Big data platform (Spark) performance acceleration

*Mentors: Tony Tan, Ning Wu, Yong Wang and **Theo Gkountouvas***

By:

Grishma Atul Thakkar

Virat Goradia

Nipun Midha

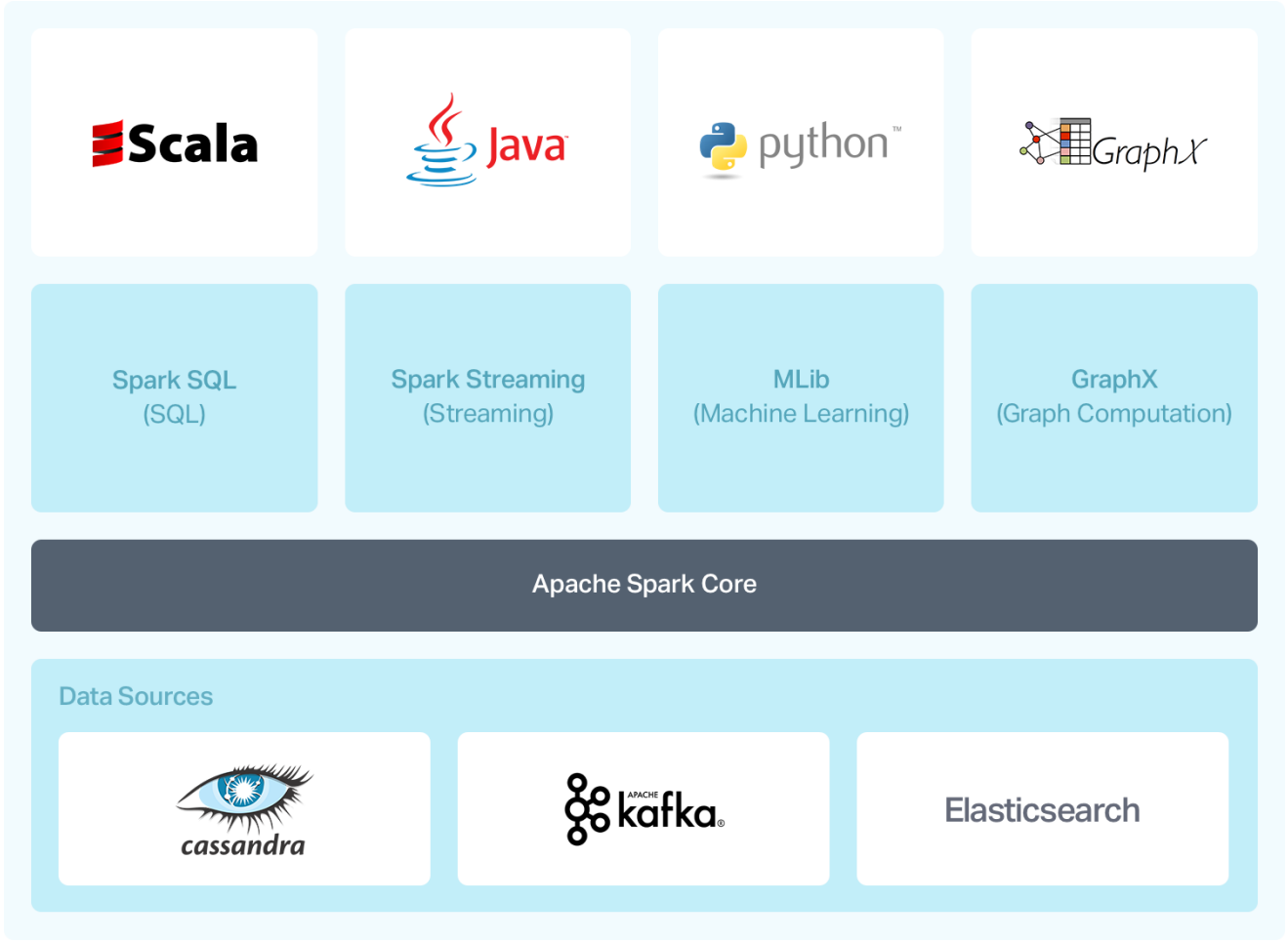
Baoshu Brady Qi

Sprint Goals

- Set-up environment
- Find the existing spark code
- Read and summarize riffle paper
- Research about the spark architecture and their phases: map, shuffle and reduce

What is Spark?

