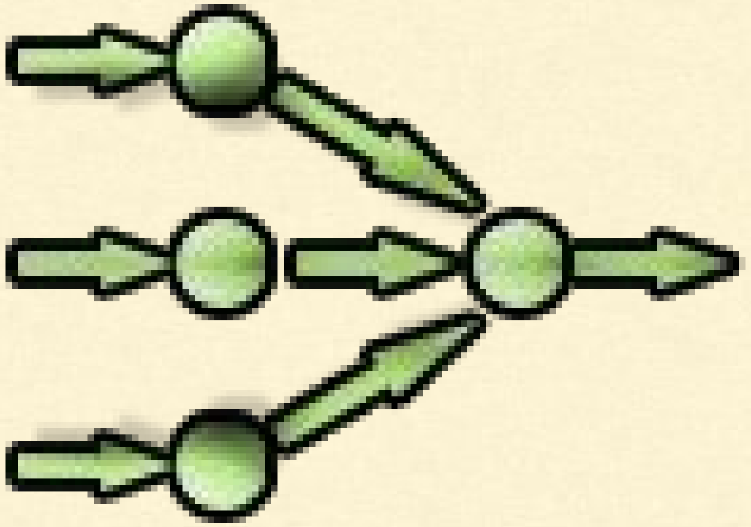


Welcome to MapReduce Session

TODAY'S CLASS

- Thinking in MapReduce
    - Word Frequency Problem
        - Solution 1 - Coding
        - Solution 2 - SQL
        - Solution 3 - Unix Pipes
        - Solution 4 - External Sort
- Map/Reduce Overview
- Visualisation
- Analogies to groupby
- Assignments

Understanding Sorting

BIG DATA PROBLEM - PROCESSING

Q: How fast can 1GHz processor sort 1TB data? This data is made up of 10 billion 100 byte size strings.

A: Around 6-10 hours

What's wrong 6-10 hours?

We need

1. Faster Sort
2. Bigger Data Sorting
3. More often

BIG DATA PROBLEM - PROCESSING

Google, 8 Sept, 2011:
Sorting 10PB took 6.5 hrs on 8000 computers

Why Sorting is such as big deal

1. Every SQL Query is impacted by Sorting:
    - **Where** clause - Index (Sorting)
    - **Group By** - Involves Sorting
    - **Joins -** immensly enhanced by Sorting
    - **Order BY**
2. Most of the algorithms depend on sorting

What is Map/Reduce?

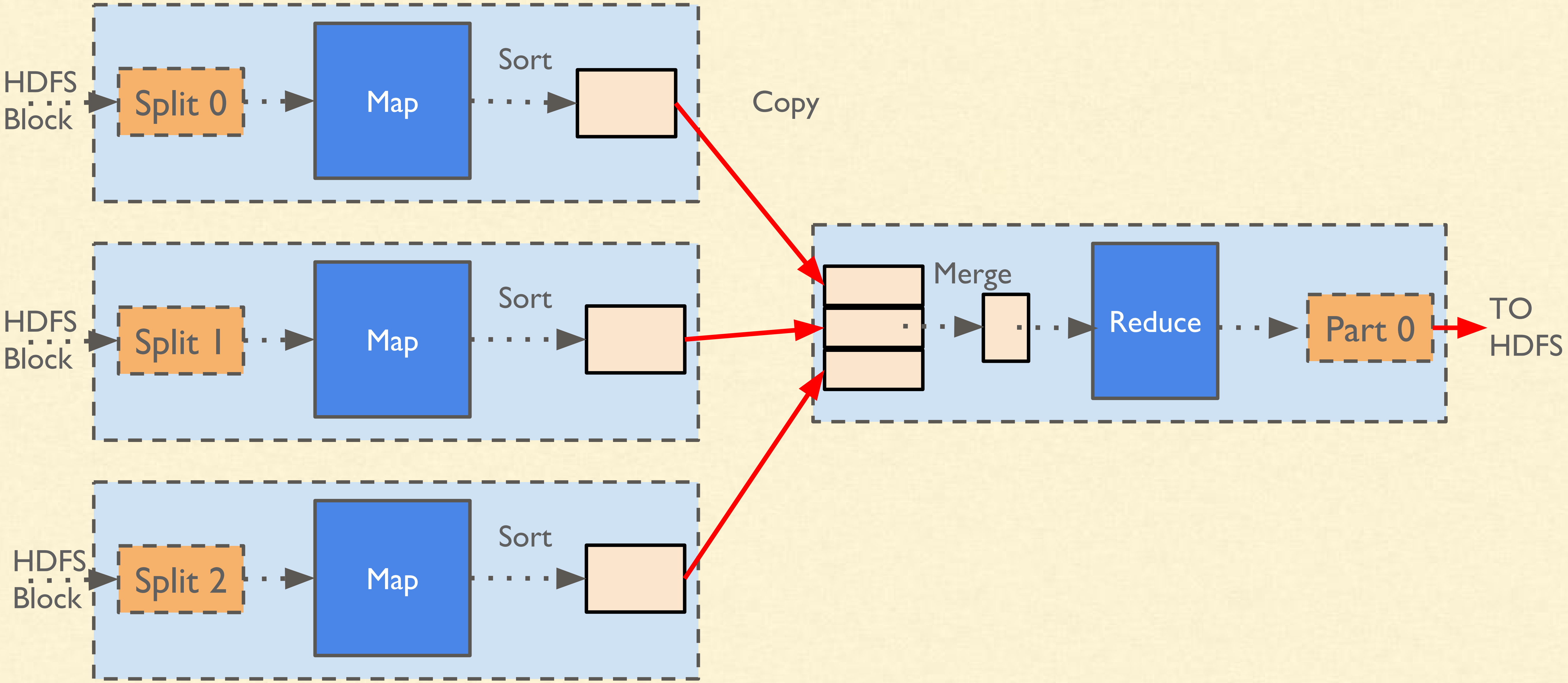
- Programming Paradigm
  - To help solve Big Data problems
  - Specifically sorting intensive jobs or disc read intensive
- You would have to code two functions:
  - Mapper - Converts Input into “key - value” pairs
  - Reducer - Aggregates all the values for a key

THINKING IN MAP / REDUCE

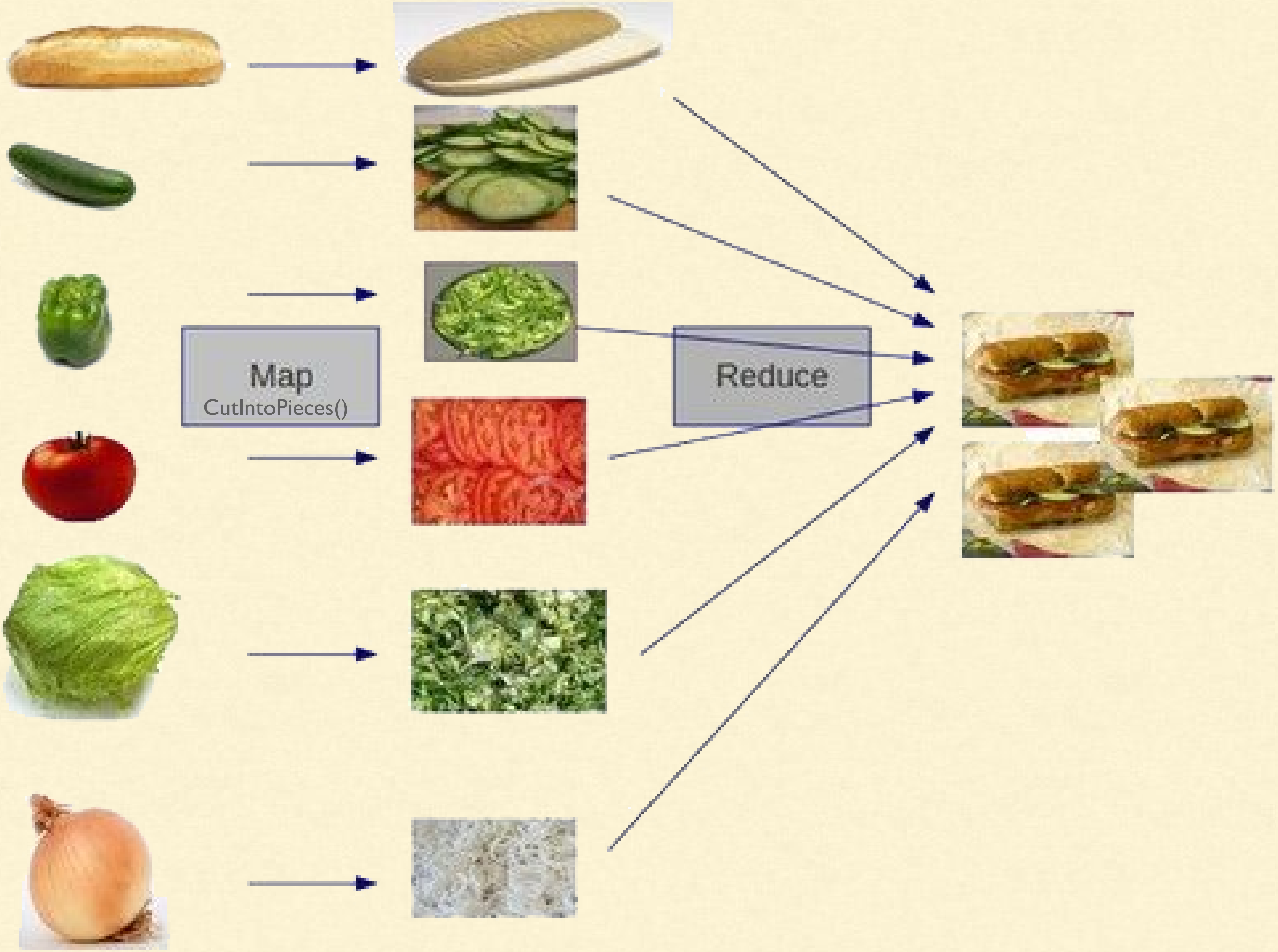
What is Map/Reduce?

- Also supported by many other systems such as
  - MongoDB / CouchDB / Cassandra
  - Apache Spark
- Mapper & Reducers in hadoop
  - can be written in Java, Shell, Python or any binary

MAP REDUCE - Multiple Reducers



MAP REDUCE



THINKING IN MAP / REDUCE

If you have the plain text file of containing 100s of text books,[500 mb] how would you find the frequencies of words?

THINKING IN MAP / REDUCE

If you have the plain text file of all the Lord Of Rings books, how would you find the frequencies of words?

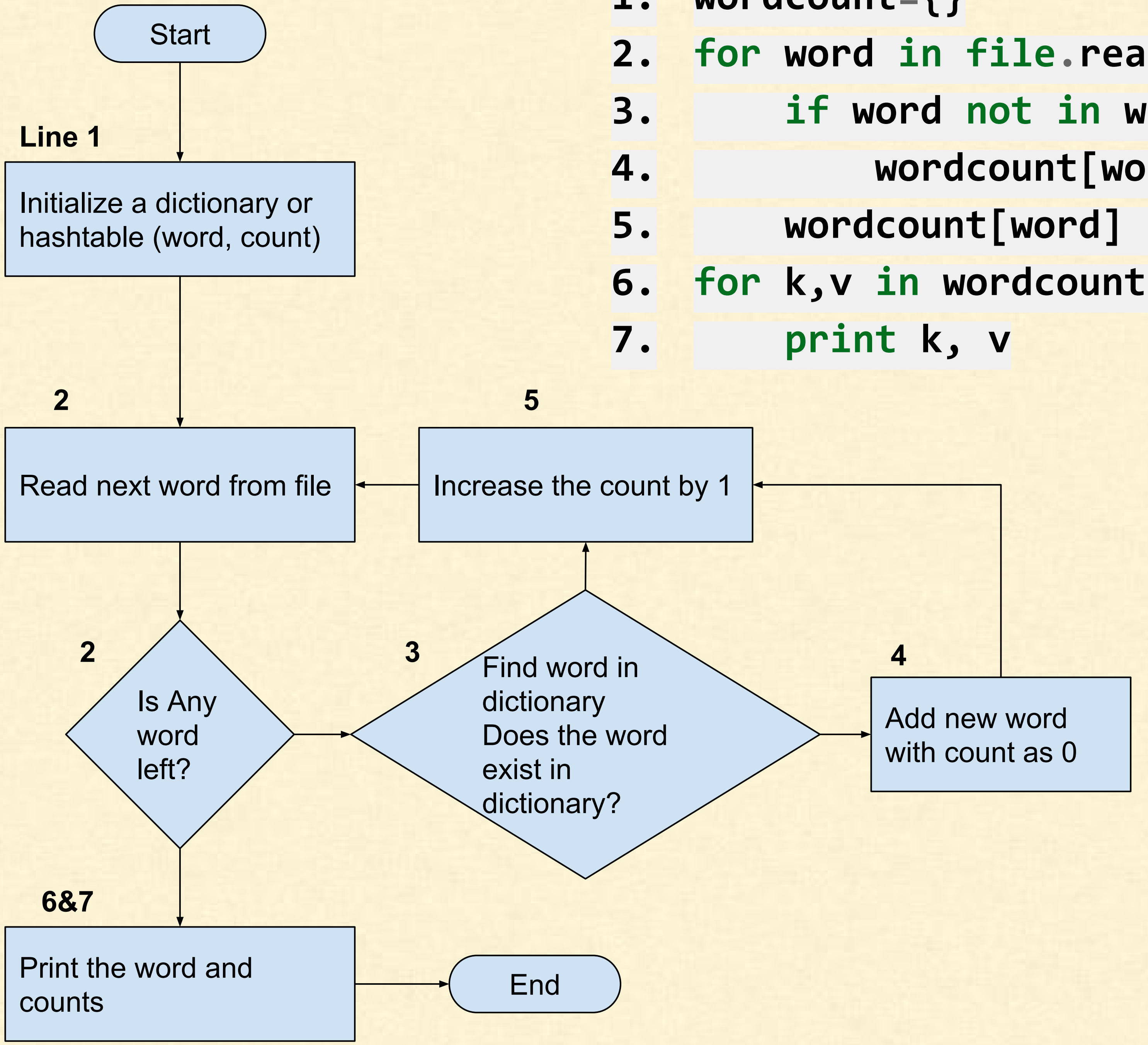
Approach 1 (Programmatic):

- Create a frequency hash table / dictionary
- For each word in the files
- Increase its frequency in the hash table
- When no more words left in file, print the hash table

Problems?

THINKING IN MAP / REDUCE

```
1.  wordcount={}
2.  for word in file.read().split():
3.      if word not in wordcount:
4.          wordcount[word] = 0
5.      wordcount[word] += 1
6.  for k,v in wordcount.items():
7.      print k, v
```


Start
Line 1
Initialize a dictionary or hashtable (word, count)
2
Read next word from file
5
Increase the count by 1
2
Is Any word left?
3
Find word in dictionary Does the word exist in dictionary?
4
Add new word with count as 0
6&7
Print the word and counts
End

Problems?

THINKING IN MAP / REDUCE

If you have the plain text file of all the Lord Of Rings books, how would you find the frequencies of words?

Approach 1 (Programmatic):

- Create a frequency hash table / dictionary
- For each word in the file
- Increase its frequency in the hash table
- When no more words left in file, print the hash table

Problems?

Can not process the data beyond RAM size.

THINKING IN MAP / REDUCE

If you have the plain text file of all the Lord Of Rings books, how would you find the frequencies of words?

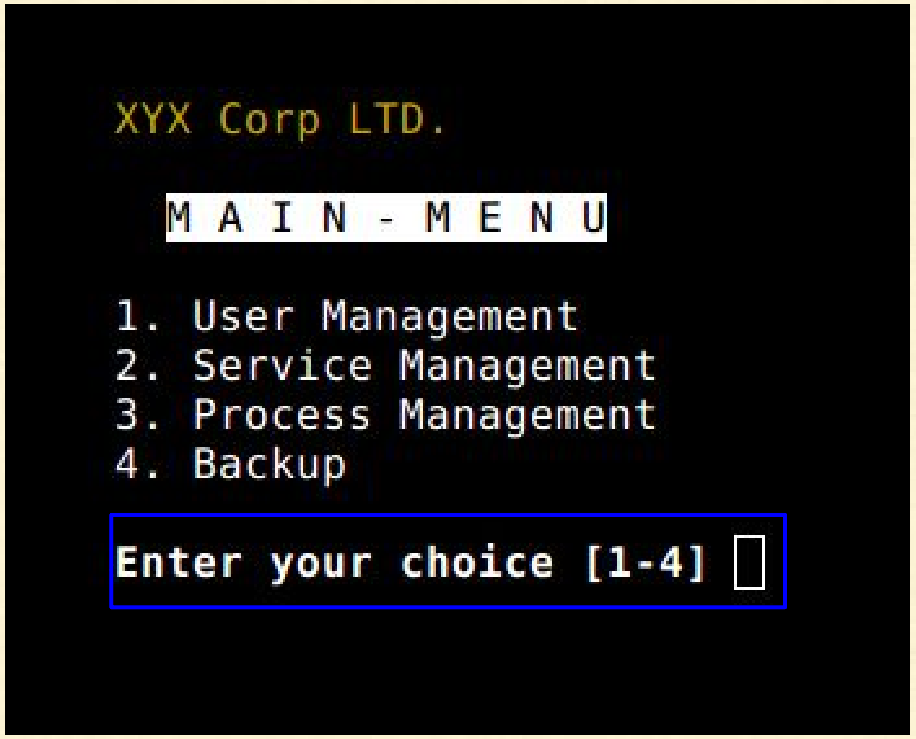
Approach2 (SQL):

- Break the books into one word per line
- Insert one word per row in database table
- Execute: *select word, count(*) from table group by word.*

