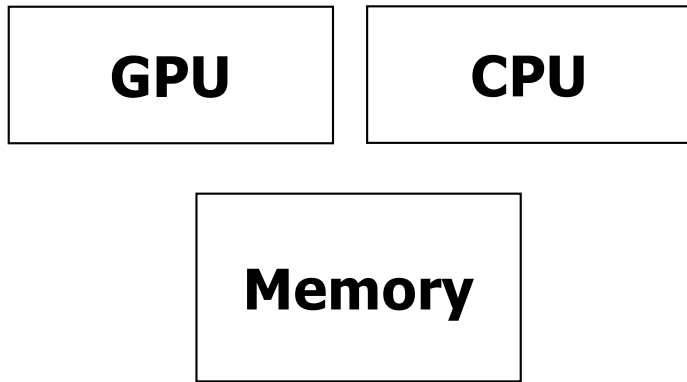


# Map-Reduce (Part I)

Yao-Yi Chiang  
Computer Science and Engineering  
University of Minnesota  
yaoyi@umn.edu

Thanks for source slides and material to: J. Leskovec, A. Rajaraman,  
J. Ullman: Mining of Massive Datasets (<http://www.mmds.org>)  
Also slides from Yijun Lin, Ann Chervenak, and Wensheng Wu

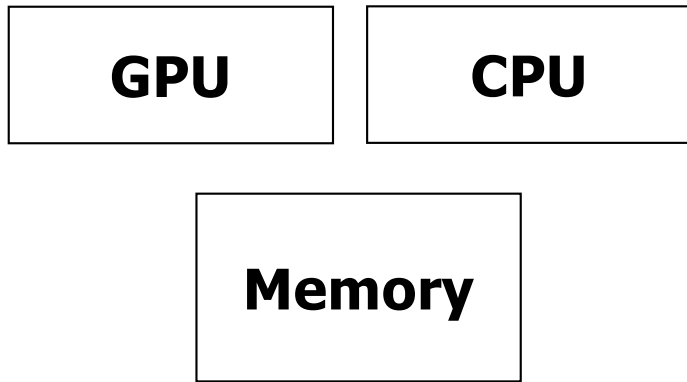
# Single Node Architecture



Machine Learning, Statistics

Computational Model of CPU/GPU and memory

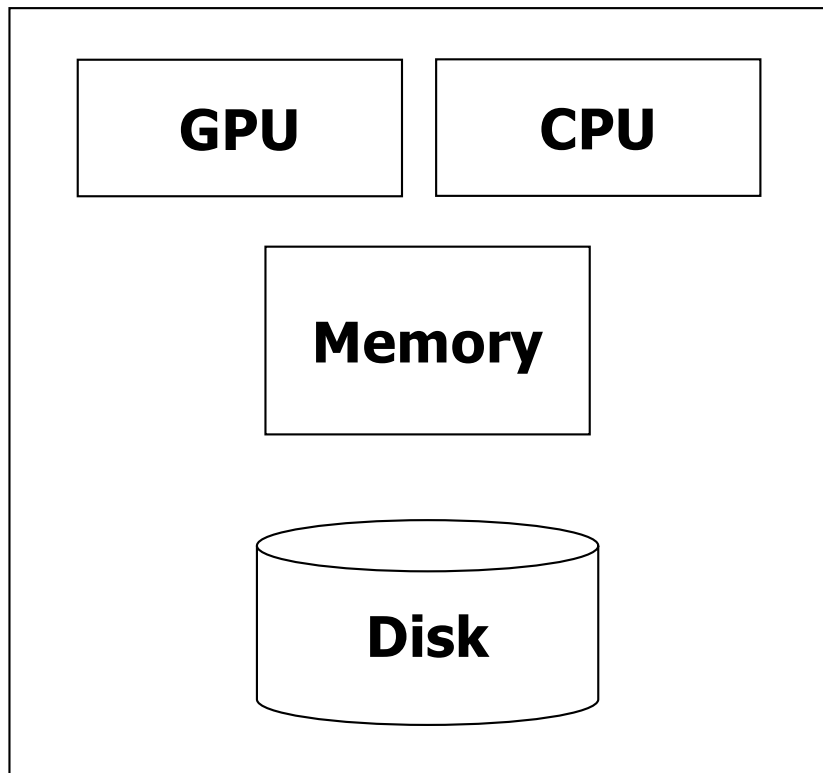
# Single Node Architecture



Machine Learning, Statistics

**What if the data can't fit in memory at the same time?**

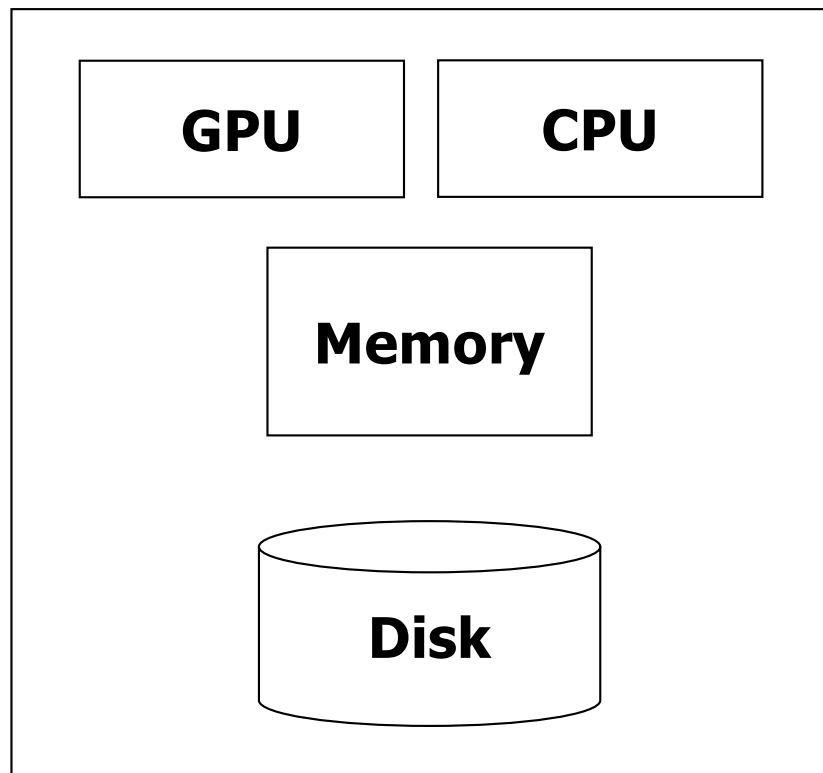
# Single Node Architecture



Machine Learning, Statistics

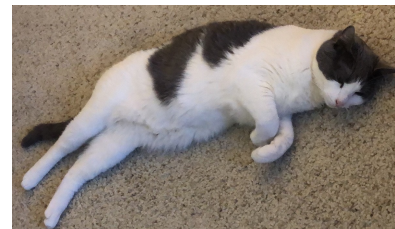
“Classical” Data Mining algorithm

# Single Node Architecture



Machine Learning, Statistics

“Classical” Data Mining algorithm



**Not sufficient !**

# Motivation: Google Example

## Crawling and indexing the web pages

- 10 billion web pages
- Average size of webpage = 20 KB
  - ⇒ 10 billion \* 20KB = 200 TB
- The data is stored on a single disk and tends to be processed in CPU

# Motivation: Google Example

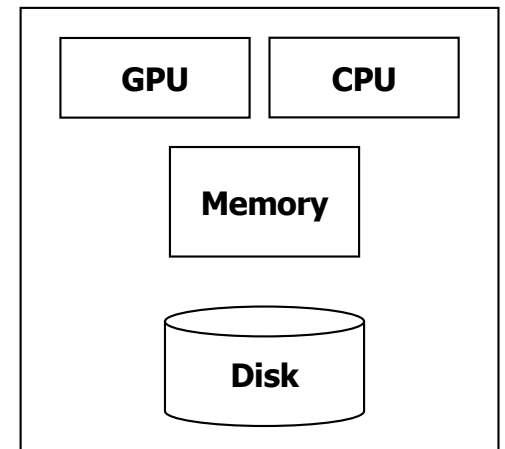
## Crawling and indexing the web pages

- 10 billion web pages
- Average size of webpage = 20 KB
  - ⇒ 10 billion \* 20KB = 200 TB
- The data is stored on a single disk and tends to be processed in CPU
- One computer reads 50 MB/sec from disk (disk read bandwidth)
  - ⇒ Time to read = 4 million seconds  $\approx$  46 days
- Even longer to do useful things with the data

# Motivation: Google Example

## Crawling and indexing the web pages

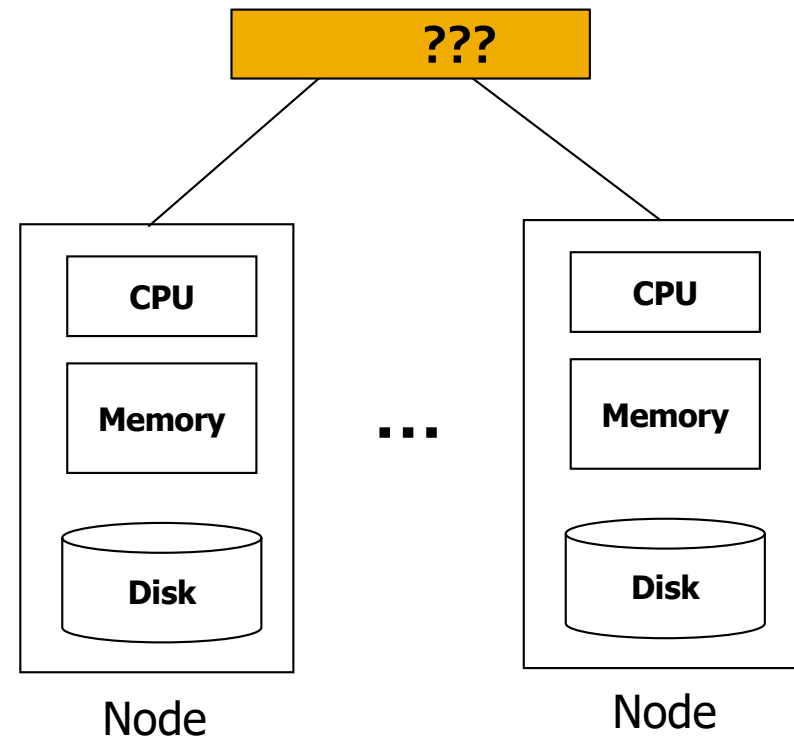
- Split the data into chunks
- Store and process the data **in parallel** in multiple disks and CPUs
- e.g., **1,000 disks and CPUs**
  - ⇒ Time to read = **4 million seconds / 1,000 = 4,000 seconds**
- **Cluster Computing**





