

# Computational Analysis of Big Data

Week 8

## MapReduce

# Upcoming timeline

Wednesday April 3, 23:59: Assignment 3

Wednesday April 3, in class: **Project validation**

Wednesday April 10, 23:59: Project videos

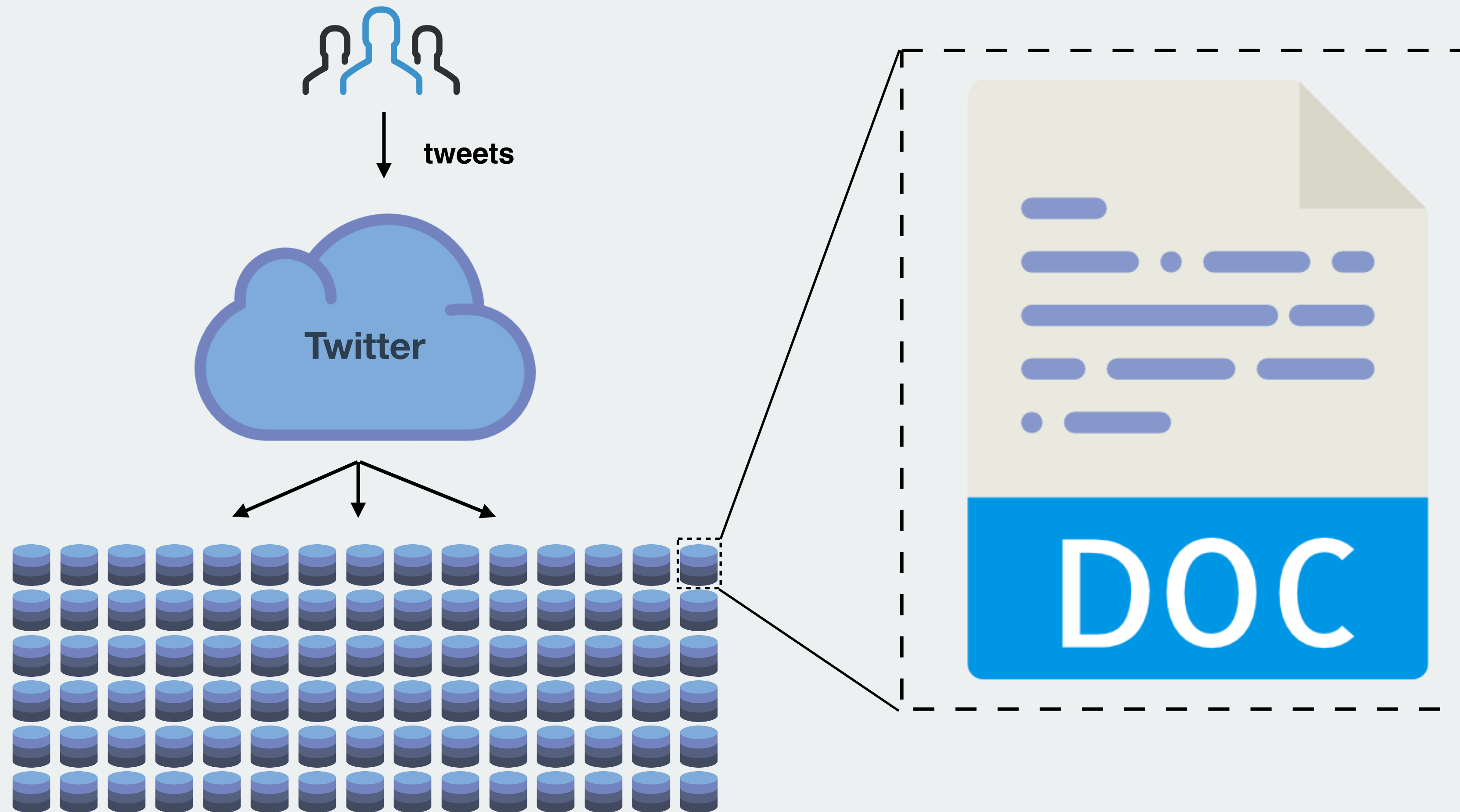
Wednesday May 2, in class: **Project presentations**

# MapReduce

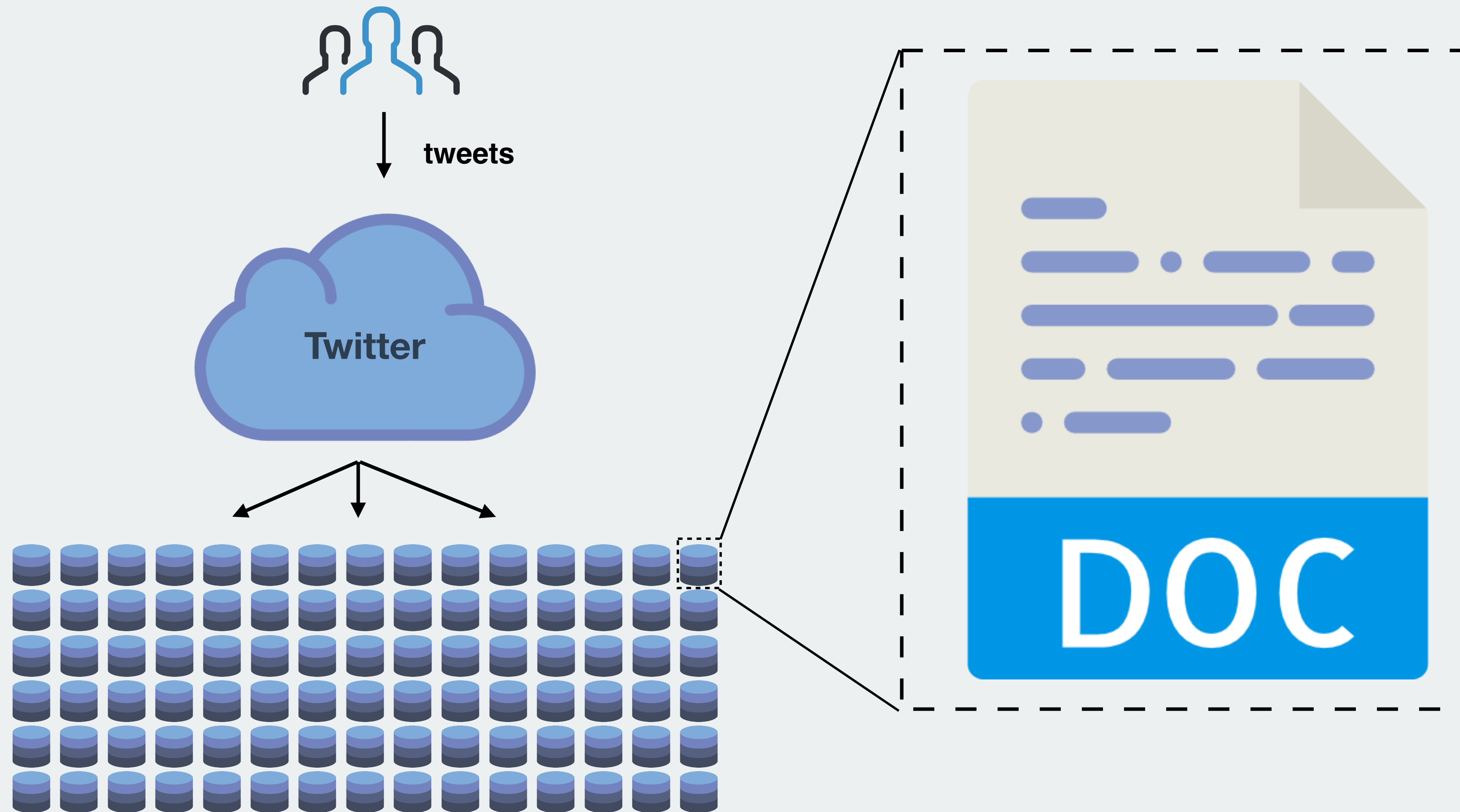
“MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster.”