Understanding Unix Pipeline

Understanding Unix Pipeline

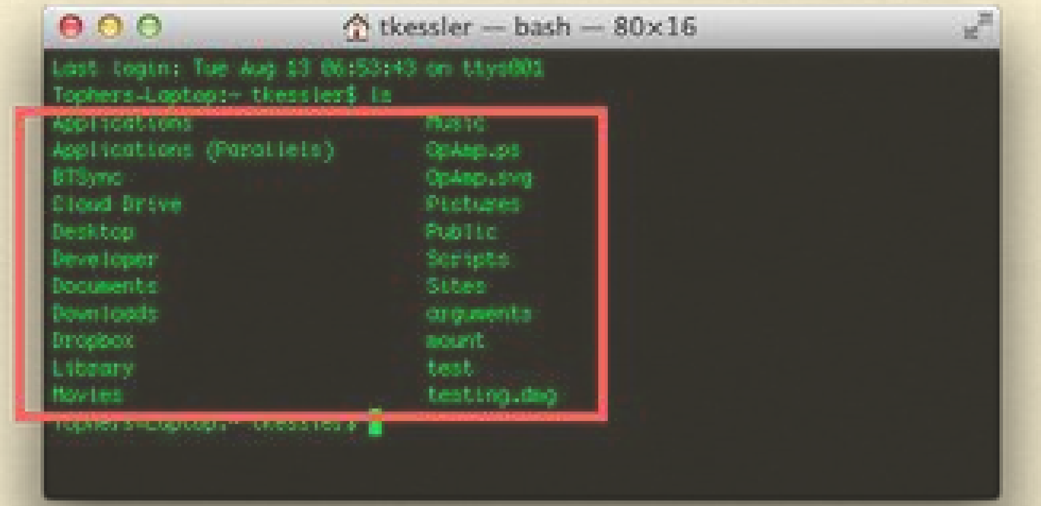
A program can take input from you.



```
XYX Corp LTD.

  M A I N - M E N U

1. User Management
2. Service Management
3. Process Management
4. Backup

Enter your choice [1-4]
```

Understanding Unix Pipeline

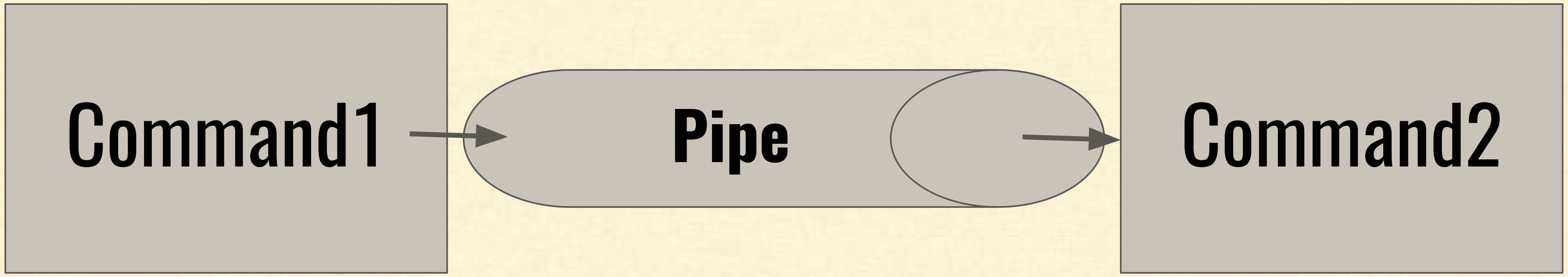
A program may also print some output



```
Last login: Tue Aug 13 06:53:43 on ttys001
Tophers-Laptop:~ tkessler$ ls
Applications                    Music
Applications (Parallels)        OpAmp.ps
BTSync                          OpAmp.svg
Cloud Drive                     Pictures
Desktop                         Public
Developer                       Scripts
Documents                       Sites
Downloads                       arguments
Dropbox                         mount
Library                         test
Movies                          testing.dmg
Tophers-Laptop:~ tkessler$
```

Understanding Unix Pipeline

command1 | command2



THINKING IN MAP / REDUCE

If you have the plain text file of all the Lord Of Rings books, how would you find the frequencies of words?

Approach 3 (Unix):

- Replace space with a newline
- Order lines with a sort command
- Then find frequencies using uniq
    - Scans from top to bottom
    - prints the count when line value changes

```
cat myfile| sed -E 's/[\t ]+/\n/g'| sort -S 1g | uniq -c
```

THINKING IN MAP / REDUCE

Problems in Approach 2 (SQL) & Approach 3 (Unix)?

THINKING IN MAP / REDUCE

Problems in Approach 2 (SQL) & Approach 3 (Unix)?

The moment the data starts going beyond RAM the time taken starts increasing. The following become bottlenecks:

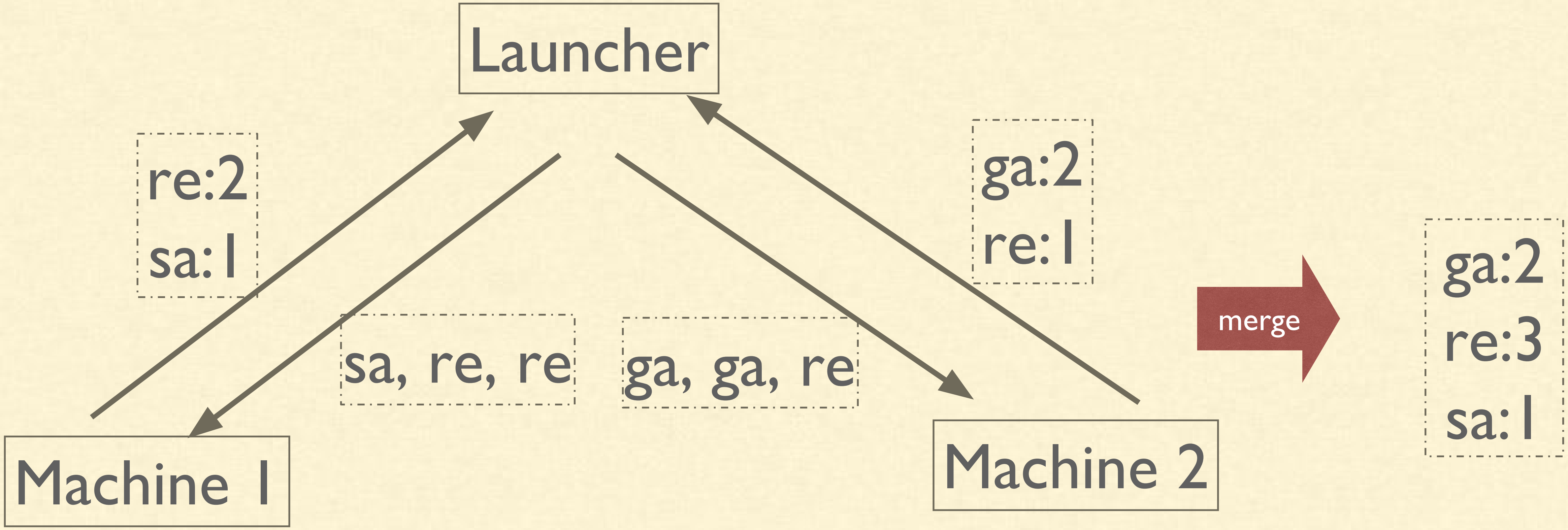
- CPU
- Disk Speed
- Disk Space

THINKING IN MAP / REDUCE

Then?

Approach 4: Use a external sort.

- Split the files to a size that fits RAM
- Use the previous approaches (2&3) to find freq
- Merge (sort -m) and sum-up frequencies



THINKING IN MAP / REDUCE

Merging

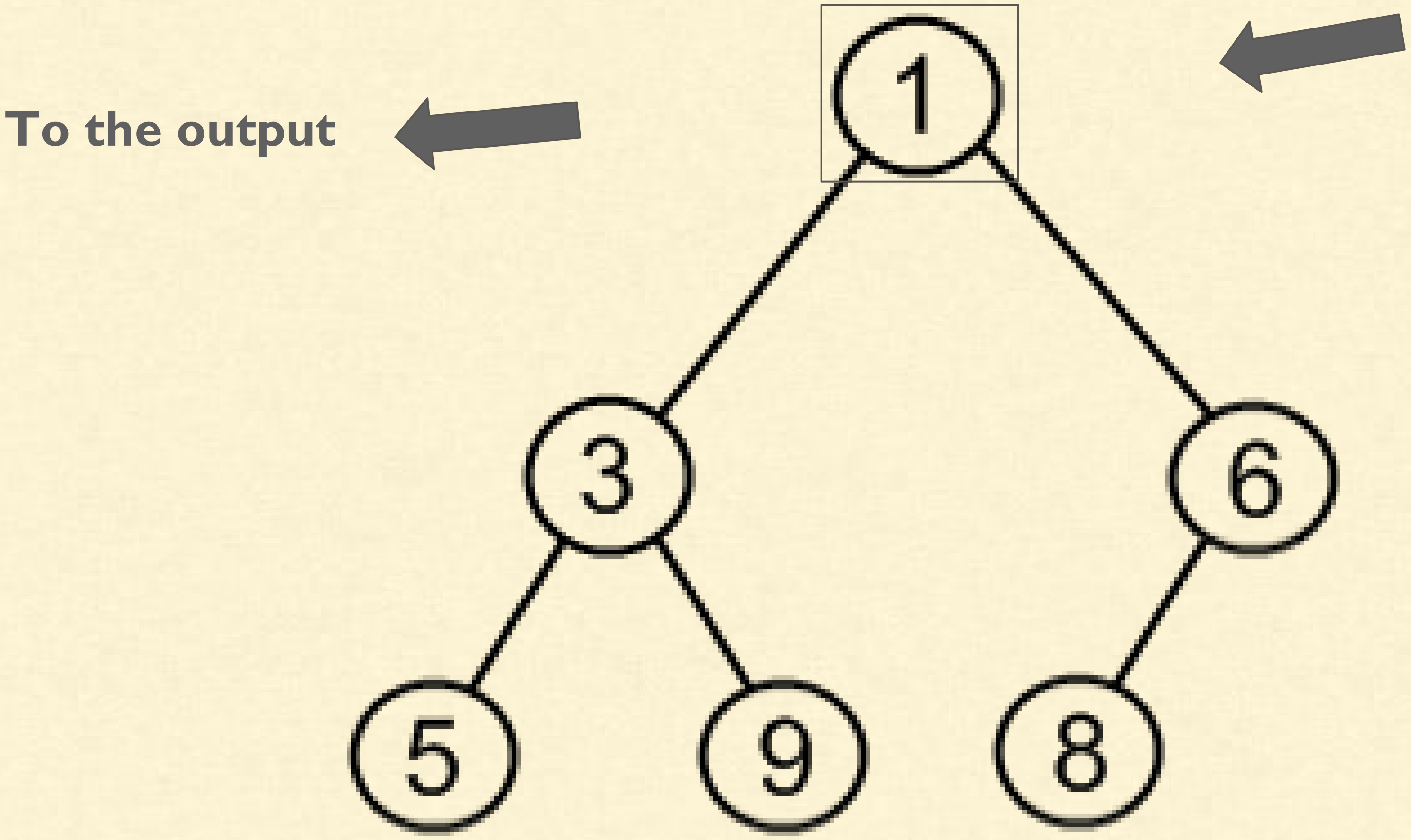
- Takes O(n) time to merge sorted data
- Or the time is proportional to the number of elements to be merged

THINKING IN MAP / REDUCE

Merging

- For more than two lists
  - ○ Use min-heap

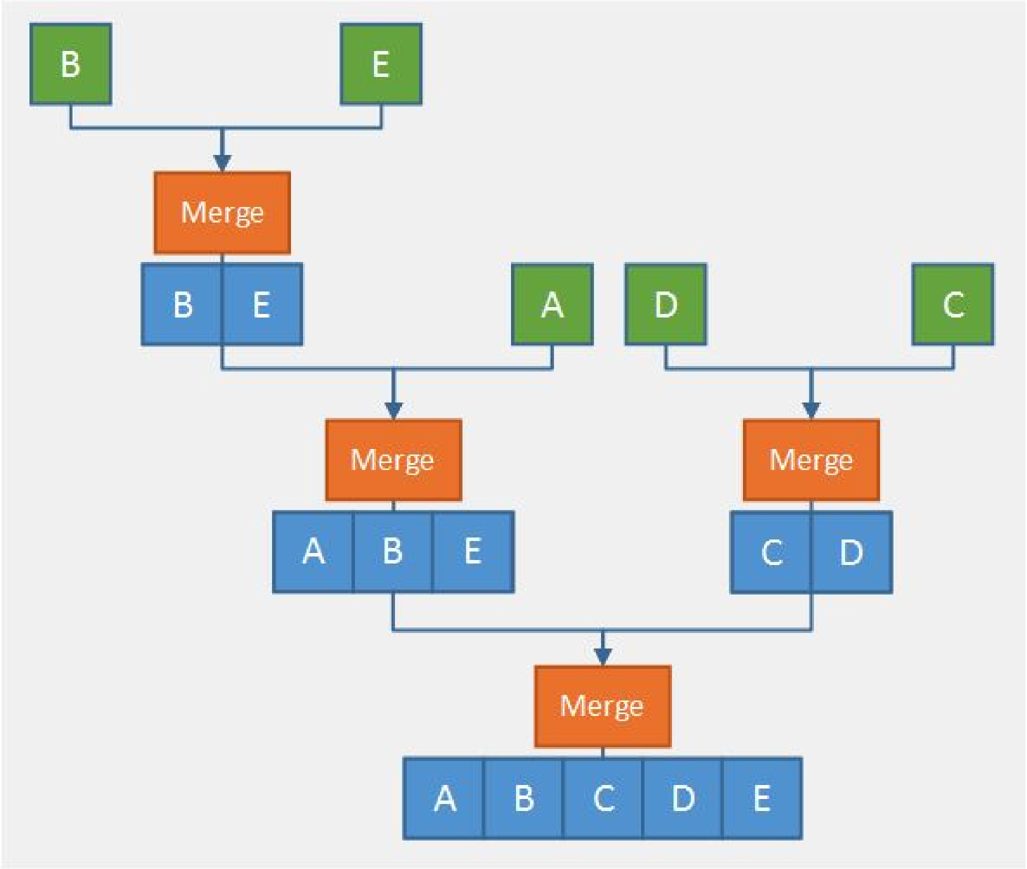
1	3	6
9	10	12
6	7	8
8	9	9
3	5	7
5	10	17



THINKING IN MAP / REDUCE

Merging

- For more than two lists
  - Or merge two at a time



THINKING IN MAP / REDUCE

Problems with Approach 4?

THINKING IN MAP / REDUCE

Problems with external Sort?

Time is consumed in transport of data.
+
For each requirement we would need to special purpose network oriented program.
+
Would Require A lot of Engineering.

Solution?
Use Map/Reduce

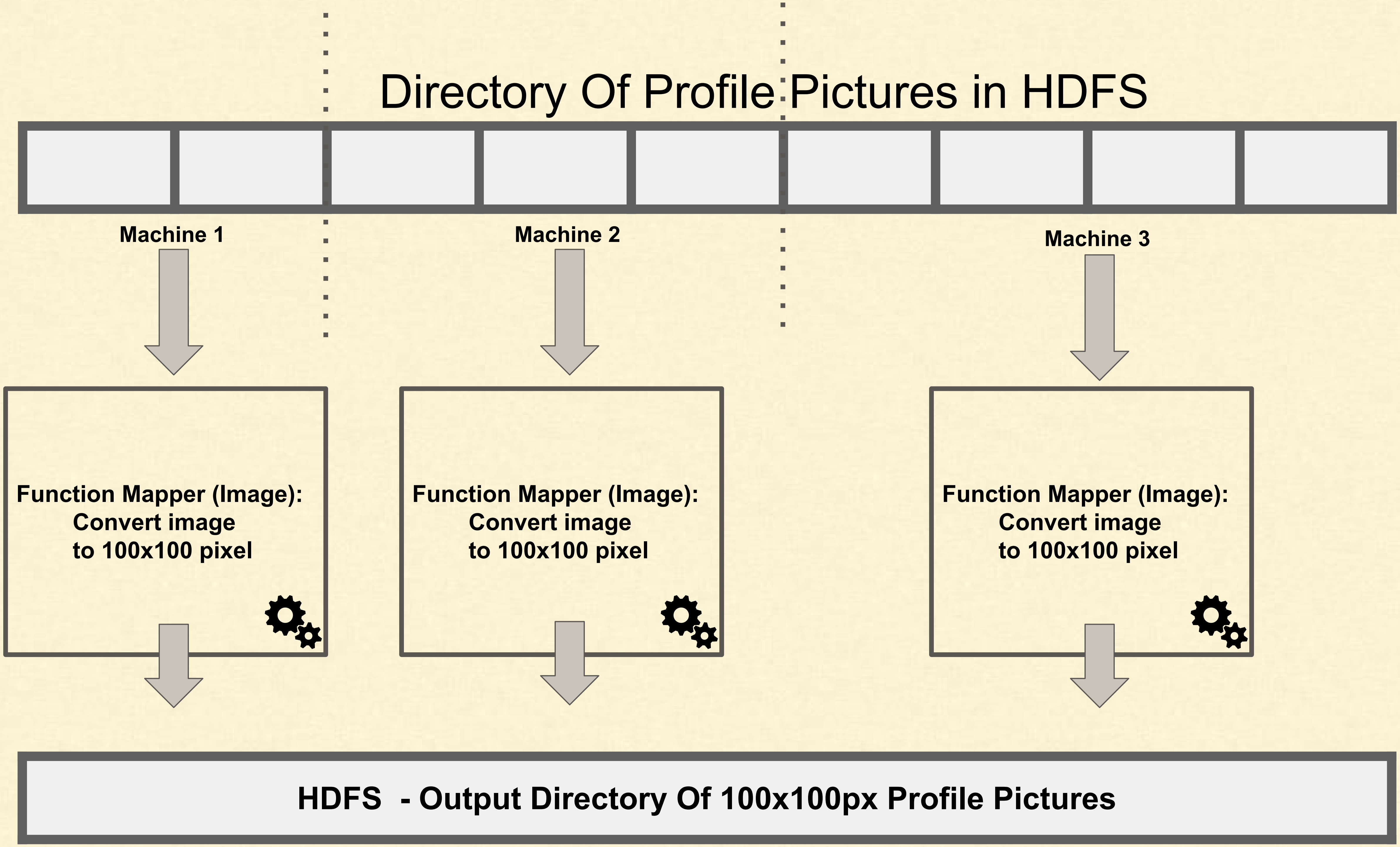
What is Map/Reduce?

- Programming Paradigm
  - To help solve Big Data problems
  - Specifically sorting intensive jobs or disc read intensive
- You would have to code two functions:
  - Mapper - Convert Input into “key - value” pairs
  - Reducer - Aggregates all the values for a key

What is Map/Reduce?