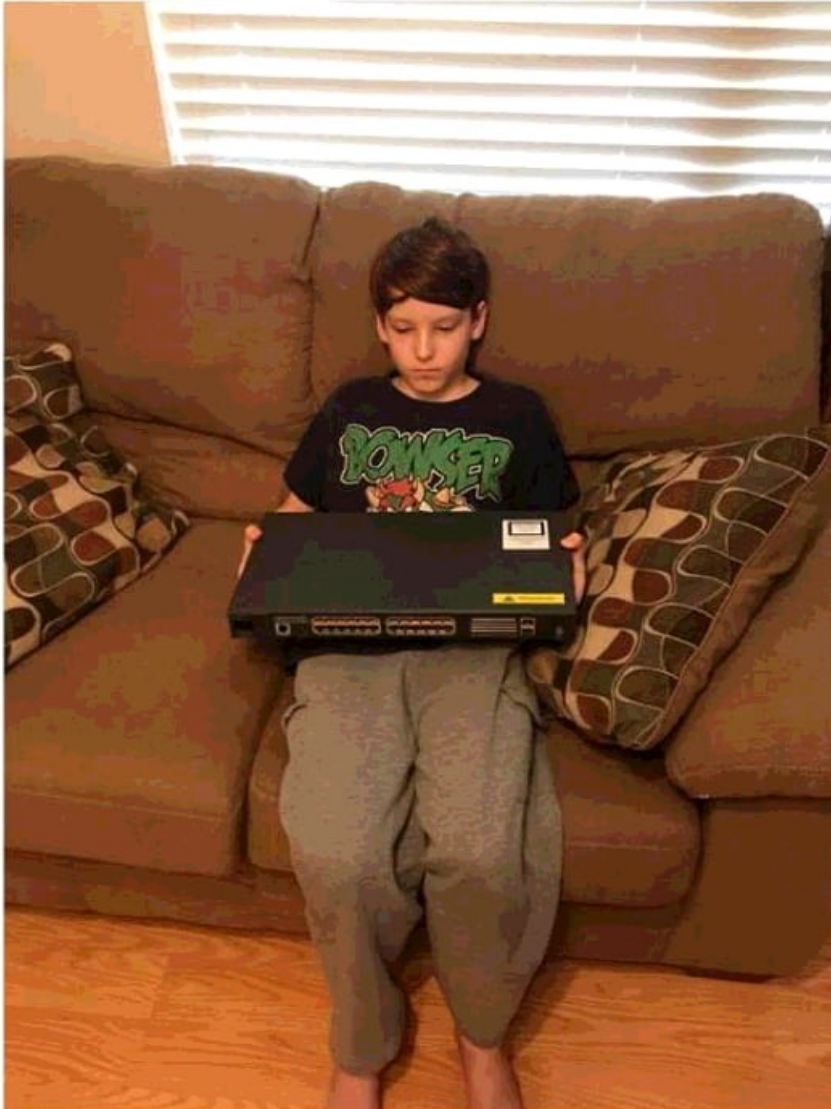
# Cluster Architecture

Each rack contains 16-64 nodes  
e.g., commodity Linux nodes

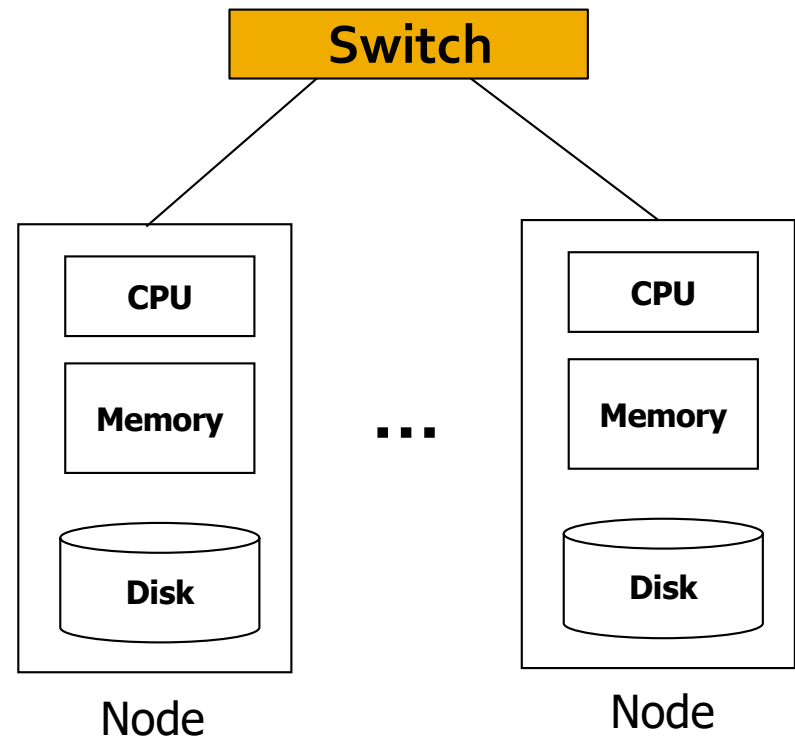


# Cluster Architecture

My son said he wanted a switch for his birthday.

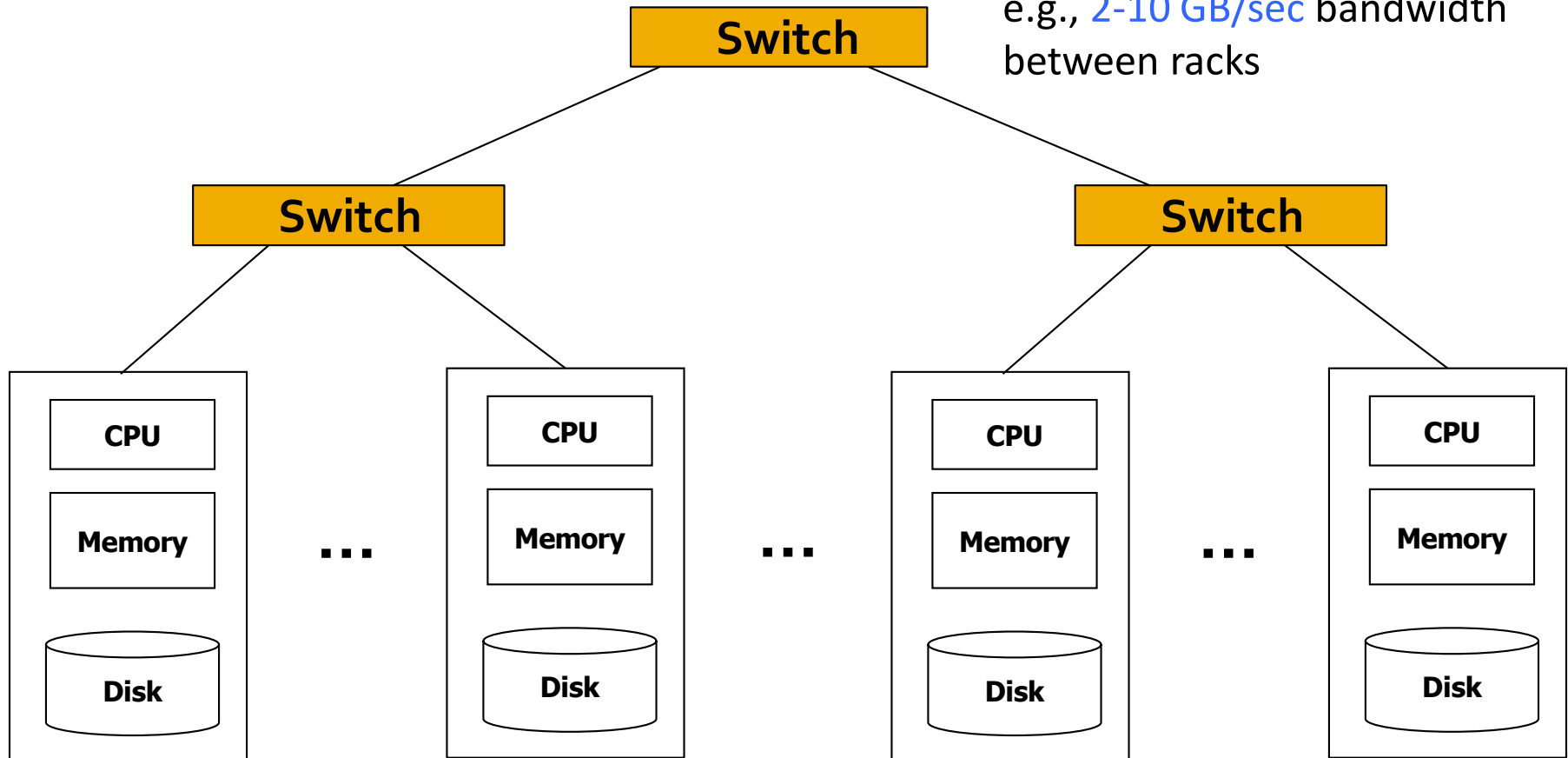


**Switch** - connect nodes  
e.g., 1 GB/sec bandwidth between any pair of nodes in a rack



# Cluster Architecture

**Backbone switch** - connect racks  
e.g., 2-10 GB/sec bandwidth  
between racks





In 2011 it was guestimated that Google had 1M machines, <http://bit.ly/Shh0RO>

# Cluster Computing Challenges

- **Node failures**

e.g., one server can stay up 3 years (1,000 days)

1,000 servers in cluster  $\Rightarrow$  1 failure/day

1M servers in cluster  $\Rightarrow$  1,000 failures/day

# Cluster Computing Challenges I

- **Node failures**

e.g., one server can stay up 3 years (1,000 days)

1,000 servers in cluster  $\Rightarrow$  1 failure/day

1M servers in cluster  $\Rightarrow$  1,000 failures/day

$\Rightarrow$  Store data **persistently** and keep it available when nodes fail

$\Rightarrow$  Deal with node failures **during a long running computation**

# Cluster Computing Challenges II

- **Network bottleneck**

e.g., network bandwidth = 1 GB/sec

moving 10TB data takes approximately 1 day

⇒ A framework that does not move data around so much while it's doing computation

# Cluster Computing Challenges III

- **Distributed/parallel programming is hard**

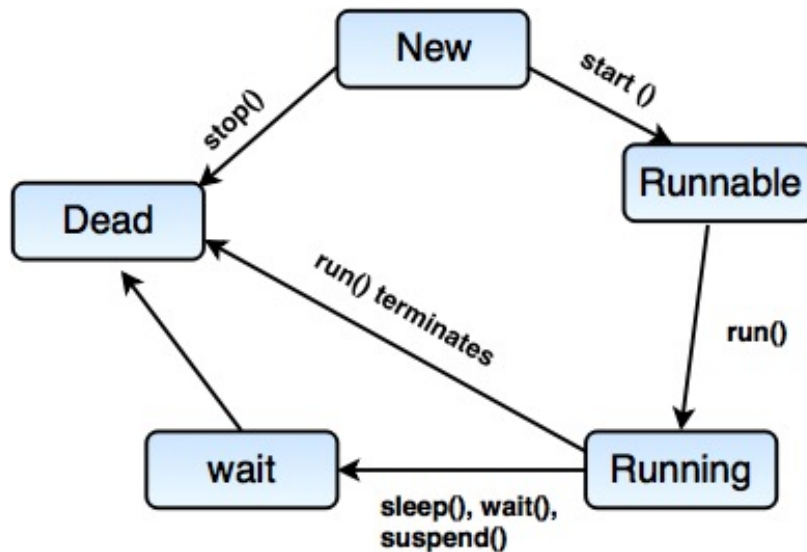


Fig: Life Cycle of Thread

<https://www.tutorialride.com/core-java/multithreading-in-java.htm>

⇒ A simple model that hides most of the complexity



# Map-Reduce

- **Map-Reduce addresses the challenges**

- **Node failure**

- Store data redundantly on multiple nodes

- **Network bottleneck**

- Move computation close to data to minimize data movement

- **Distributed programming**

- Map function and Reduce functions

# Redundant Storage Infrastructure

- **Distributed File System**
  - Store data multiple times across a cluster
  - Provide global file namespace
  - E.g., Google GFS; Hadoop HDFS



