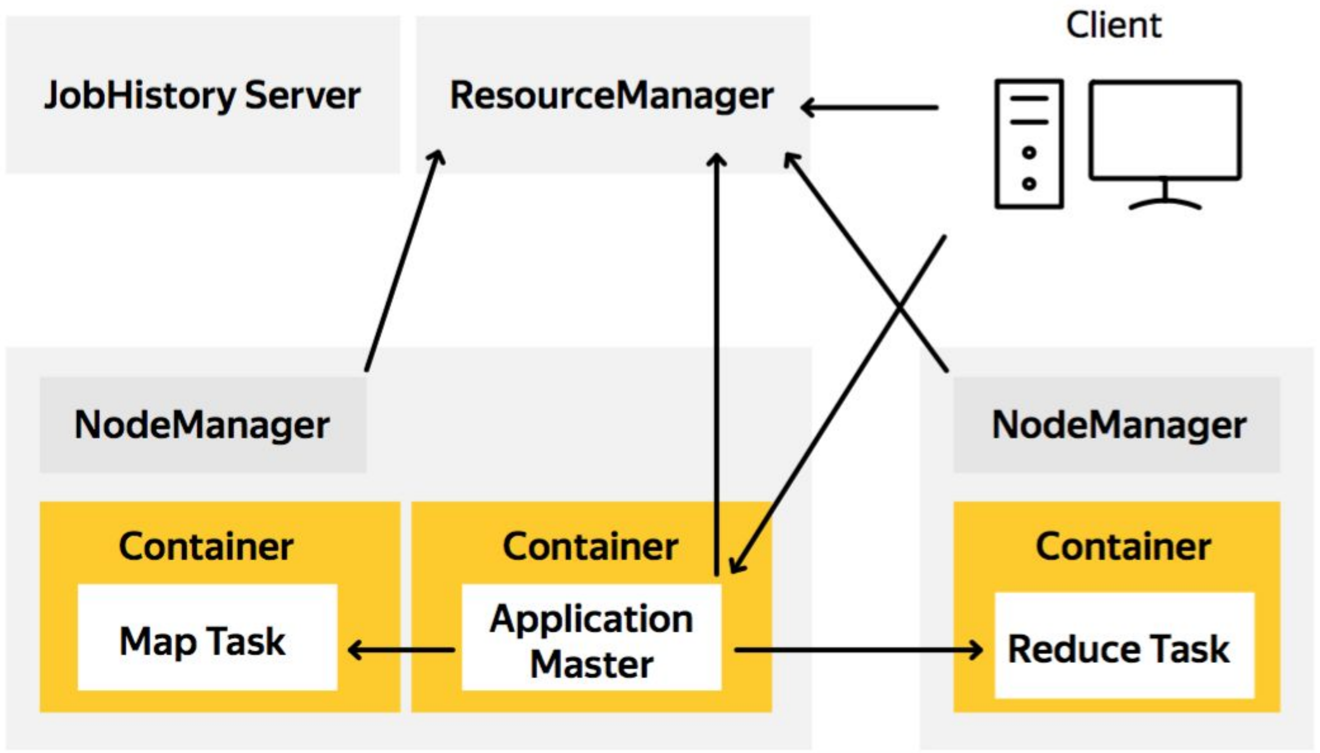
Fault Tolerance



Fault Tolerance

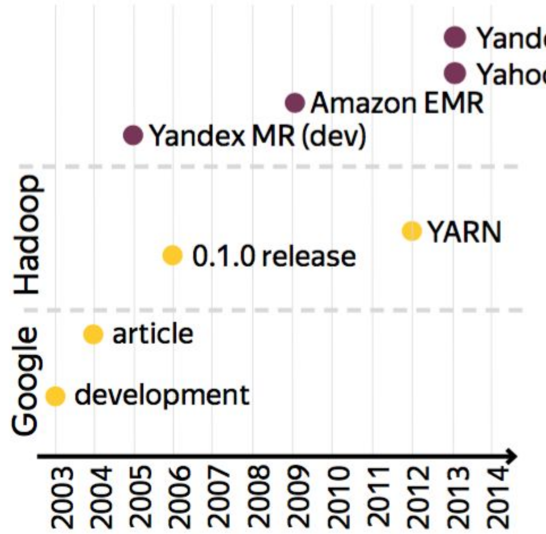


MRv1 vs YARN


Client
JobHistory Server
ResourceManager
NodeManager
NodeManager
Container
Container
Container
Map Task
Application Master
Reduce Task