- Also supported by many other systems such as
  - MongoDB / CouchDB / Cassandra
  - Apache Spark
- Mapper & Reducers in hadoop
  - can be written in Java, Shell, Python or any binary

EXAMPLE OF ONLY MAPPER



Directory Of Profile Pictures in HDFS

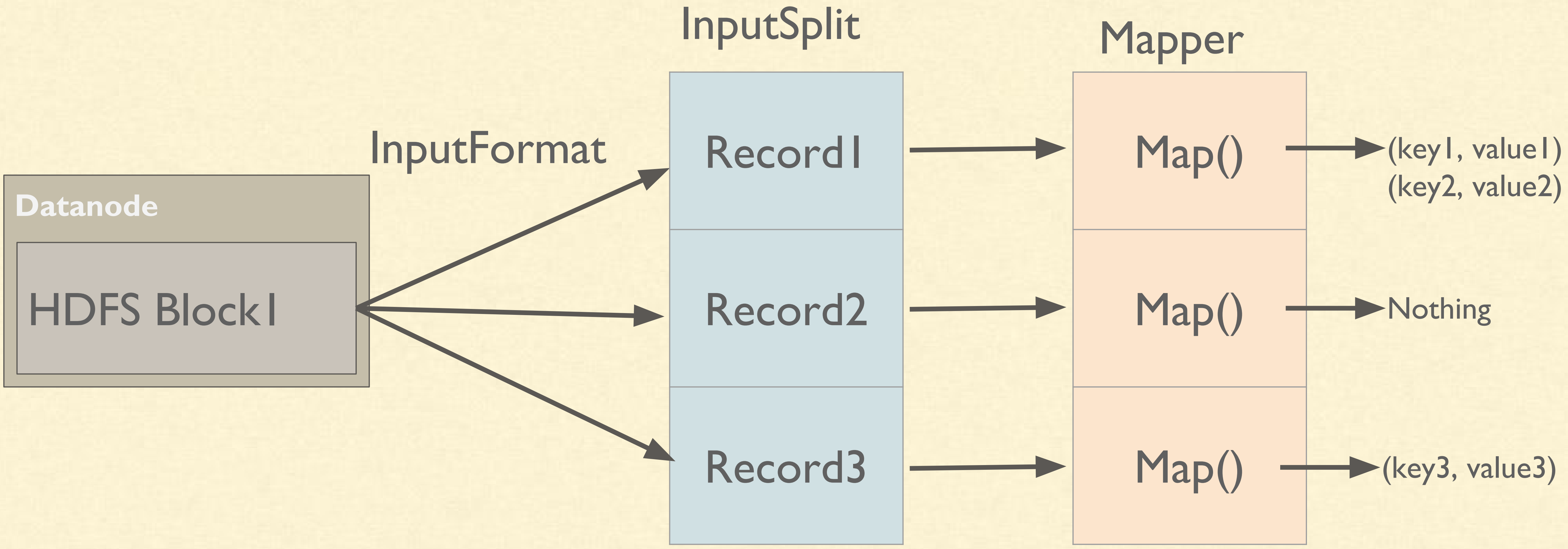
Function Mapper (Image):
Convert image
to 100x100 pixel

Function Mapper (Image):
Convert image
to 100x100 pixel

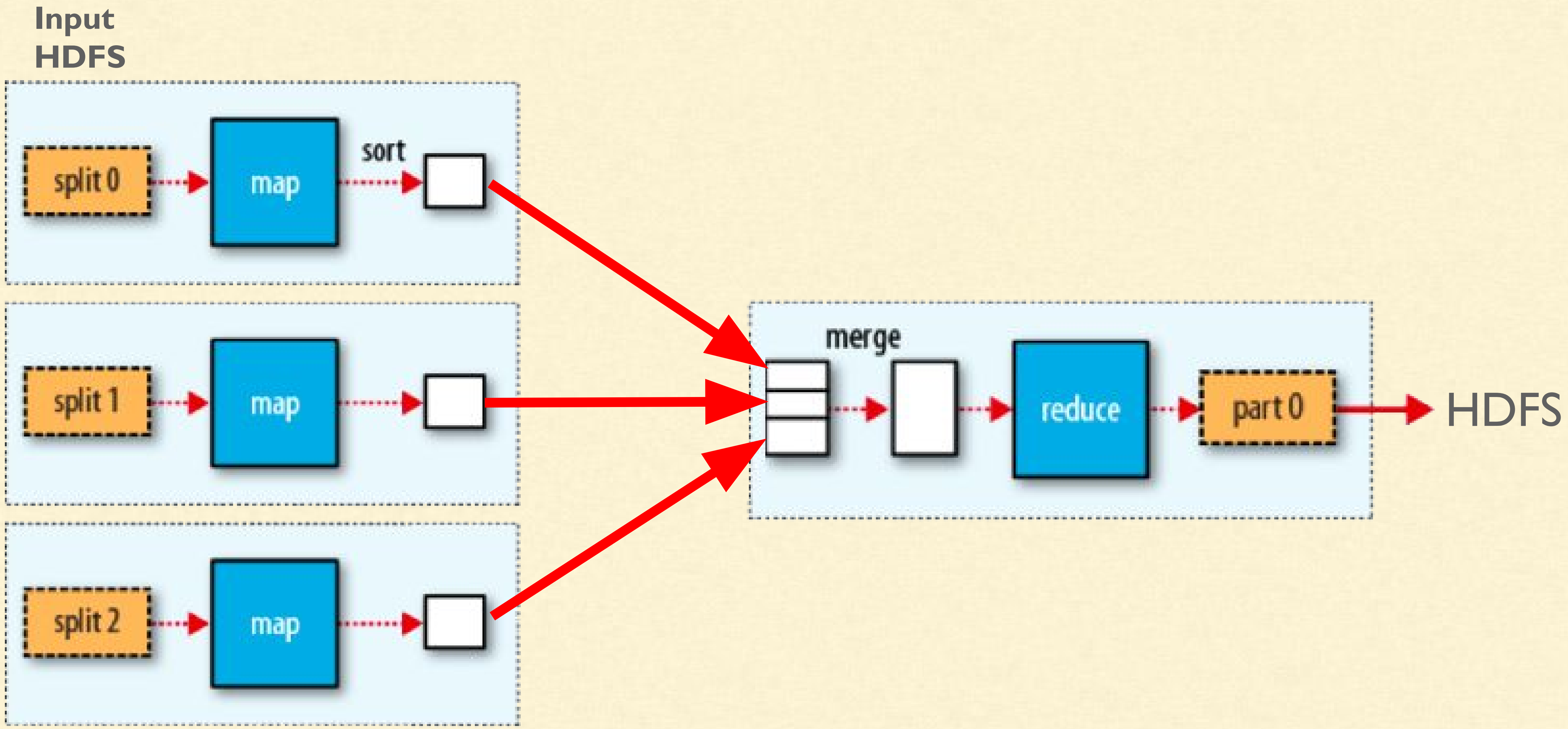
Function Mapper (Image):
Convert image
to 100x100 pixel

HDFS - Output Directory Of 100x100px Profile Pictures

Input Split

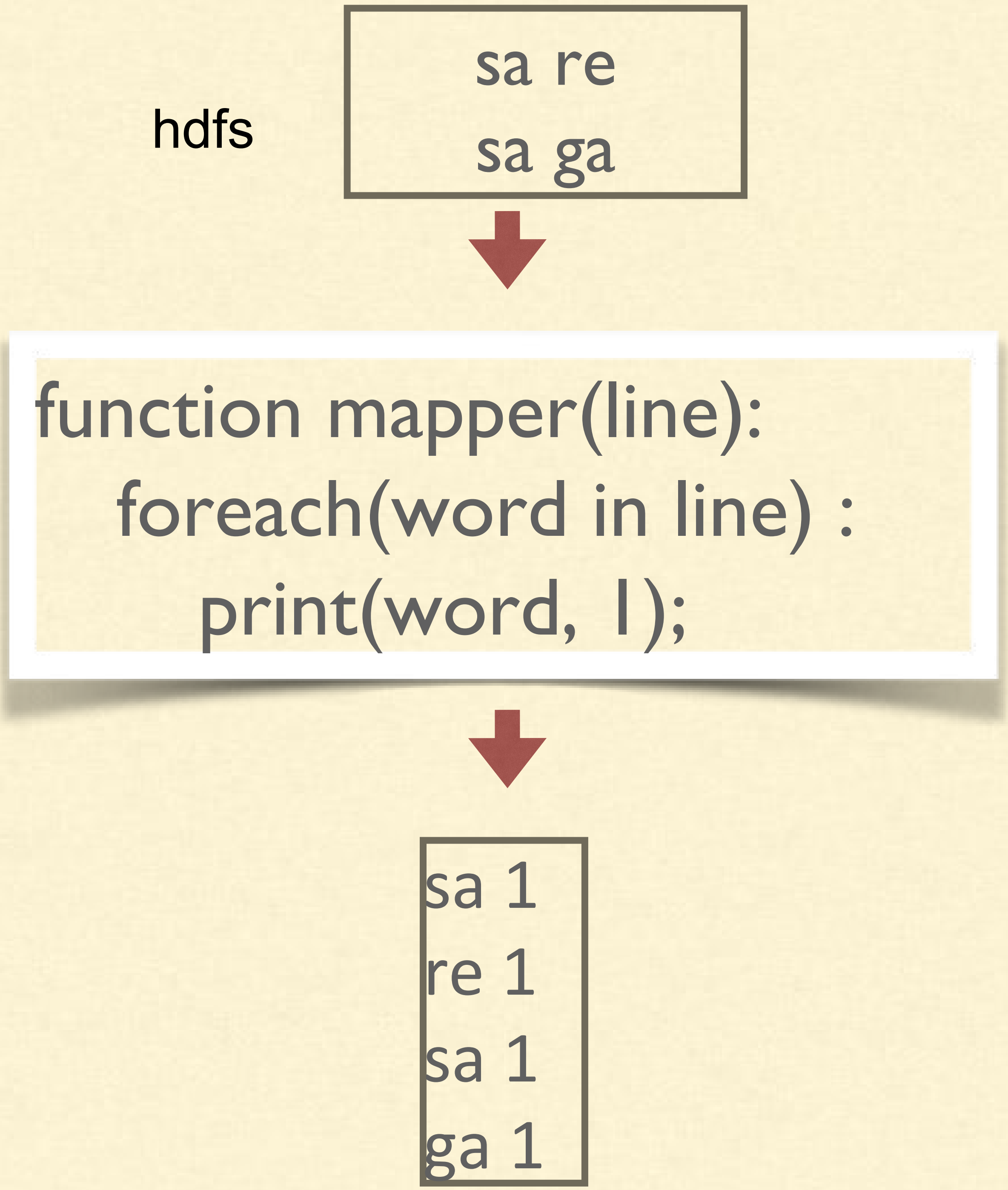

InputSplit
Mapper
InputFormat
Datanode
HDFS Block1
Record1
Record2
Record3
Map()
Map()
Map()
(key1, value1)
(key2, value2)
Nothing
(key3, value3)

With Both mapper() & Reducer() code



MAP / REDUCE

Mapper/Reducer for word frequency problem.



hdfs

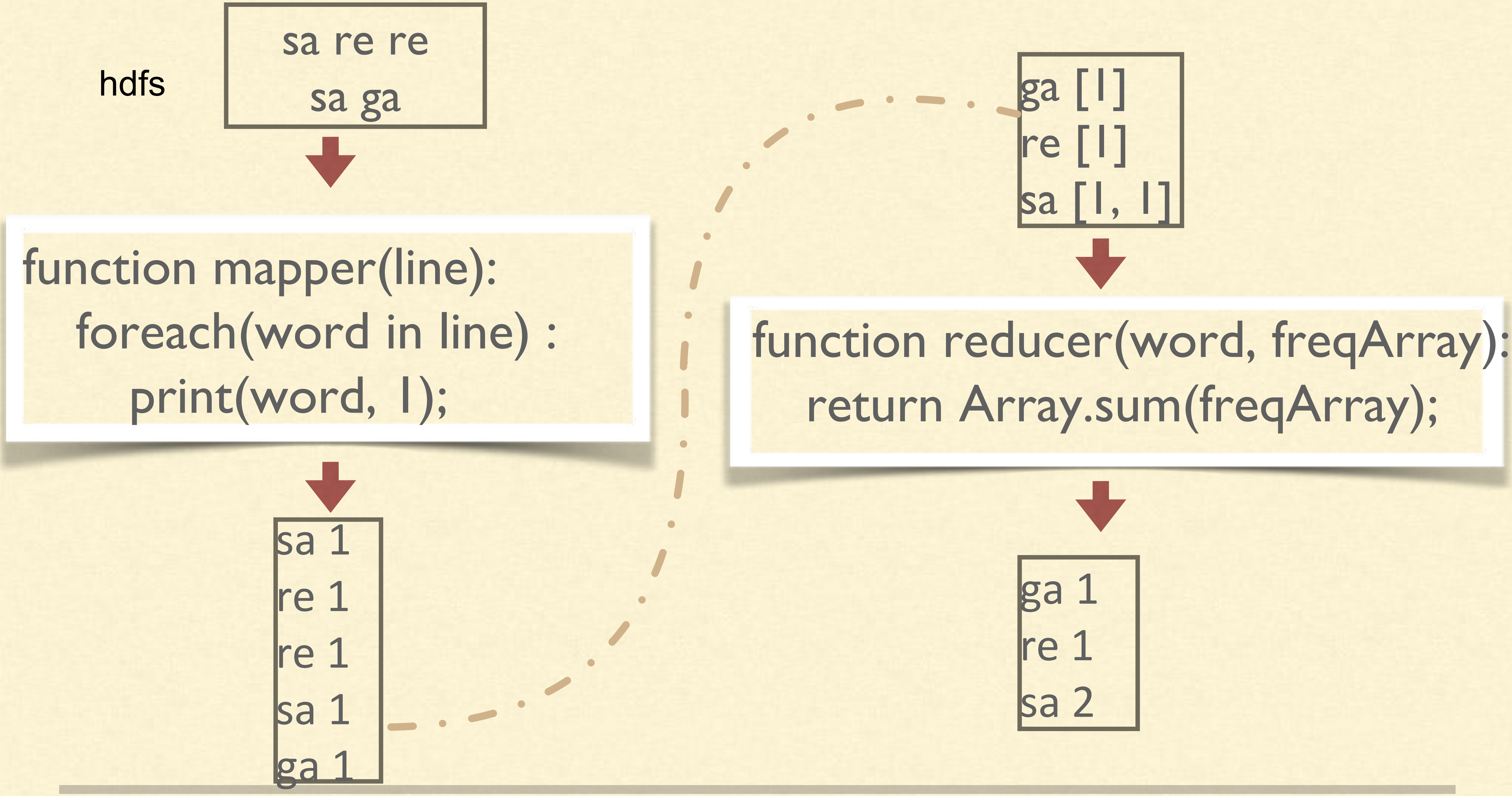
sa re
sa ga

```
function mapper(line):
    foreach(word in line) :
        print(word, 1);
```

sa 1
re 1
sa 1
ga 1

MAP / REDUCE

Mapper/Reducer for word frequency problem.



hdfs

sa re re
sa ga

```
function mapper(line):
    foreach(word in line) :
        print(word, 1);
```

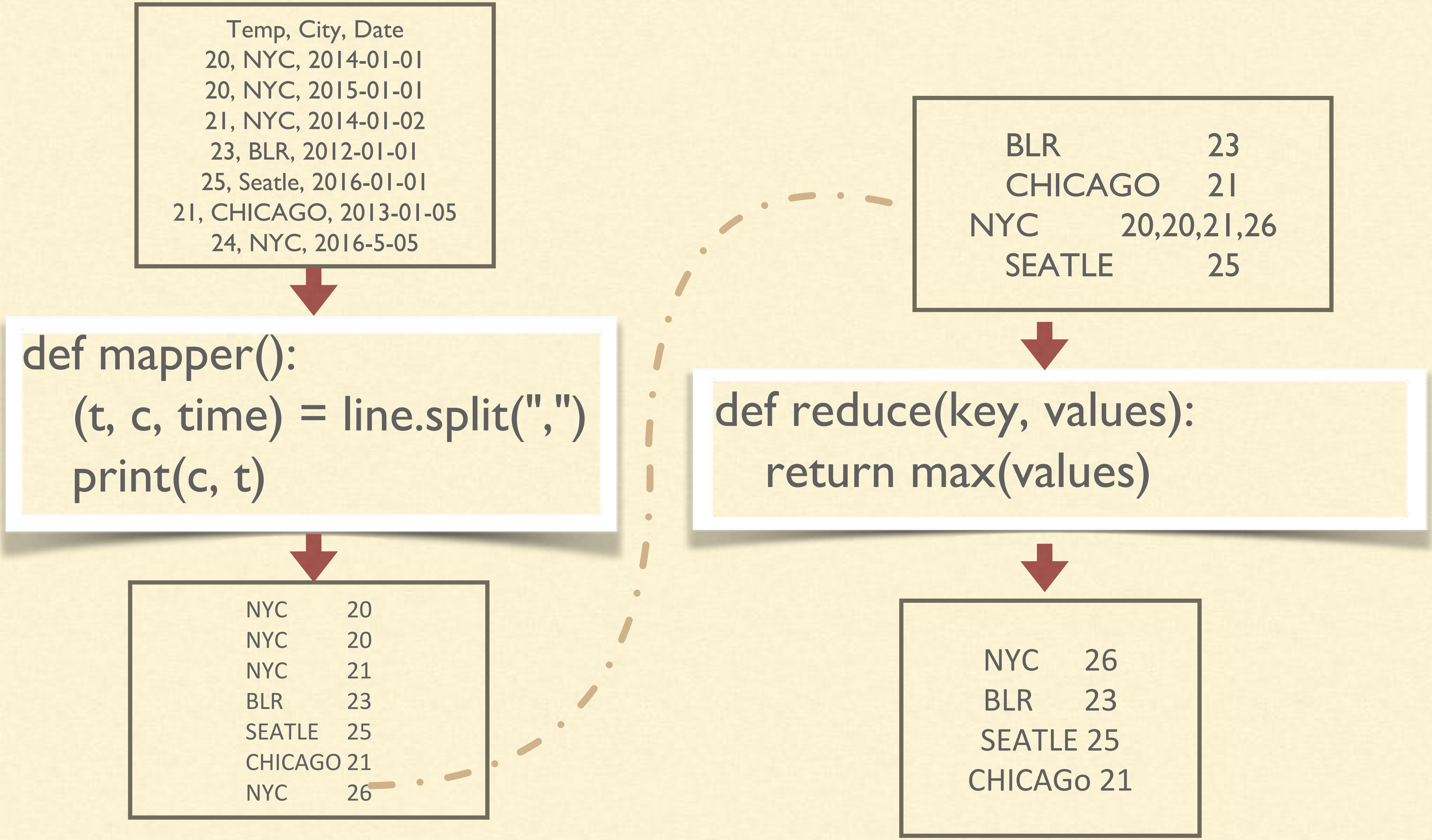
sa 1
re 1
re 1
sa 1
ga 1

ga [1]
re [1]
sa [1, 1]

```
function reducer(word, freqArray):
    return Array.sum(freqArray);
```

ga 1
re 1
sa 2

Mapper/Reducer for computing max temp



```
Temp, City, Date
20, NYC, 2014-01-01
20, NYC, 2015-01-01
21, NYC, 2014-01-02
23, BLR, 2012-01-01
25, Seatle, 2016-01-01
21, CHICAGO, 2013-01-05
24, NYC, 2016-5-05
```

```
def mapper():
    (t, c, time) = line.split(",")
    print(c, t)
```

```
NYC       20
NYC       20
NYC       21
BLR       23
SEATLE    25
CHICAGO 21
NYC       26
```