- **Typical usage pattern**
  - Huge files (100s of GB to TB)
  - Data is rarely updated in place
  - Reads and appends are common



# Distributed File System

- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines
- E.g., 4 chunk servers



Chunk server 1



Chunk server 2



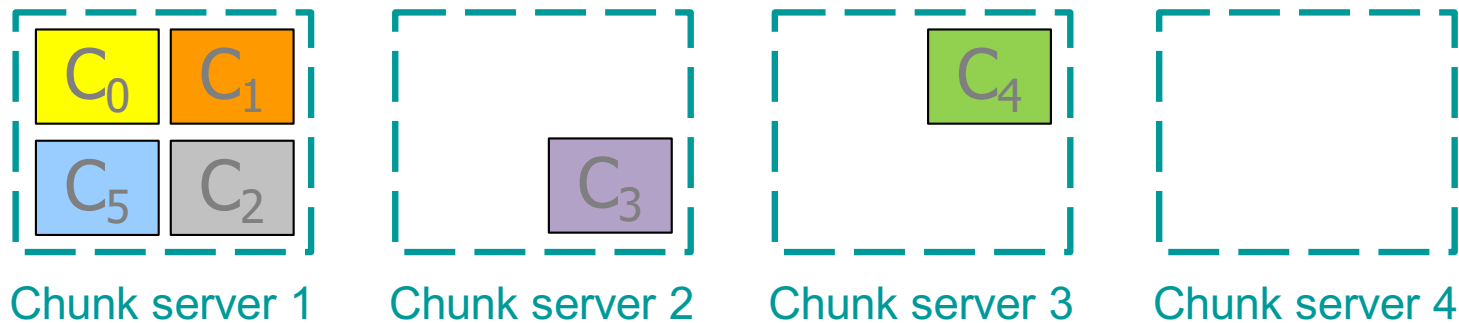
Chunk server 3



Chunk server 4

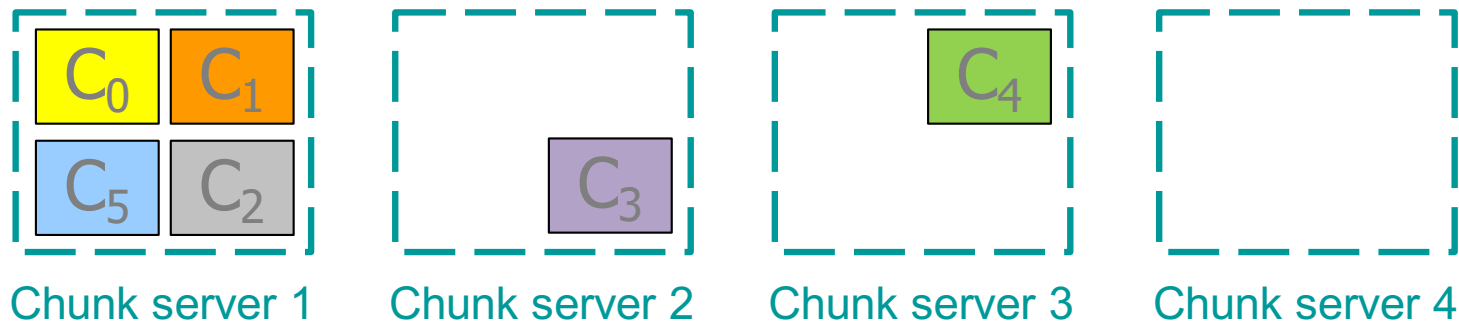
# Distributed File System

- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines
- E.g., 4 chunk servers , **file 1** is split into **6 chunks**



# Distributed File System

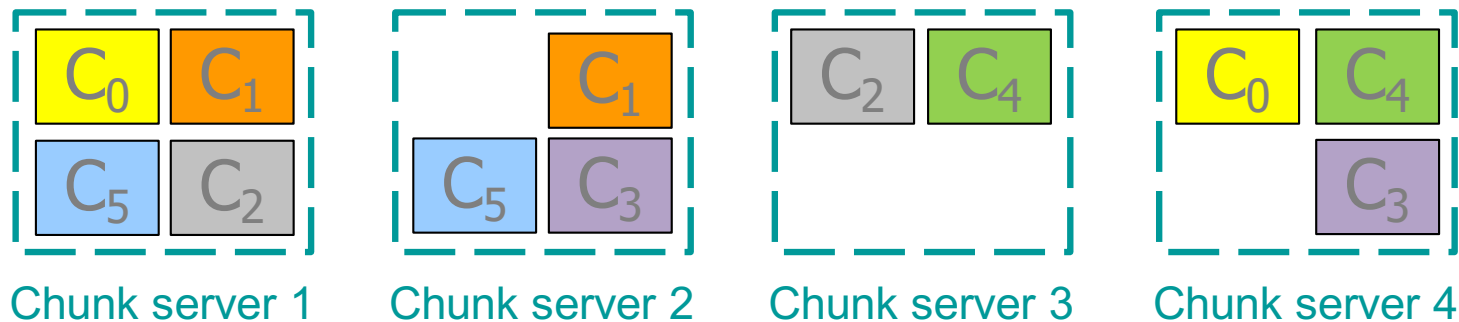
- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines
- E.g., 4 chunk servers , **file 1** is split into **6 chunks**



**Not sufficient ! Need multiple copies of each chunk.**

# Distributed File System

- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines
- E.g., 4 chunk servers , **file 1** is split into **6 chunks**

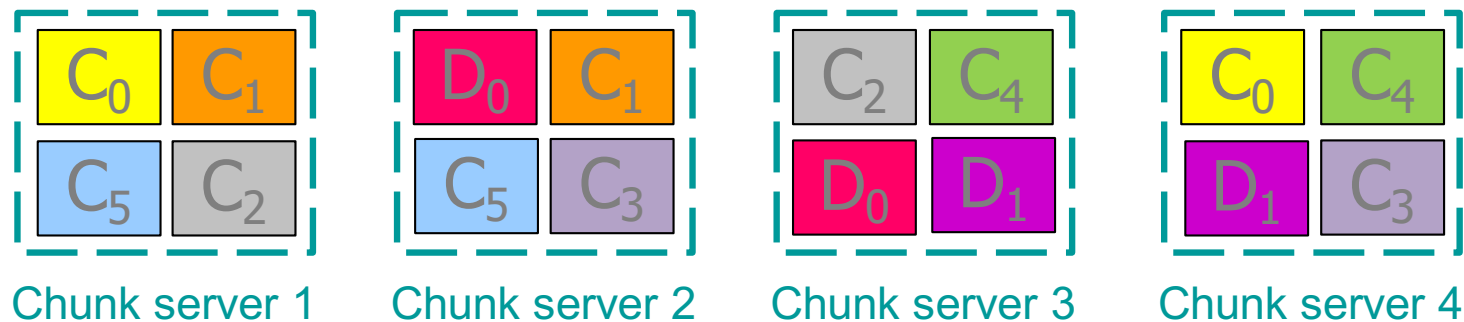


**Each chunk is replicated twice, and the replicas of a chunk are never on the same chunk server.**

# Distributed File System

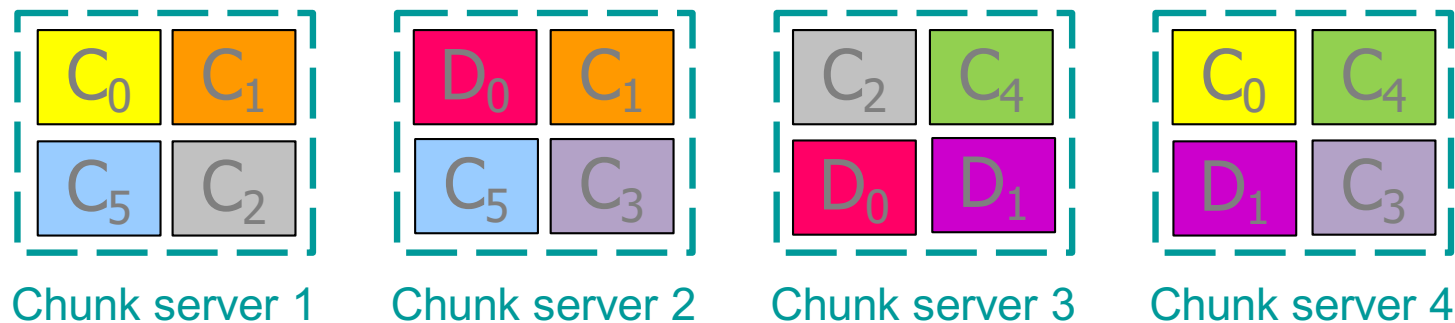
- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines
- E.g., 4 chunk servers , file 1 is split into 6 chunks

Another file 2 has 2 chunks,  $D_0$  and  $D_1$



# Distributed File System

- Data is kept in “chunks” spread across machines (**chunk servers**)
- Each chunk replicated on different machines



Chunk servers also serve as compute servers

Bring computation to data!



# Components of Distributed File System

- **Chunk servers**
  - File is split into **contiguous chunks** (typically 16-64 MB)
  - Each chunk is replicated (usually 2x or 3x)
  - Try to keep replicas in different racks (**Why?**)

