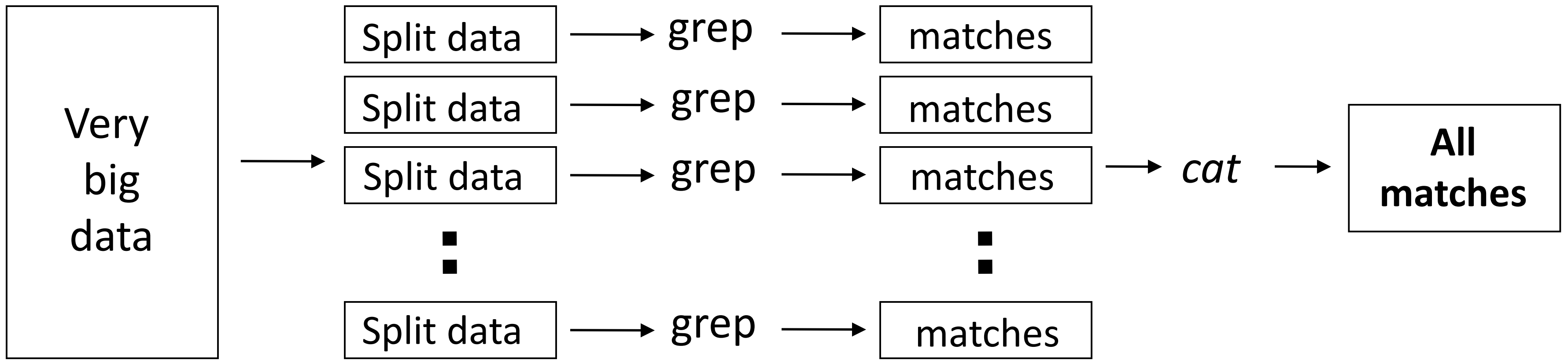
Map Reduce

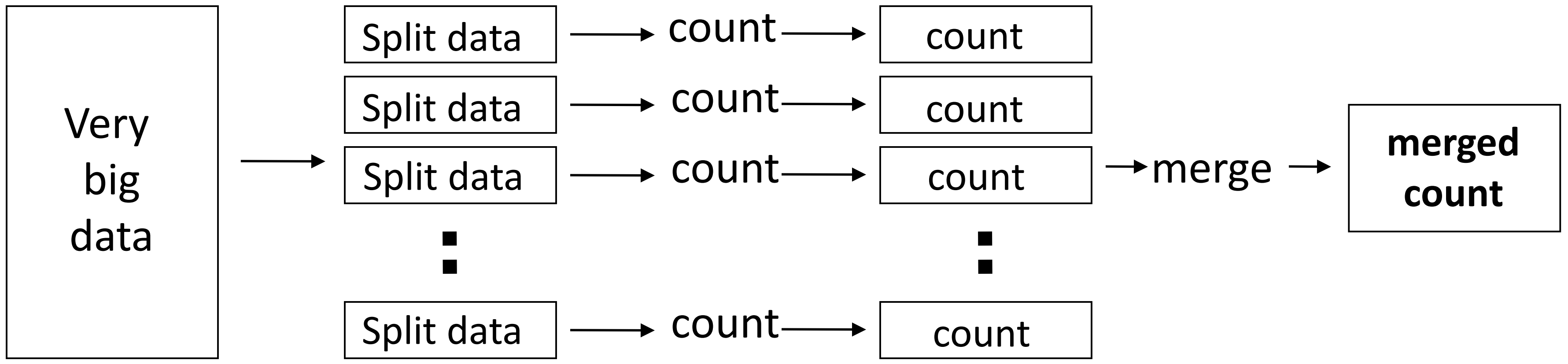
What is MapReduce?