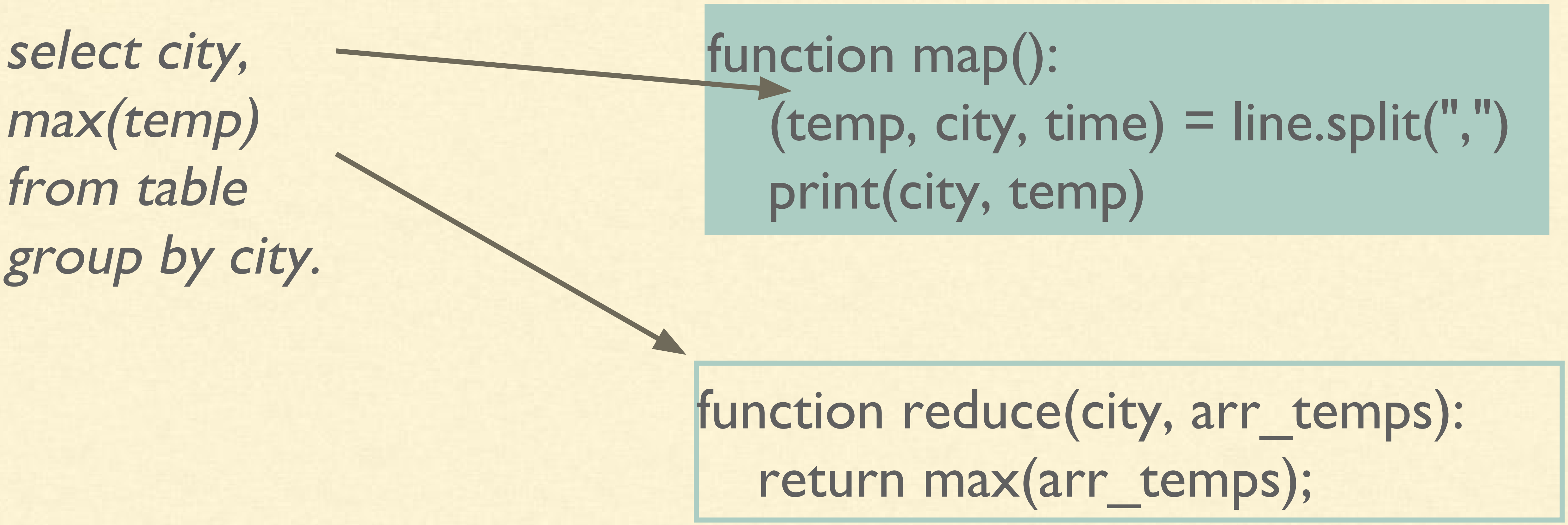
```
BLR          23
CHICAGO      21
NYC      20,20,21,26
SEATLE       25
```

```
def reduce(key, values):
    return max(values)
```

```
NYC     26
BLR     23
SEATLE 25
CHICAGo 21
```

MAP / REDUCE

Analogous to Group By



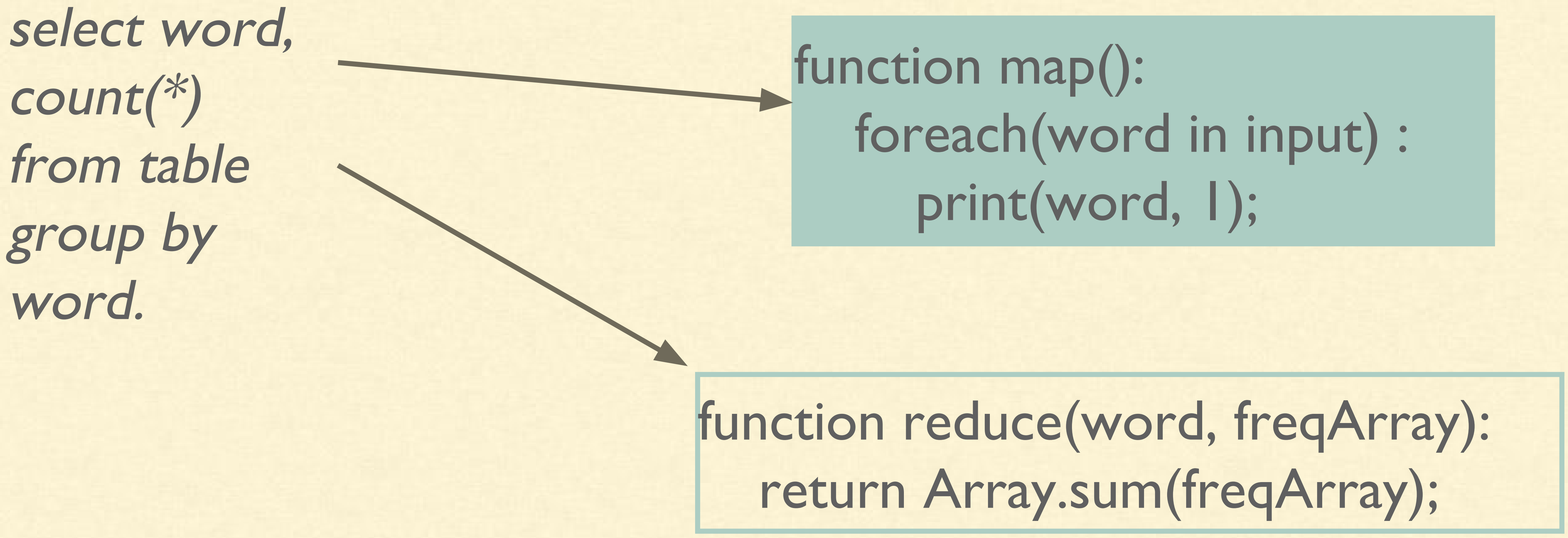
select city,
max(temp)
from table
group by city.

```
function map():
    (temp, city, time) = line.split(",")
    print(city, temp)
```

```
function reduce(city, arr_temps):
    return max(arr_temps);
```

MAP / REDUCE

Analogous to Group By

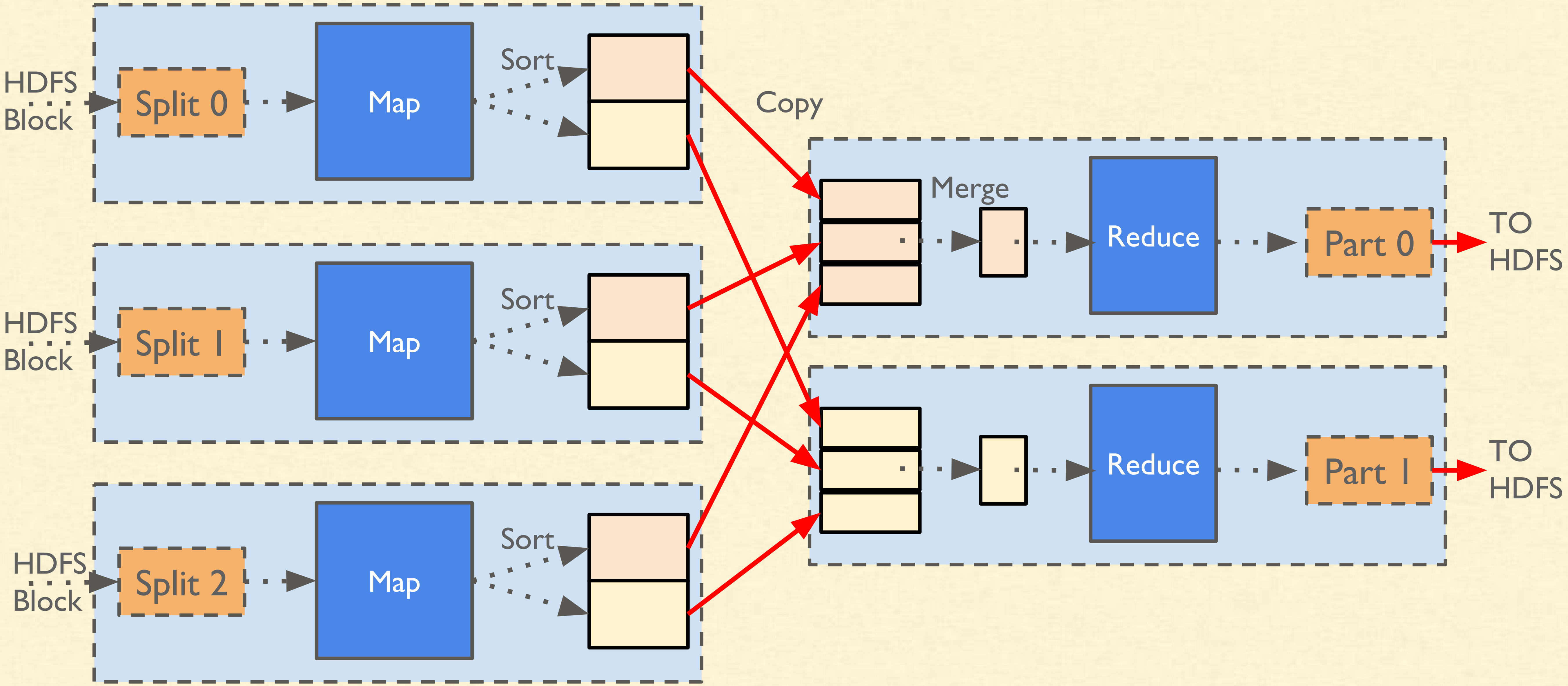


select word,
count(*)
from table
group by
word.

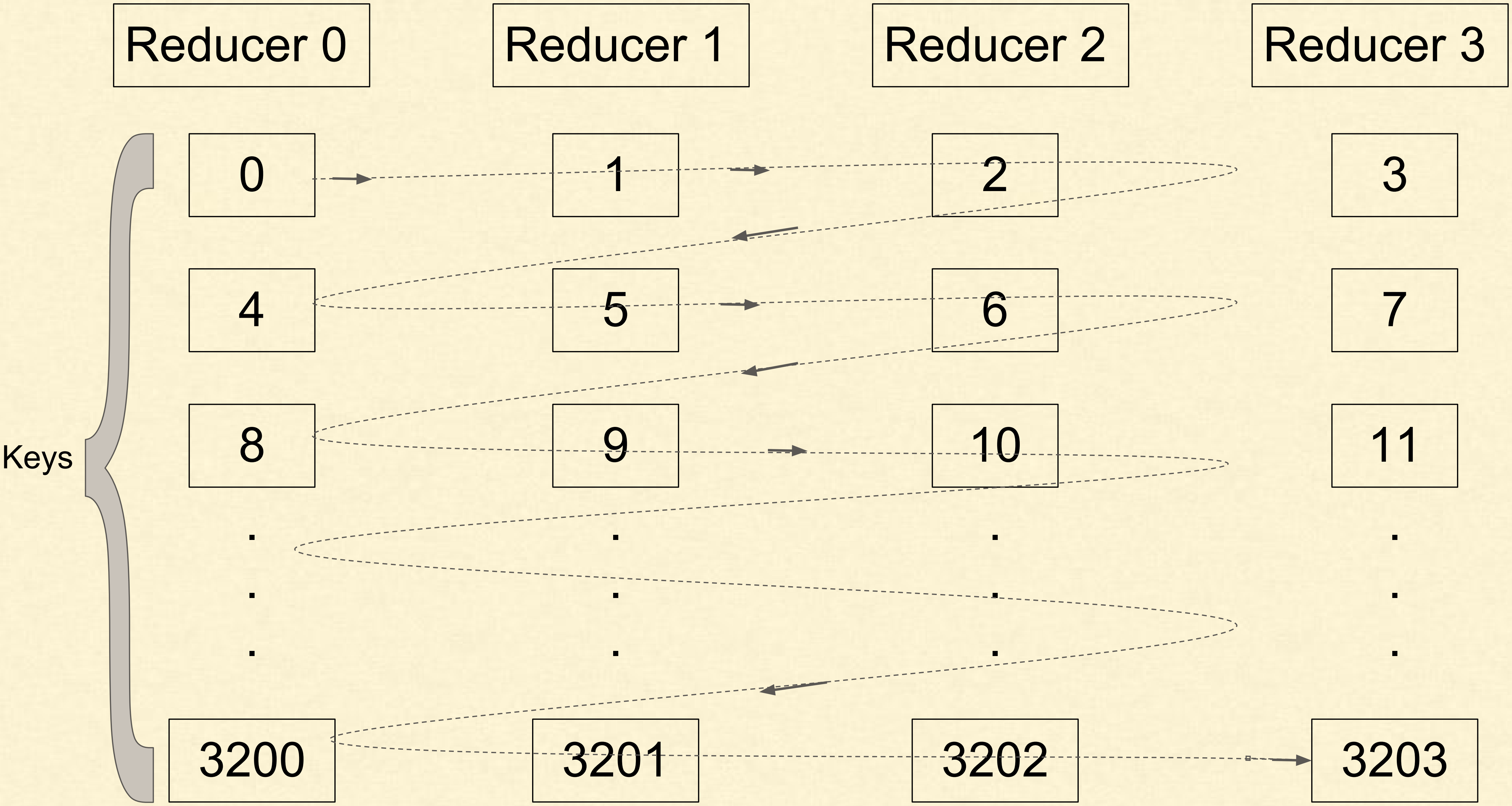
```
function map():
   foreach(word in input) :
      print(word, 1);
```

```
function reduce(word, freqArray):
   return Array.sum(freqArray);
```

MAP REDUCE - Multiple Reducers

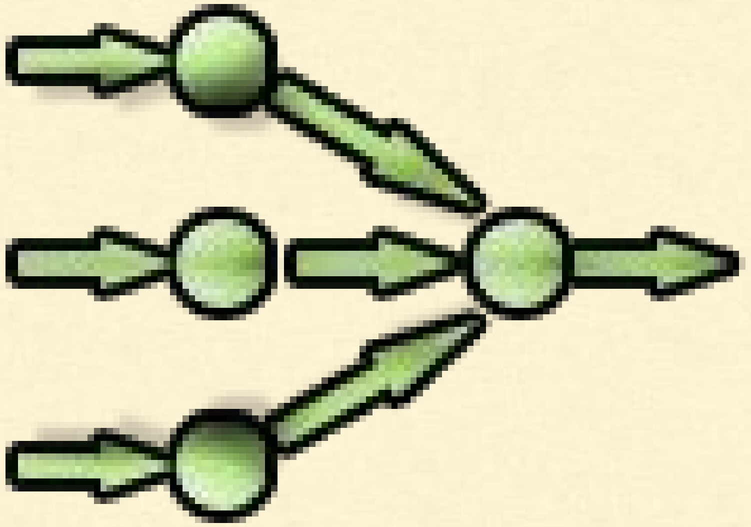


MAP REDUCE - Paritioning



Reducer 0	Reducer 1	Reducer 2	Reducer 3
0	1	2	3
4	5	6	7
8	9	10	11
.	.	.	.
.	.	.	.
.	.	.	.
3200	3201	3202	3203

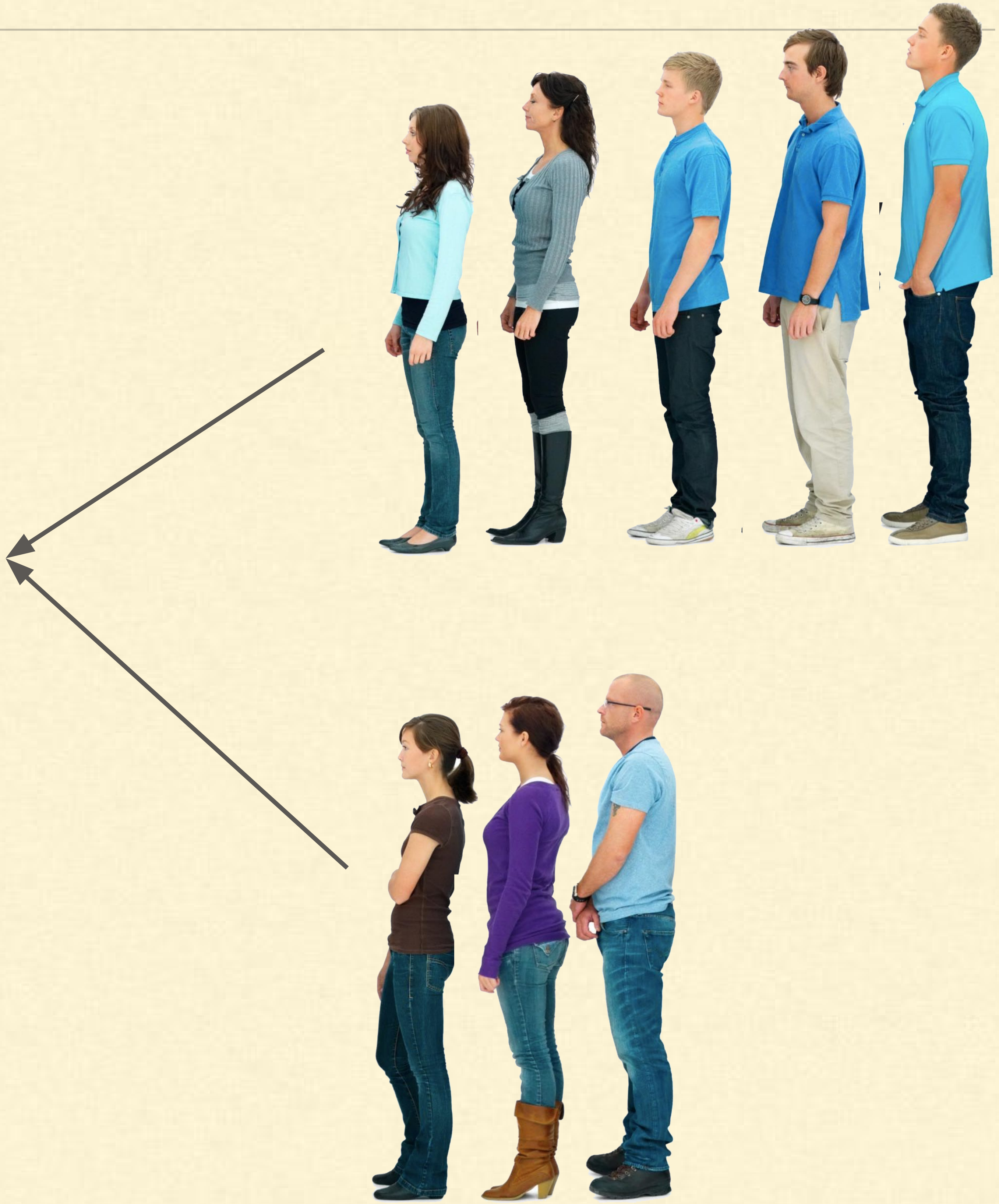
Key k will go to this reducer: hashcode(k) % total_reducers