# Components of Distributed File System

- **Chunk servers**

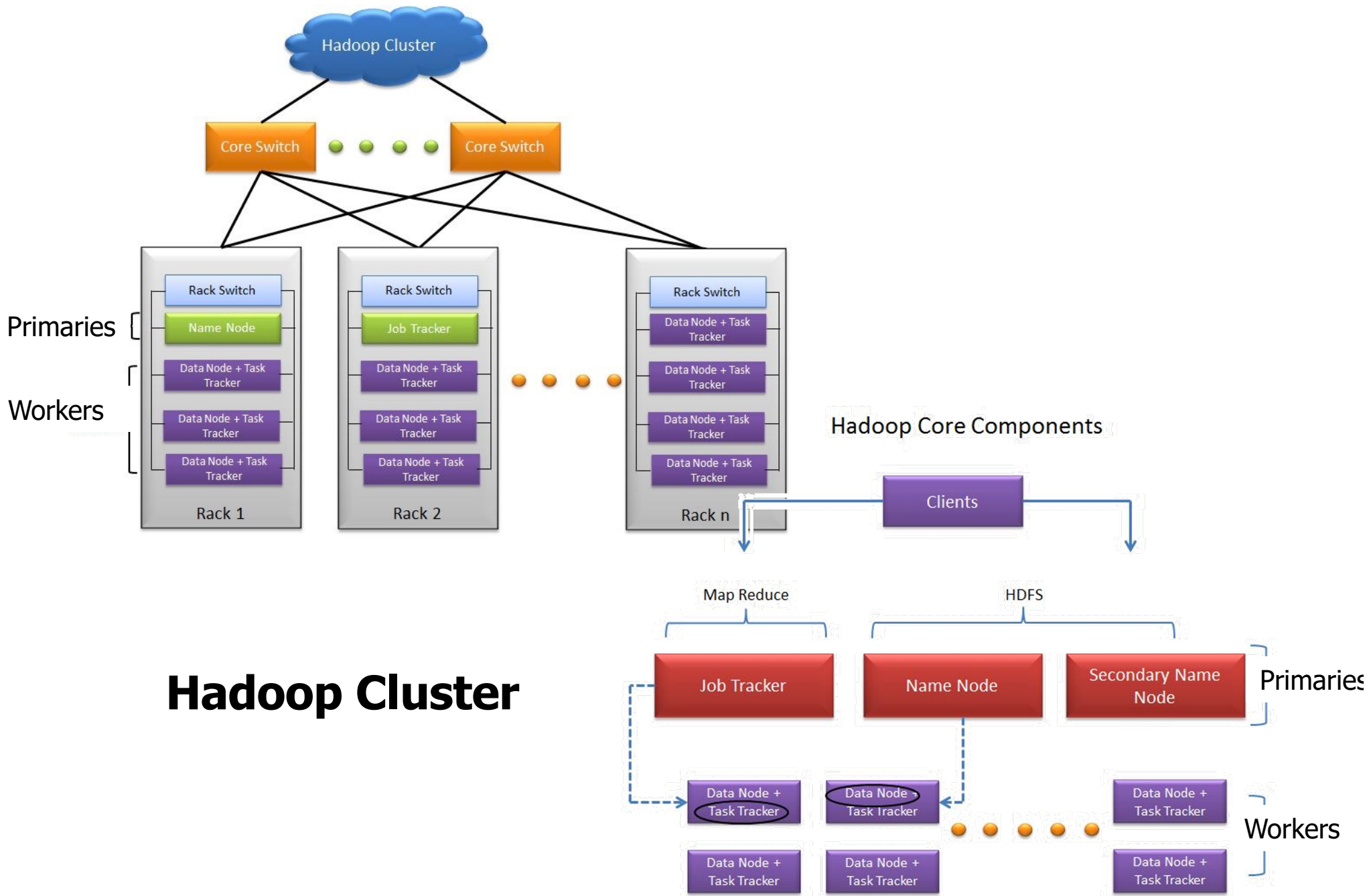
- File is split into **contiguous chunks** (typically 16-64 MB)
- Each chunk is replicated (usually 2x or 3x)
- Try to keep replicas in different racks
  - **In case that the switch on a rack can fail and entire rack becomes inaccessible**

# Components of Distributed File System

- **Primary node (Master node in the textbook)**
  - a.k.a. **Name Node** in Hadoop HDFS
  - Stores metadata about where files are stored
    - e.g., it will know that file 1 is divided into 6 chunks, the locations of each of the 6 chunks and the locations of the replicas
  - Might be replicated
    - Otherwise it might become a single point of failure

# Components of Distributed File System

- **Client library for file access**
  - Talks to primary to find chunk servers that store the chunks
  - **Connects directly to chunk servers** to access data without going through the primary node



# Hadoop Cluster

# Programming Model: Map-Reduce

- **Warm-up task**
  - We have a huge text document
  - Count the number of times each distinct word appears in the file
  - Sample application:
    - Analyze web server logs to find popular URLs

# Task: Word Count

- **Case 1**

- File is too large for memory; all <word, count> pairs fit in memory

**How to solve ?**

# Task: Word Count

- **Case 1**

- File is too large for memory, but all <word, count> pairs fit in memory
- Solution:
  - Use a **hash table** (word -> count) to store the number of times that word appears
  - Make a simple sweep through the file and will have the word count pairs for every unique word.



# Task: Word Count

- **Case 2**
  - Even the <word, count> pairs do not fit in memory
  - Solution:
    - **Map-Reduce**

# Map-Reduce: Overview

- **Map**
  - Extract something (e.g., word) from each record (as **keys**)
  - Output one or multiple things for each record
- **Group by key**
  - Sort and shuffle
- **Reduce**
  - Aggregate, summarize, filter or transform
  - Output the result

# Map-Reduce: Overview

- **Map**
  - Extract something (e.g., word) from each record (as **keys**)
  - Output one or multiple things for each record
- **Group by key**
  - Sort and shuffle
- **Reduce**
  - Aggregate, summarize, filter or transform
  - Output the result

Outline stays the same, **Map** and **Reduce** change to fit the problem

# Map-Reduce: Word Count

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

**Big document**

# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

(The, 1)  
(crew, 1)  
(of, 1)  
(the, 1)  
(space, 1)  
(shuttle, 1)  
(Endeavor, 1)  
(recently, 1)  
.....

**Big document**

**(key, value)**

# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

## Group by key:

Collect all pairs with same key

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

(The, 1)  
(crew, 1)  
(of, 1)  
(the, 1)  
(space, 1)  
(shuttle, 1)  
(Endeavor, 1)  
(recently, 1)  
....

(crew, 1)  
(crew, 1)  
(space, 1)  
(the, 1)  
(the, 1)  
(the, 1)  
(shuttle, 1)  
(recently, 1)  
...

**Big document**

**(key, value)**

**(key, value)**

# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

(The, 1)  
(crew, 1)  
(of, 1)  
(the, 1)  
(space, 1)  
(shuttle, 1)  
(Endeavor, 1)  
(recently, 1)  
....

(key, value)

## Group by key:

Collect all pairs with same key

(crew, 1)  
(crew, 1)  
(space, 1)  
(the, 1)  
(the, 1)  
(the, 1)  
(shuttle, 1)  
(recently, 1)  
...

(key, value)

Provided by the programmer

## Reduce:

Collect all values belonging to the key and output

(crew, 2)  
(space, 2)  
(the, 3)  
(shuttle, 1)  
(recently, 1)  
...

(key, value)

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

Big document

# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

## Group by key:

Collect all pairs with same key

Provided by the programmer

## Reduce:

Collect all values belonging to the key and output

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

(The, 1)  
(crew, 1)  
(of, 1)  
(the, 1)  
(space, 1)  
(shuttle, 1)  
(Endeavor, 1)  
(recently, 1)  
....

(crew, 1)  
(crew, 1)  
(space, 1)  
(the, 1)  
(the, 1)  
(the, 1)  
(shuttle, 1)  
(recently, 1)  
...

(crew, 2)  
(space, 2)  
(the, 3)  
(shuttle, 1)  
(recently, 1)  
...

Big document

(key, value)

(key, value)

(key, value)



# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

(The, 1)

(crew, 1)

(of, 1)

(the, 1)

(space, 1)

(shuttle, 1)

(Endeavor, 1)

(recently, 1)

....

(key, value)

## Group by key:

Collect all pairs with same key

(crew, 1)

(crew, 1)

(space, 1)

(the, 1)

(the, 1)

(the, 1)

(shuttle, 1)

(recently, 1)

...

(key, value)

Provided by the programmer

## Reduce:

Collect all values belonging to the key and output

(crew, 2)

(space, 2)

(the, 3)

(shuttle, 1)

(recently, 1)

...

(key, value)

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need  
.....

Big document

# Map-Reduce: Word Count

Provided by the programmer

## MAP:

Read input and produces a set of key-value pairs

(The, 1)

(crew, 1)

(of, 1)

(the, 1)

(space, 1)

(shuttle, 1)

(Endeavor, 1)

(recently, 1)

....

(key, value)

## Group by key:

Collect all pairs with same key

(crew, 1)

(crew, 1)

(space, 1)

(the, 1)

(the, 1)

(the, 1)

(shuttle, 1)

(recently, 1)

...

(key, value)

Provided by the programmer

## Reduce:

Collect all values belonging to the key and output

(crew, 2)

(space, 2)

(the, 3)

(shuttle, 1)

(recently, 1)

...

(key, value)

The crew of the space shuttle Endeavor recently returned to Earth as ambassadors, harbingers of a new era of space exploration. Scientists at NASA are saying that the recent assembly of the Dextre bot is the first step in a long-term space-based man/mache partnership. "The work we're doing now -- the robotics we're doing -- is what we're going to need .....

Big document

# More formally...

- **Input:** a set of key-value pairs
- Programmer need to specify two methods:
  - **Map(k, v)**  $\rightarrow \langle k', v' \rangle^*$ 
    - Takes a key-value pair and outputs a set of key-value pairs  
E.g., key is the filename, value is a single line in the file
    - There is one Map call for every (k, v) pair
  - **Reduce(k',  $\langle v' \rangle^*$ )**  $\rightarrow \langle k', v'' \rangle^*$ 
    - **All values v' with same key k' are reduced together**
    - There is one Reduce function call per unique key k'

# Map-Reduce: Word Count

## pseudo-code

**map(key, value) :**

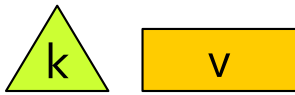
```
// key: document name; value: text of the document
  for each word w in value:
    emit(w, 1)
```

**reduce(key, values) :**

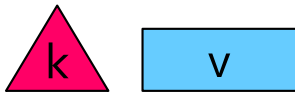
```
// key: a word; value: an iterator over counts
  result = 0
  for each count v in values:
    result += v
  emit(key, result)
```

# Map-Reduce: The Map Step

Input: key-value pairs



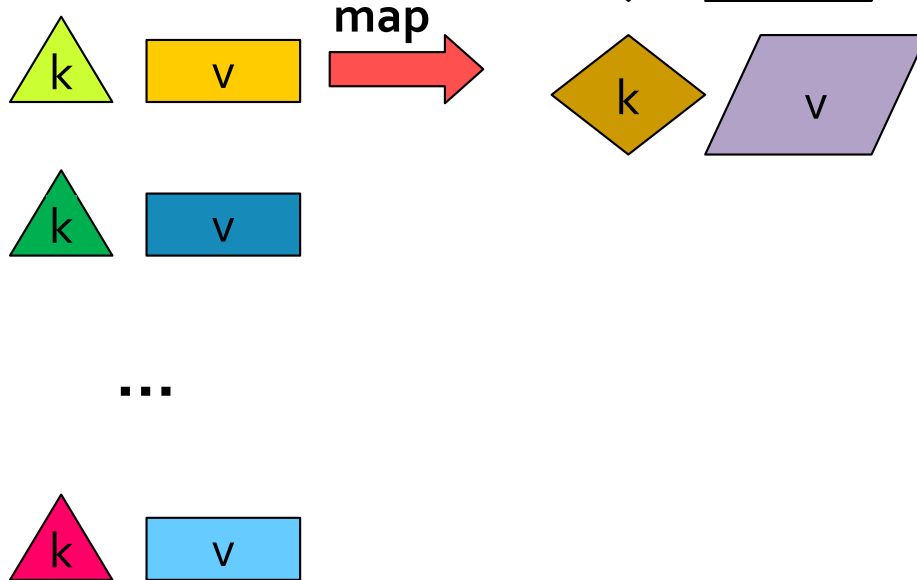
...