MapReduce Frameworks (Timeline)

- [2003] Google MapReduce (development)
- [2004] Google MapReduce (article)
- [2005] Yandex MapReduce (development)
- [2006] Hadoop 0.1.0 release
- [2009] Amazon EMR (Hadoop inside)
- [2012] MapReduce —> YARN
- [2013] Yahoo! YARN deployed in production
- [2013] Yandex YT…
- MapReduce in MongoDB, Riak, …



MapReduce Game Time

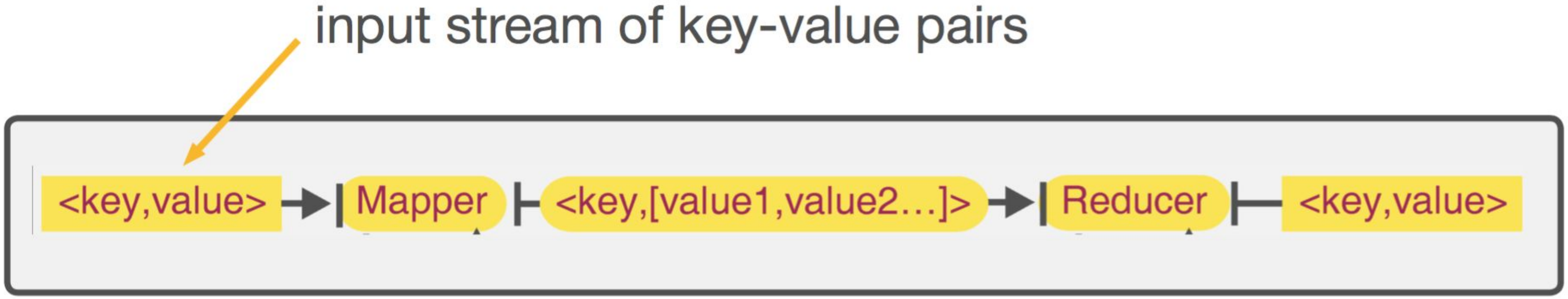
BIGDATA TEAM