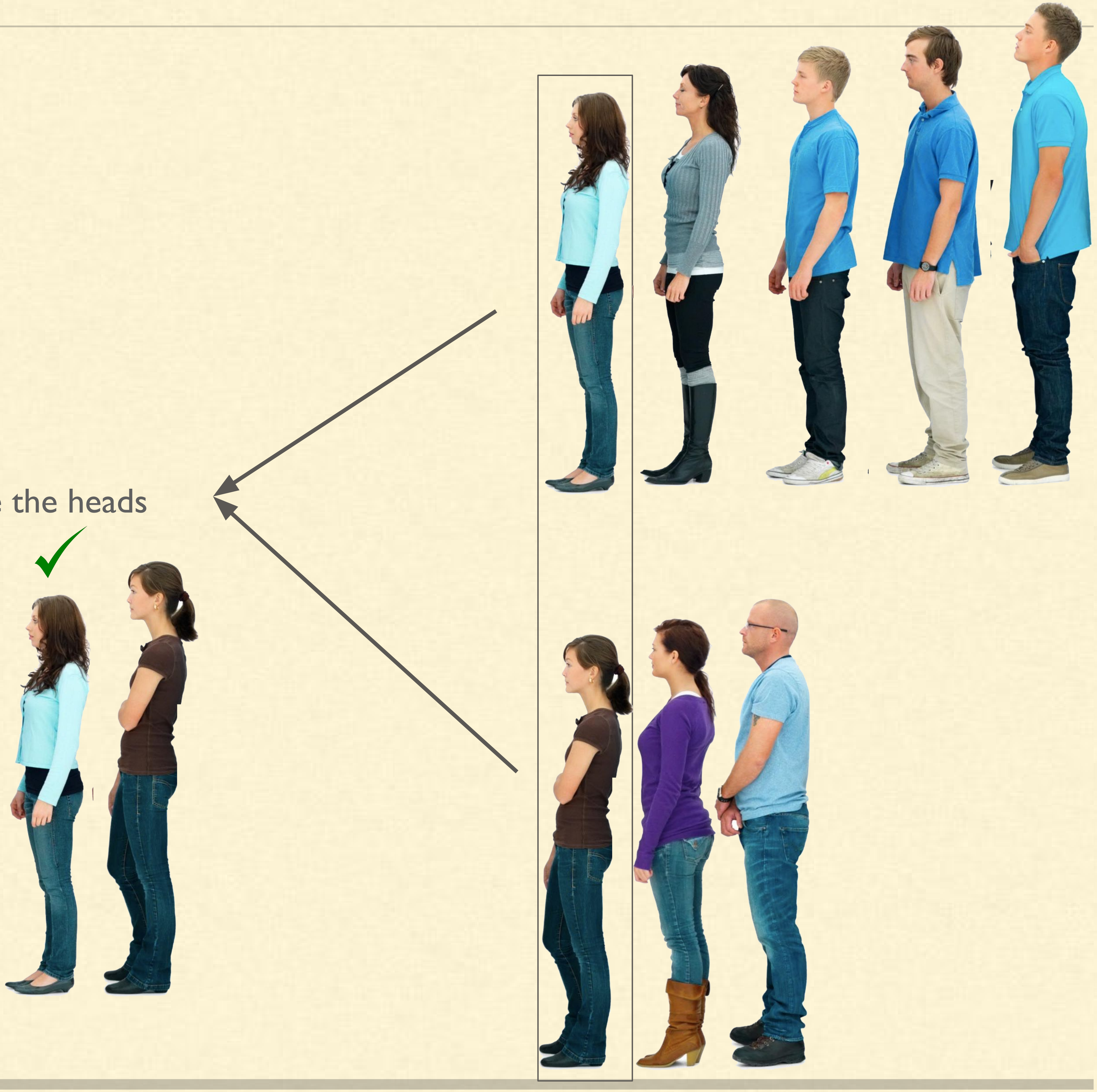
Thank you

Merging

Merge the two sorted queues to form another sorted queue



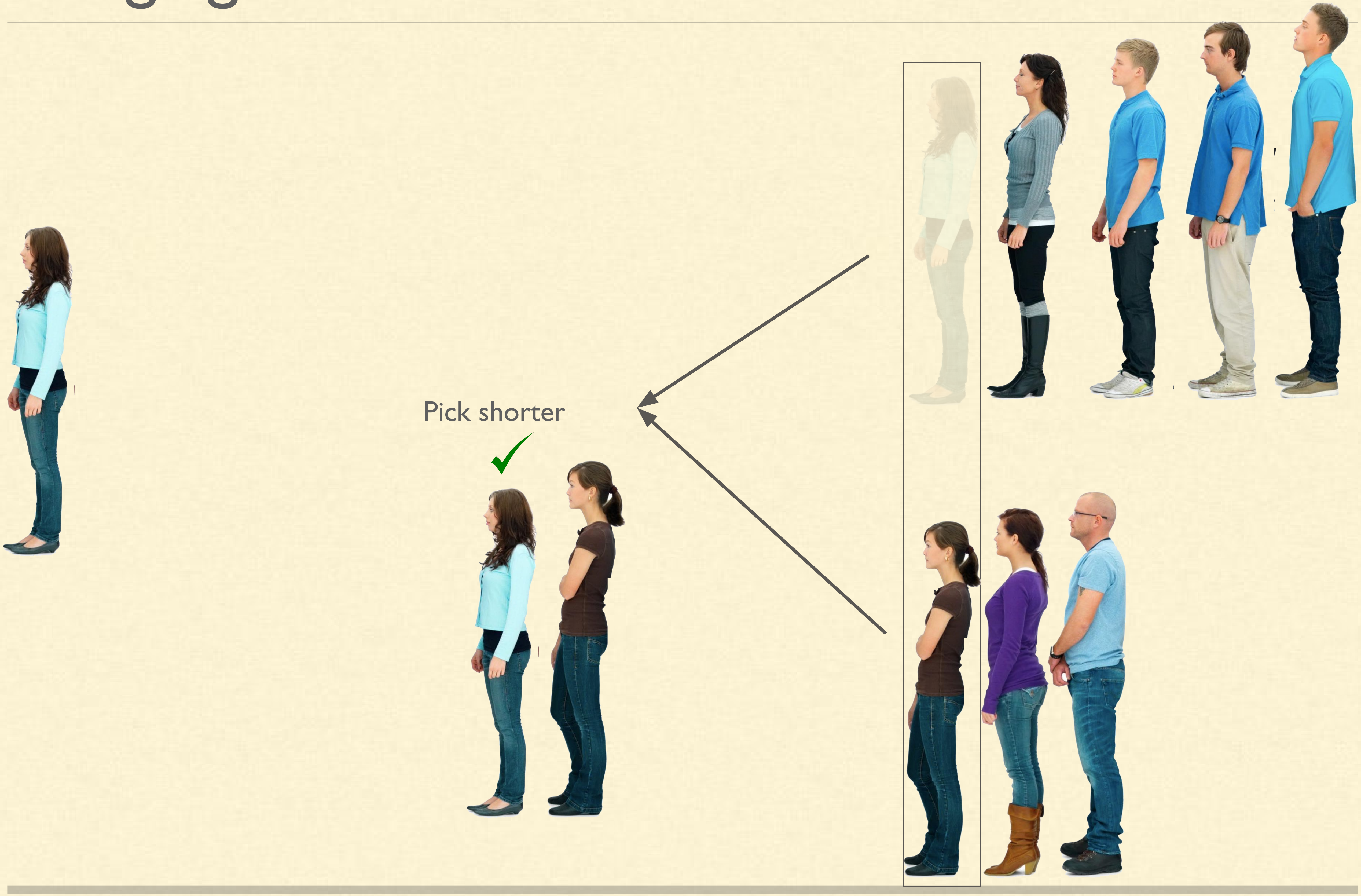
Merging



Compare the heads

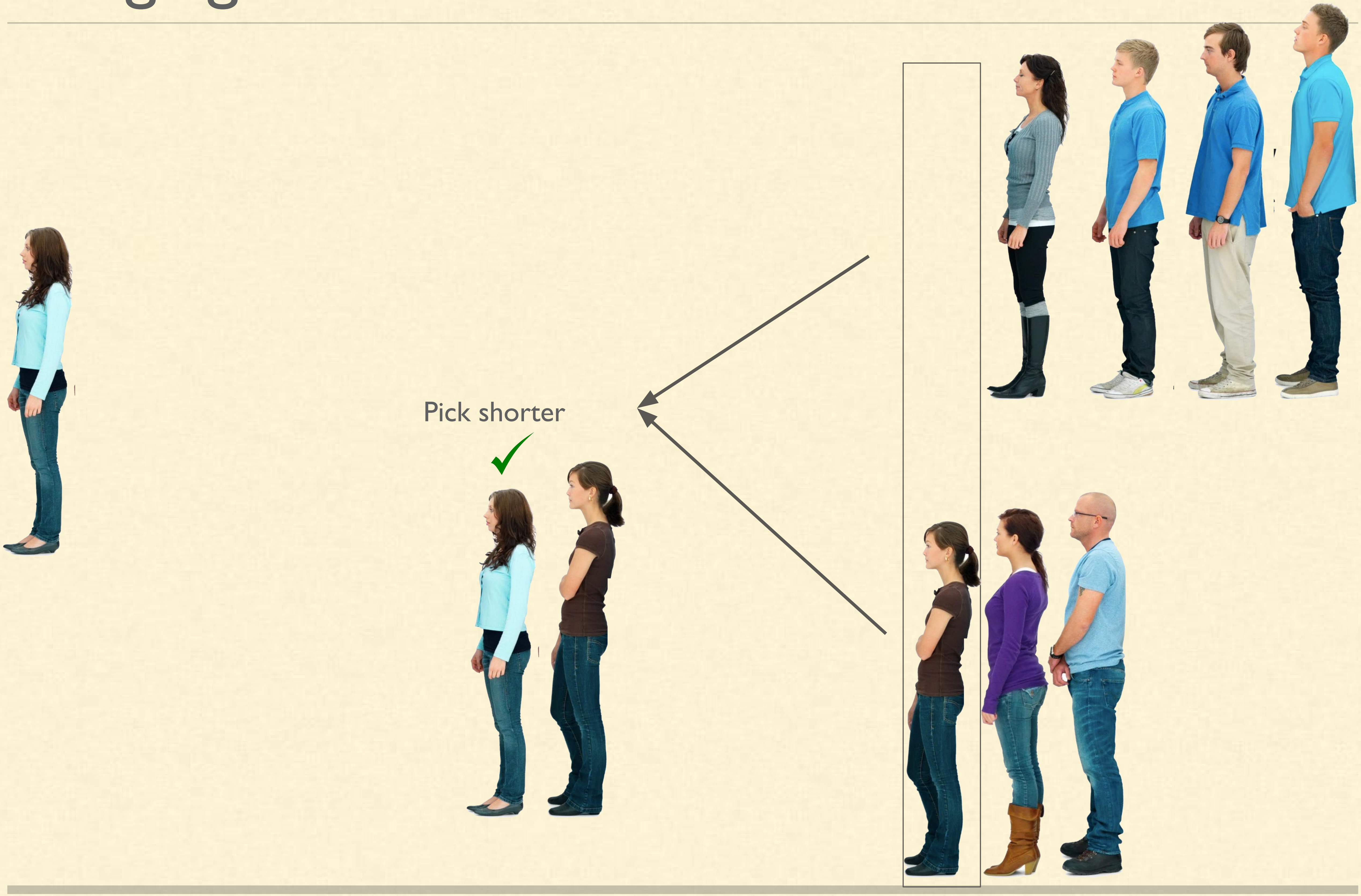
✓

Merging

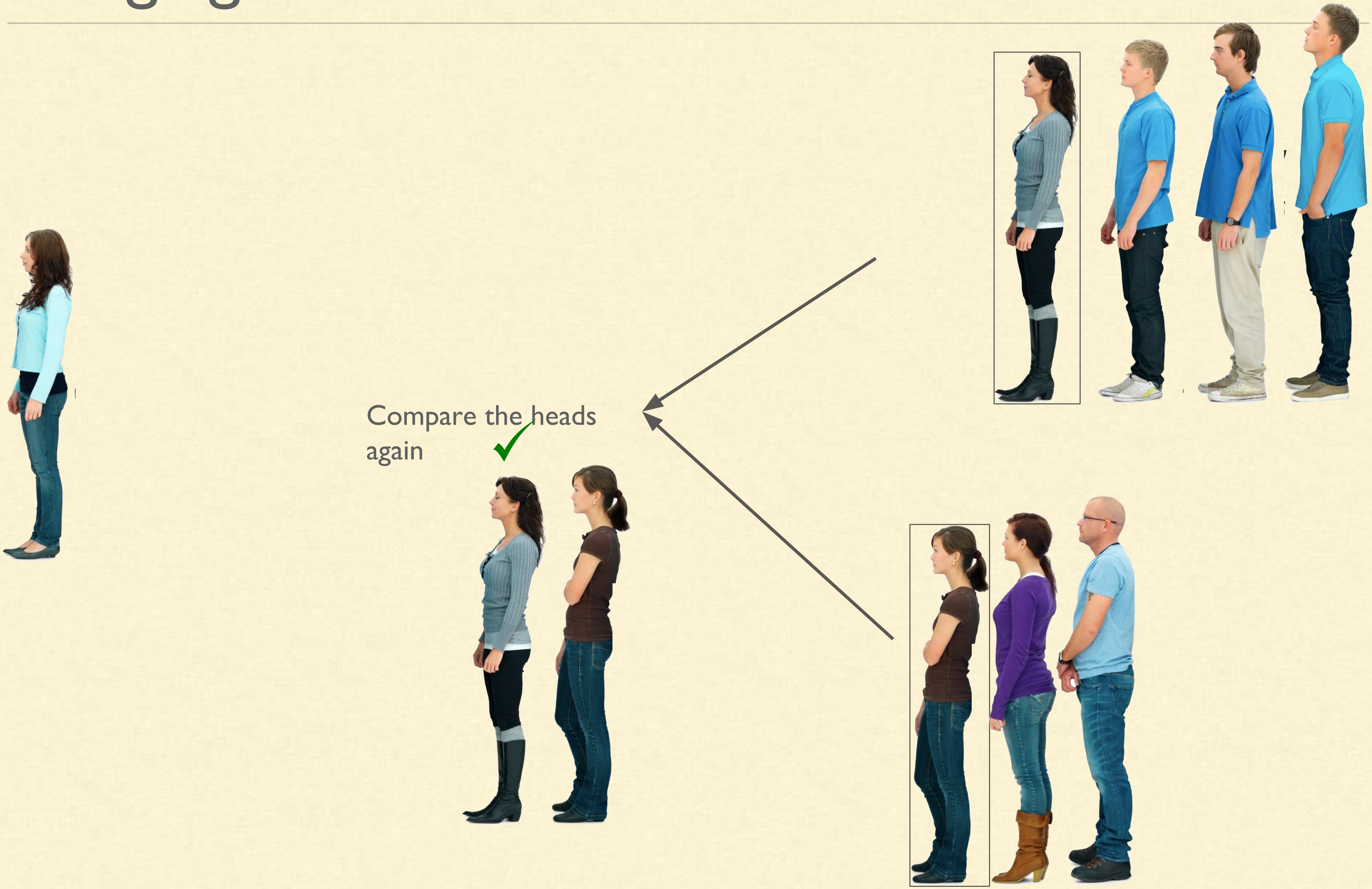


Pick shorter ✓

Merging

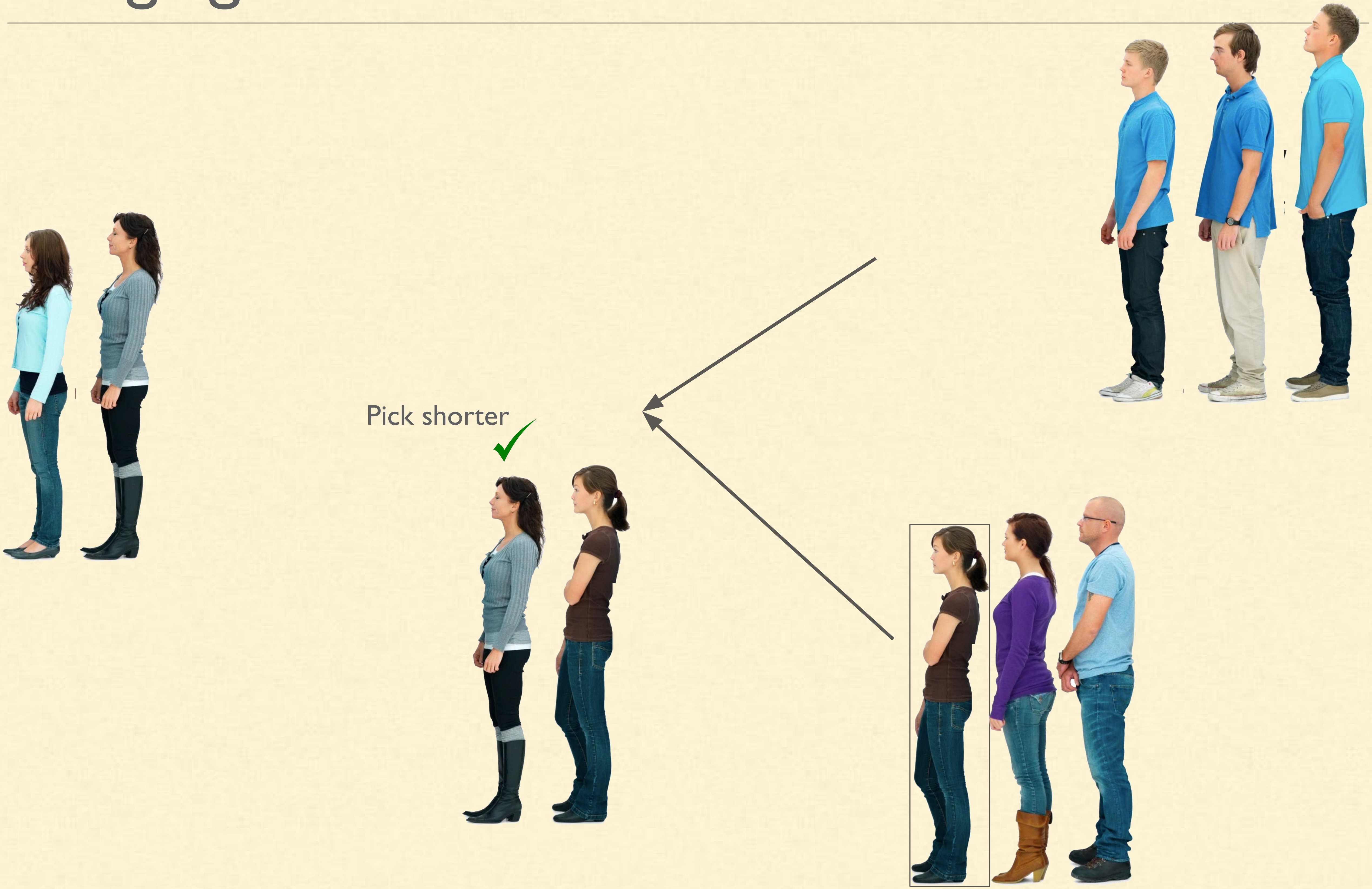


Merging



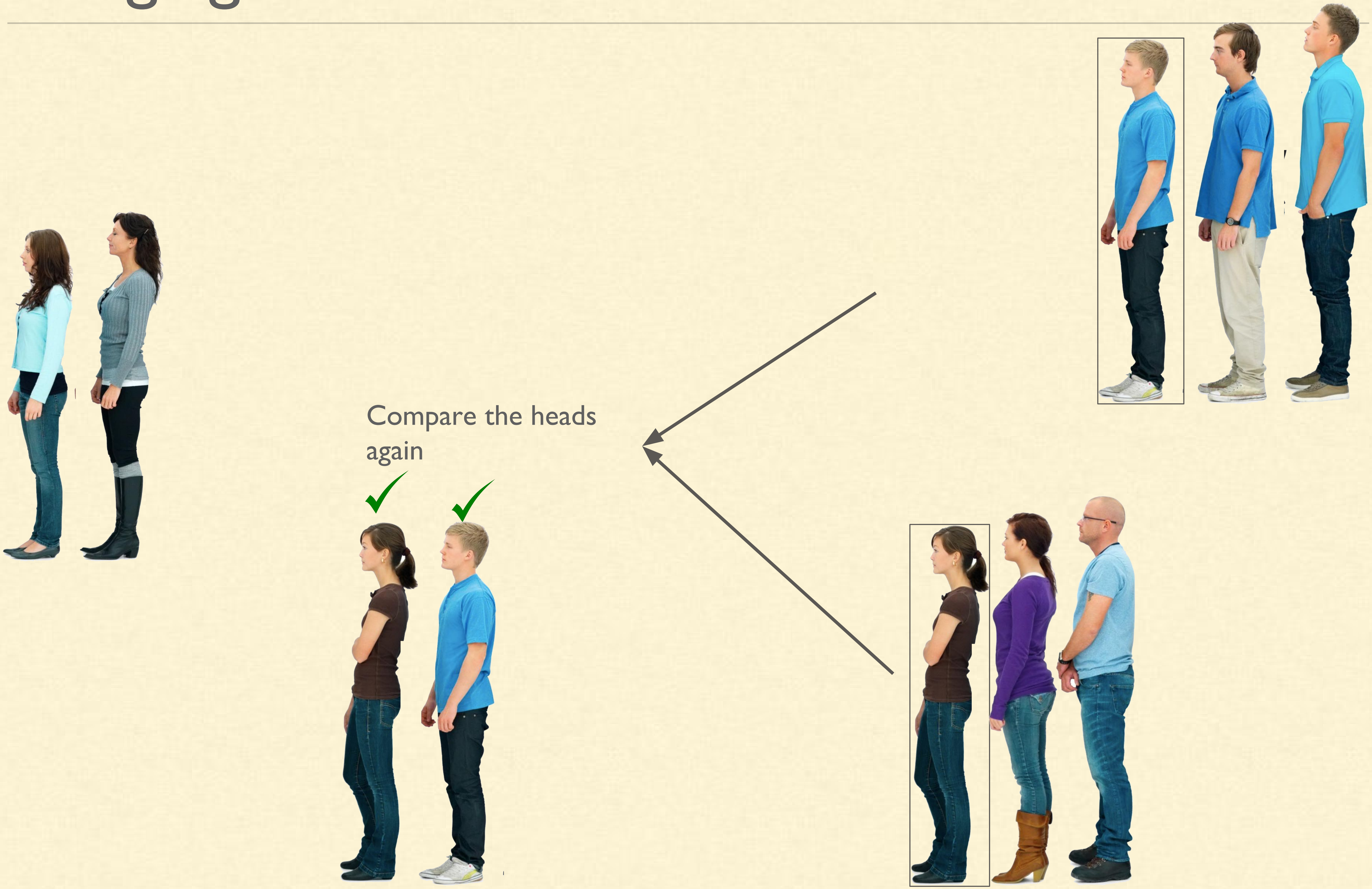
Compare the heads again ✓

Merging



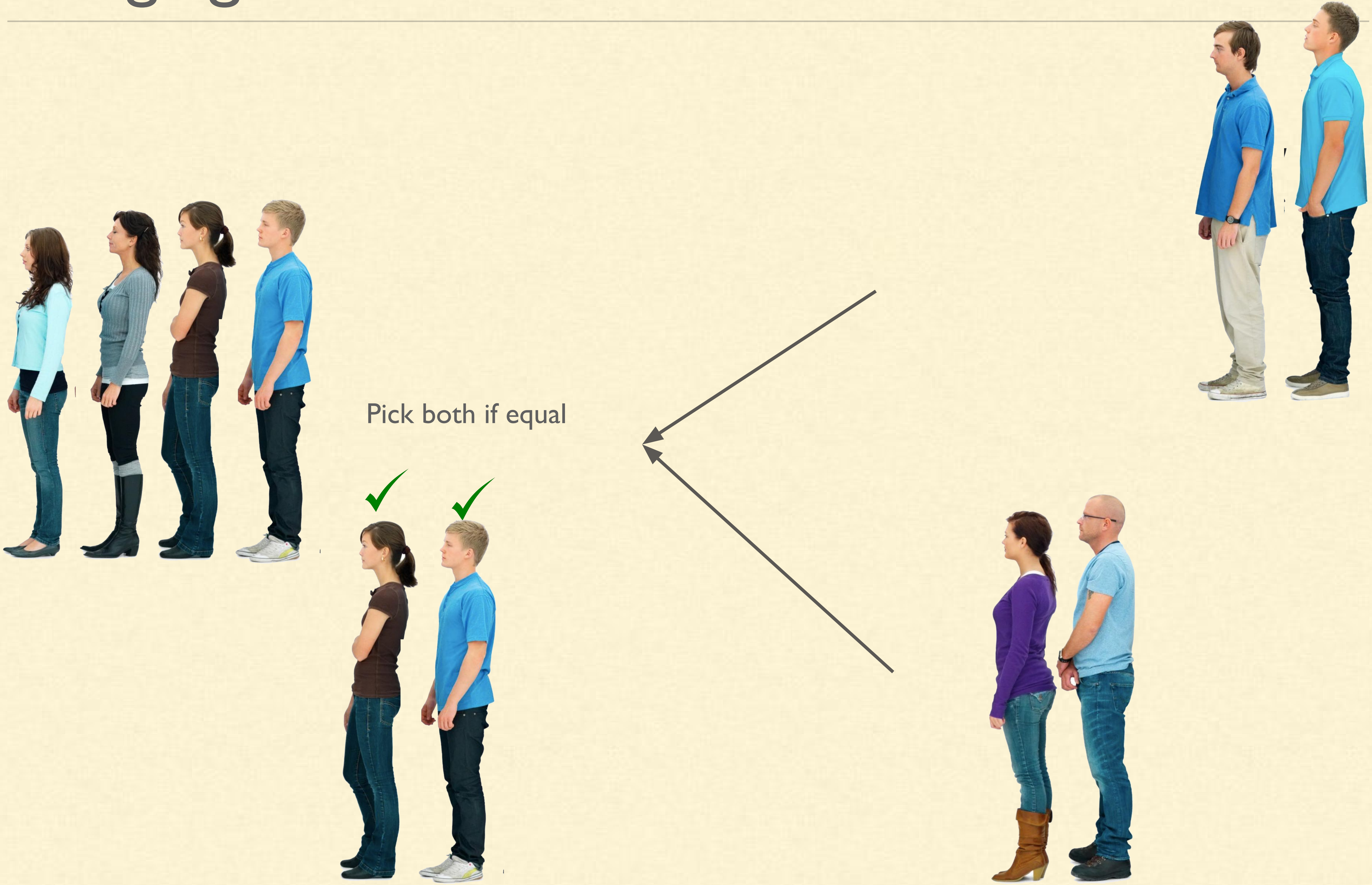
Pick shorter ✓

Merging



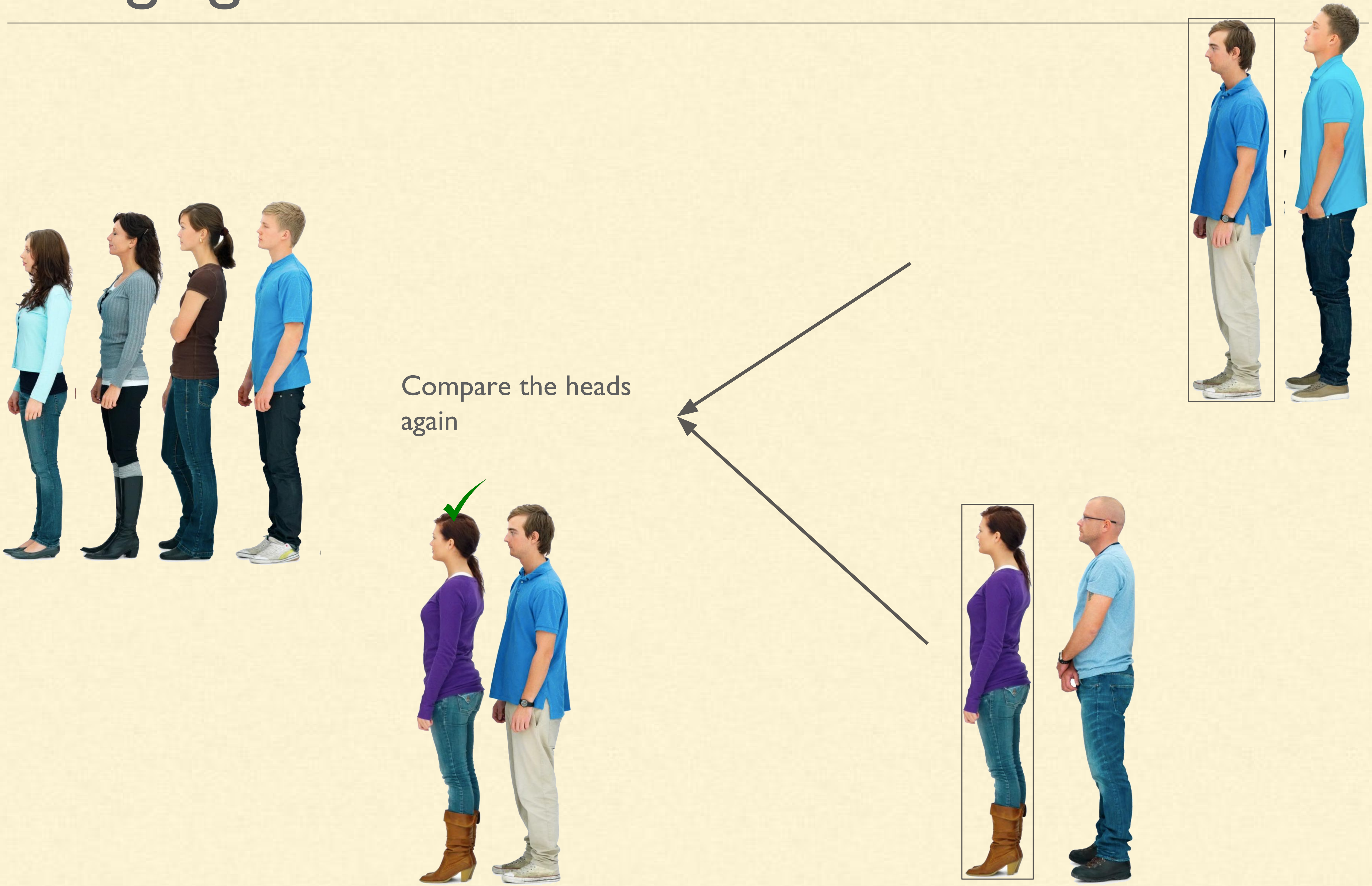
Compare the heads again

Merging



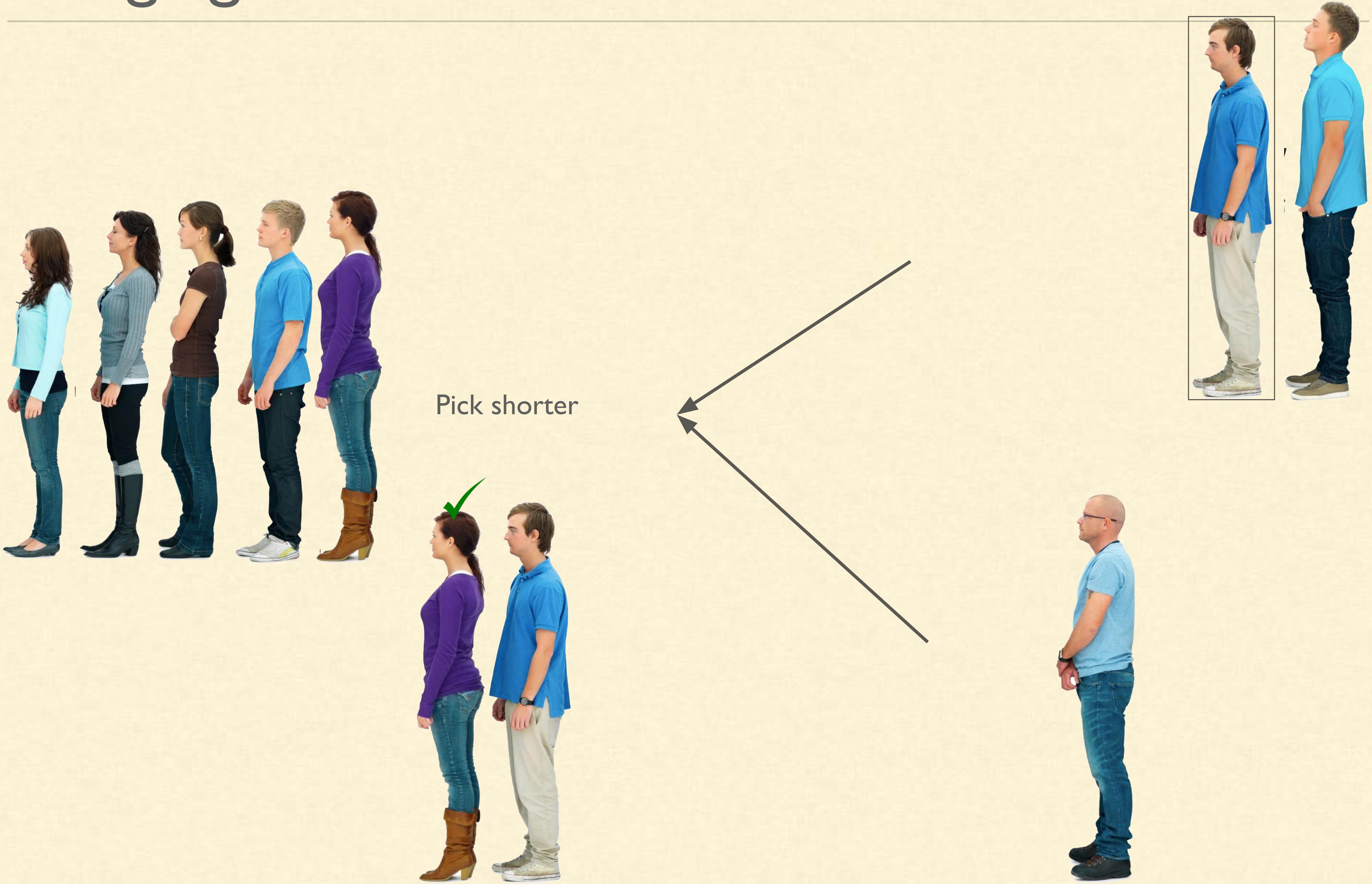
Pick both if equal

Merging



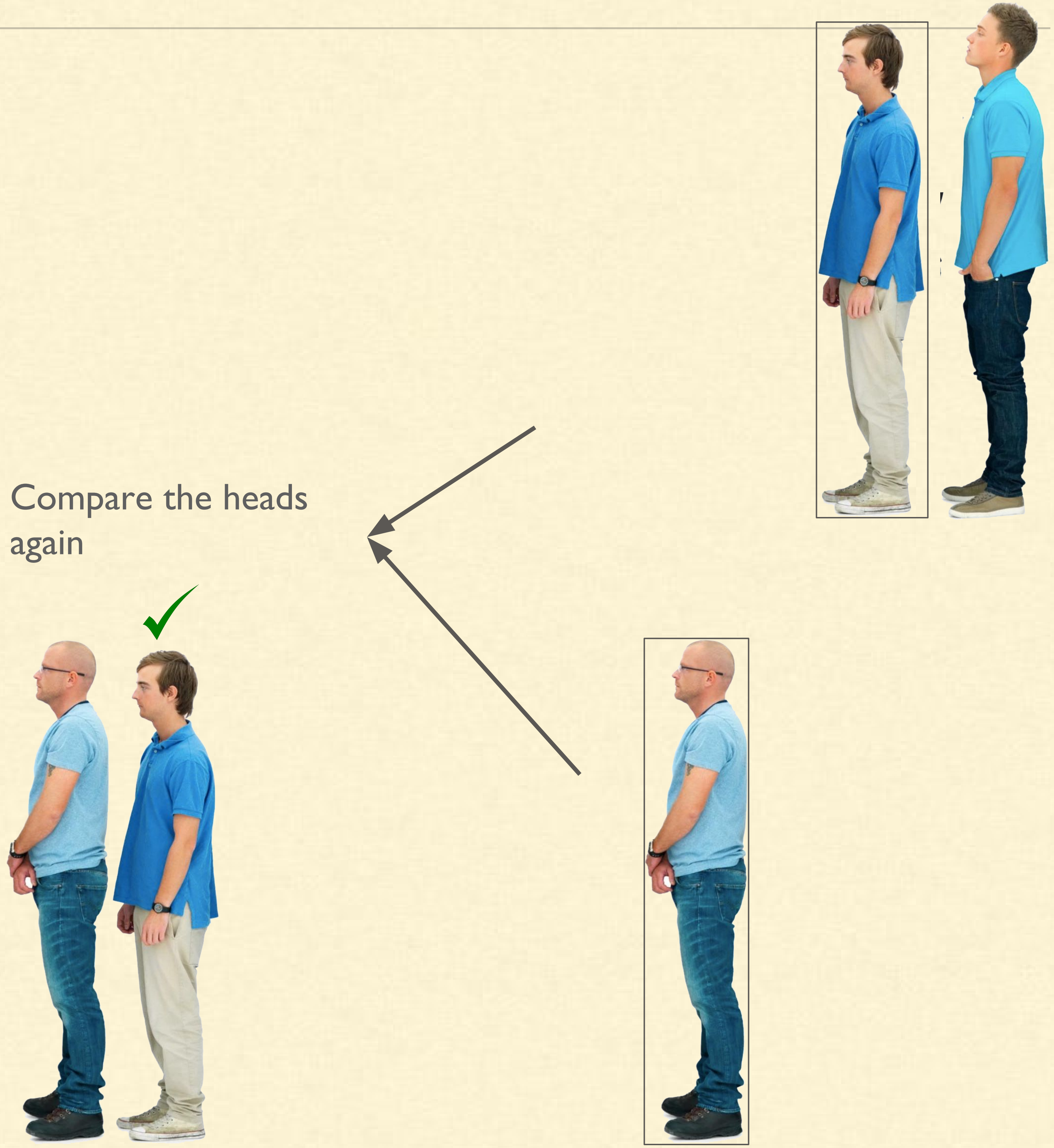
Compare the heads again

Merging



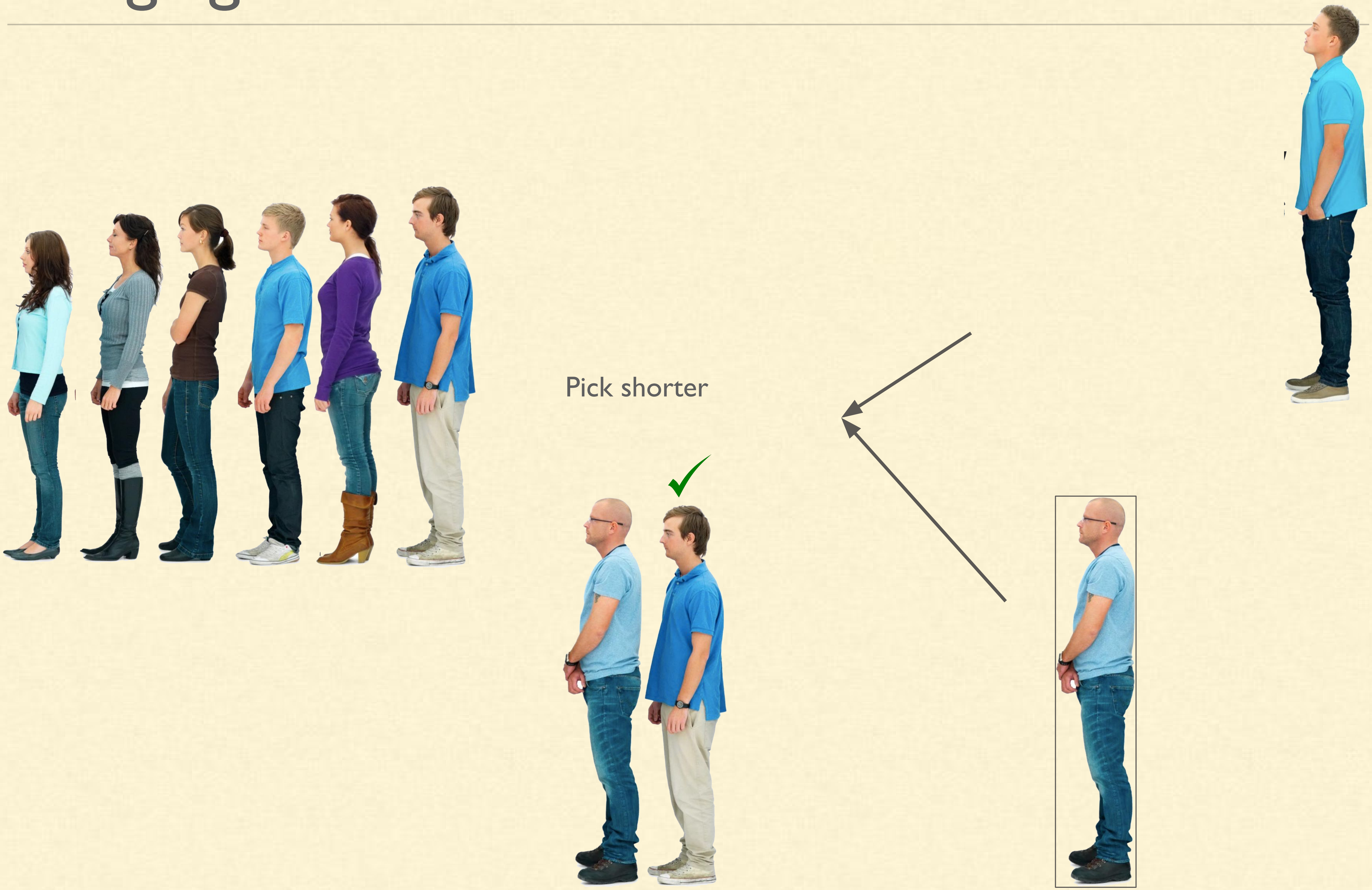
Pick shorter

Merging



Compare the heads again

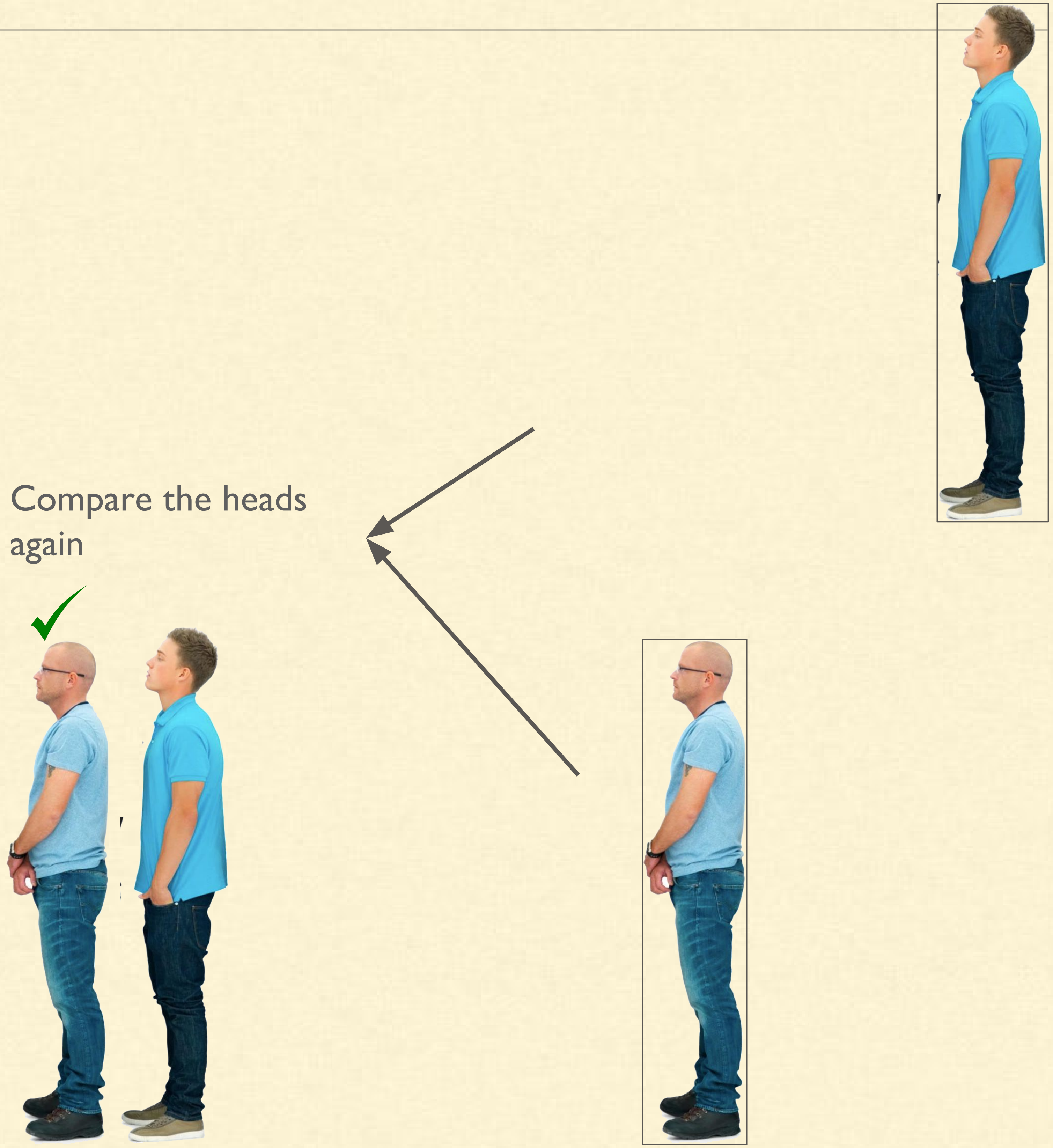
Merging



Pick shorter

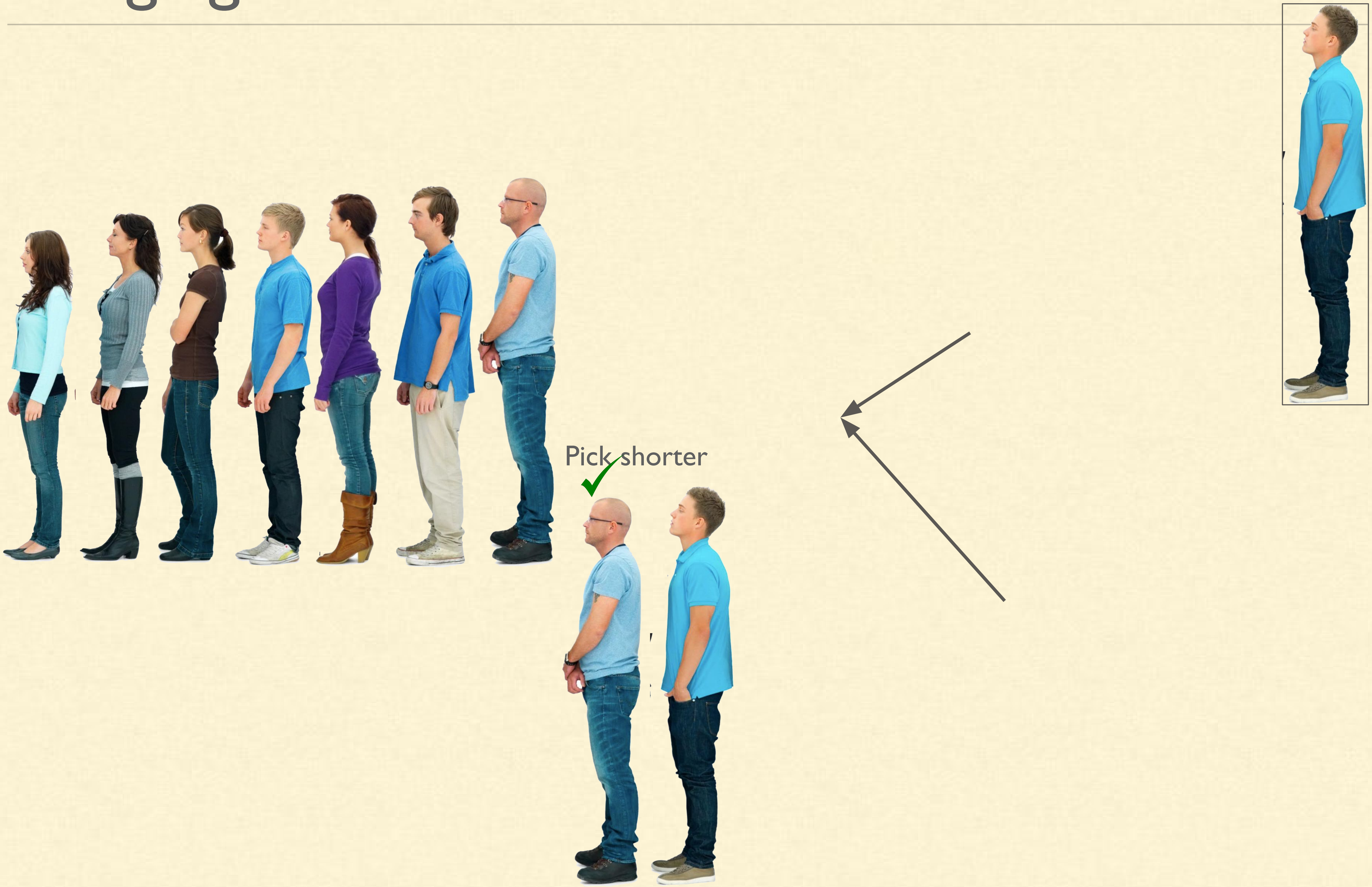
✓

Merging



Compare the heads again

Merging

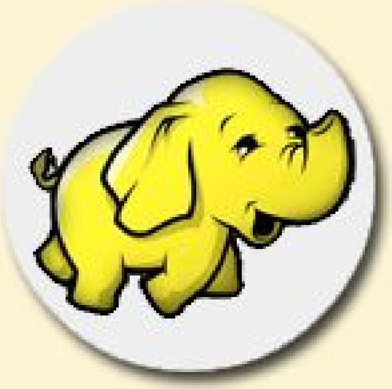


Merging

Since no one is left on second queue.
Put remaining from first

Merging

This merges the two queues into one



Hadoop & Spark

Thank you.

+1 419 665 3276 (US)
+91 803 959 1464 (IN)

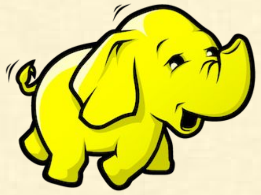
support@knowbigdata.com