# Map-Reduce: The Map Step

Intermediate key-value pairs

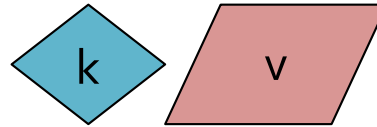
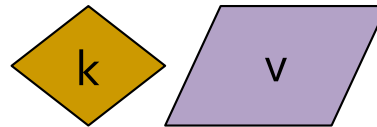
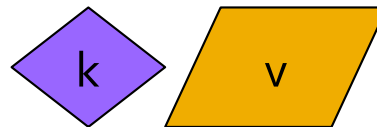
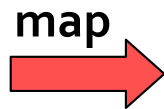
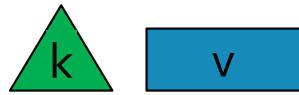
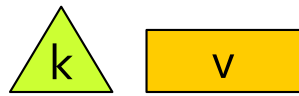
Input: key-value pairs



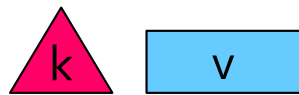
# Map-Reduce: The Map Step

Intermediate key-value pairs

Input: key-value pairs



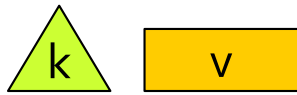
...



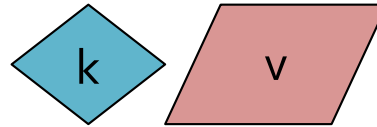
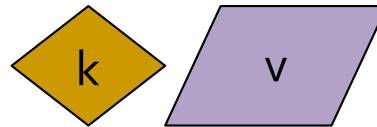
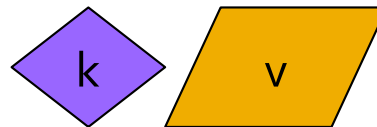
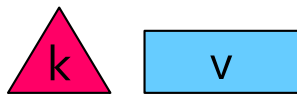
# Map-Reduce: The Map Step

Intermediate key-value pairs

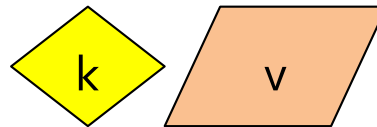
Input: key-value pairs



...



...



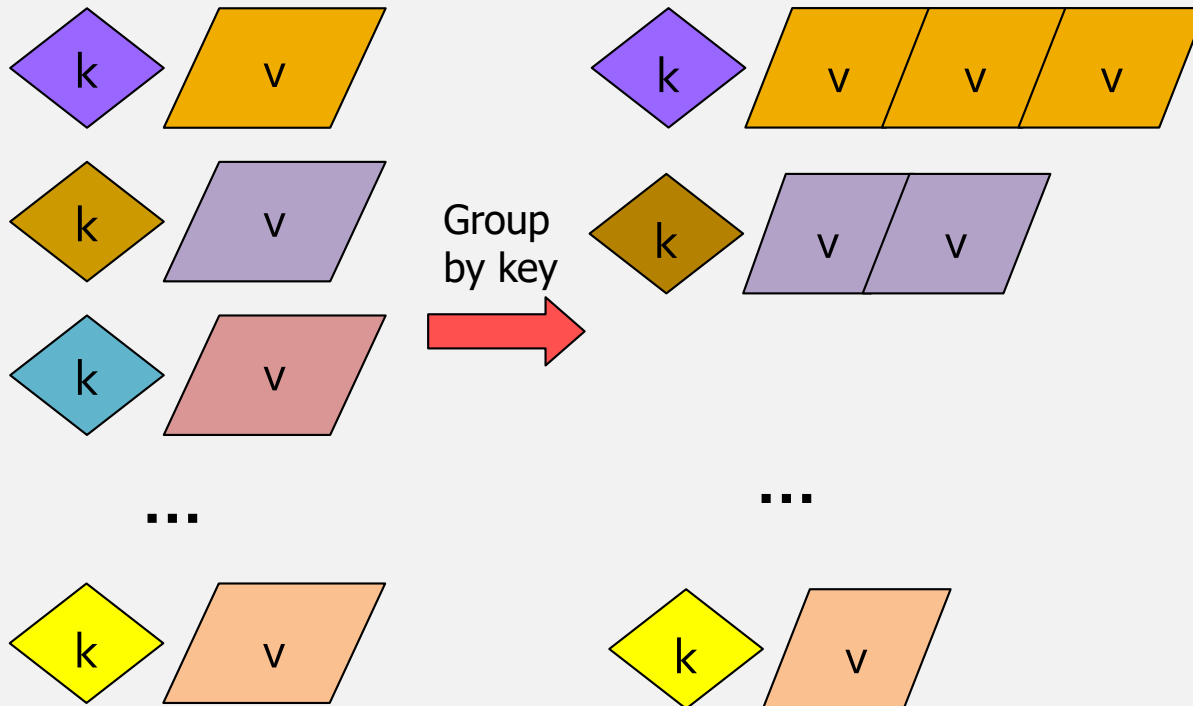


# Map-Reduce: The Reduce Step

## Shuffle/Group by Key Step

Intermediate key-value pairs

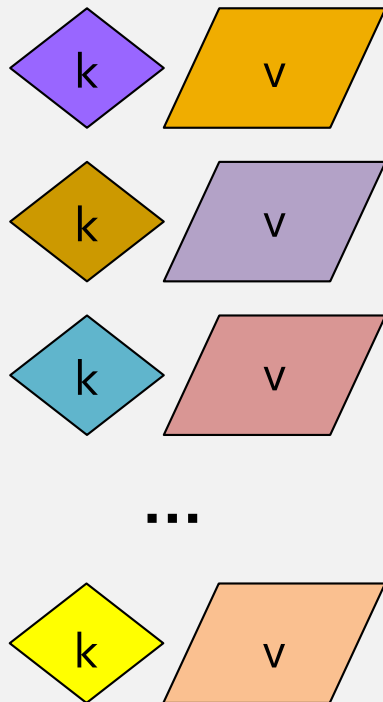
Key-value groups



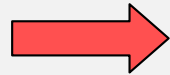
# Map-Reduce: The Reduce Step

## Shuffle/Group by Key Step

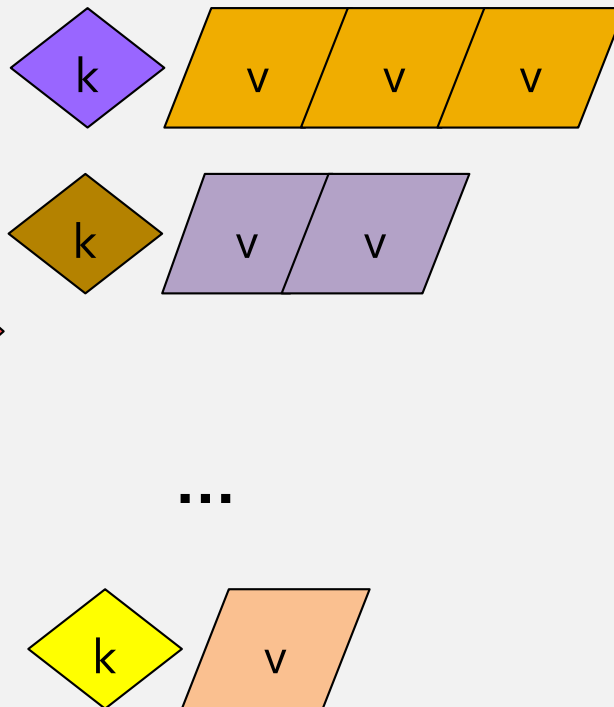
Intermediate key-value pairs



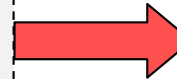
Group  
by key



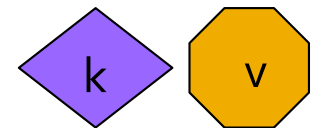
Key-value groups



reduce



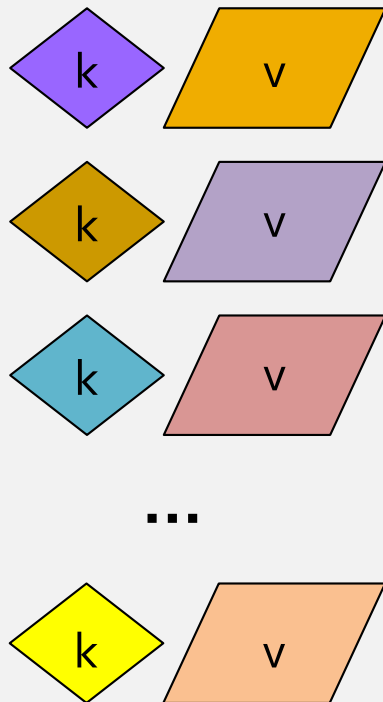
Output key-value pairs



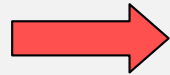
# Map-Reduce: The Reduce Step

## Shuffle/Group by Key Step

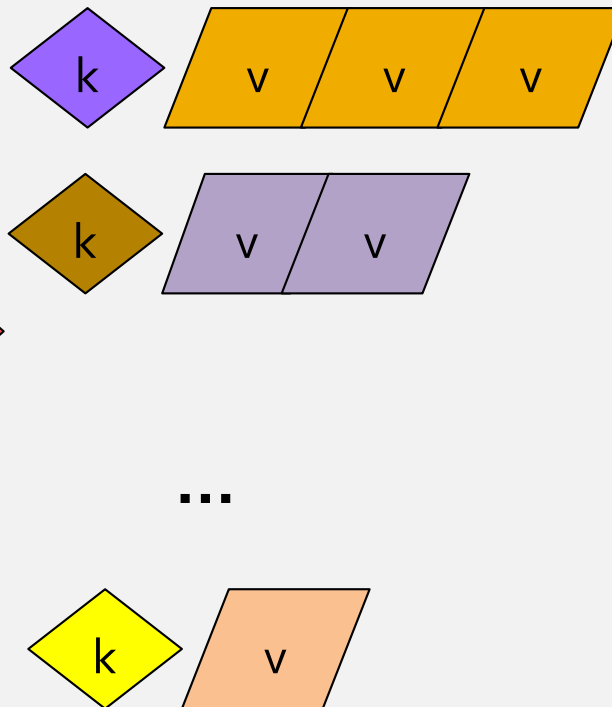
Intermediate key-value pairs



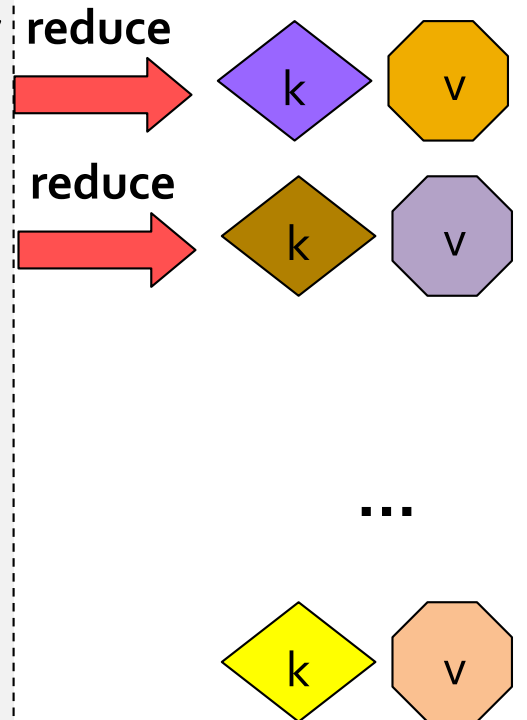
Group  
by key



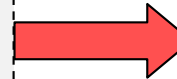
Key-value groups



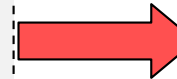
Output key-value pairs



reduce



reduce



# Map-Reduce example

## Build an inverted index

- (ID, content) => (content, List[IDs])
- Application: Search Engines, supporting full text searches

### Input:

tweet1, ("I love pancakes for breakfast")  
tweet2, ("I dislike pancakes")  
tweet3, ("What should I eat for breakfast?")  
tweet4, ("I love to eat")

### Desired output:

"pancakes", (tweet1, tweet2)  
"breakfast", (tweet1, tweet3)  
"eat", (tweet3, tweet4)  
"love", (tweet1, tweet4)  
...