- *Wikipedia*

# Problem: Massive computation

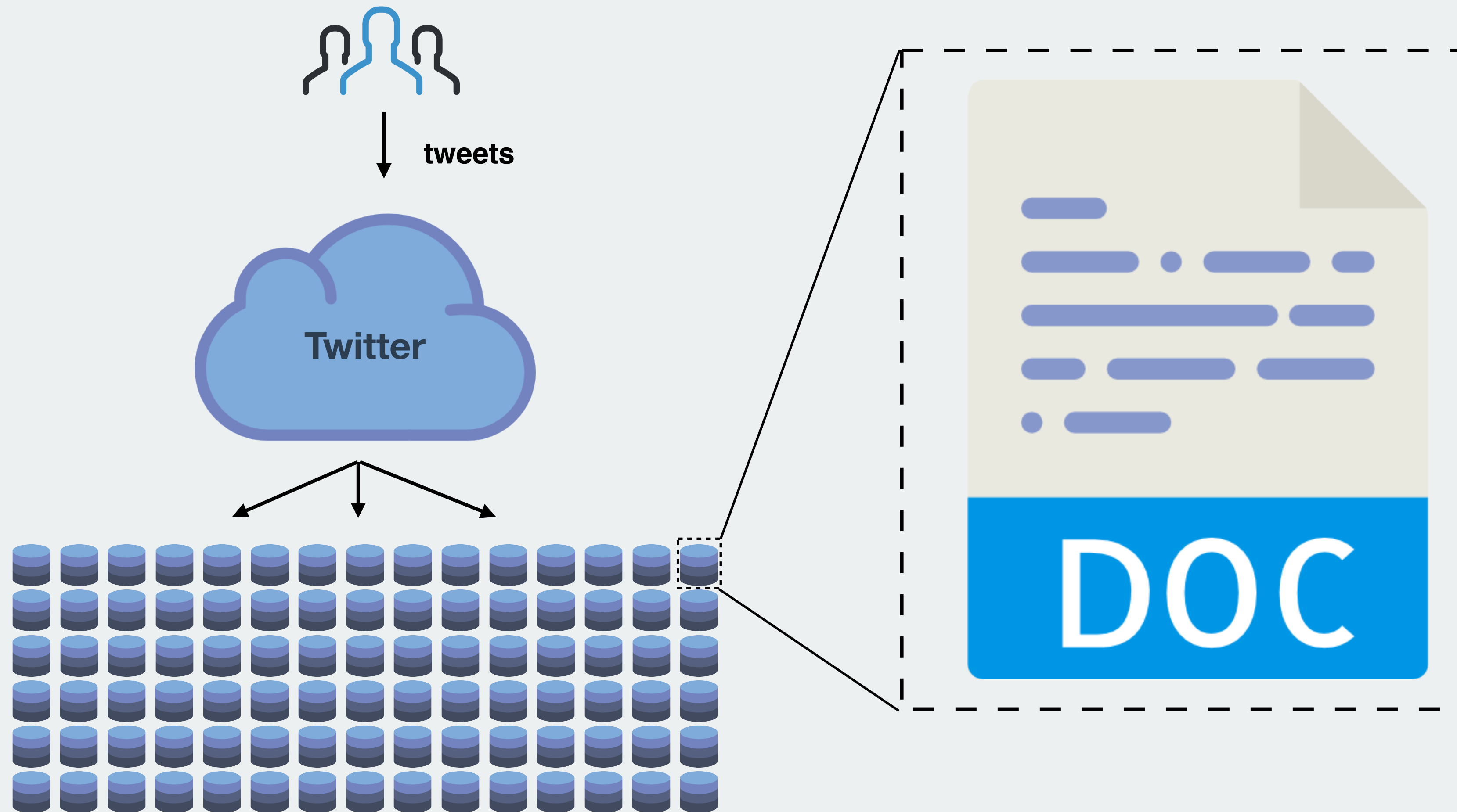


# Problem: Massive computation



**Objective:** Count number of occurrences for each word used on twitter

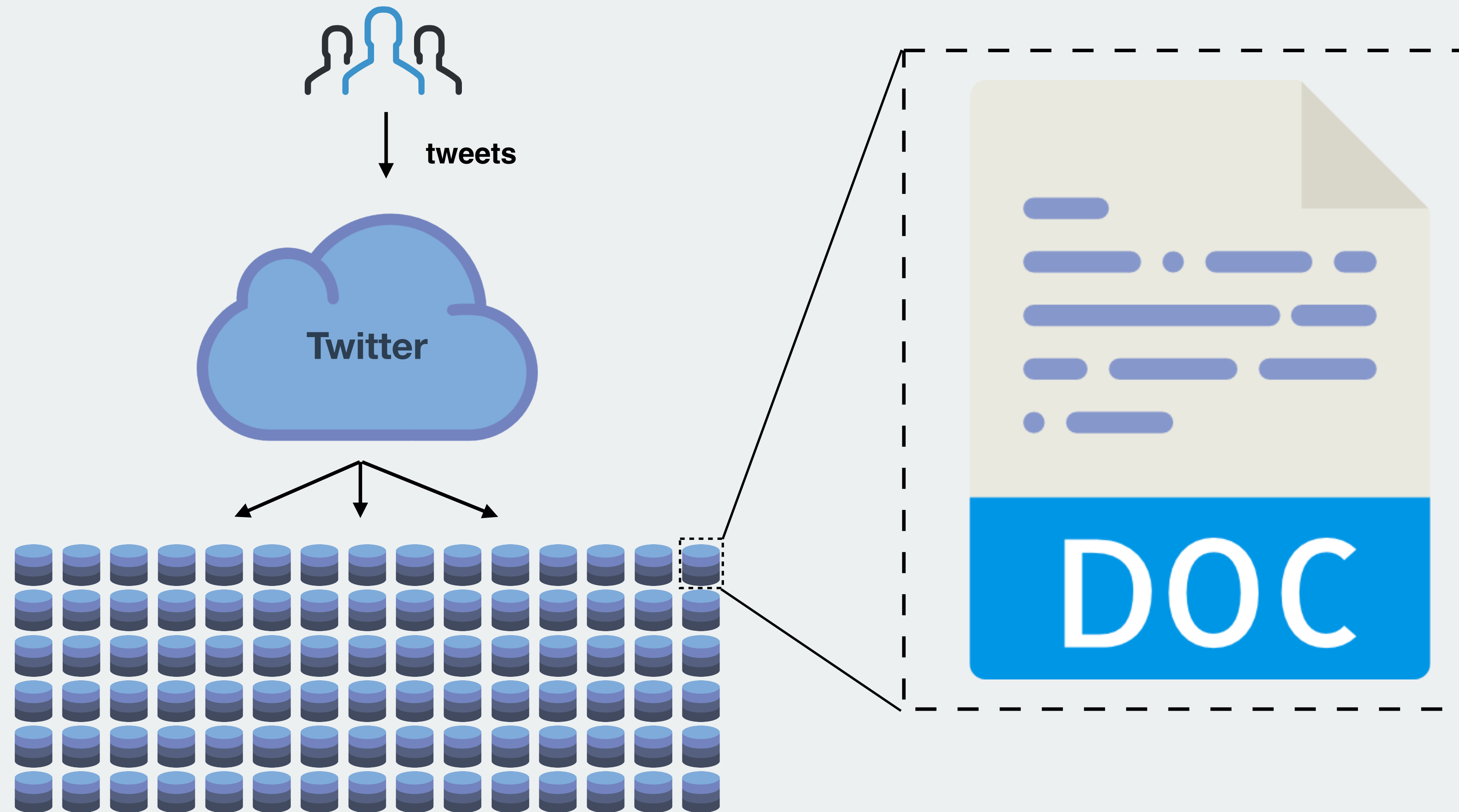
# Problem: Massive computation



**Objective:** Count number of occurrences for each word used on twitter

**Sequential:** Set a Python script to loop over databases and for each word in each document, increment the word's Counter .

# Problem: Massive computation

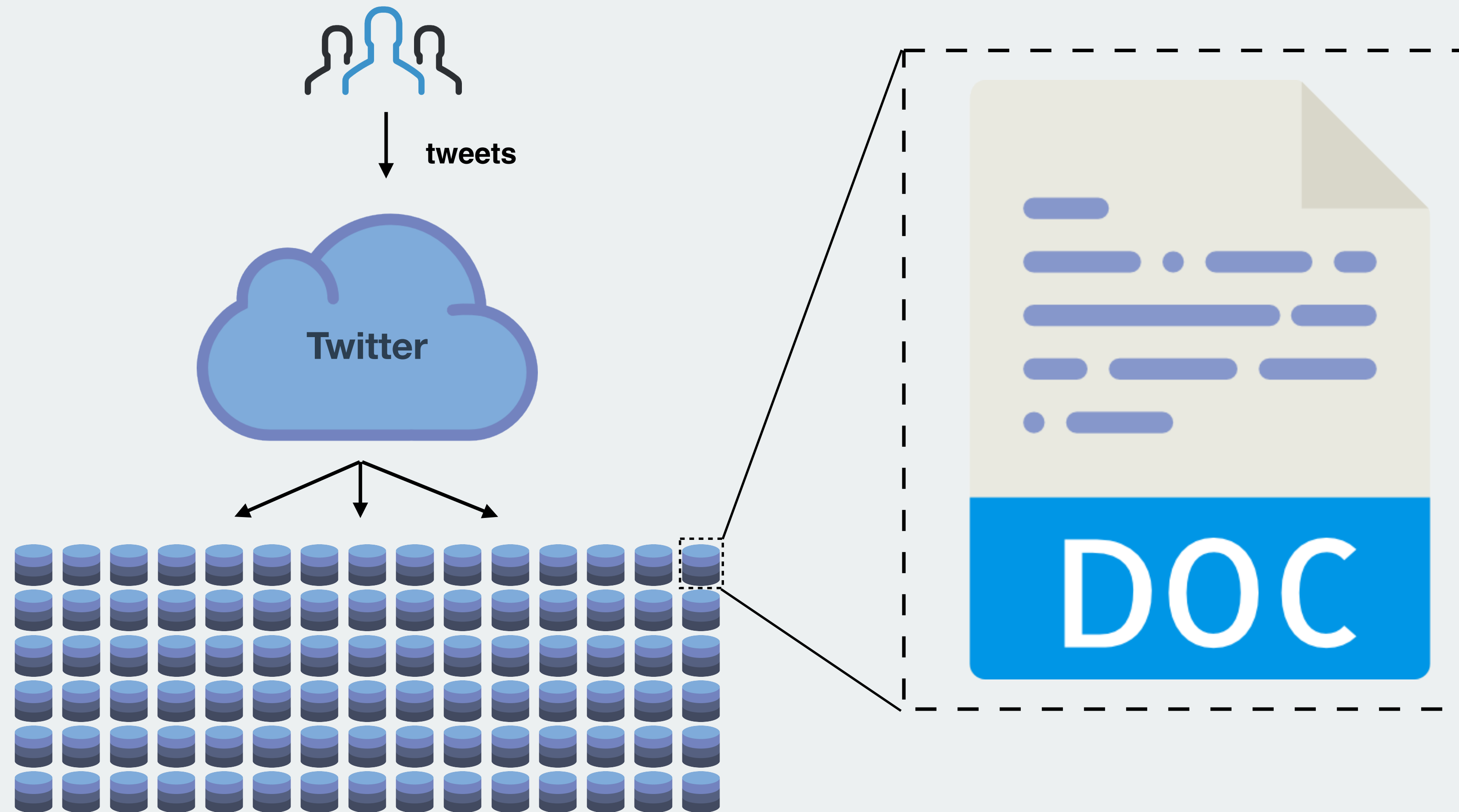


**Objective:** Count number of occurrences for each word used on twitter

**Sequential:** Set a Python script to loop over databases and for each word in each document, increment the word's Counter .

**SLOW**

# Problem: Massive computation



**Objective:** Count number of occurrences for each word used on twitter

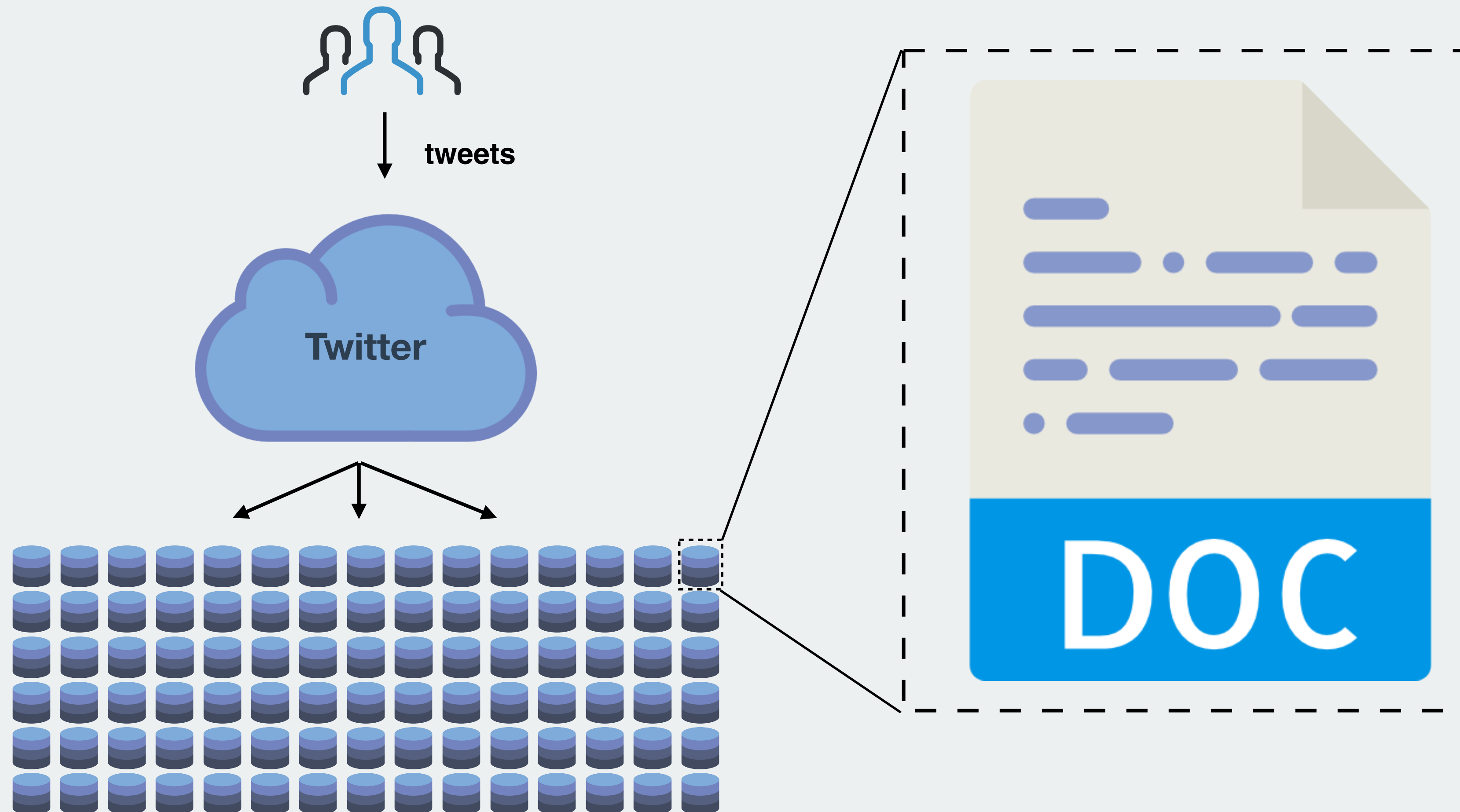
**Sequential:** Set a Python script to loop over databases and for each word in each document, increment the word's Counter .