Subscribe to our Youtube channel for latest videos - https://www.youtube.com/channel/UCxugRFe5wETYA7nMH6VGyEA

MAP / REDUCE - RECAP

MAP / REDUCE

The data generated by the mapper is given to reducer and then it is sorted / shuffled [Yes/No]?



MAP / REDUCE

The data generated by the mapper is given to reducer and then it is sorted / shuffled [Yes/No]?

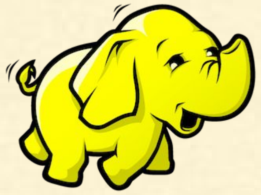
No. The output of mapper is first shuffled/sorted and then given to reducers.

MAP / REDUCE

The mapper can only generate a single key value pair for an input value [True/False]?

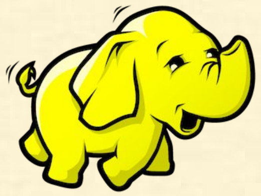
MAP / REDUCE

The mapper can only generate a single key value pair for an input value [True/False]?



False. Mapper can generate as many key-value pair as it wants for an input.

MAP / REDUCE



A mapper always have to generate at least a key-value pair[Correct/Wrong]?

MAP / REDUCE

A mapper always generates at least a key-value pair[Correct/Wrong]?

Wrong

MAP / REDUCE

By default there is only one reducer in case of streaming job [Yes/No]?

MAP / REDUCE

By default there is only one reducer in case of streaming job [Yes/No]?

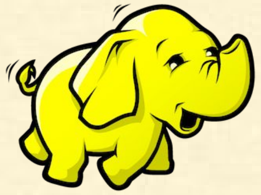
Yes. By default there is a single reducer job but it can be split by specifying cmd option : mapred.reduce.tasks.

MAP / REDUCE

In hadoop 1.0, What is the role of job tracker?