- Open-source distributed general-purpose cluster-computing framework
- Unified analytics engine for big data processing, with built-in modules for streaming, SQL, machine learning and graph processing.
- Provides an interface for programming entire clusters with implicit data parallelism and fault tolerance.

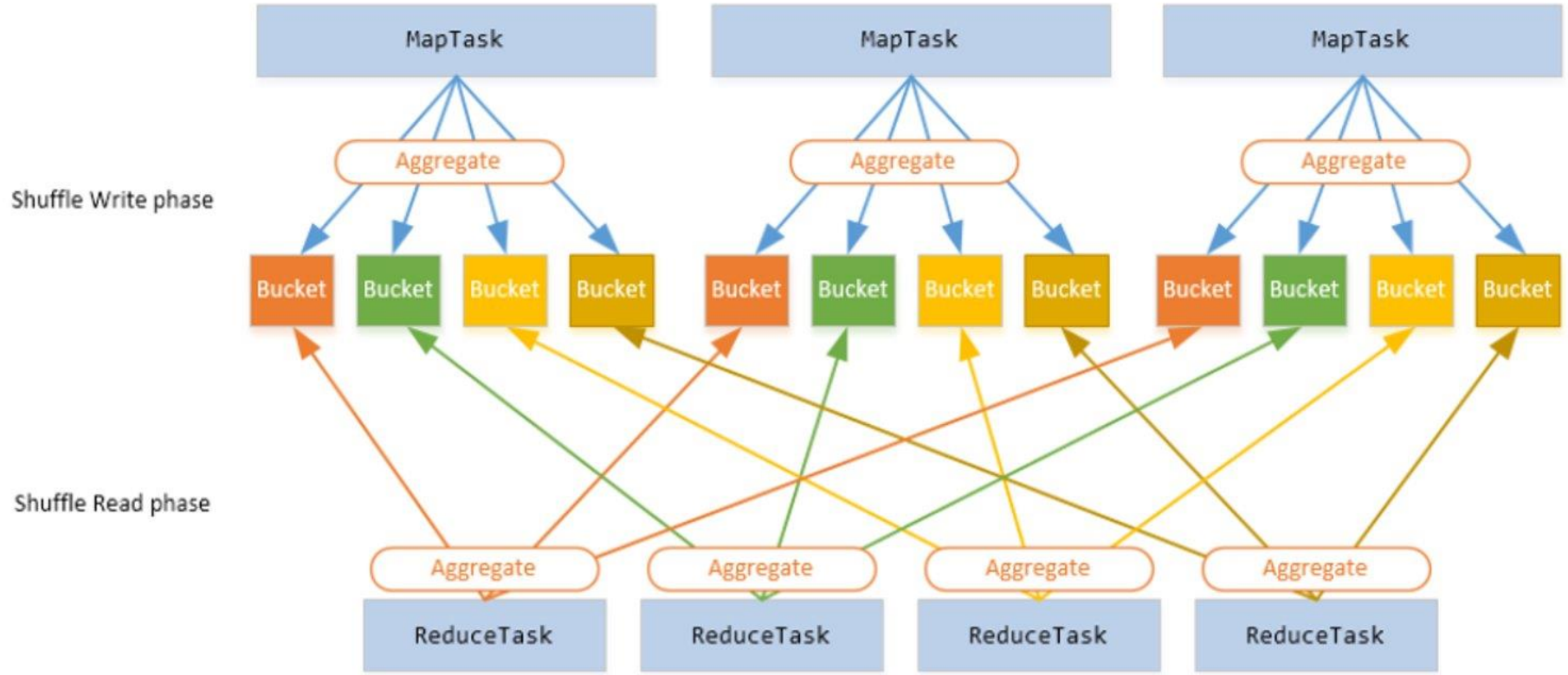


Key difference between MapReduce and Spark

- Processing: Spark can do it in-memory, while Hadoop MapReduce has to read from and write to a disk.