Tea / Coffee Break

BIGDATA TEAM

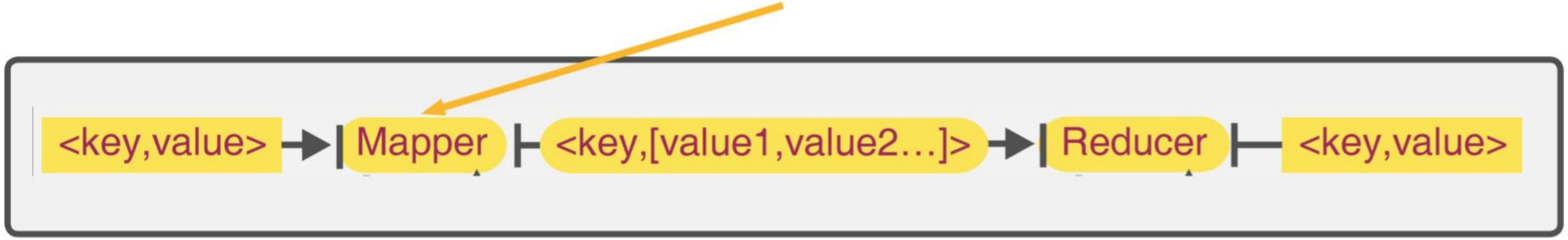
MapReduce Streaming

MapReduce Streaming

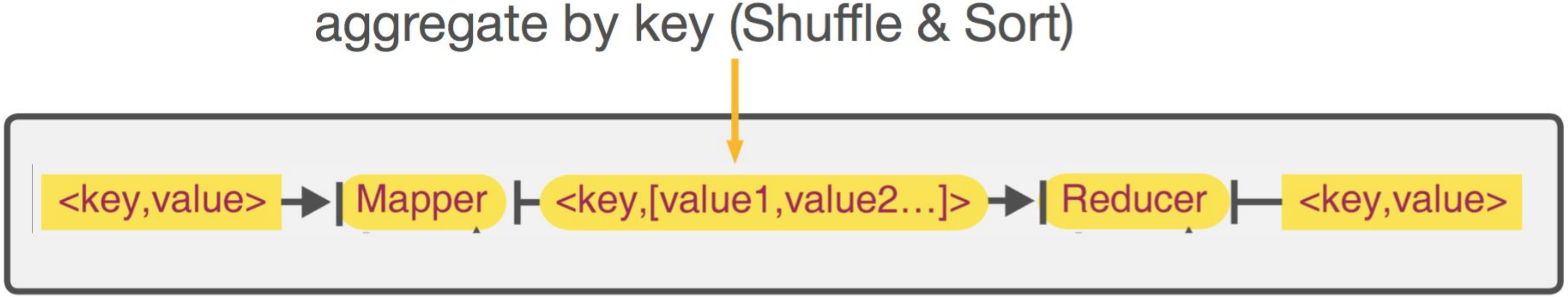


MapReduce Streaming

```
map: (k_in, v_in) --> [(k_interm, v_interm), …]
```

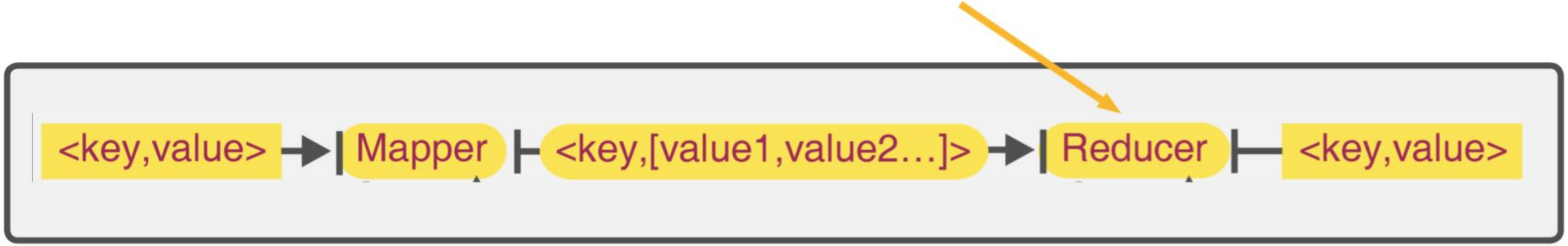


MapReduce Streaming

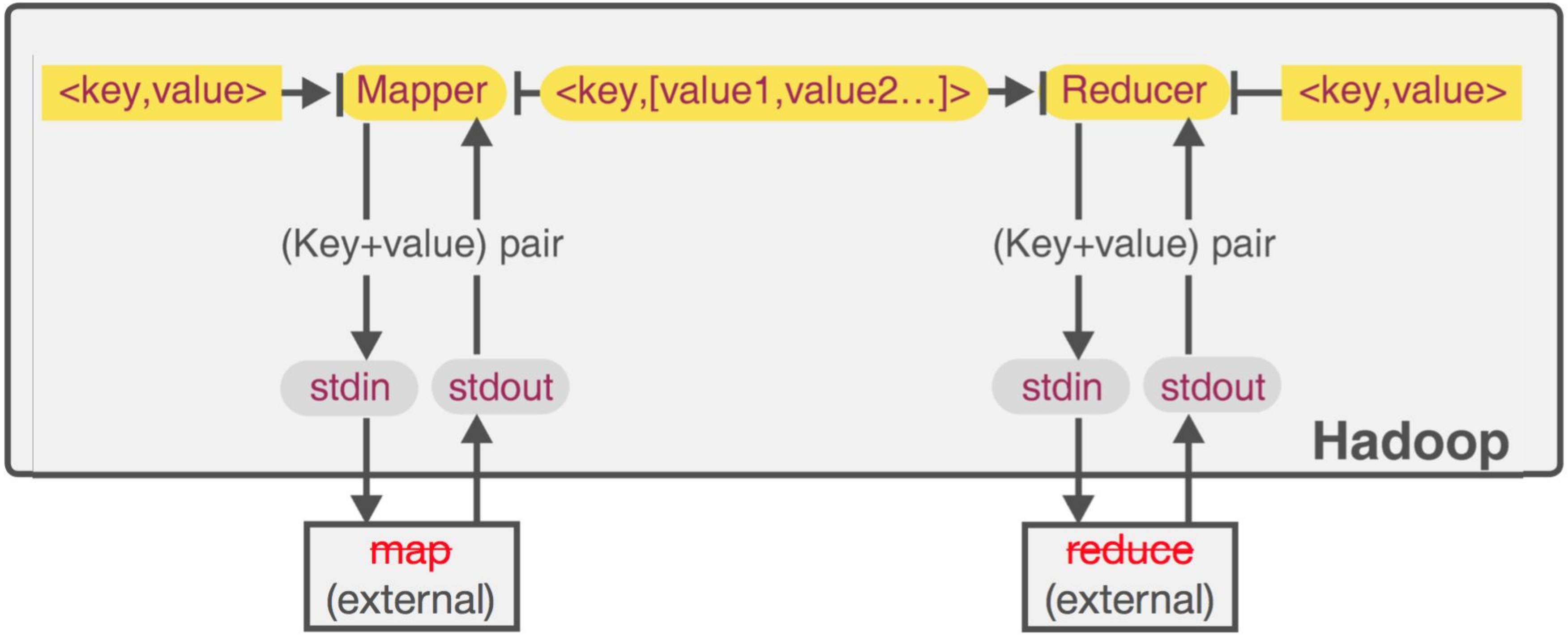


MapReduce Streaming

```
reduce: (k_interm, [(v_interm, …)] ) --> [(k_out, v_out), …]
```