### Map task:

What intermediate (key, value) pairs produced?

### Reduce task:

?

# Map-Reduce example

## Build an inverted index

- (ID, content) => (content, List[IDs])
- Application: Search Engines, supporting full text searches

### Input (key, value):

tweet1, ("I love pancakes for breakfast")  
tweet2, ("I dislike pancakes")  
tweet3, ("What should I eat for breakfast?")  
tweet4, ("I love to eat")

### Desired output:

"pancakes", (tweet1, tweet2)  
"breakfast", (tweet1, tweet3)  
"eat", (tweet3, tweet4)  
"love", (tweet1, tweet4)  
...

### Map task:

For each word in input value, emit (word, tweet\_ID) as intermediate (key, value) pair

### Reduce task

?

# Map-Reduce example

## Build an inverted index

- (ID, content) => (content, List[IDs])
- Application: Search Engines, supporting full text searches

### Input (key, value):

tweet1, ("I love pancakes for breakfast")  
tweet2, ("I dislike pancakes")  
tweet3, ("What should I eat for breakfast?")  
tweet4, ("I love to eat")

### Desired output:

"pancakes", (tweet1, tweet2)  
"breakfast", (tweet1, tweet3)  
"eat", (tweet3, tweet4)  
"love", (tweet1, tweet4)  
...

### Map task:

For each word in input value, emit (word, tweet\_ID) as intermediate (key, value) pair

### Reduce task

Reduce function emits key and list of tweet\_IDs associated with that key

# Map-Reduce example

## Social Network Analysis: Count Friends

- In a social network (Facebook, Instagram, etc.), how many friends does each person have?

### Input (key, value)

(Jim, Sue)  
(Sue, Jim)  
(Lin, Joe)  
(Joe, Lin)  
(Jim, Kai)  
(Kai, Jim)  
(Jim, Lin)  
(Lin, Jim)

### Desired Output

(Jim, 3)  
(Lin, 2)  
(Sue, 1)  
(Kai, 1)  
(Joe, 1)


**Map task: ? Reduce task: ?**

# Map-Reduce example

## Social Network Analysis: Count Friends

- In a social network (Facebook, Instagram, etc.), how many friends does each person have?

Input (key, value)

(Jim, Sue)		Jim, 1
(Sue, Jim)		Sue, 1
(Lin, Joe)		Lin, 1
(Joe, Lin)		Joe, 1
(Jim, Kai)		Jim, 1
(Kai, Jim)		Kai, 1
(Jim, Lin)		Jim, 1
(Lin, Jim)		Lin, 1

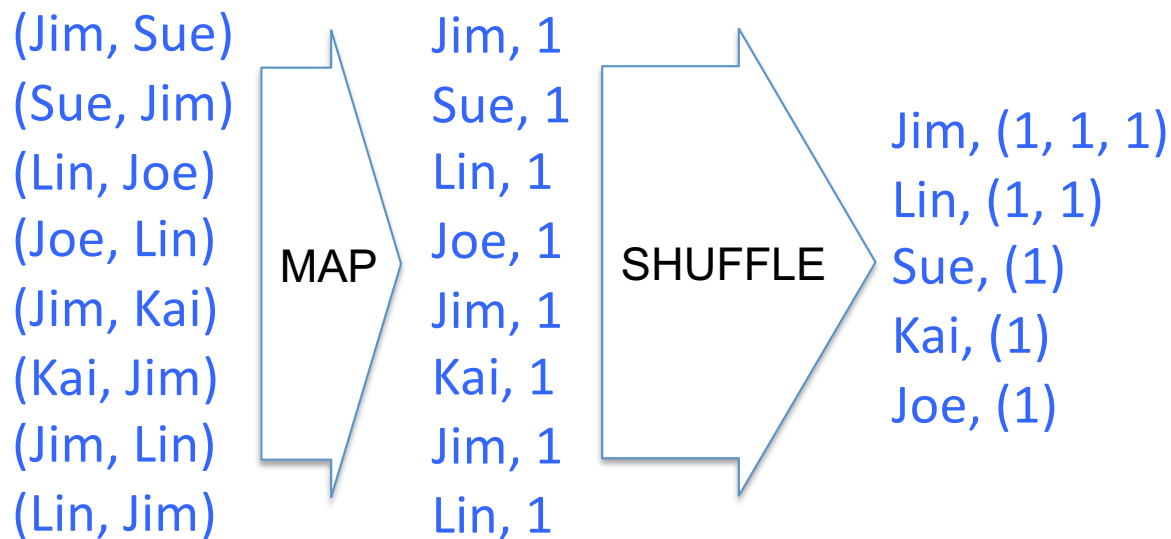


# Map-Reduce example

## Social Network Analysis: Count Friends

- In a social network (Facebook, Instagram, etc.), how many friends does each person have?

Input (key, value)



# Map-Reduce example

## Social Network Analysis: Count Friends

- In a social network (Facebook, Instagram, etc.), how many friends does each person have?

Input (key, value)

(Jim, Sue)  
(Sue, Jim)  
(Lin, Joe)  
(Joe, Lin)  
(Jim, Kai)  
(Kai, Jim)  
(Jim, Lin)  
(Lin, Jim)

MAP

Jim, 1  
Sue, 1  
Lin, 1  
Joe, 1  
Jim, 1  
Kai, 1  
Jim, 1  
Lin, 1

SHUFFLE

Jim, (1, 1, 1)  
Lin, (1, 1)  
Sue, (1)  
Kai, (1)  
Joe, (1)

REDUCE

Output

(Jim, 3)  
(Lin, 2)  
(Sue, 1)  
(Kai, 1)  
(Joe, 1)

# Map-Reduce example 0

## Integers divisible by 7

- Design a Map-Reduce algorithm that takes a very large file of integers and produces as output all unique integers from the original file that are evenly divisible by 7
- The large file of integers cannot fit in the memory of node

### Map task:

Each Map task gets a chunk of the file of integers and processes it ...

### Reduce task:

?

# Map-Reduce example

## Integers divisible by 7

- Design a Map-Reduce algorithm that takes a very large file of integers and produces as **output all unique integers** from the original file that are **evenly divisible by 7**
- The large file of integers cannot fit in the memory of node

```
map(key, value_list):  
    for v in value_list:  
        emit(v, 1)
```

# Map-Reduce example

## Integers divisible by 7

- Design a Map-Reduce algorithm that takes a very large file of integers and produces as **output all unique integers** from the original file that are **evenly divisible by 7**
- The large file of integers cannot fit in the memory of node

```
map(key, value_list) :  
    for v in value_list:  
        emit(v, 1)  
  
reduce(key, values) :  
    // Eliminate duplicates  
    if (v % 7) == 0 :  
        emit (key, 1)
```

# Map-Reduce example

## Integers divisible by 7

- Design a Map-Reduce algorithm that takes a very large file of integers and produces as **output all unique integers** from the original file that are **evenly divisible by 7**
- The large file of integers cannot fit in the memory of node

```
map(key, value_list):  
    for v in valuelist:  
        if (v % 7) == 0:  
            emit(v, 1)
```

```
reduce(key, values):  
    // Eliminate duplicates  
    emit (key, 1)
```

**Question: Why check whether divisible by 7 in the Map task rather than the Reduce task?**

# Map-Reduce example

## Integers divisible by 7

- Design a Map-Reduce algorithm that takes a very large file of integers and produces as **output all unique integers** from the original file that are **evenly divisible by 7**
- The large file of integers cannot fit in the memory of node

```
map(key, value_list):  
    for v in valuelist:  
        if (v % 7) == 0 :  
            emit(v, 1)
```

```
reduce(key, values):  
    // Eliminate duplicates  
    emit (key, 1)
```

**Question: Why check whether divisible by 7 in the Map task rather than the Reduce task?**

**Reduce communication: send less data over network**

# Map-Reduce example 1

## Find largest integer (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the largest integer** as output
- The large file of integers cannot fit in the memory of node

### Map task:

Each Map task gets a chunk of the file of integers and processes it ...

### Reduce task:

?



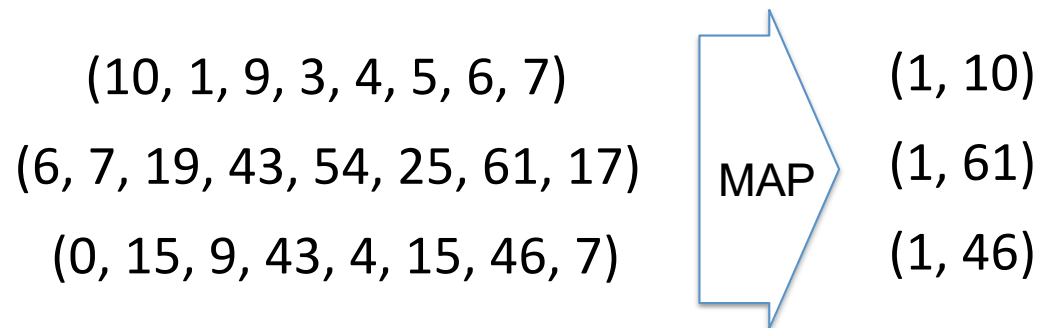
# Map-Reduce example

## Find largest integer (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the largest integer** as output
- The large file of integers cannot fit in the memory of node

### Map task:

Map task produces **(1, largest-integer)** of the largest value in that chunk as (key, value) pair

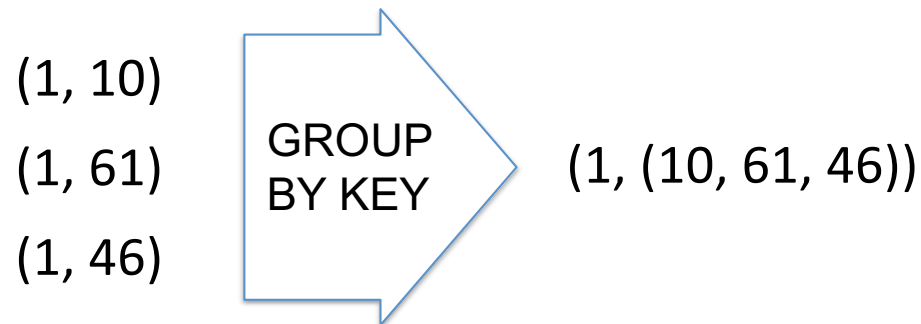


# Map-Reduce example

## Find largest integer (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the largest integer** as output
- The large file of integers cannot fit in the memory of node

### Group by key:



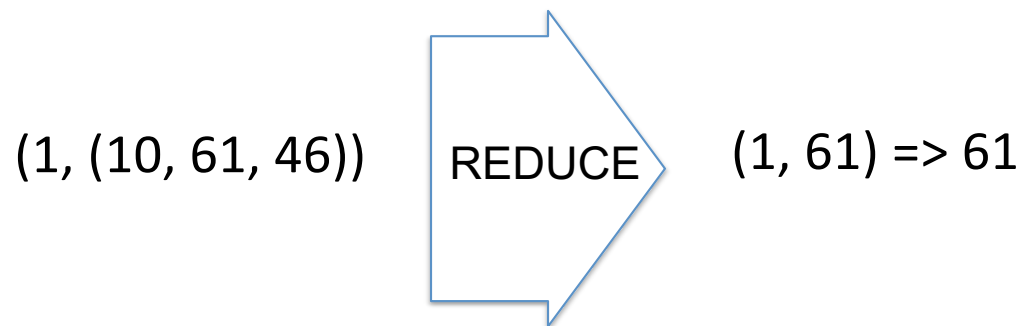
# Map-Reduce example

## Find largest integer (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the largest integer** as output
- The large file of integers cannot fit in the memory of node

### Reduce task:

Single Reduce task is needed to pick the largest integer



# Map-Reduce example 2

## Get unique integers (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **a set of unique integers** as output
- The large file of integers cannot fit in the memory of node

### Map task:

Each Map task gets a chunk of the file of integers and processes it ...