**SLOW**

**Parallel:** Send Python script to multiple databases at a time, which loops over words in the document and increments a central Counter.



# Problem: Massive computation



**Objective:** Count number of occurrences for each word used on twitter

**Sequential:** Set a Python script to loop over databases and for each word in each document, increment the word's Counter .

**SLOW**

**Parallel:** Send Python script to multiple databases at a time, which loops over words in the document and increments a central Counter.

**READ/WRITE ERRORS**

# Solution: MapReduce

**Map step**

**Reduce step**



# Solution: MapReduce

**Map step**

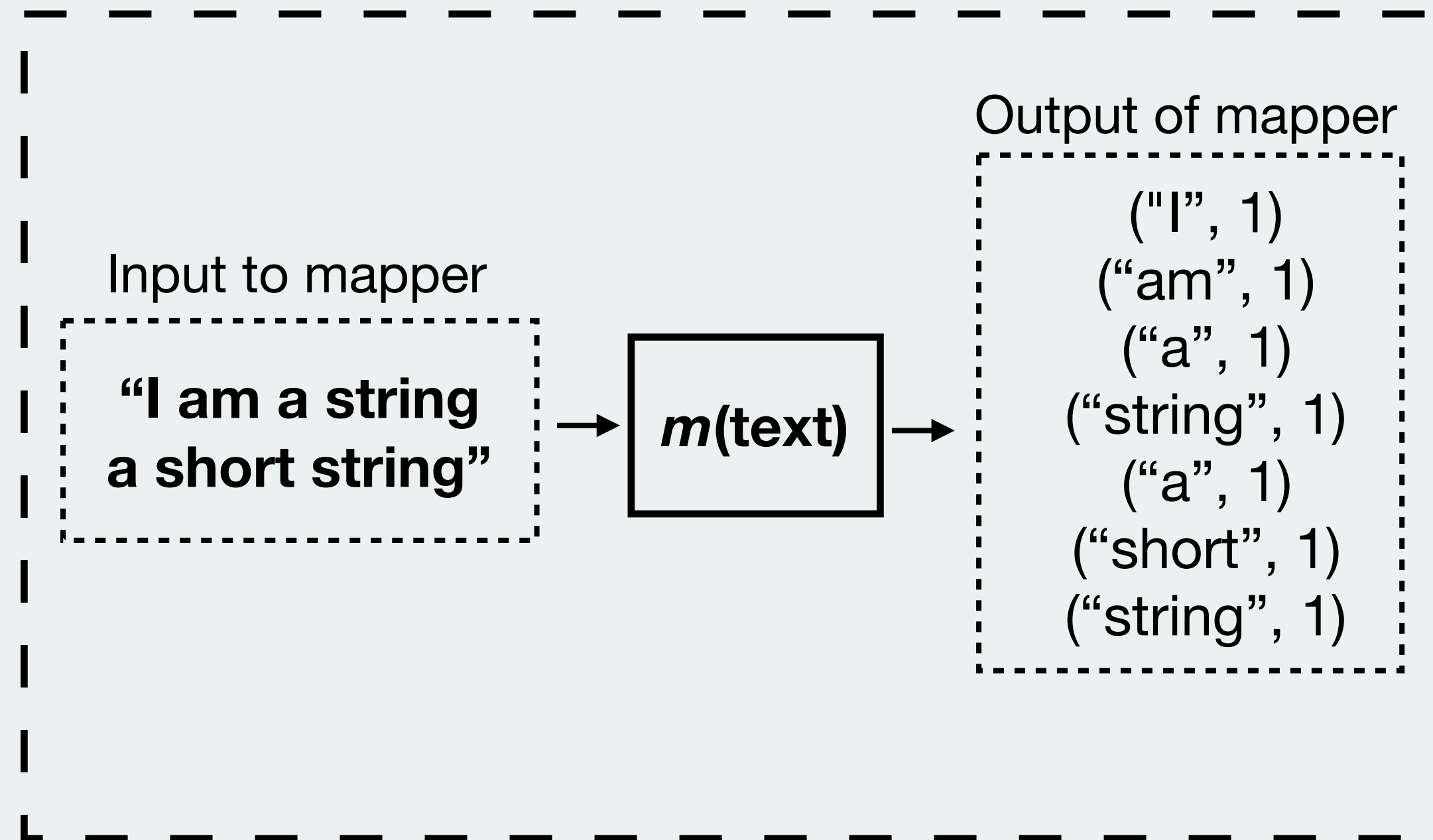
**“I am a string  
a short string”**

**Reduce step**

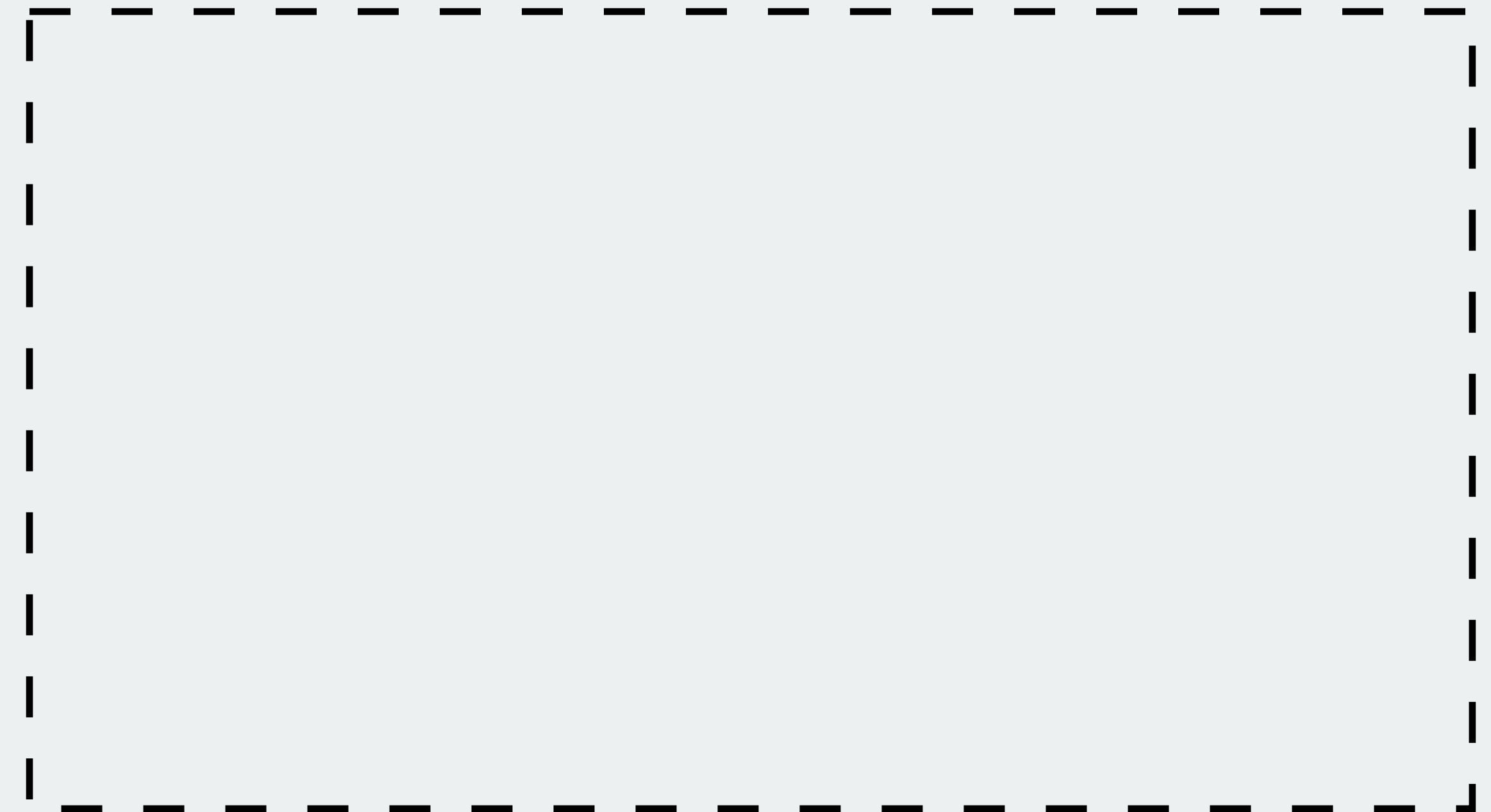


# Solution: MapReduce

## Map step

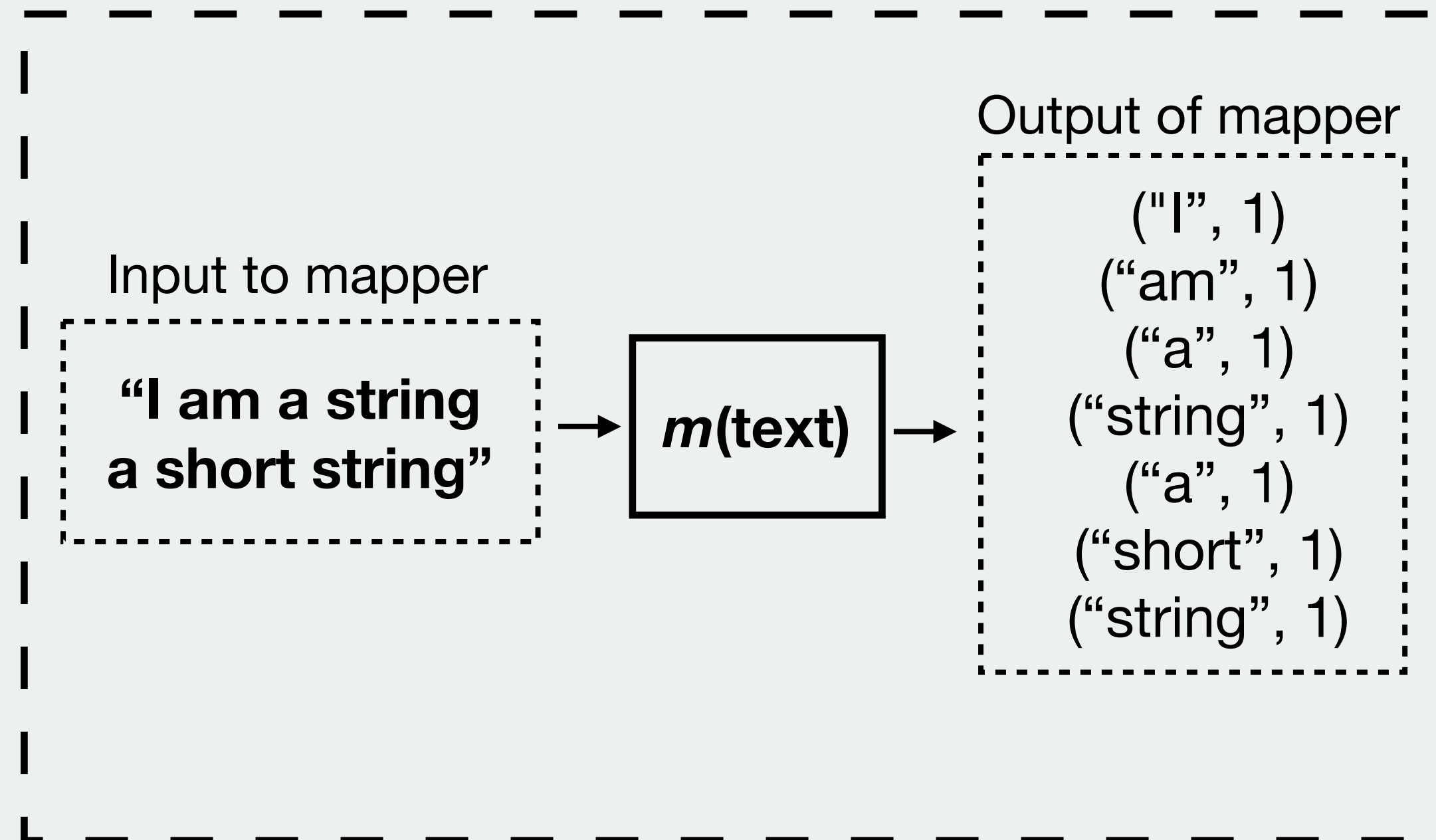


## Reduce step



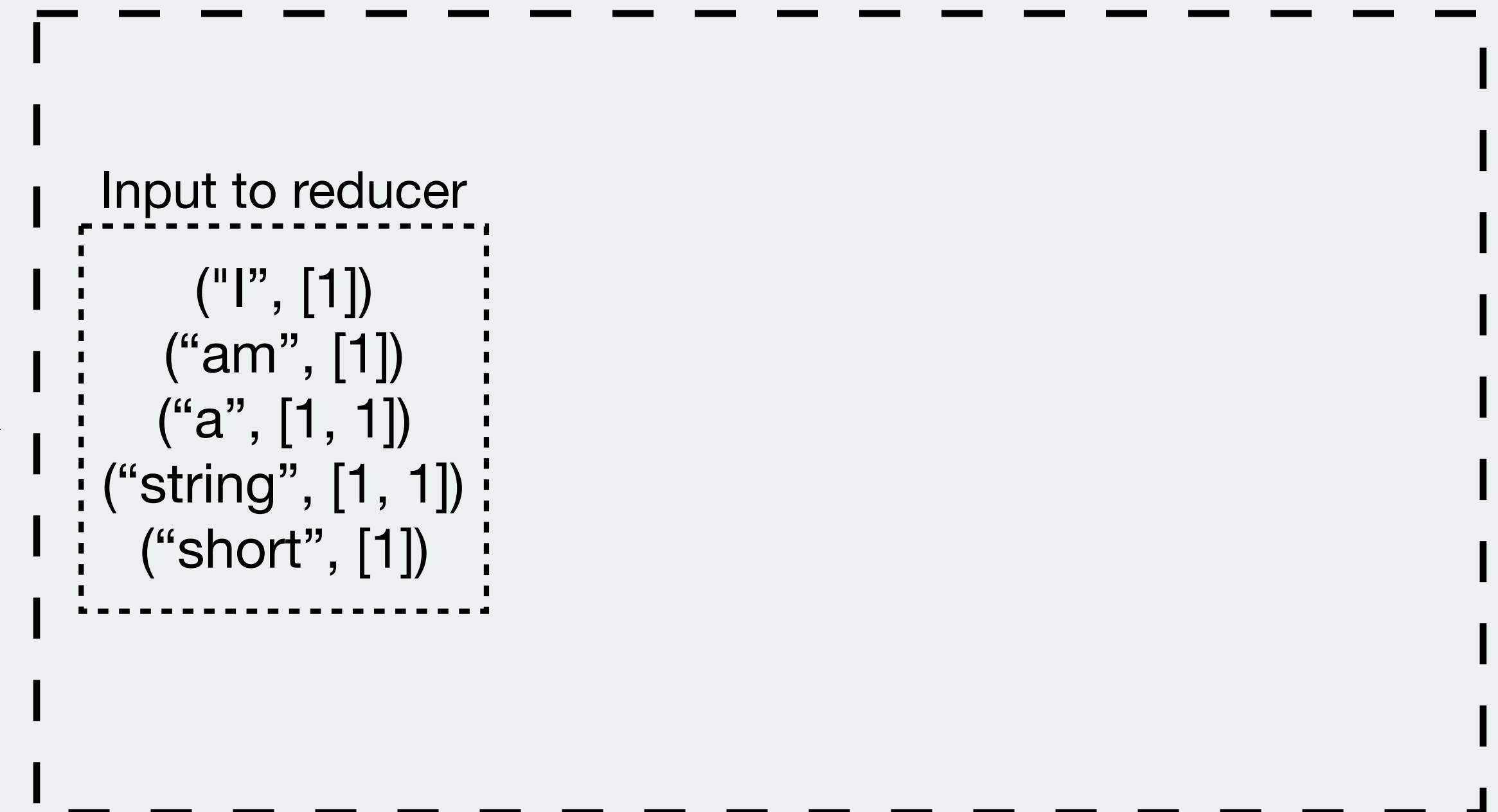
# Solution: MapReduce

## Map step



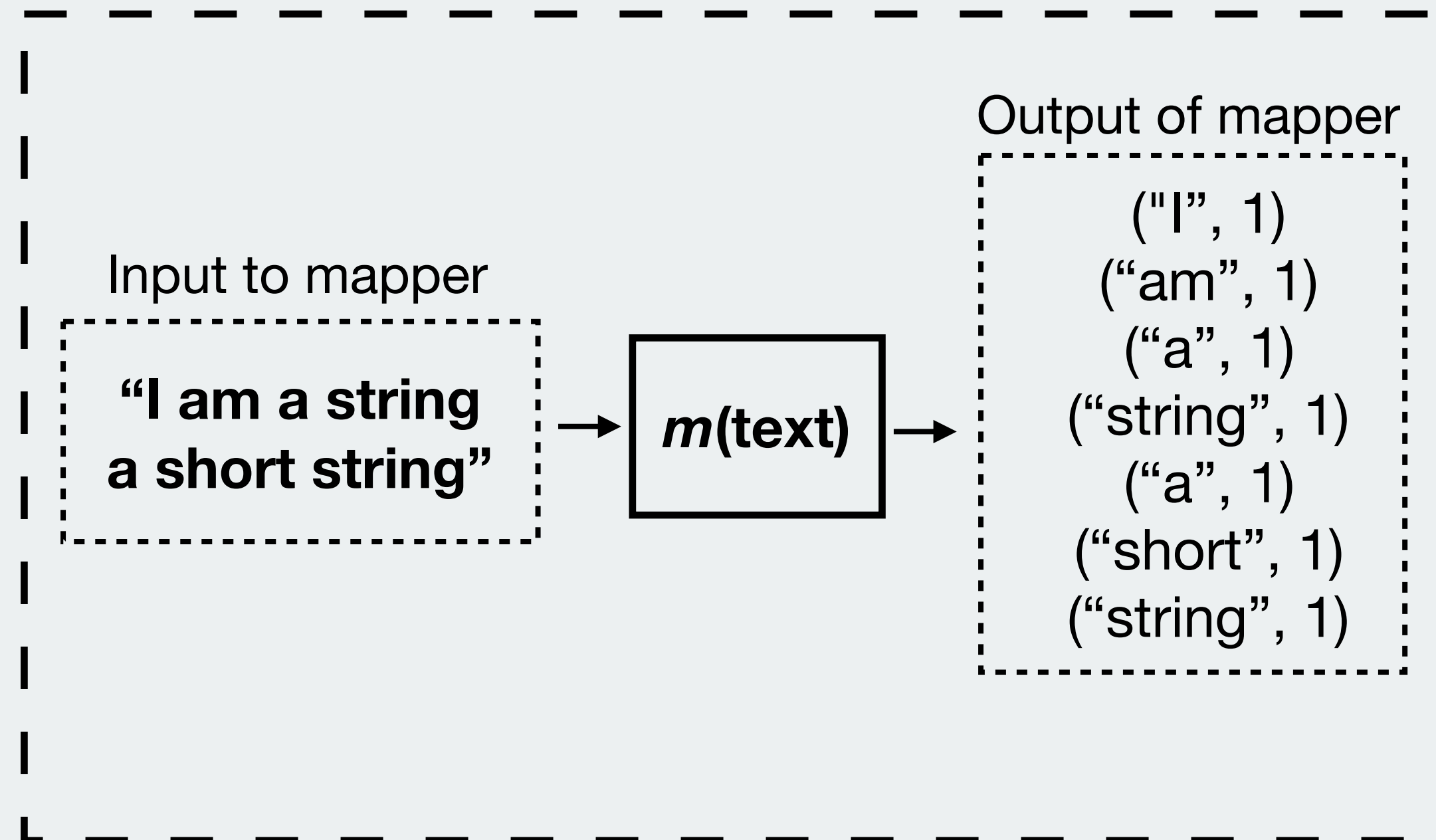