A: Executing the Map/Reduce Logic
B: Delegate the Map/Reduce Logic to task tracker.

MAP / REDUCE

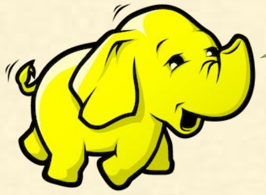
What is the role of job tracker?
A: Executing the Map/Reduce Logic
B: Delegate the Map/Reduce Logic to task tracker.



B.

MAP / REDUCE

Q: The Map logic is executed preferably on the nodes that have the required data [Yes/No]?



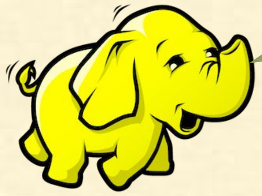
MAP / REDUCE

Q: The Map logic is executed preferably on the nodes that have the required data [Yes/No]?

Yes.

MAP / REDUCE

Q: The Map logic is *always* executed on the nodes that have the required data [Correct/Wrong]?



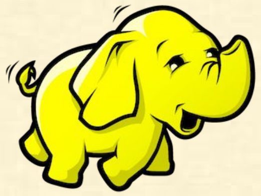
MAP / REDUCE

Q: The Map logic is *always* executed on the nodes that have the required data [Correct/Wrong]?

Wrong

MAP / REDUCE

Where does Hadoop Store the result of reducer?
In HDFS or Local File System?



MAP / REDUCE

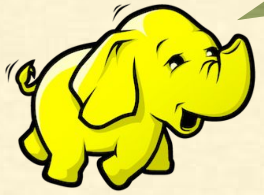
Where does Hadoop Store the result of reducer?
In HDFS or Local File System?

In HDFS.

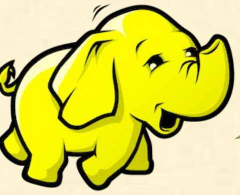
MAP / REDUCE

Where does Hadoop Store the intermediate data such as output of Map Tasks?

In HDFS or Local File System or Memory?



MAP / REDUCE



Where does Hadoop Store the intermediate data such as output of Map Tasks?
In HDFS or File System or Memory?