- A programming model (& its associated implementation)
- For processing large data set
- Exploits large set of commodity computers
- Executes process in distributed manner
- Offers high degree of transparencies

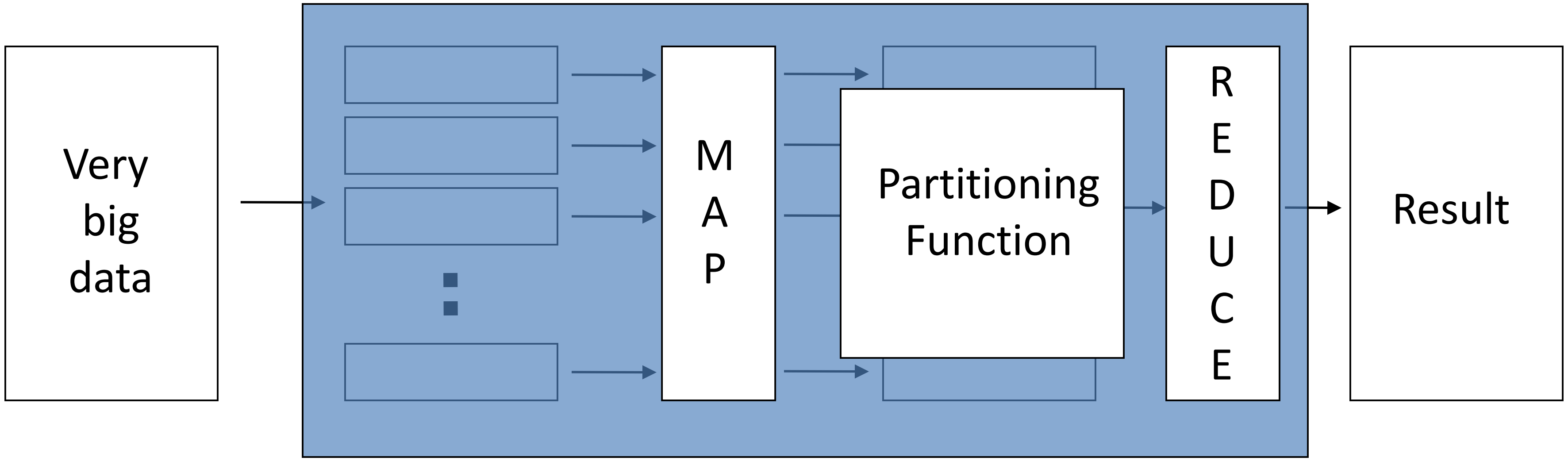
Distributed Grep



Distributed Word Count

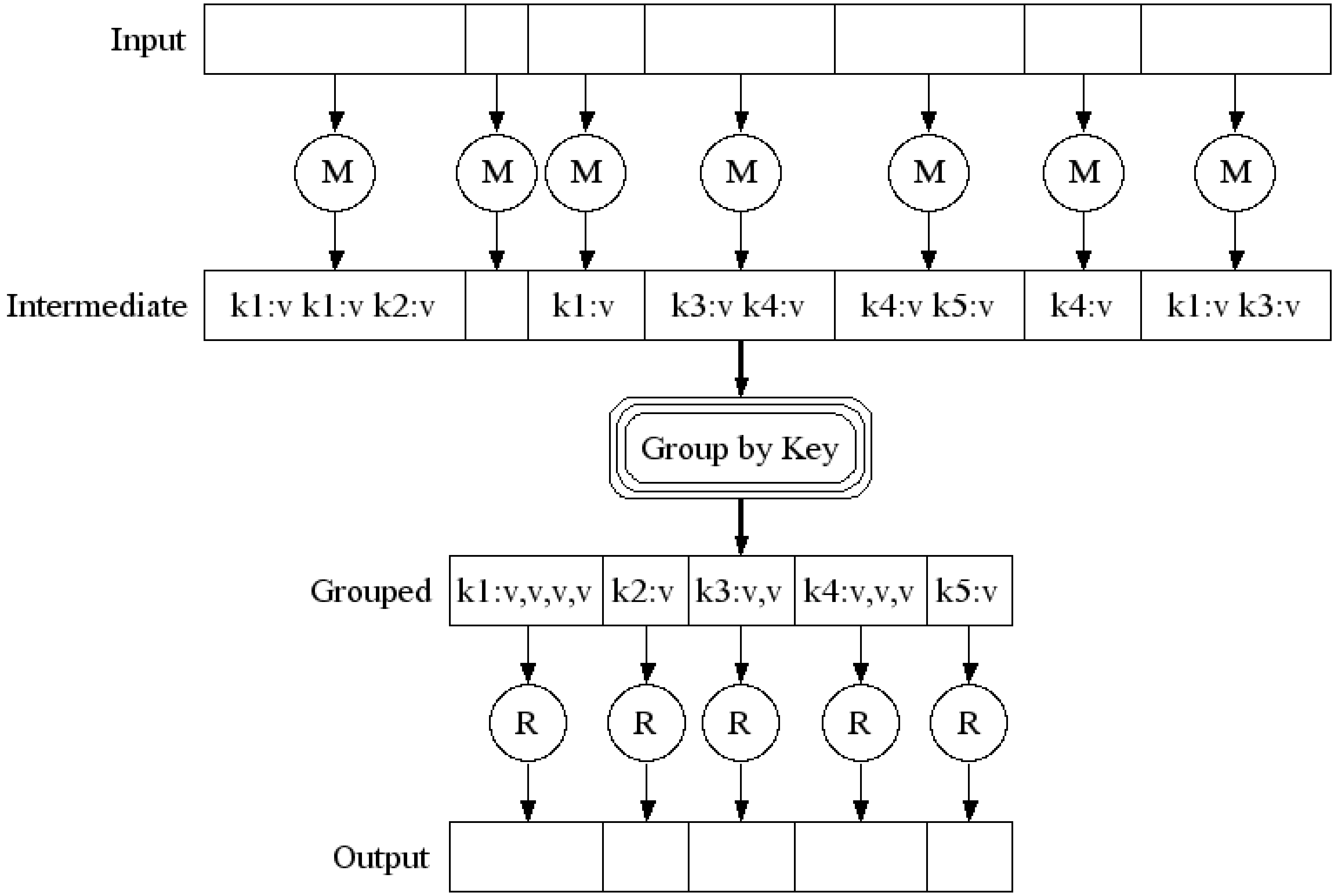


Map Reduce



- Map:
  - Accepts *input* key/value pair
  - Emits *intermediate* key/value pair
- Reduce :
  - Accepts *intermediate* key/value* pair
  - Emits *output* key/value pair

Partitioning Function



Example for MapReduce

- Page 1: the weather is good
- Page 2: today is good
- Page 3: good weather is good.

Map output

- Worker 1:
  - (the 1), (weather 1), (is 1), (good 1).
- Worker 2:
  - (today 1), (is 1), (good 1).
- Worker 3:
  - (good 1), (weather 1), (is 1), (good 1).