group  
by keys

## Reduce step



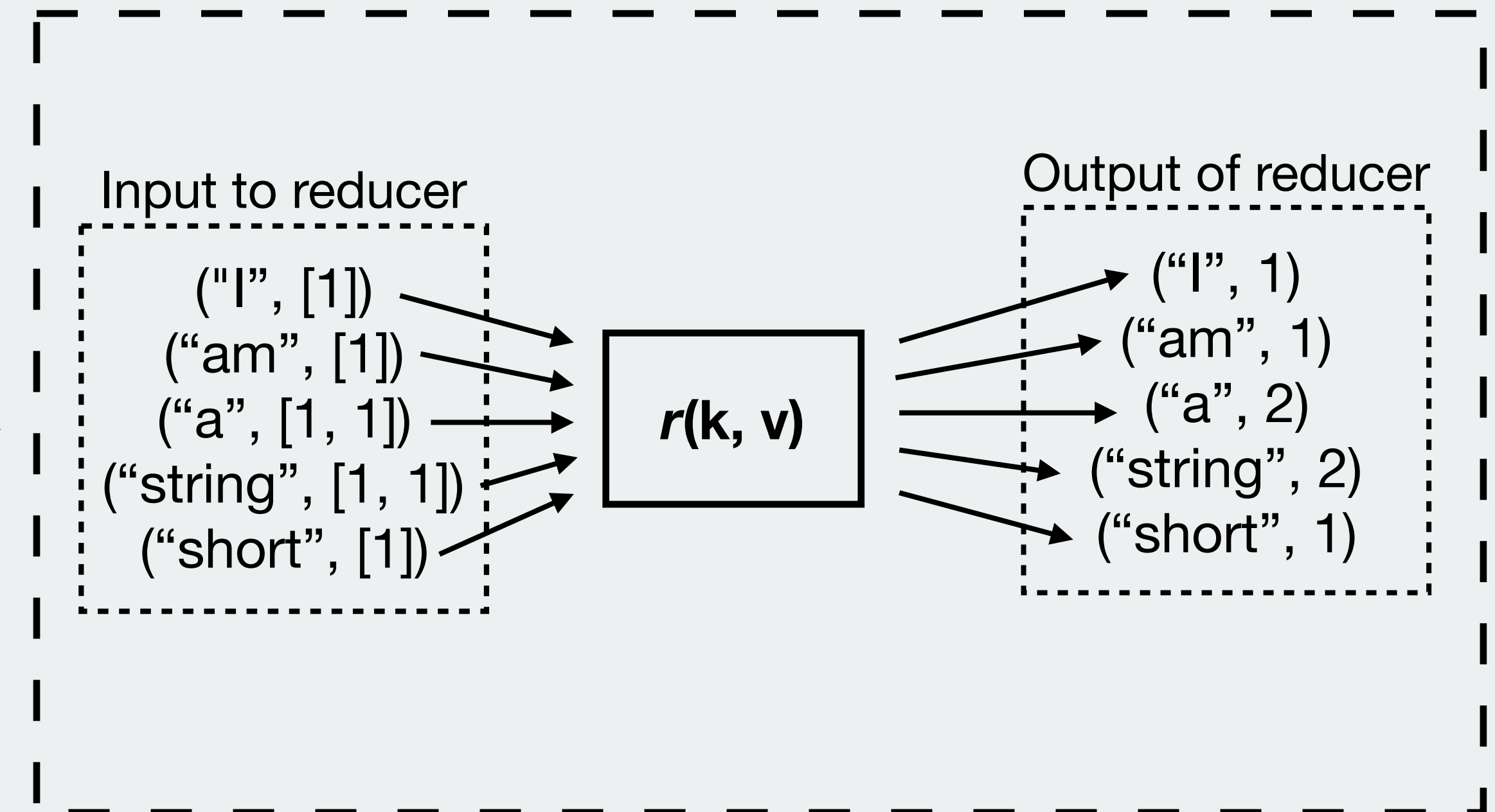
# Solution: MapReduce

## Map step

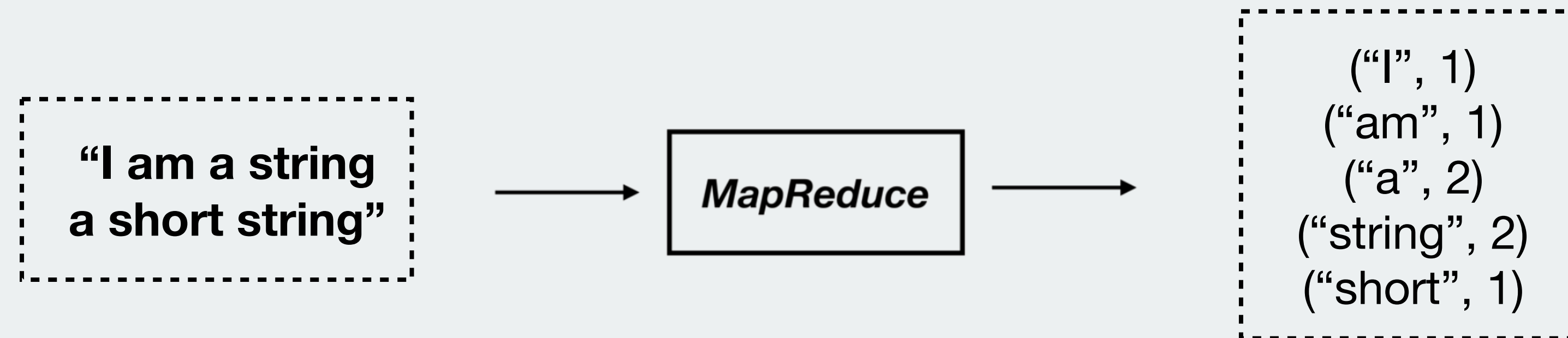


group  
by keys

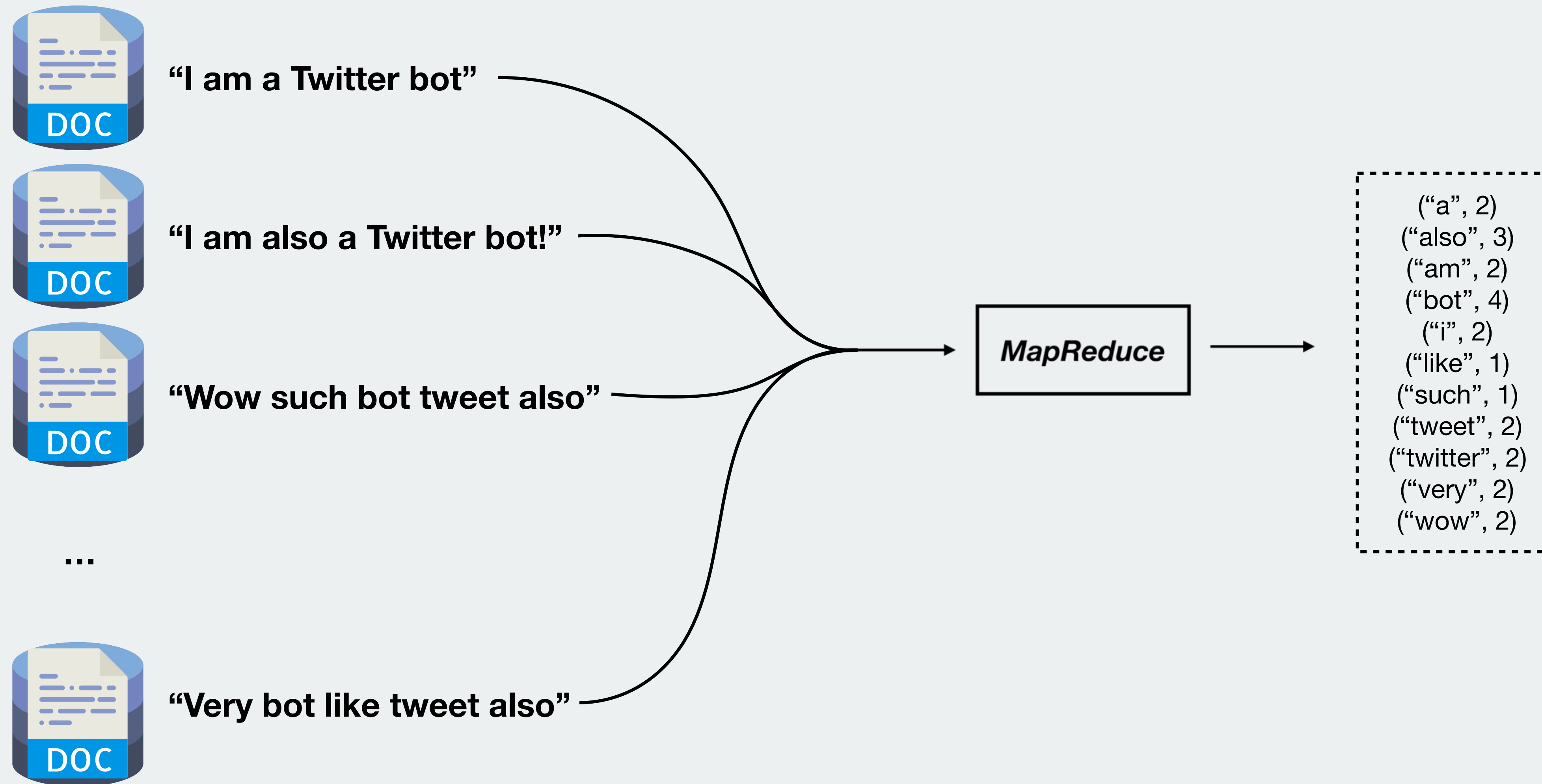
## Reduce step



# Solution: MapReduce



# Word count with multiple documents





# Map step



**“I am a Twitter bot”**



**“I am also a Twitter bot!”**



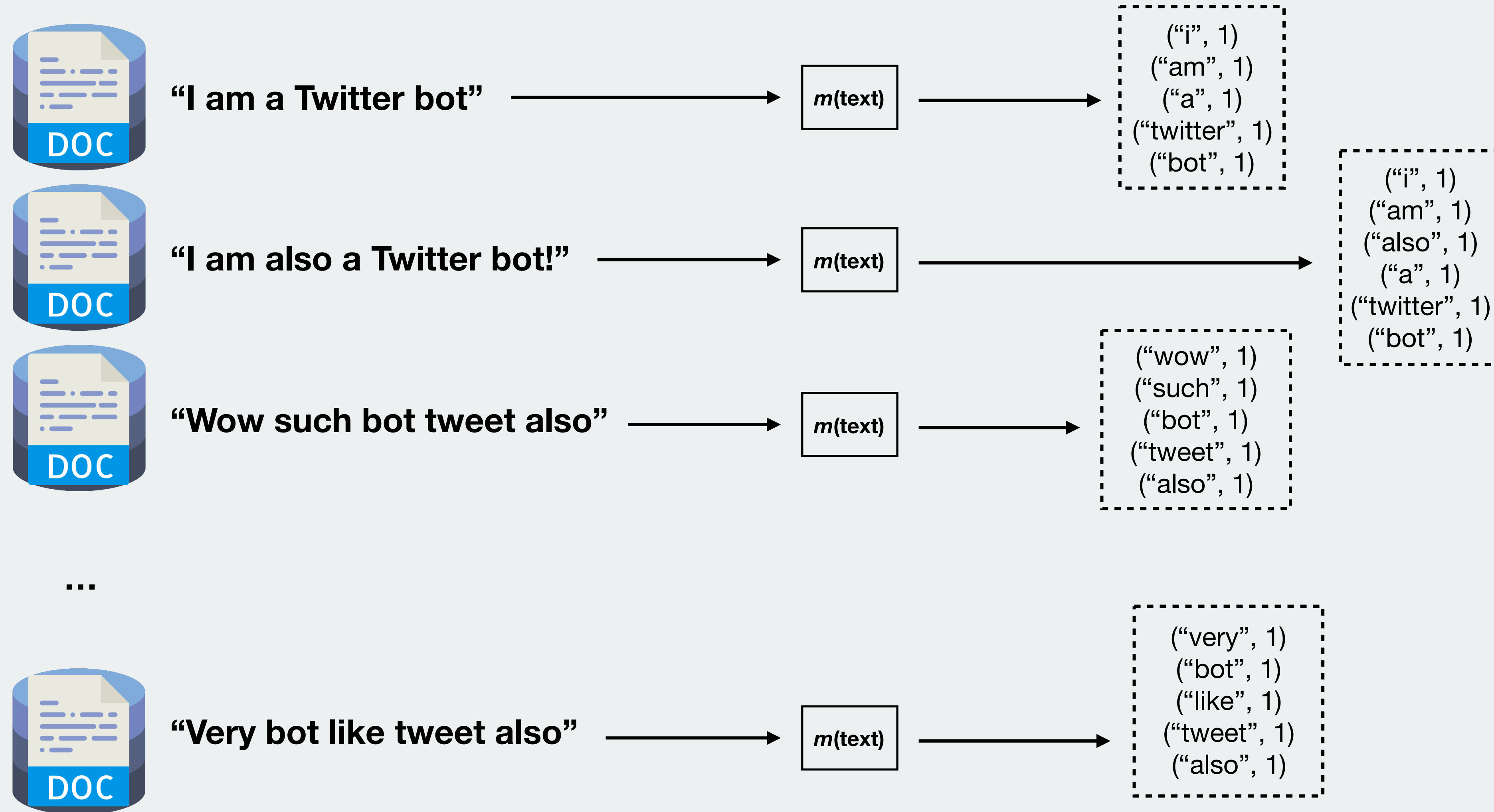
**“Wow such bot tweet also”**

...