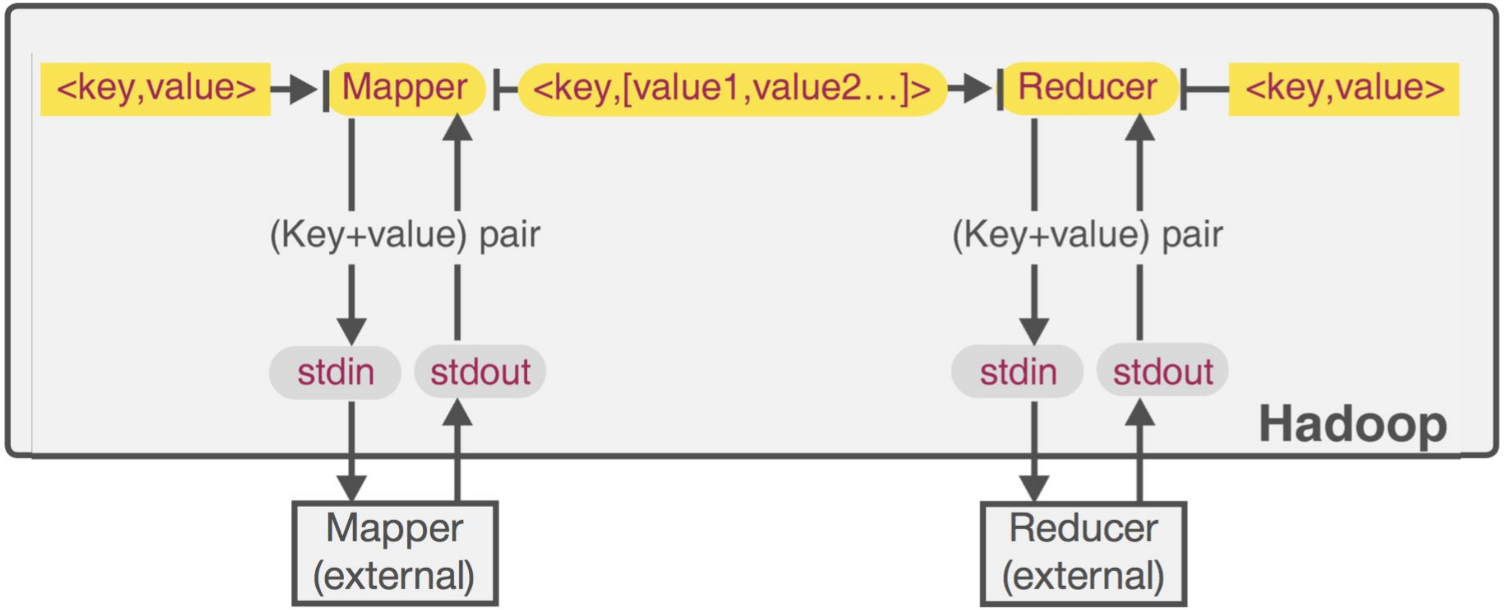


MapReduce Streaming



BIGDATA TEAM

MapReduce Streaming



- define input format
- process data
- define output format

- define input format
- aggregate sorted data by key
- process data
- define output format

MapReduce Examples

```
<article id> <tab> <article content>
```

Line Count?

MapReduce Examples

```
$ man locate
```

/opt/cloudera/parcels/CDH-5.9.0-1.cdh5.9.0.p0.23/lib/hadoop-mapreduce/hadoop-streaming.jar

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
         -mapper 'wc -l' \
         -numReduceTasks 0 \
         -input /data/wiki/en_articles \
         -output wc_mr
```

MapReduce Examples

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
            -mapper 'wc -l' \
            -numReduceTasks 0 \
            -input /data/wiki/en_articles \