### Reduce task:

?

# Map-Reduce example

## Get unique integers (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **a set of unique integers** as output
- The large file of integers cannot fit in the memory of node

### Map task:

**Emit (integer, 1)** once only for each unique integer for each chunk

e.g., maintain an array or a map to track whether have seen/emitted that integer before

# Map-Reduce example

## Get unique integers (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **a set of unique integers** as output
- The large file of integers cannot fit in the memory of node

### Shuffle step:

Shuffle step will group together all values for the same integer:  
(integer, (1, 1, 1, 1, ...))

Same integer might **appear in multiple chunks** from Map tasks and **each integer key** will only go to one Reduce task

# Map-Reduce example

## Get unique integers (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **a set of unique integers** as output
- The large file of integers cannot fit in the memory of node

### Reduce task:

Each Reduce task eliminates duplicates (also ignore list of 1's) for each integer key and emit (integer)

Combine the output from multiple reduce tasks (not required to be in any order)

# Map-Reduce example 3

Count the number of unique integers (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the count of the number of distinct integers** as output
- The large file of integers cannot fit in the memory of node

## Map task:

Each Map task gets a chunk of the file of integers and processes it ...

## Reduce task:

?



# Map-Reduce example

Count the number of unique integers (Exercise 2.3.1 in book)

- **Two stages needed**
  - **First phase: eliminate duplicates in large input file**
    - Map task 1: just emit (integer, 1) for unique integers  
e.g., Mapper 1 (chunk1, (7, 7, 8, 8, ...)) => (7, 1), (8,1), ...  
Mapper 2 (chunk1, (7, 7, 9, ...)) => (7, 1), (9,1), ...
    - Reduce task 1: Eliminates duplicates (across chunks)  
e.g., (7, (1, 1)) => (7, 1) or just 7  
(8, (1)) => (8, 1) or just 8

# Map-Reduce example

Count the number of unique integers (Exercise 2.3.1 in book)

- **Two stages needed**
  - **Second phase: count unique numbers**
    - Map task 2: each map task gets some unique integers from the previous Reduce phase as the input and output the key-value pair like **(some key, count)** (key could be 1)  
  
e.g., (7, 8, 9, 12, 13) => (1, 5)
    - Reduce task 2: single Reducer, sums all counts from map tasks and produces overall count

# Combiners

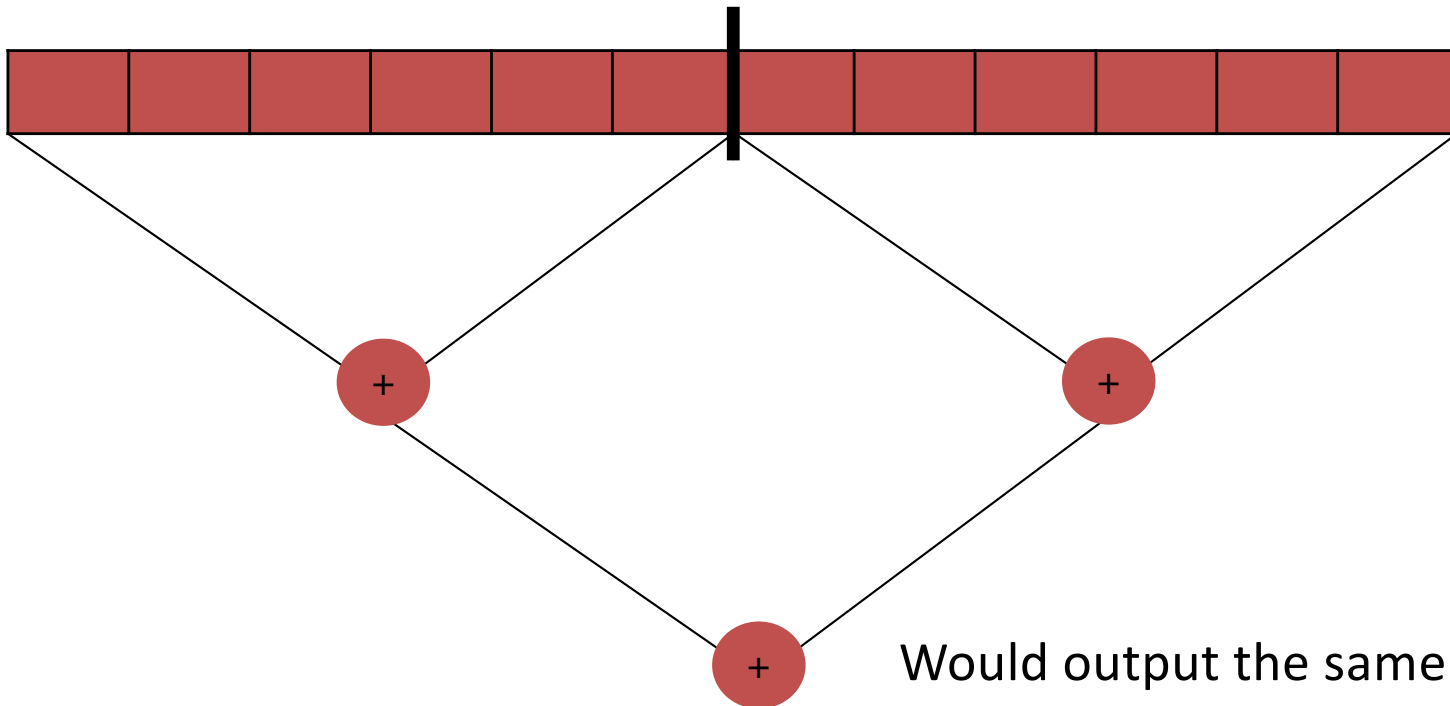
- Often a Map task will produce many pairs of the form  $(k, v_1), (k, v_2), \dots$  for **the same key  $k$** 
  - e.g., popular words “the” would appear thousands of times in the word count example, **generate thousands of (“the”, 1) tuples in each mapper, and ship to reducers**
  - Instead of producing many pairs (“the”, 1), (“the”, 1), ... can sum the  $n$  occurrences of “the” (each word) and **emit ( $w, n$ ) in Map task before shipping to reducers**
  - Now each node **only sends a single value** for each word
- Can save network time by **pre-aggregating values in the mapper**

# Combiners

- Combiners run **after the Mappers** and **before the Reducers**
- Combiner receives all data emitted by the Mapper as input
- The output of the Combiners is then sent to the Reducers
- Usage of the Combiner is **optional (When to use ?)**
- If combiner is suitable for the job, **the instances of the Combiner are run on every node that has run map tasks (Why?)**
- The Combiner is **a "mini-reduce" process** (usually the same as the reduce function)

# Combiners

- If a Reduce function is both **commutative** and **associative**, then it can be used as a Combiner
  - e.g., **sum** (add up all the input values)  
Commutative:  $a + b = b + a$   
Associative:  $(a + b) + c = a + (b + c)$



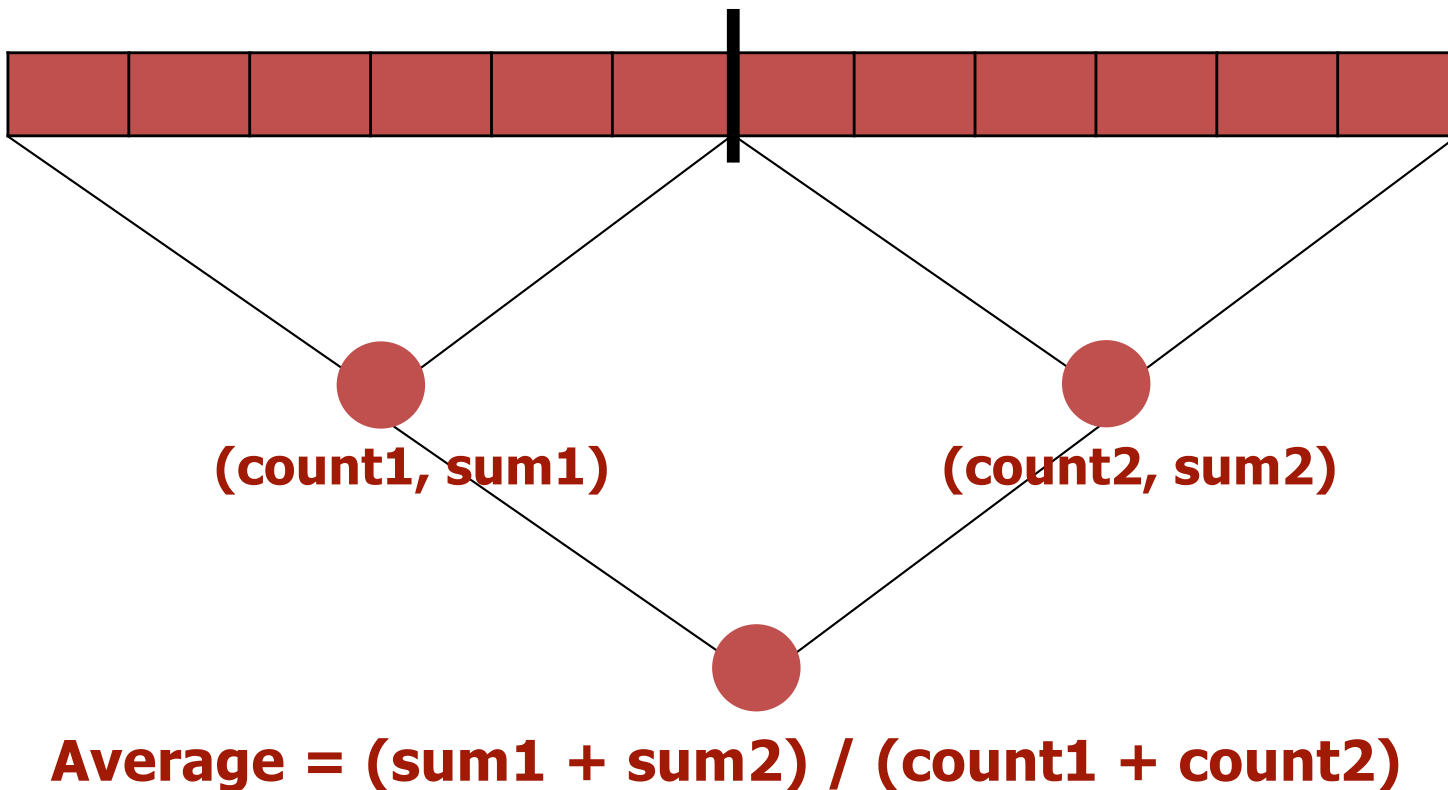
Would output the same result after adding the intermediate sum

