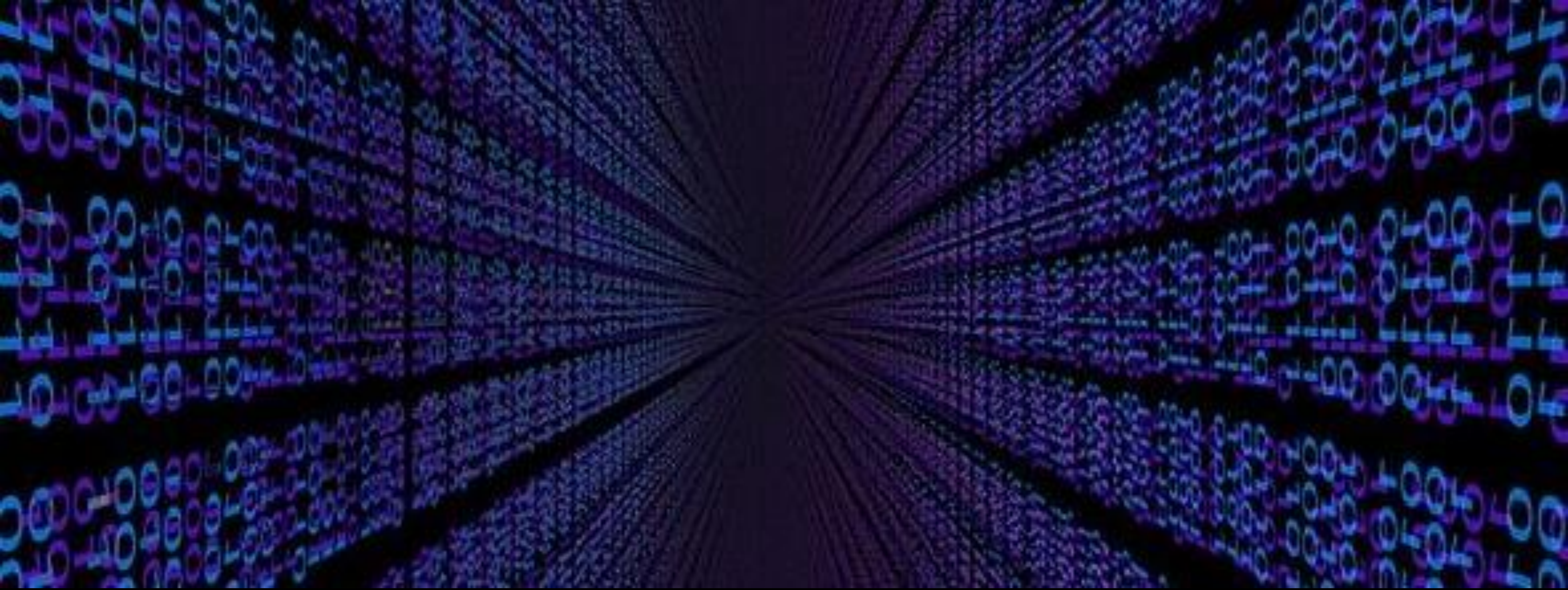


Intro to Apache Spark

XQuery Working Group

Text Mining at Scale

Fall 2019

What is it?

Apache Spark is an open-source distributed general-purpose cluster-computing framework

What does that even mean?

Break it down

- open-source (non-proprietary, works well with others)
- distributed (more than one computer, often remote, entails communication costs)
- general-purpose (can do any big data operation)
- cluster-computing (handle parallel programming)

MapReduce Model

- Grew out of needs of Google to process big data quickly
- Across a distributed parallel computer cluster

- TL; DR
  - Input data
  - Map (filtering and sorting operations)
  - Reduce (summary operations like counting)
  - Output data

MapReduce Model: Input

- Start with a set of key/value pairs

- Where have we seen key/value pairs?
  - JSON
  - Need to convert XML to JSON (or JSONL)

MapReduce Model: Map

Map the input pairs to a function and output intermediate pairs

map (input_key, input_value) -> list(output_key, intermediate_value)

Function typically a sorting or filtering function

Output an intermediate file

MapReduce Model: Shuffle

- Hidden phase where the intermediate output from the Map function is transferred to the Reduce function
- Apache Spark optimize the shuffle part, so you only have to focus on the map and reduce parts
- Apache Spark handles managing task scheduling and fault tolerance