            -output wc_mr
```

```
ERROR streaming.StreamJob: Error Launching job : Output directory
hdfs://virtual-master.atp-fivt.org:8020/user/adral/wc_mr already exists
Streaming Command Failed!
```

```
$ hdfs dfs -rm -r wc_mr
```

BIGDATA TEAM

MapReduce Examples

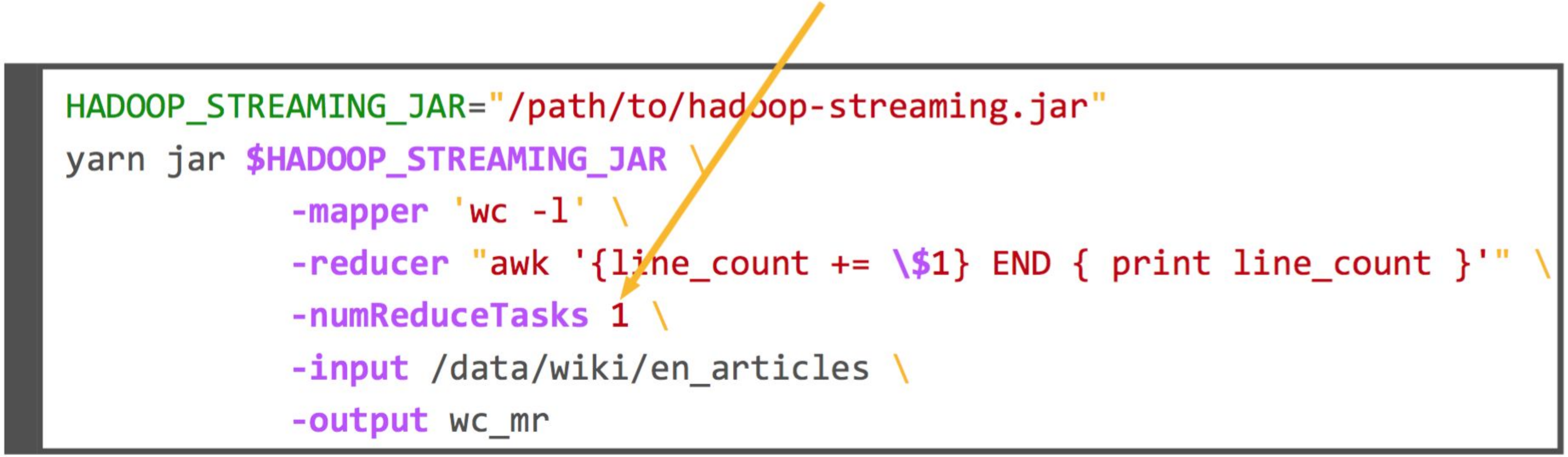
```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
           -mapper 'wc -l' \
           -numReduceTasks 0 \
           -input /data/wiki/en_articles \
           -output wc_mr
```

```
$ hdfs dfs -ls wc_mr
Found 3 items
-rw-r--r--   3 adral adral          0 2017-03-21 14:48 wc_mr/_SUCCESS
-rw-r--r--   3 adral adral          6 2017-03-21 14:48 wc_mr/part-00000
-rw-r--r--   3 adral adral          6 2017-03-21 14:48 wc_mr/part-00001
```

```
$ hdfs dfs -text wc_mr/*
1986
2114
```