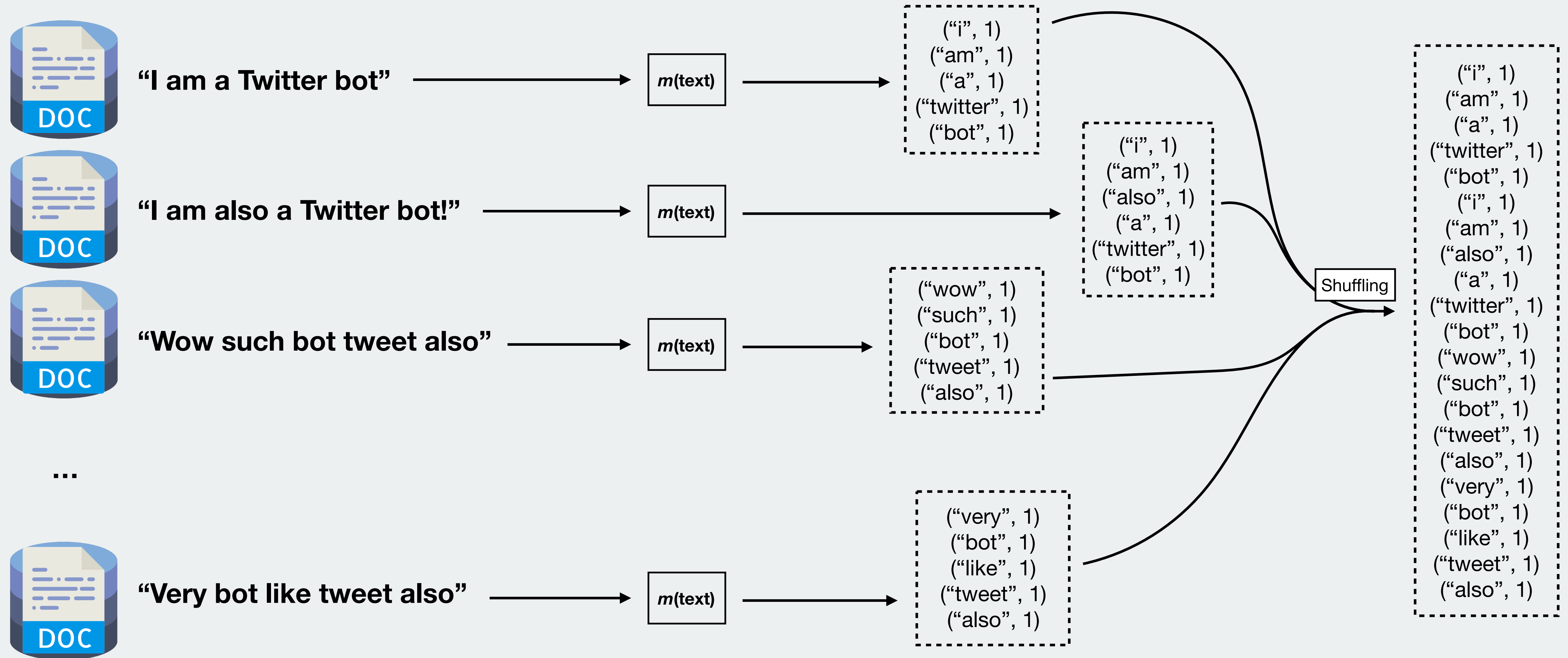


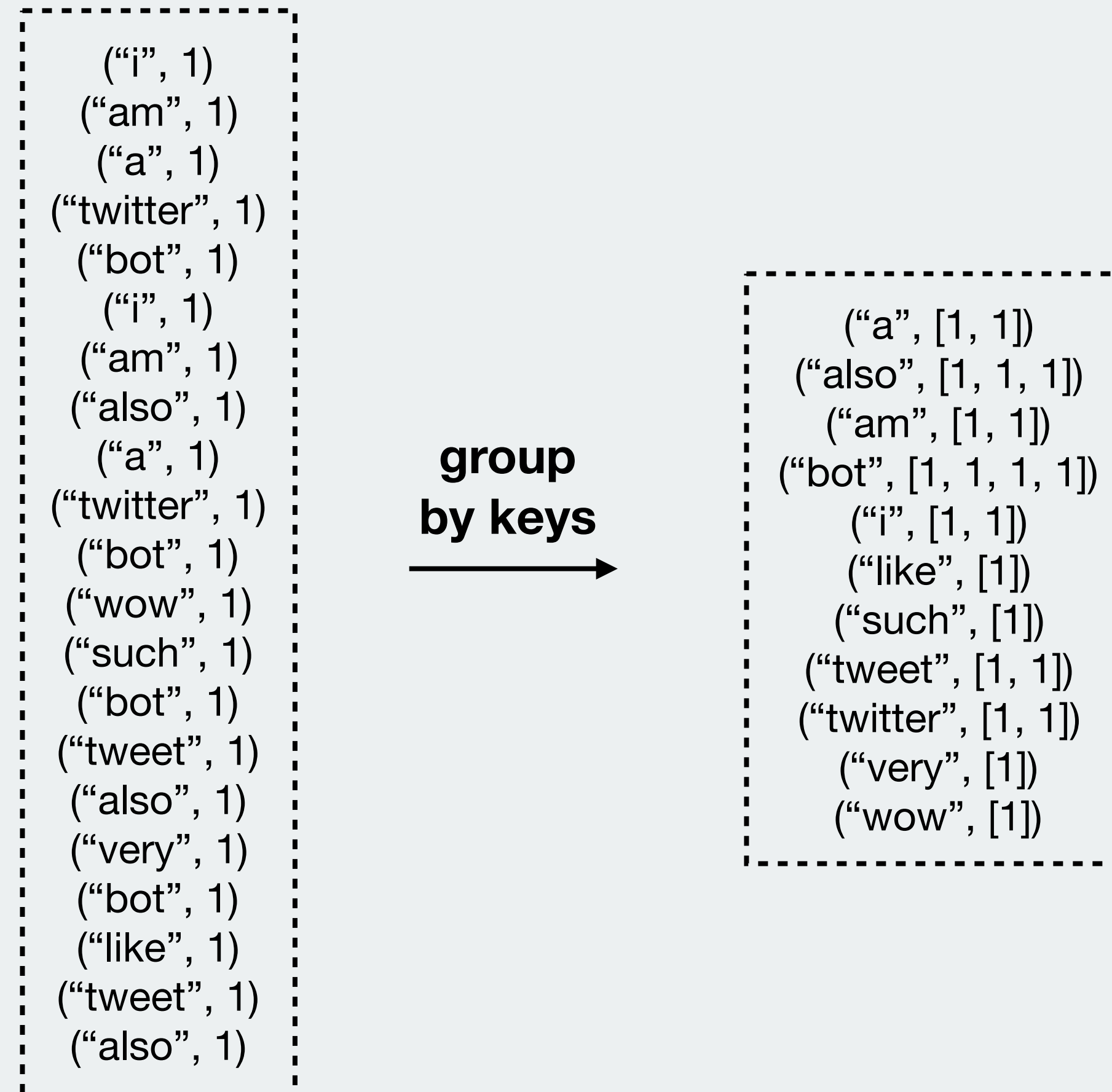
**“Very bot like tweet also”**

# Map step

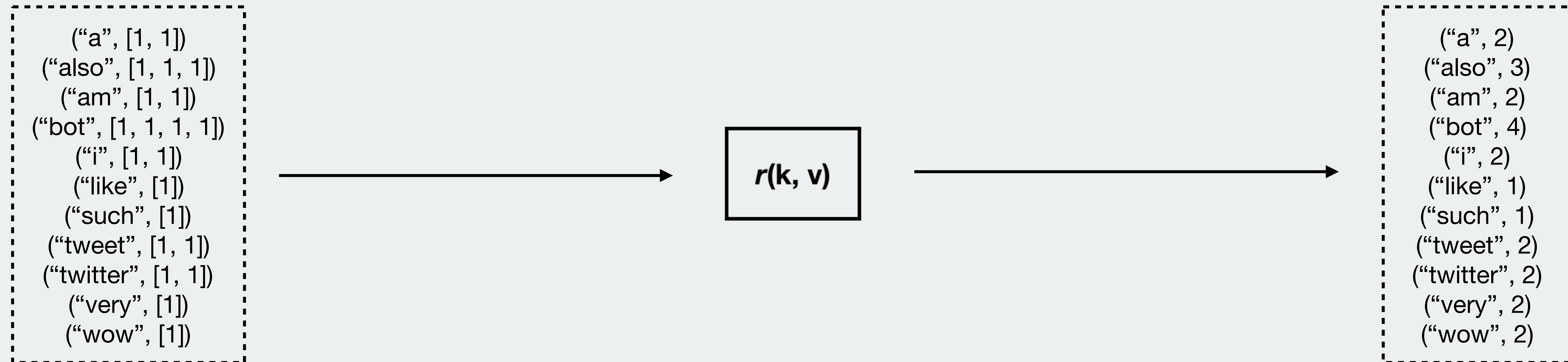


# Map step

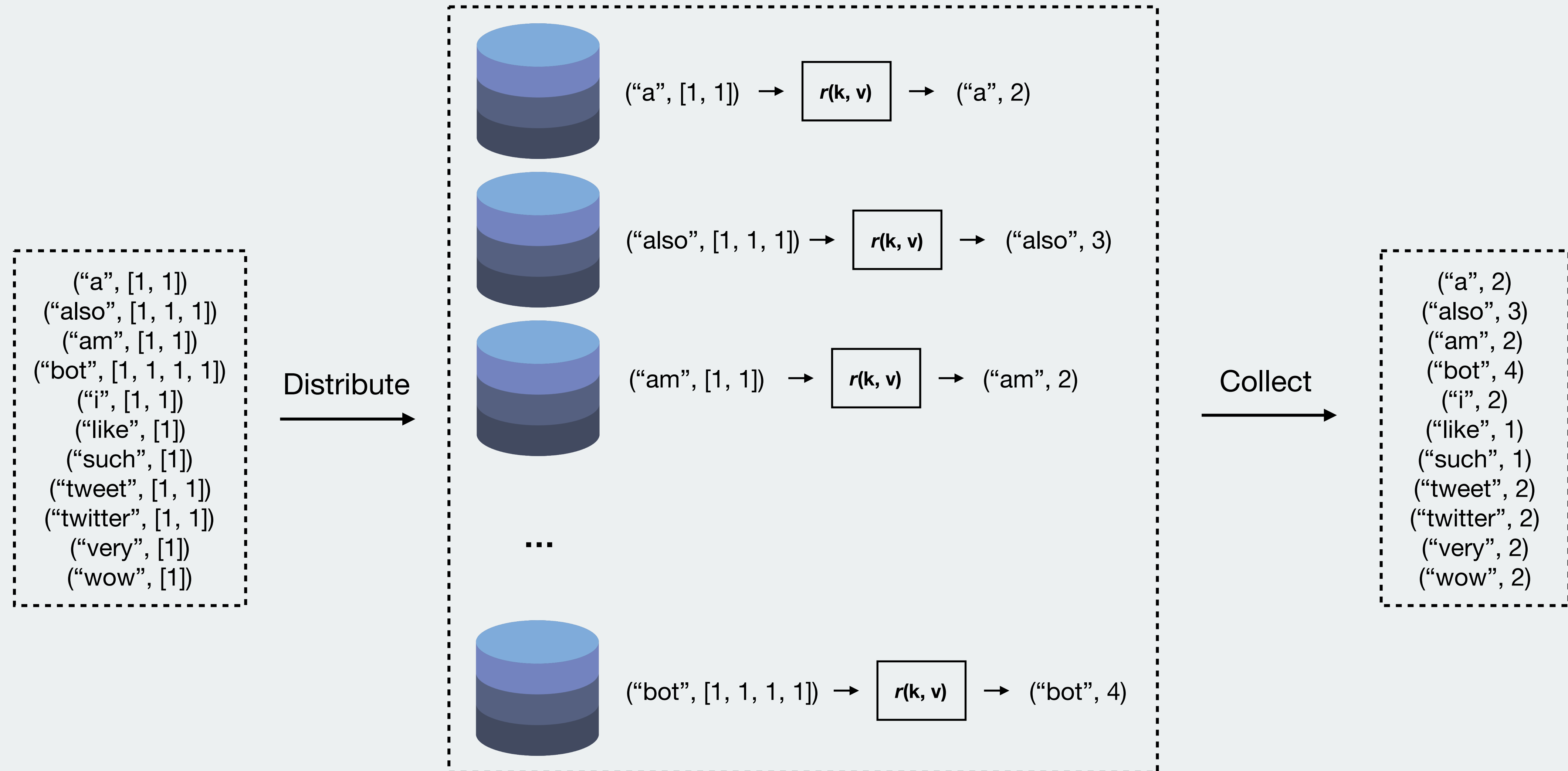




# Reduce step

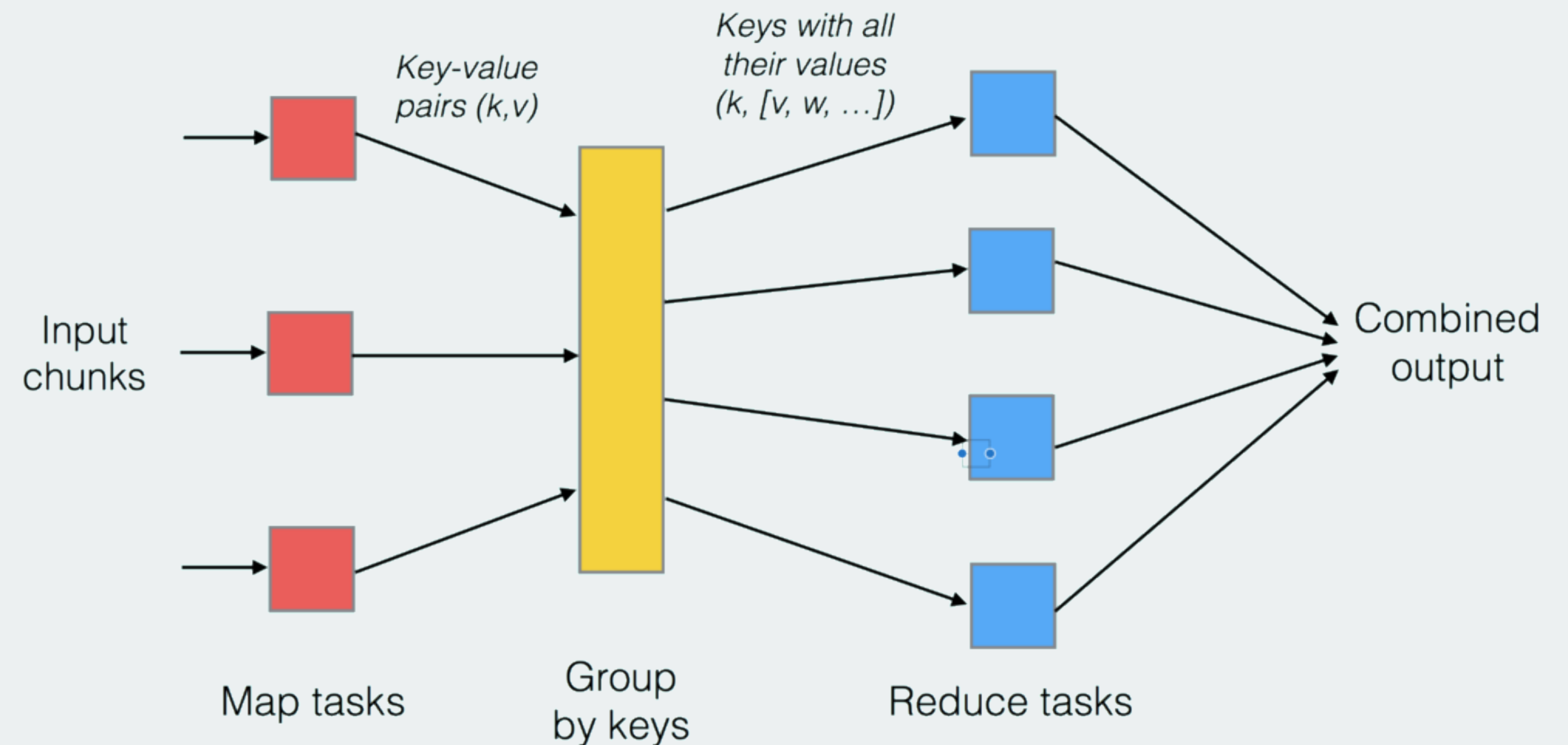


# Reduce step

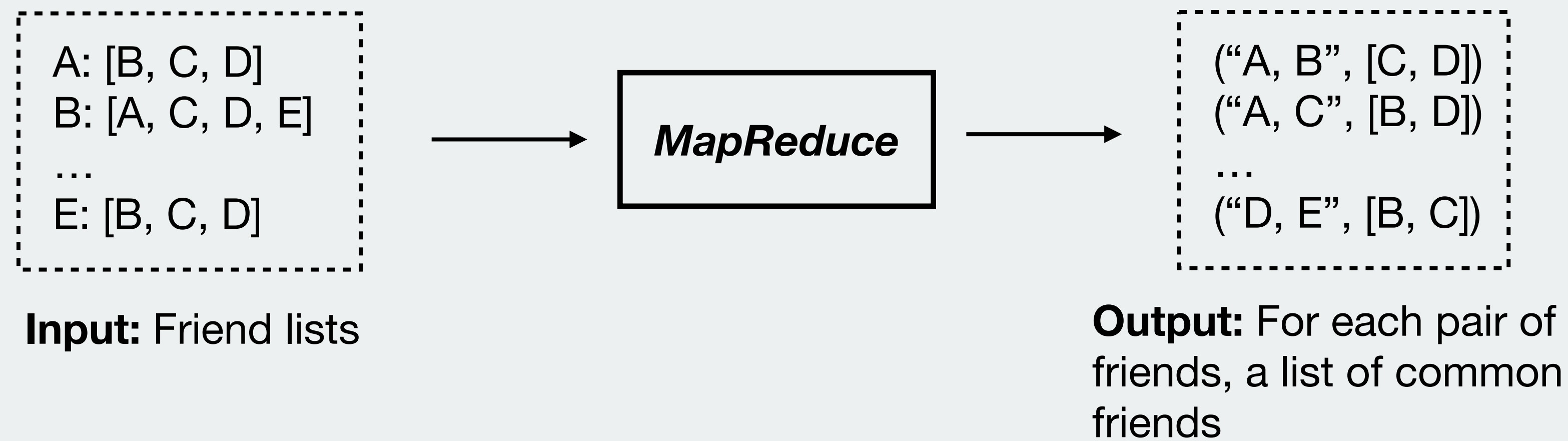


# Summary

1. Map input **key-value** pairs
2. Group values by their keys
3. Perform an operation on each key's list of values

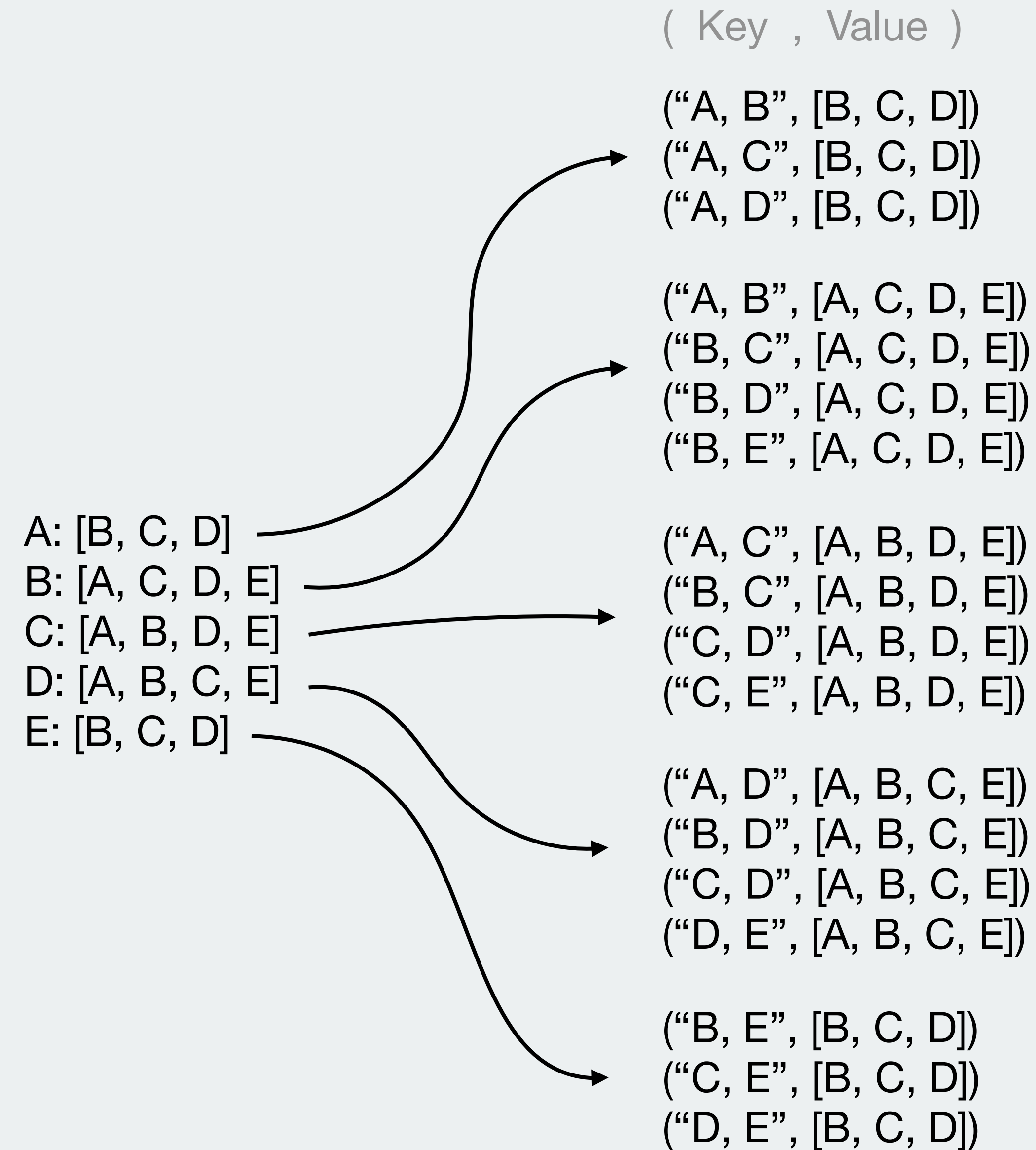


# Advanced example: Common friends





# Map step



# Group by key

( Key , Value )

(“A, B”, [B, C, D])  
 (“A, C”, [B, C, D])  
 (“A, D”, [B, C, D])  
 (“A, B”, [A, C, D, E])  
 (“B, C”, [A, C, D, E])  
 (“B, D”, [A, C, D, E])  
 (“B, E”, [A, C, D, E])  
 (“A, C”, [A, B, D, E])  
 (“B, C”, [A, B, D, E])  
 (“C, D”, [A, B, D, E])  
 (“C, E”, [A, B, D, E])  
 (“A, D”, [A, B, C, E])  
 (“B, D”, [A, B, C, E])  
 (“C, D”, [A, B, C, E])  
 (“D, E”, [A, B, C, E])  
 (“B, E”, [B, C, D])  
 (“C, E”, [B, C, D])  
 (“D, E”, [B, C, D])



( Key , [Value1, Value2] )

(“A, B”, [[B, C, D], [A, C, D, E]])  
 (“A, C”, [[B, C, D], [A, B, D, E]])  
 (“A, D”, [[B, C, D], [A, B, C, E]])  
 (“B, C”, [[A, C, D, E], [A, B, D, E]])  
 (“B, D”, [[A, C, D, E], [A, B, C, E]])  
 (“B, E”, [[A, C, D, E], [B, C, D]])  
 (“C, D”, [[A, B, D, E], [A, B, C, E]])  
 (“C, E”, [[A, B, D, E], [B, C, D]])  
 (“D, E”, [[A, B, C, E], [B, C, D]])

# Reduce step

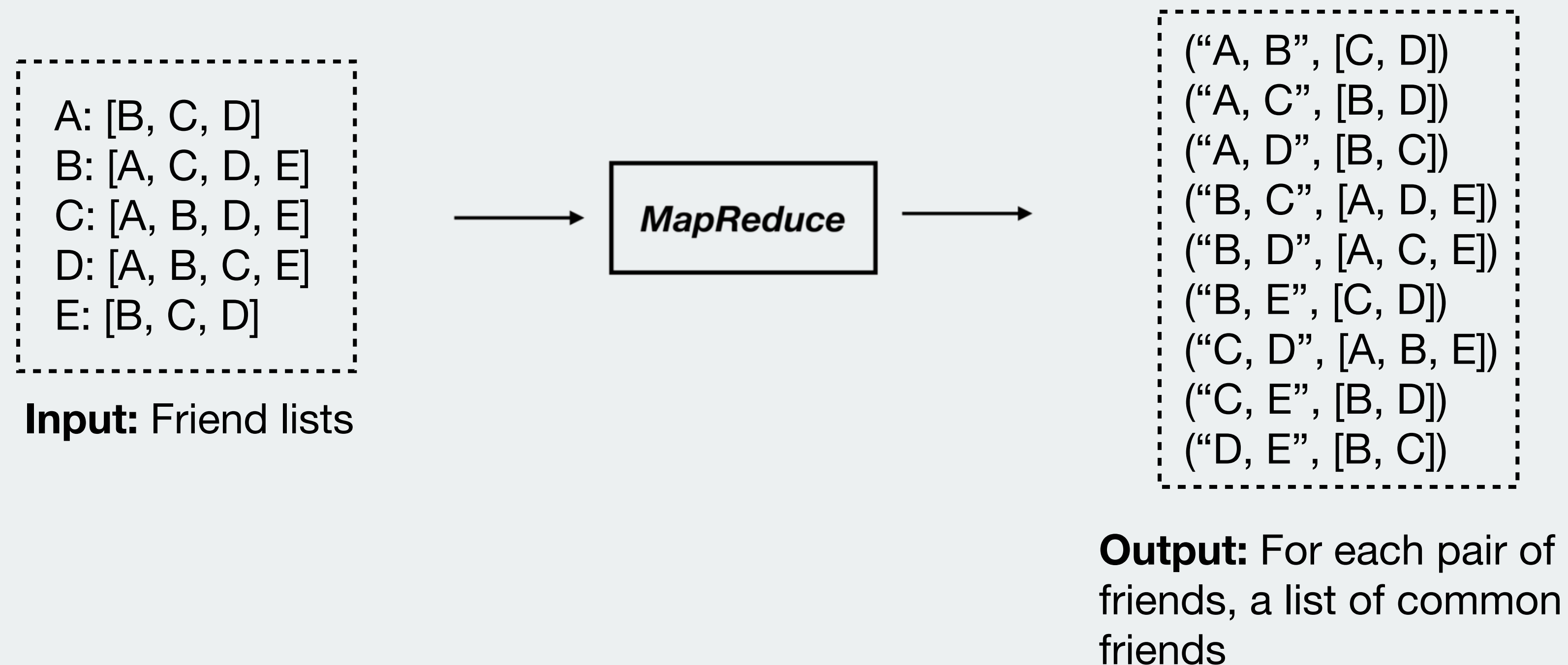
(“A, B”, [[**B, C, D**], [A, **C, D, E**]])  