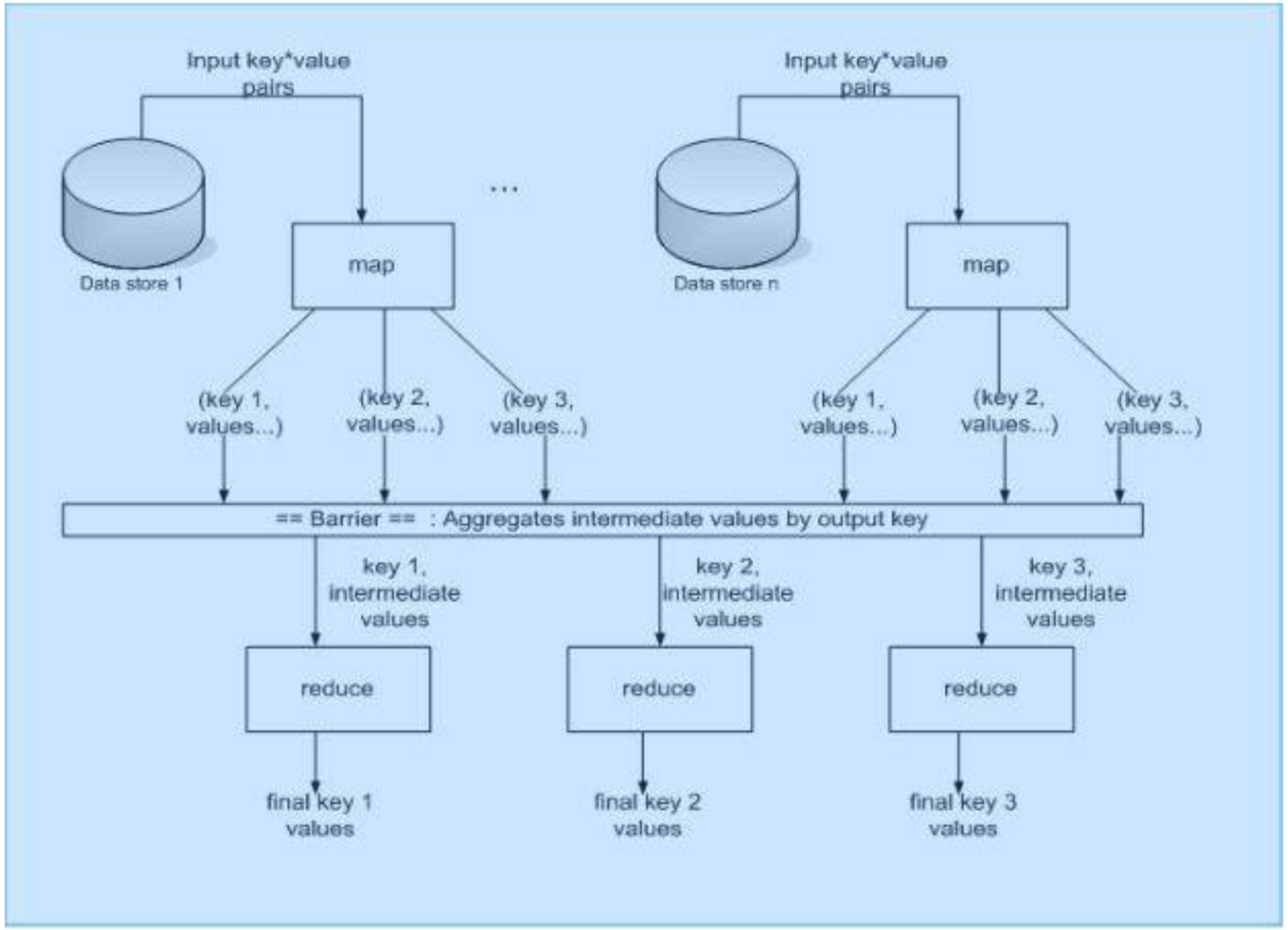
MapReduce Model: Reduce

Reduce the intermediate pairs to the output key/value pair

```
reduce (output_key, list(intermediate_value)) -> list(output_value)
```

Collect the intermediate outputs and perform a summary function on them up to produce a single result

Summary functions include sum, count, max, min, avg



https://mindmajix.com/mapreduce/googles-mapreduce-programming-model

Trade-offs

- Trade off between computation and communication costs
- Communication costs usually more expensive than computational costs

∴ Spark isn’t always the fastest, but handles big data the best

Pseudo-Code Example: Word Count

```
Map_function  (String name, String document):
    for each word w in document:
        return (w, 1)

Reduce function (word, partialCounts):
    sum = 0
    for each pc in partialCounts:
        sum += pc
        return (word, sum)
```

RumbleDB

- Can query big data on the Spark cluster
- Contained in JSONL files
- Using JSONiq query language
- And familiar FLWOR expressions

- http://rumbledb.org/

Language Game

https://namethatlanguage.org/ntl

Language Game Dataset

- 16 million records of each guess, one JSON record per line:

```
{ "target": "Turkish",
  "sample": "af0e25c7637fb0dcdc56fac6d49aa55e",
  "choices": [
    "Hindi",
    "Lao",
    "Maltese",
    "Turkish"
  ],
  "guess": "Maltese",
  "date": "2013-08-19",
  "country": "AU"
}
```