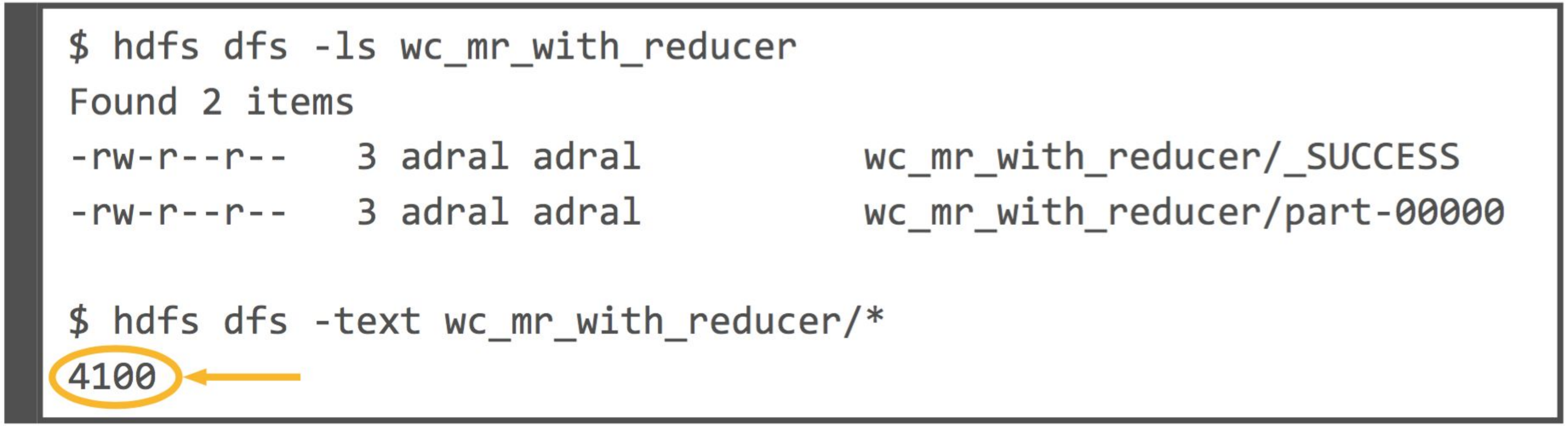
$1968 + 2114 = 4100$

MapReduce Examples



```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
            -mapper 'wc -l' \
            -reducer "awk '{line_count += \$1} END { print line_count }'" \
            -numReduceTasks 1 \
            -input /data/wiki/en_articles \
            -output wc_mr
```

MapReduce Examples



```
$ hdfs dfs -ls wc_mr_with_reducer
Found 2 items
-rw-r--r--   3 adral adral          wc_mr_with_reducer/_SUCCESS
-rw-r--r--   3 adral adral          wc_mr_with_reducer/part-00000

$ hdfs dfs -text wc_mr_with_reducer/*
4100
```

MapReduce Examples

reducer.sh

```
#!/usr/bin/env bash
awk '{line_count += $1} END { print line_count }'
```