- As a result, the speed of processing differs significantly – Spark may be up to 100 times faster. However, the volume of data processed also differs: Hadoop MapReduce is able to work with far larger data sets than Spark.

How does Spark work?



Demo

```
val conf = new SparkConf().setAppName("Link Analysis")
val sc = new SparkContext(conf)
val textFile = sc.textFile(args(0))

val counts = textFile.map(line => line.split( regex = ",")(1))
  .map(folowee => (folowee, 1))
  .reduceByKey(_ + _)
```

	follower	followee
0	1	11553
1	1	8762940
2	1	8762941
3	1	688136
4	1	8762942

Duration	Tasks succeeded / total	Input	Output	Shuffle read	Shuffle write
10 s	20 / 20		77.1 MiB	90.7 MiB	
46 s	20 / 20	1.2 GiB			90.7 MiB

Stage ID	Status	Description
1	Completed	runJob at SparkHadoopWriter.scala:78
0	Completed	map at LinkAnalysis.scala:25

Current Issues

ALL-TO-ALL COMMUNICATION REQUIRED BETWEEN MAP AND REDUCE.

WAIT TIME OF REDUCE PHASE.

The **number of disk I/o operations done by the reduce phase equals the number of map outputs generated.**

Observations

WASTE OF TIME TO WAIT BY REDUCE PHASE.

ONE CAN MERGE **FRAGMENTED INTERMEDIATE SHUFFLE FILES** INTO LARGER **BLOCK FILES, AND THUS** CONVERT **SMALL, RANDOM DISK** I/O REQUESTS **INTO LARGE, SEQUENTIAL ONES.**

REDUCE WILL NOW ACCESS THESE MERGE OUTPUTS, WHICH CAN DECREASE THE NUMBER OF I/O OPERATIONS.

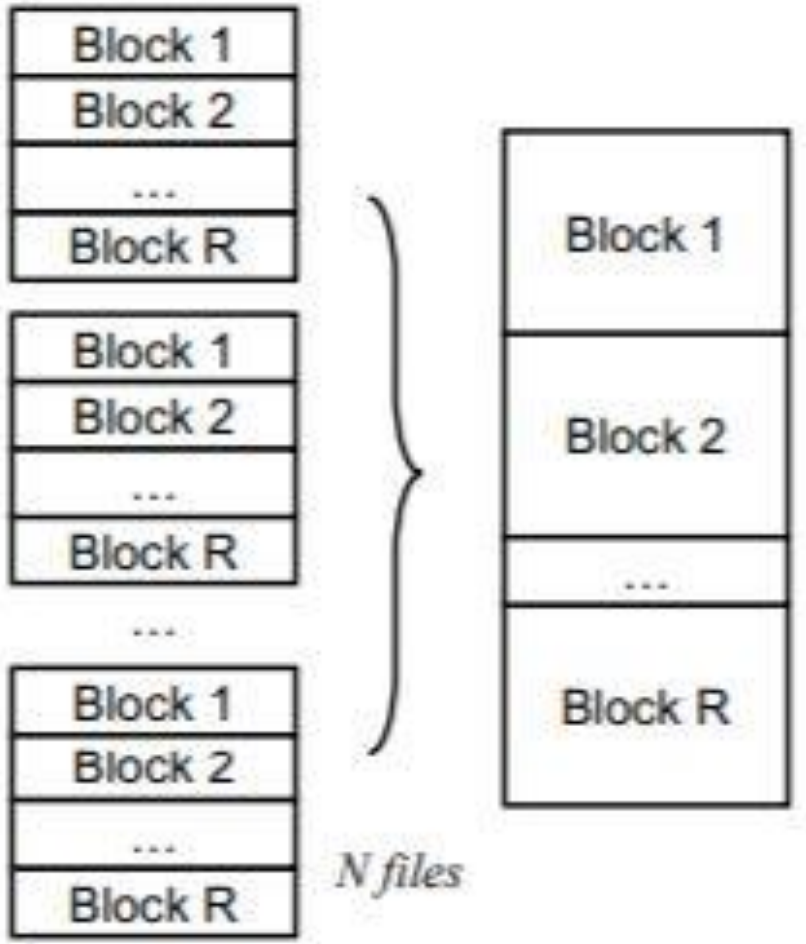
Riffle

RIFFLE IS THE SHUFFLE MERGE SERVICE THAT KEEPS TRACK OF INTERMEDIATE SHUFFLE FILES AND DYNAMICALLY COORDINATES MERGE OPERATIONS.