# Combiners

- If a Reduce function is both **commutative** and **associative**, then it can be used as a Combiner
  - e.g., **Sum (add up all the input values)**  
Commutative:  $a + b = b + a$   
Associative:  $(a + b) + c = a + (b + c)$
- If the Reducer cannot be used directly as a Combiner because of commutativity or associativity
  - You still be able to write a third class to use as a Combiner
  - e.g., **Average**

# Combiners

- If the Reducer cannot be used directly as a Combiner because of commutativity or associativity
  - You still be able to write a third class to use as a Combiner
  - e.g., [Average](#)



# Map-Reduce example (with combiner)

## Compute average (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the average of all the integers** as output
- The large file of integers cannot fit in the memory of node

### Map task:

Each Map task gets a chunk of the file of integers and processes it ...

### Reduce task:

?



# Map-Reduce example (with combiner)

## Compute average (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the average of all the integers** as output
- The large file of integers cannot fit in the memory of node

### Map task:

Map task produces **(key, (number of integers, sum of integers))** for each chunk as output

- Key could be set to the same value for all mappers, e.g., 1
- Value is the (number, sum) tuple

# Map-Reduce example (with combiner)

## Compute average (Exercise 2.3.1 in book)

- Design MapReduce algorithms to take a very large file of integers and produce **the average of all the integers** as output
- The large file of integers cannot fit in the memory of node

### Reduce task:

A single Reduce task **sums all the sums of integers** and **sums number of integers**, calculate average; emit (average, 1)

Since each Map task summarizes a large chunk of data by a **single (key, (number, sum)) key-value pair**, this should be able to use a **single Reducer** even with thousands of Map tasks

# Map-Reduce Example (with combiner)

## Word length histogram

Goal: How many “big” (10+ letters), “medium” (5-9 letters), “small” (2-4 letters), and “tiny” (1 letter) words are used?

### Abridged Declaration of Independence

A Declaration By the Representatives of the United States of America, in General Congress Assembled.  
When in the course of human events it becomes necessary for a people to advance from that subordination in which they have hitherto remained, and to assume among powers of the earth the equal and independent station to which the laws of nature and of nature's god entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the change.  
We hold these truths to be self-evident; that all men are created equal and independent; that from that equal creation they derive rights inherent and inalienable, among which are the preservation of life, and liberty, and the pursuit of happiness; that to secure these ends, governments are instituted among men, deriving their just power from the consent of the governed; that whenever any form of government shall become destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying it's foundation on such principles and organizing it's power in such form, as to them shall seem most likely to effect their safety and happiness. Prudence indeed will dictate that governments long established should not be changed for light and transient causes: and accordingly all experience hath shewn that mankind are more disposed to suffer while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, begun at a distinguished period, and pursuing invariably the same object, evinces a design to reduce them to arbitrary power, it is their right, it is their duty, to throw off such government and to provide new guards for future security. Such has been the patient sufferings of the colonies; and such is now the necessity which constrains them to expunge their former systems of government. the history of his present majesty is a history of unremitting injuries and usurpations, among which no one fact stands single or solitary to contradict the uniform tenor of the rest, all of which have in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world, for the truth of which we pledge a faith yet unsullied by falsehood

# Map-Reduce Example (with combiner)

## Word length histogram

### Abridged Declaration of Independence

"Big" (Yellow) = 10+ letters

"Medium" (Red) = 5..9 letters

"Small" (Blue) = 2..4 letters

"Tiny" (Pink) = 1 letter

Map task: ?

Reduce task: ?

A Declaration By the Representatives of the United States of America, in General Congress Assembled.

When in the course of human events it becomes necessary for a people to advance from that subordination in which they have hitherto remained, and to assume among powers of the earth the equal and independent station to which the laws of nature and of nature's god entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the change.

We hold these truths to be self-evident; that all men are created equal and independent; that from that equal creation they derive rights inherent and inalienable, among which are the preservation of life, and liberty, and the pursuit of happiness; that to secure these ends, governments are instituted among men, deriving their just power from the consent of the governed; that whenever any form of government shall become destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying it's foundation on such principles and organizing it's power in such form, as to them shall seem most likely to effect their safety and happiness. Prudence indeed will dictate that governments long established should not be changed for light and transient causes: and accordingly all experience hath shewn that mankind are more disposed to suffer while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, begun at a distinguished period, and pursuing invariably the same object, evinces a design to reduce them to arbitrary power, it is their right, it is their duty, to throw off such government and to provide new guards for future security. Such has been the patient sufferings of the colonies; and such is now the necessity which constrains them to expunge their former systems of government. the history of his present majesty is a history of unremitting injuries and usurpations, among which no one fact stands single or solitary to contradict the uniform tenor of the rest, all of which have in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world, for the truth of which we pledge a faith yet unsullied by falsehood.

# Map-Reduce Example (with combiner)

## Word length histogram

### Abridged Declaration of Independence

A Declaration By the Representatives of the United States of America, in General Congress Assembled.

When in the course of human events it becomes necessary for a people to advance from that subordination in which they have hitherto remained, and to assume among powers of the earth the equal and independent station to which the laws of nature and of nature's god entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the change.

We hold these truths to be self-evident; that all men are created equal and independent; that from that equal creation they derive rights inherent and inalienable, among which are the preservation of life, and liberty, and the pursuit of happiness; that to secure these ends, governments are instituted among men, deriving their just power from the consent of the governed; that whenever any form of government shall become destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying it's foundation on such principles and organizing it's power in such form, as to them shall seem most likely to effect their safety and happiness. Prudence indeed will

dictate that governments long established should not be changed for light and transient causes: and accordingly all experience hath shewn that mankind are more disposed to suffer while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, begun at a distinguished period, and pursuing invariably the same object, evinces a design to reduce them to arbitrary power, it is their right, it is their duty, to throw off such government and to provide new guards for future security. Such has been the patient sufferings of the colonies; and such is now the necessity which constrains them to expunge their former systems of government. the history of his present majesty is a history of unremitting injuries and usurpations, among which no one fact stands single or solitary to contradict the uniform tenor of the rest, all of which have in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world, for the truth of which we pledge a faith yet unsullied by falsehood.

#### Chunk 1

Split the document into multiple chunks and process each chunk on different nodes

#### Chunk 2

# Map-Reduce Example (with combiner)

## Word length histogram

### Abridged Declaration of Independence

A Declaration By the Representatives of the United States of America, in General Congress Assembled.

When in the course of human events it becomes necessary for a people to advance from that subordination in which they have hitherto remained, and to assume among powers of the earth the equal and independent station to which the laws of nature and of nature's god entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the change.

We hold these truths to be self-evident; that all men are created equal and independent; that from that equal creation they derive rights inherent and inalienable, among which are the preservation of life, and liberty, and the pursuit of happiness; that to secure these ends, governments are instituted among men, deriving their just power from the consent of the governed; that whenever any form of government shall become destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying it's foundation on such principles and organizing it's power in such form, as to them shall seem most likely to effect their safety and happiness. Prudence indeed will

(key, value)

(yellow, 17)  
(red, 77)  
(blue, 107)  
(pink, 3)

Map Task 2  
(204 words)

dictate that governments long established should not be changed for light and transient causes: and accordingly all experience hath shewn that mankind are more disposed to suffer while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, begun at a distinguished period, and pursuing invariably the same object, evinces a design to reduce them to arbitrary power, it is their right, it is their duty, to throw off such government and to provide new guards for future security. Such has been the patient sufferings of the colonies; and such is now the necessity which constrains them to expunge their former systems of government. the history of his present majesty is a history of unremitting injuries and usurpations, among which no one fact stands single or solitary to contradict the uniform tenor of the rest, all of which have in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world, for the truth of which we pledge a faith yet unsullied by falsehood.

(yellow, 20)  
(red, 71)  
(blue, 93)  
(pink, 6 )

Map Task 2  
(190 words)

# Map-Reduce Example (with combiner)

## Word length histogram

### Map task 1

A Declaration By the Representatives of the United States of America, in General Congress Assembled.  
 When in the course of human events it becomes necessary for a people to advance from that subordination in which they have hitherto remained, and to assume among powers of the earth the equal and independent station to which the laws of nature and of nature's god entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the change.  
 We hold these truths to be self-evident; that all men are created equal and independent; that from that equal creation they derive rights inherent and inalienable, among which are the preservation of life, and liberty, and the pursuit of happiness; that to secure these ends, governments are instituted among men, deriving their just power from the consent of the governed; that whenever any form of government shall become destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying it's foundation on such principles and organizing it's power in such form, as to them shall seem most likely to effect their safety and happiness. Prudence indeed will

(yellow, 17)  
 (red, 77)  
 (blue, 107)  
 (pink, 3)

### Map task 2

dictate that governments long established should not be changed for light and transient causes: and accordingly all experience hath shewn that mankind are more disposed to suffer while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, begun at a distinguished period, and pursuing invariably the same object, evinces a design to reduce them to arbitrary power, it is their right, it is their duty, to throw off such government and to provide new guards for future security. Such has been the patient sufferings of the colonies; and such is now the necessity which constrains them to expunge their former systems of government. the history of his present majesty is a history of unremitting injuries and usurpations, among which no one fact stands single or solitary to contradict the uniform tenor of the rest, all of which have in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world, for the truth of which we pledge a faith yet unshaken by falsehood.

(yellow, 20)  
 (red, 71)  
 (blue, 93)  
 (pink, 6 )

### Shuffle step

(yellow, 17)  
 (yellow, 20)

(red, 77)  
 (red, 71)

(blue, 93)  
 (blue, 107)

(pink, 6)  
 (pink, 3)

### Reduce tasks

(yellow, 37)

(red, 148)

(blue, 200)

(pink, 9)

# Map-Reduce: Summary

- **Map tasks**

- Input is in “key-value” format, e.g., key = file locations, value = text
- Map code written by the user
- Processes chunks and produces sequence of key-value pairs, Notice: not the “key” in usual sense, do not have to be unique

- **Group by Key/Shuffle**

- Collects key-value pairs from each Map task
- Values associated with each key are formed into a list of values
- All key-value pairs with same key go to same Reduce task

- **Reduce task**

- Reduce code written by user
- Produces output key-value pairs