BIGDATA TEAM

MapReduce Examples

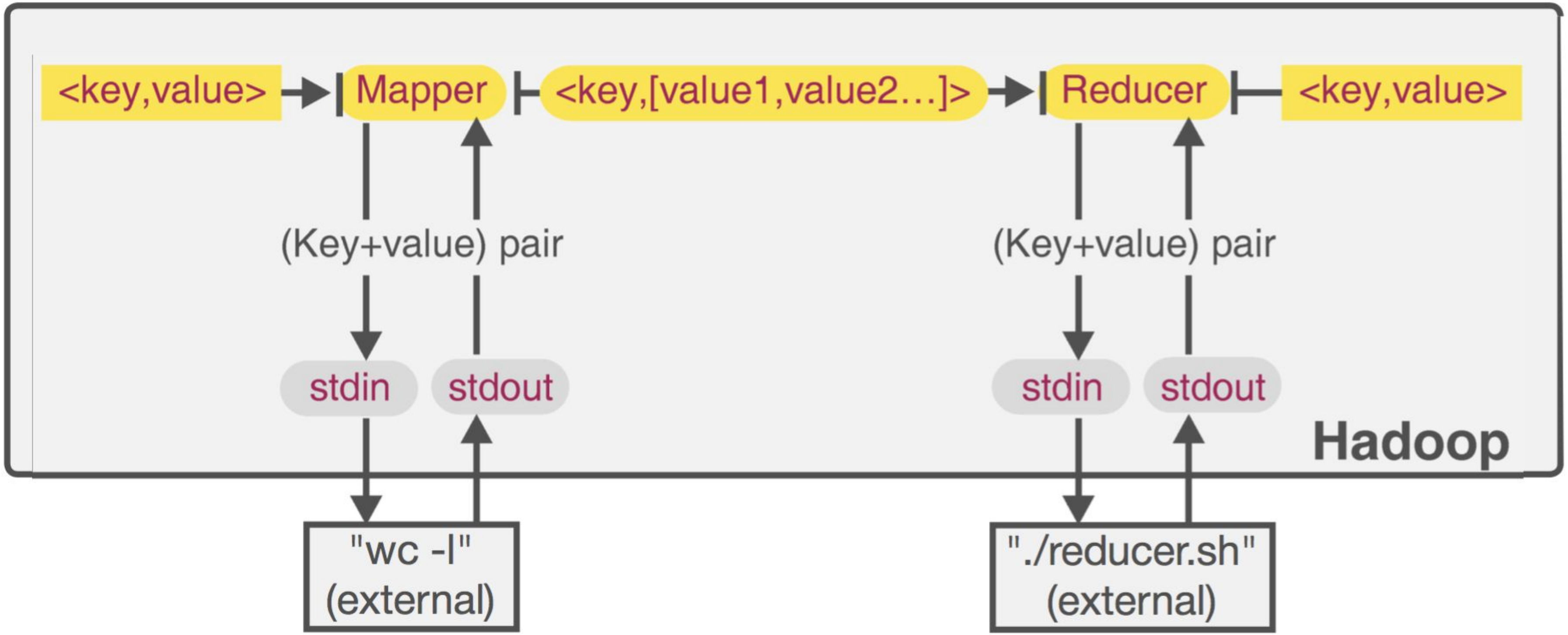
reducer.sh

```
#!/usr/bin/env bash
awk '{line_count += $1} END { print line_count }'
```

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
         -mapper 'wc -l' \
         -reducer './reducer.sh' \
         -file reducer.sh \
         -numReduceTasks 1 \
         -input /data/wiki/en_articles \
         -output wc_mr_with_reducer
```

BIGDATA TEAM

MapReduce Examples



Резюме

- ▶️ Вы можете объяснить, что происходит когда “умирает” Mapper или Reducer
- ▶️ Вы знаете, за что отвечают ResourceManager и NodeManager в YARN
- ▶️ Вы знаете 3 фазы MapReduce (Map, Shuffle & Sort, Reduce)
- ▶️ Вы знаете, что такое MapReduce Streaming и как он работает (примеры: distributed grep, wc, LineCount, WordCount)

KEEP
CALM
AND
TRY
CODING

BIGDATA TEAM

Thank you! Questions?

Feedback: http://rebrand.ly/mf2019q2_feedback_02_mr

Dral Alexey, aadral@bigdatateam.org
CEO at BigData Team, http://bigdatateam.org/
https://www.linkedin.com/in/alexey-dral
https://www.facebook.com/bigdatateam/