(“A, C”, [[**B, C, D**], [A, **B, D, E**]])  
(“A, D”, [[**B, C, D**], [A, **B, C, E**]])  
(“B, C”, [[**A, C, D, E**], [**A, B, D, E**]])  
(“B, D”, [[**A, C, D, E**], [**A, B, C, E**]])  
(“B, E”, [[A, **C, D, E**], [B, **C, D**]])  
(“C, D”, [[**A, B, D, E**], [**A, B, C, E**]])  
(“C, E”, [[A, **B, D, E**], [**B, C, D**]])  
(“D, E”, [[A, **B, C, E**], [**B, C, D**]])

Intersection



(“A, B”, [C, D])  
(“A, C”, [B, D])  
(“A, D”, [B, C])  
(“B, C”, [A, D, E])  
(“B, D”, [A, C, E])  
(“B, E”, [C, D])  
(“C, D”, [A, B, E])  
(“C, E”, [B, D])  
(“D, E”, [B, C])

# Advanced example: Common friends



# MapReduce in Python: mrjob

- Python package that lets you write MapReduce jobs in pure Python.
- Runs on your local machine as well as a Hadoop cluster
- Can also be used to write Spark jobs.

```
from mrjob.job import MRJob

class MRWordCounter(MRJob):

    def mapper(self, _, line):
        for word in line.split():
            yield word, 1

    def reducer(self, key, values):
        yield key, sum(values)

if __name__ == '__main__':
    MRWordCounter.run()
```



# MapReduce in Python: mrjob

```

from mrjob.job import MRJob

class MRWordCounter(MRJob):

    def mapper(self, _, line):
        for word in line.split():
            yield word, 1

    def reducer(self, key, values):
        yield key, sum(values)

if __name__ == '__main__':
    MRWordCounter.run()

```

my\_script.py

```

i am a twitter bot
i am also a twitter bot
wow such bot tweet also
very bot like tweet also

```

text\_file.txt

```