KEEPS TRACK OF TASK EXECUTION PROGRESS AND SCHEDULES MERGE OPERATIONS BASED ON CONFIGURABLE STRATEGIES AND POLICIES.

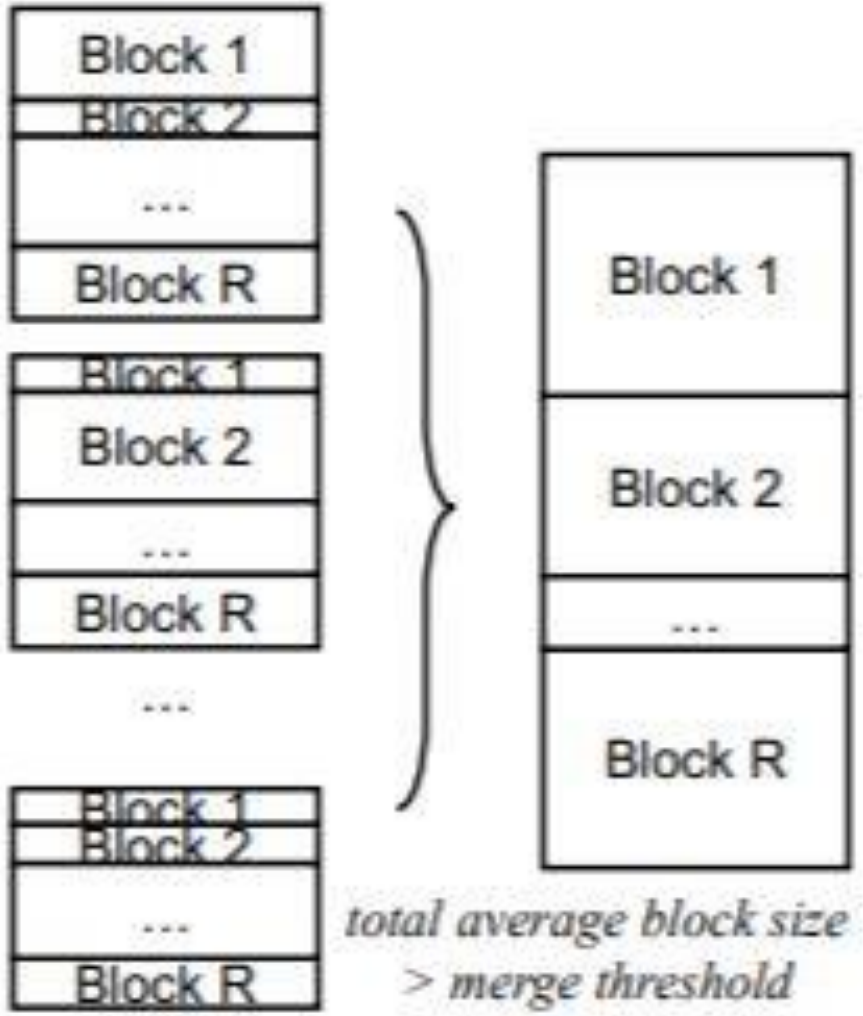
THE INTERMEDIATE FILES ARE SOON GARBAGE COLLECTED AFTER JOB COMPLETION, SO THEY OCCUPY DISK SPACE ONLY TEMPORARILY

Merge with fixed Number of file



(a) Greedy N-way Merge.

Merge with fixed block size



(b) Fixed Block Size Merge.

Next Sprint Goals

- Understand existing spark code
- Research on ways to implement the N-Way merge
- Find the appropriate data set
- Start implementing the N-Way merge algorithm

Any Questions?

Thank You!