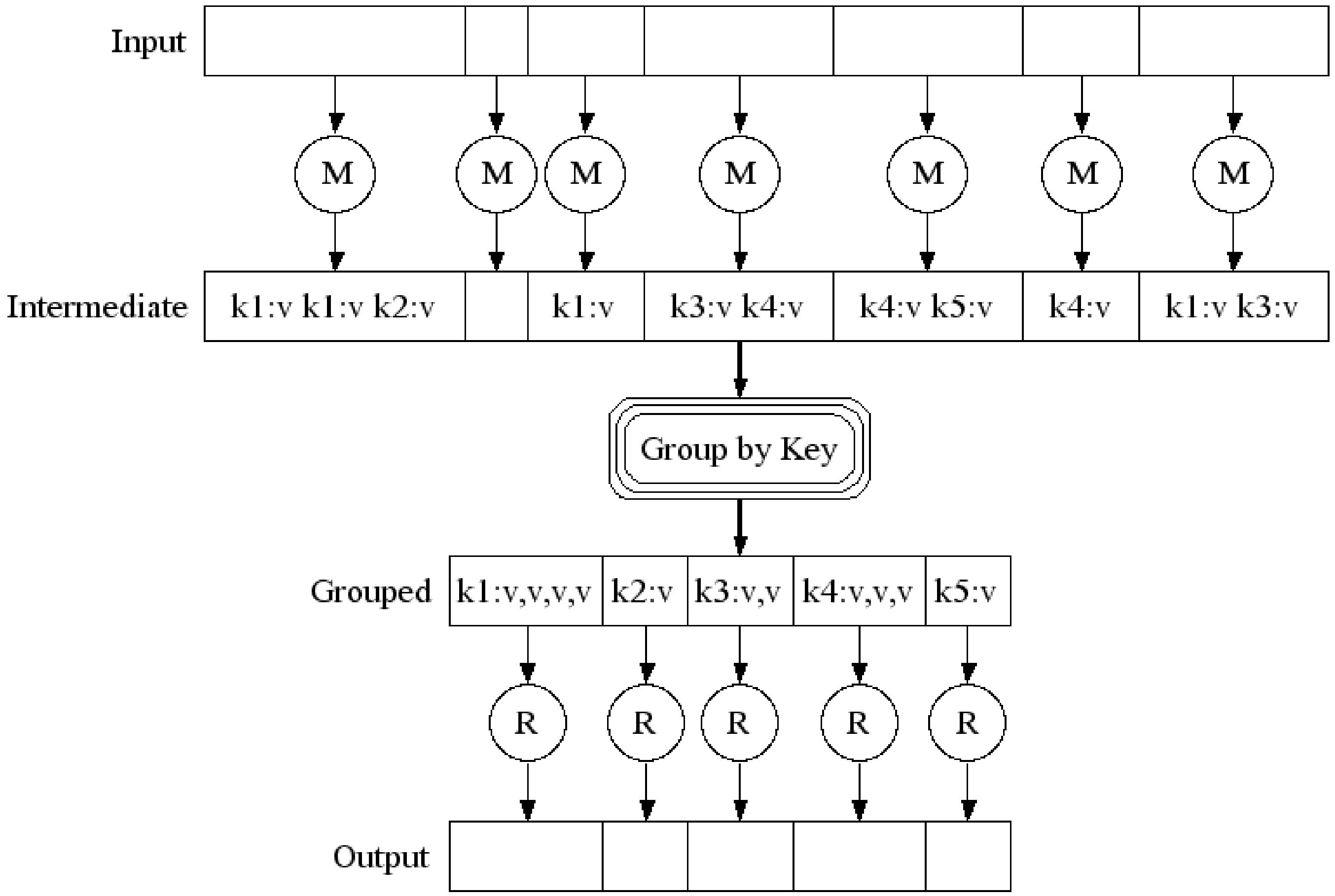
Reduce Input

- Worker 1:
  - (the 1)
- Worker 2:
  - (is 1), (is 1), (is 1)
- Worker 3:
  - (weather 1), (weather 1)
- Worker 4:
  - (today 1)
- Worker 5:
  - (good 1), (good 1), (good 1), (good 1)