Desktop — ulfaslak@UAM — ~/Desktop — zsh — 80x35
➔ Desktop python my_script.py text_file.txt
no configs found; falling back on auto-configuration
no configs found; falling back on auto-configuration
creating tmp directory /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_scrip
t.ulfaslak.20171023.230714.054830
writing to /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_script.ulfaslak.2
0171023.230714.054830/step-0-mapper_part-00000
Counters from step 1:
  (no counters found)
writing to /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_script.ulfaslak.2
0171023.230714.054830/step-0-mapper-sorted
> sort /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_script.ulfaslak.20171
023.230714.054830/step-0-mapper_part-00000
writing to /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_script.ulfaslak.2
0171023.230714.054830/step-0-reducer_part-00000
Counters from step 1:
  (no counters found)
Moving /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_script.ulfaslak.20171
023.230714.054830/step-0-reducer_part-00000 -> /var/folders/1q/f3jgbgs96f120psg_
srrjw1r0000gn/T/my_script.ulfaslak.20171023.230714.054830/output/part-00000
Streaming final output from /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_
script.ulfaslak.20171023.230714.054830/output
"a"      2
"also"   3
"am"     2
"bot"    4
"i"      2
"like"   1
"such"   1
"tweet"  2
"twitter"      2
"very"   1
"wow"    1
removing tmp directory /var/folders/1q/f3jgbgs96f120psg_srrjw1r0000gn/T/my_scrip
t.ulfaslak.20171023.230714.054830

```