First in Memory and purged to Local File System.
Output of mapper is saved in HDFS directly only if there is no reduce phase.

Assignment For Tomorrow

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

2. Find anagrams in a huge text. An anagram is basically a different arrangement of letters in a word. Anagram does not need have a meaning.

Input:

"the cat act in tic tac toe"

Output:

cat, tac, act
the
toe
in
tic

MAP / REDUCE

Assignment For Tomorrow

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

Assignment For Tomorrow

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 ACCT"

Output:

User1, User2, User4
User3, User 5
User6

MAP / REDUCE

Assignment For Tomorrow

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

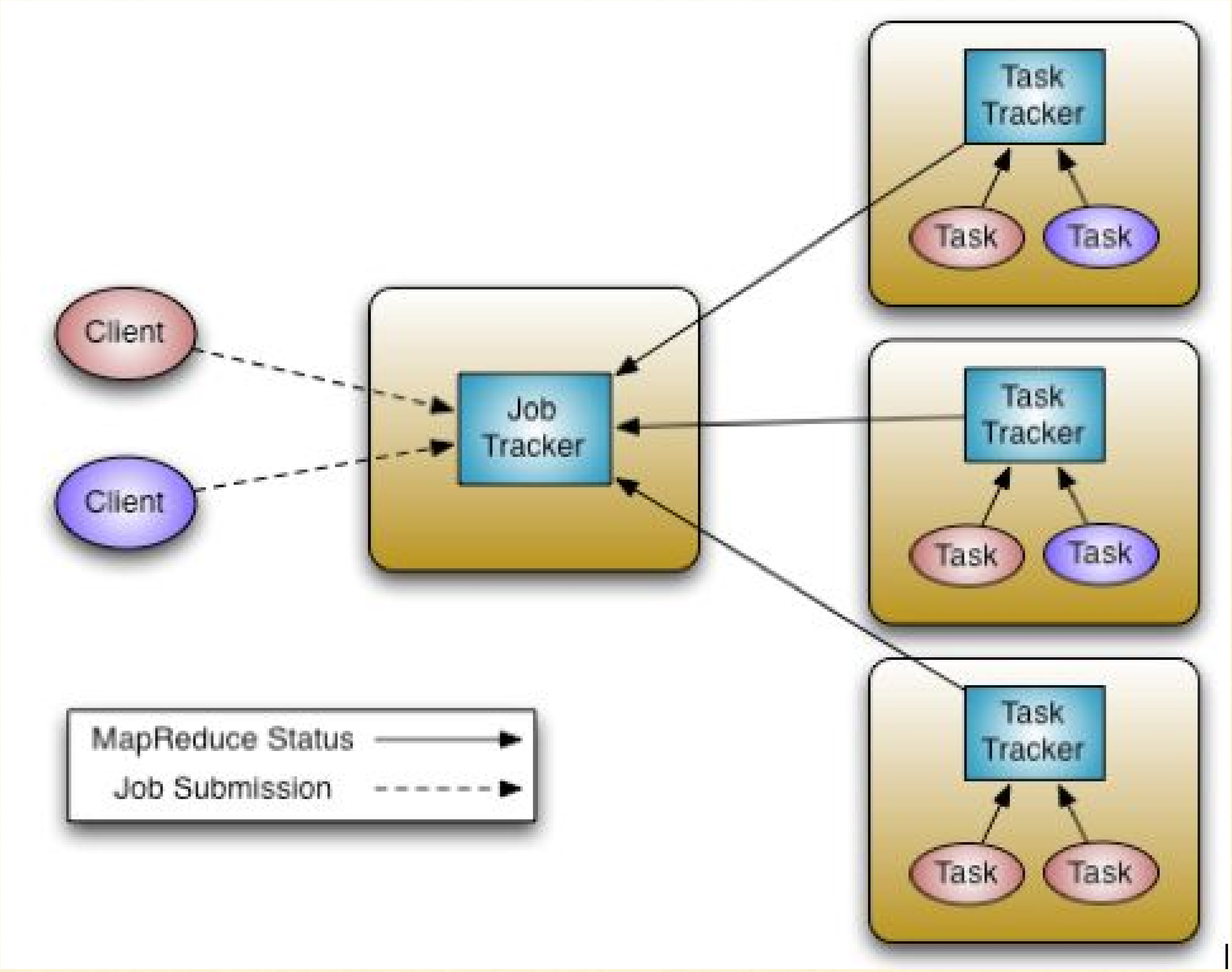
List2

Person	*Worth*
A	5
B	1
C	11

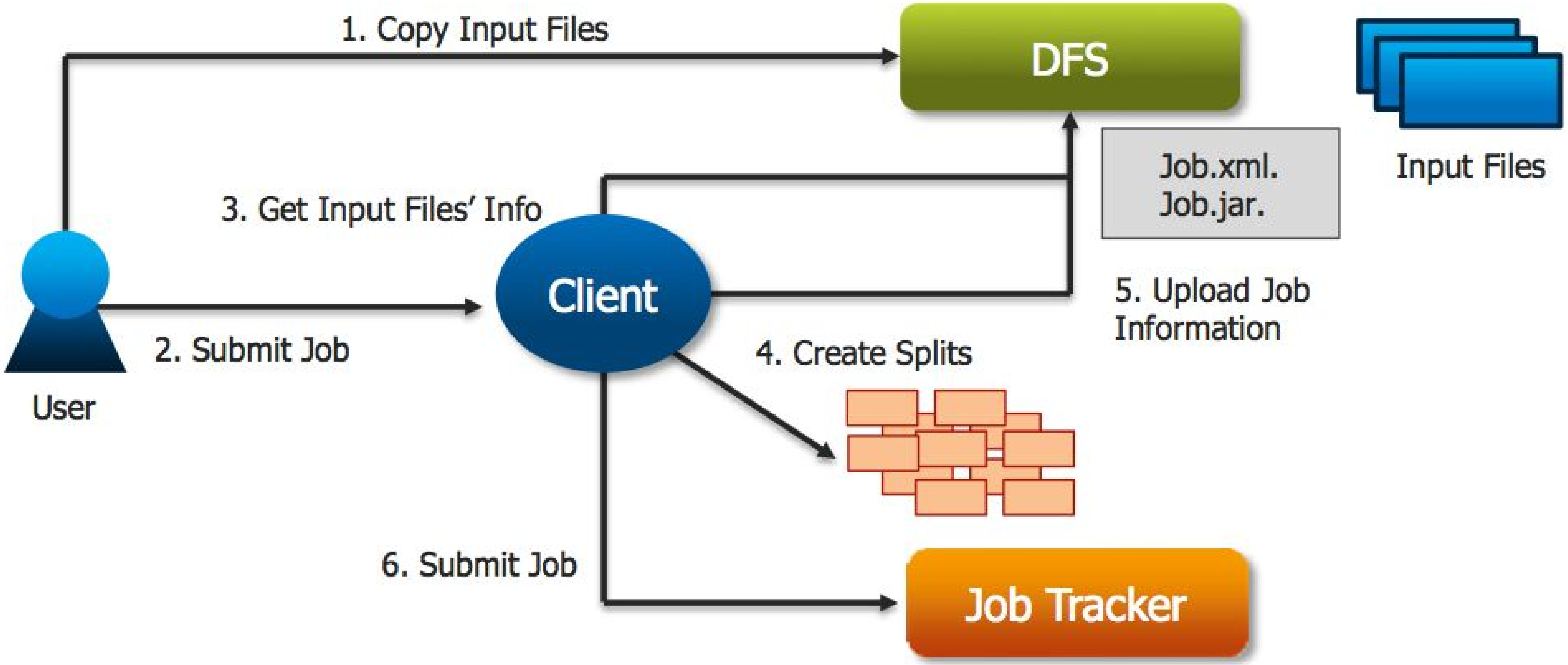
Result

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

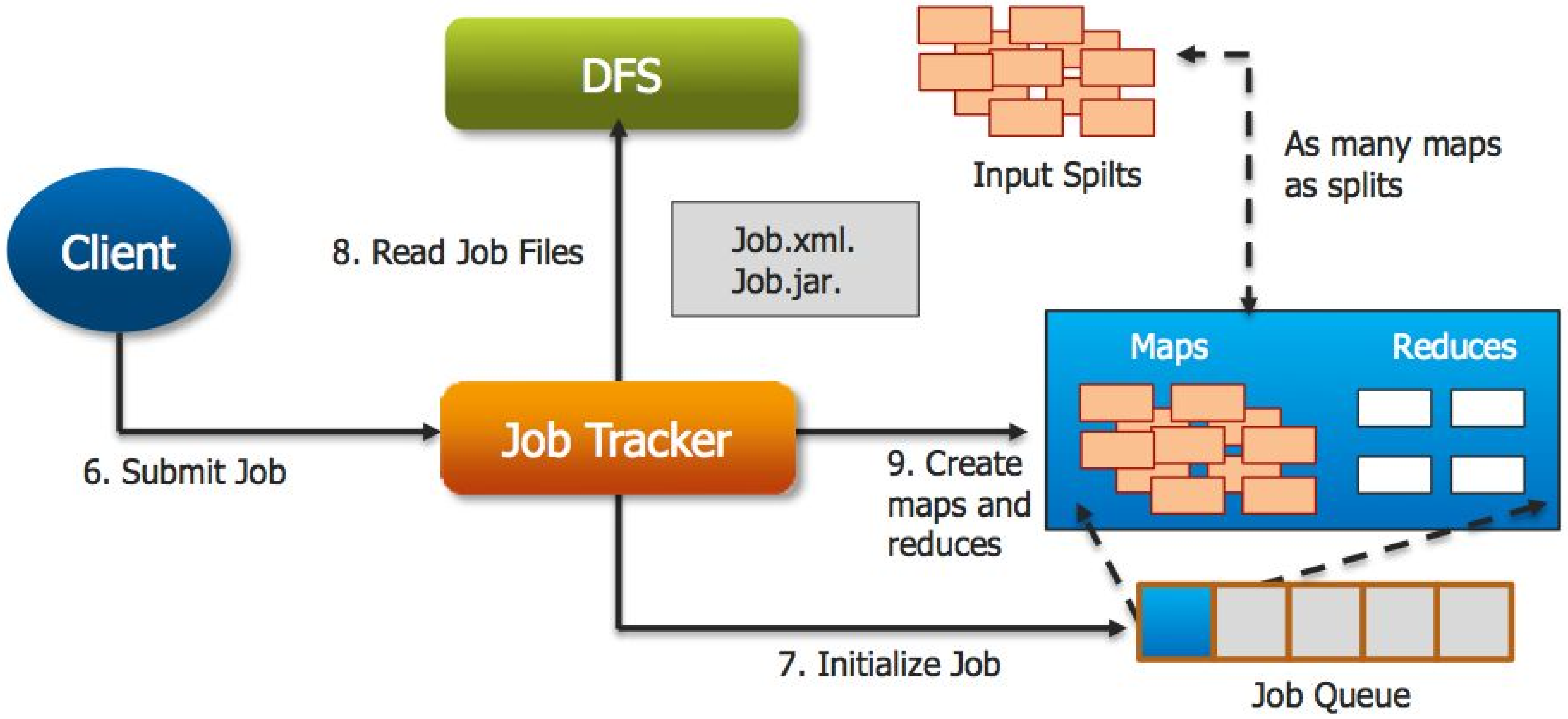
JOB TRACKER


Task Tracker
Task
Task
Client
Job Tracker
Client
Task Tracker
Task
Task
Task Tracker
Task
Task
MapReduce Status
Job Submission

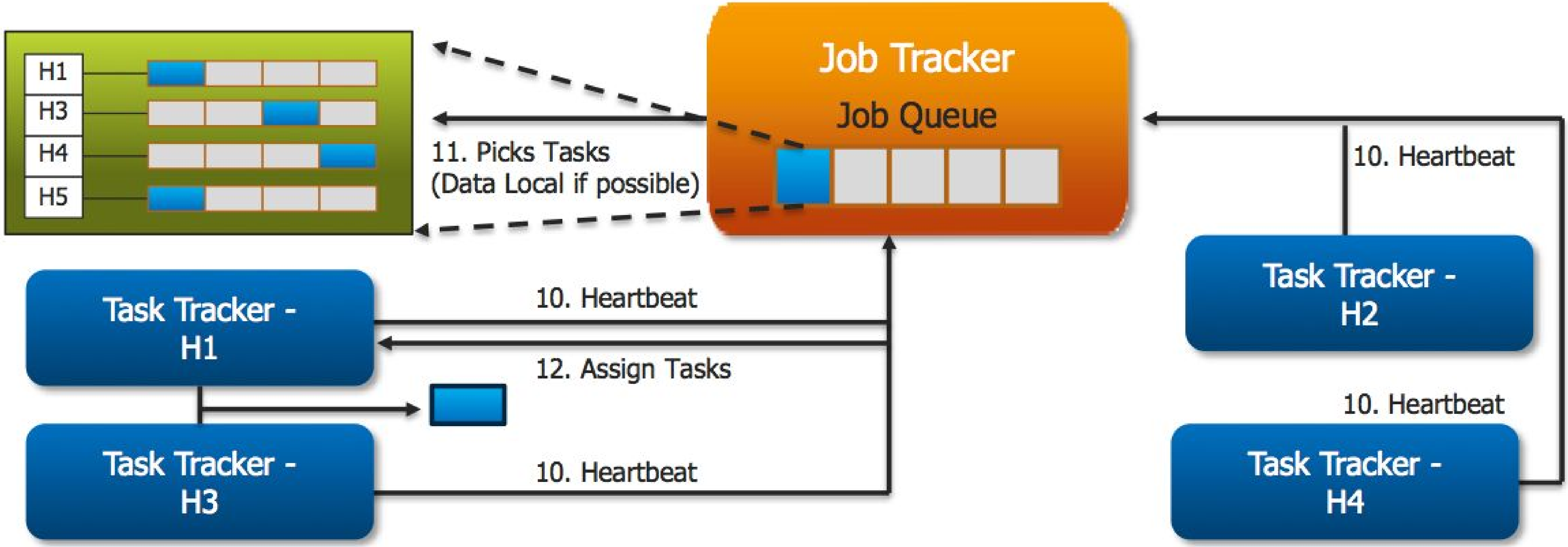
JOB TRACKER (DETAILED)

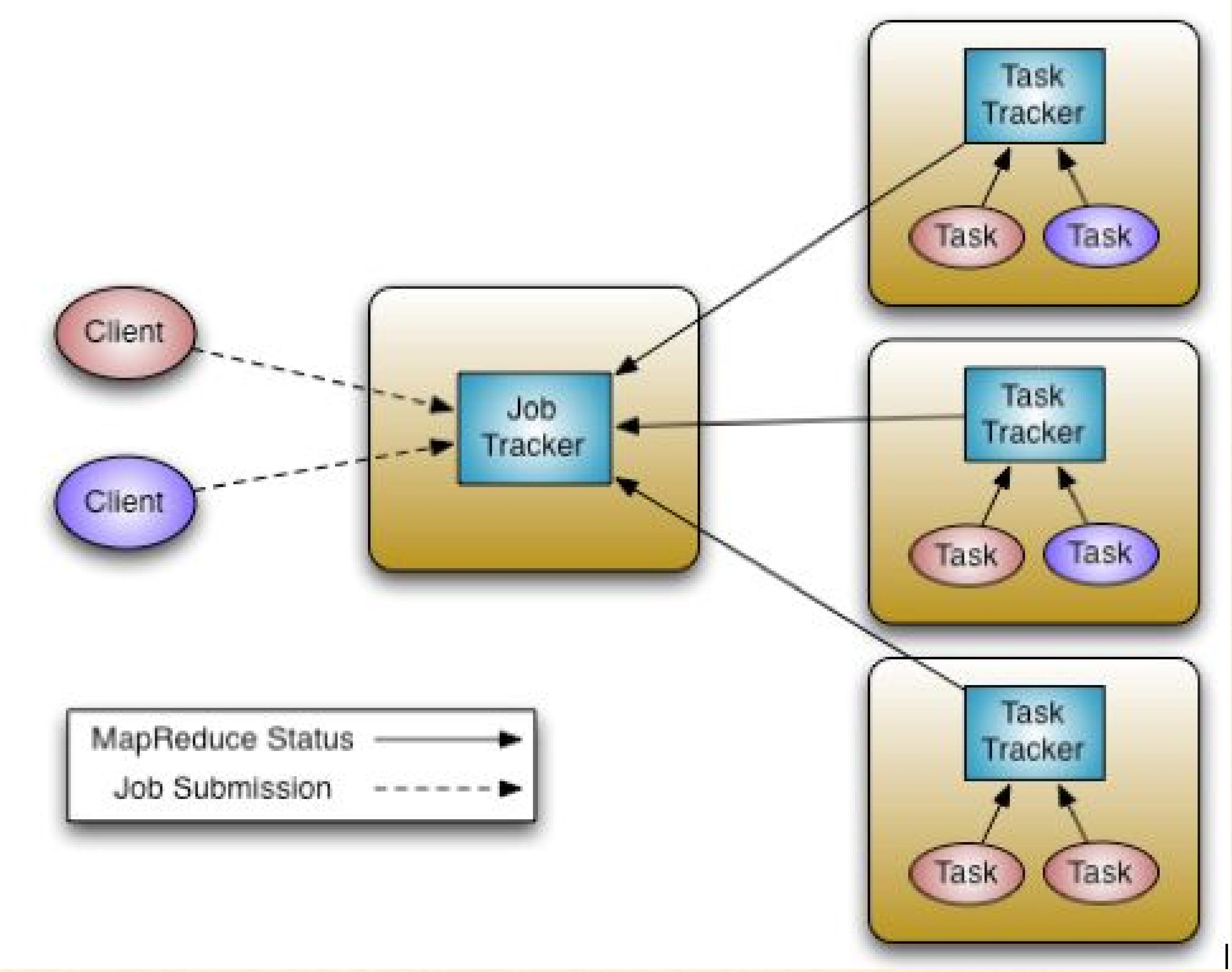


JOB TRACKER (CONT.)



JOB TRACKER (CONT.)




Task Tracker
Task
Task
Client
Job Tracker
Client
Task Tracker
Task
Task
MapReduce Status
Job Submission
Task Tracker
Task
Task

QUICK - CLUSTER HANDS ON

MapReduce Command

The Example is available here

Remove old output directory

```
hadoop fs -rm -r /user/student/wordcount/output
```

Execute the mapReduce Command:

```
hadoop jar /usr/hdp/2.3.4.0-3485/hadoop-mapreduce/hadoop-mapreduce-examples.jar
wordcount /data/mr/wordcount/input mrout
```