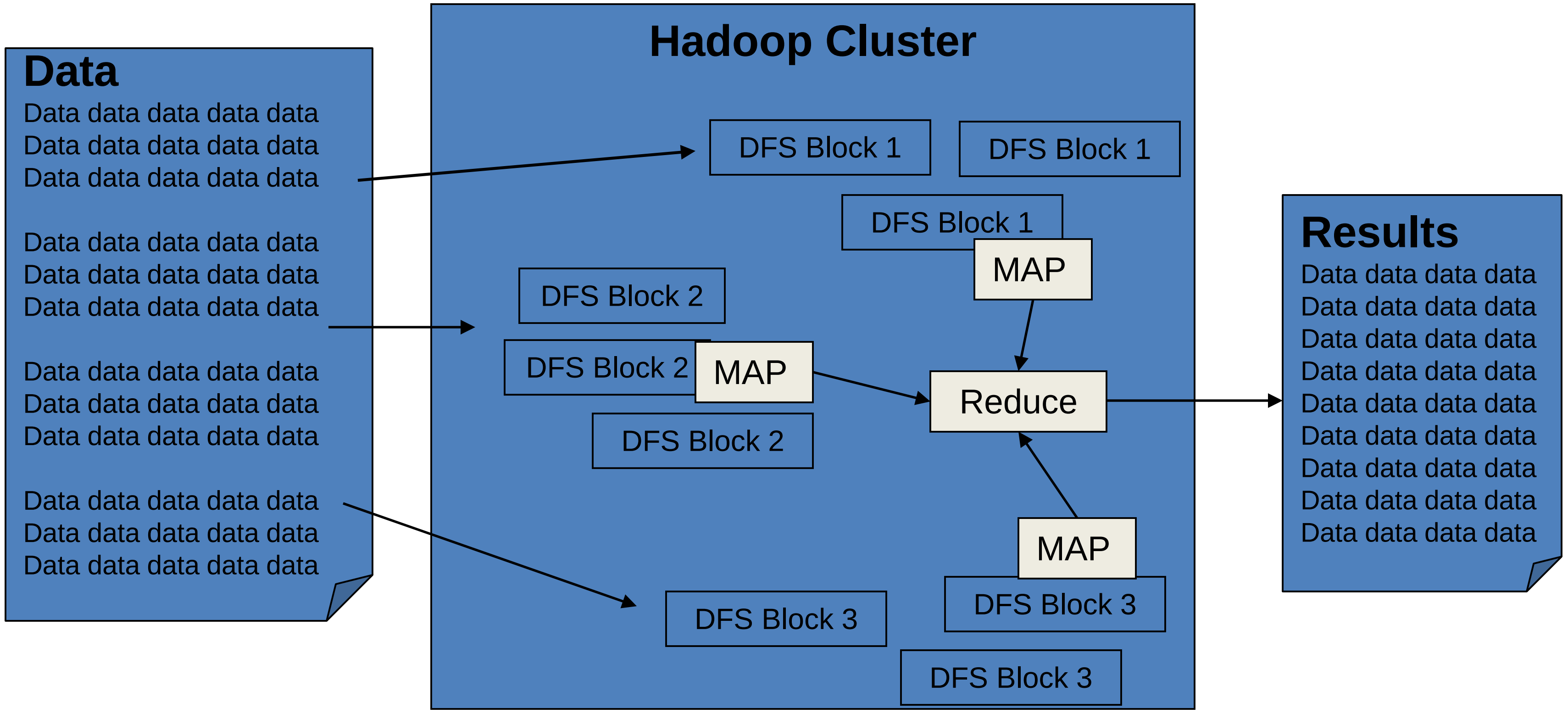
Reduce Output

- Worker 1:
  - (the 1)
- Worker 2:
  - (is 3)
- Worker 3:
  - (weather 2)
- Worker 4:
  - (today 1)
- Worker 5:
  - (good 4)

MapReduce Architecture



Hadoop Architecture



Sample Hadoop Code

- **Sample text-files as input:**

-
  ```
  $ bin/hadoop dfs -ls /usr/joe/wordcount/input/
  /usr/joe/wordcount/input/file01
  /usr/joe/wordcount/input/file02

  $ bin/hadoop dfs -cat /usr/joe/wordcount/input/file01
  Hello World, Bye World!

  $ bin/hadoop dfs -cat /usr/joe/wordcount/input/file02
  Hello Hadoop, Goodbye to hadoop.
  ```
- **Run the application:**

-
  ```
  $ bin/hadoop jar /usr/joe/wordcount.jar org.myorg.WordCount
  /usr/joe/wordcount/input /usr/joe/wordcount/output
  ```
- **Output:**

-
  ```
  $ bin/hadoop dfs -cat /usr/joe/wordcount/output/part-00000
  Bye 1
  Goodbye 1
  Hadoop, 1
  Hello 2
  World! 1
  World, 1
  hadoop. 1
  to 1
  ```

Contd…

- Notice that the inputs differ from the first version we looked at, and how they affect the outputs.
- Now, lets plug-in a pattern-file which lists the word-patterns to be ignored,
- $ hadoop dfs -cat /user/joe/wordcount/patterns.txt

```
\.
\,
\!
to
```

- Run it again, this time with more options:
- $ bin/hadoop jar /usr/joe/wordcount.jar org.myorg.WordCount –
- Dwordcount.case.sensitive=true /usr/joe/wordcount/input
- /usr/joe/wordcount/output -skip /user/joe/wordcount/patterns.txt
- As expected, the output:
- $ bin/hadoop dfs -cat /usr/joe/wordcount/output/part-00000

```
Bye 1
Goodbye 1
Hadoop 1
Hello 2
World 2
hadoop 1
```

Contd…

- Run it once more, this time switch-off case-sensitivity:

-
  ```
  $ bin/hadoop jar /usr/joe/wordcount.jar
  org.myorg.WordCount -Dwordcount.case.sensitive=false
  /usr/joe/wordcount/input  /usr/joe/wordcount/output  -skip
  /user/joe/wordcount/patterns.txt
  ```
- Sure enough, the output:

-
  ```
  $ bin/hadoop dfs -cat /usr/joe/wordcount/output/part-00000
  bye 1
  goodbye 1
  hadoop 2
  hello 2
  world 2
  ```