

第七周领教直播 · Spark（下篇）

张语

目录

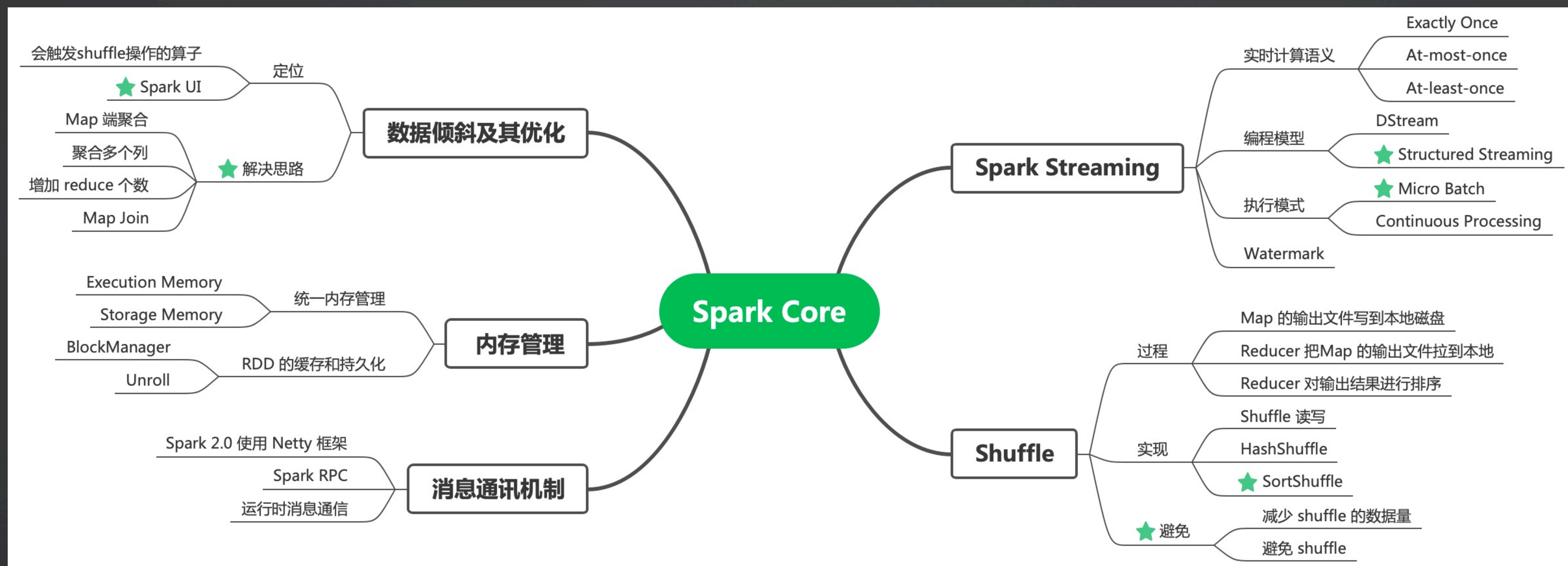
1 重点内容回顾

2 作业讲解

3 Structured Streaming

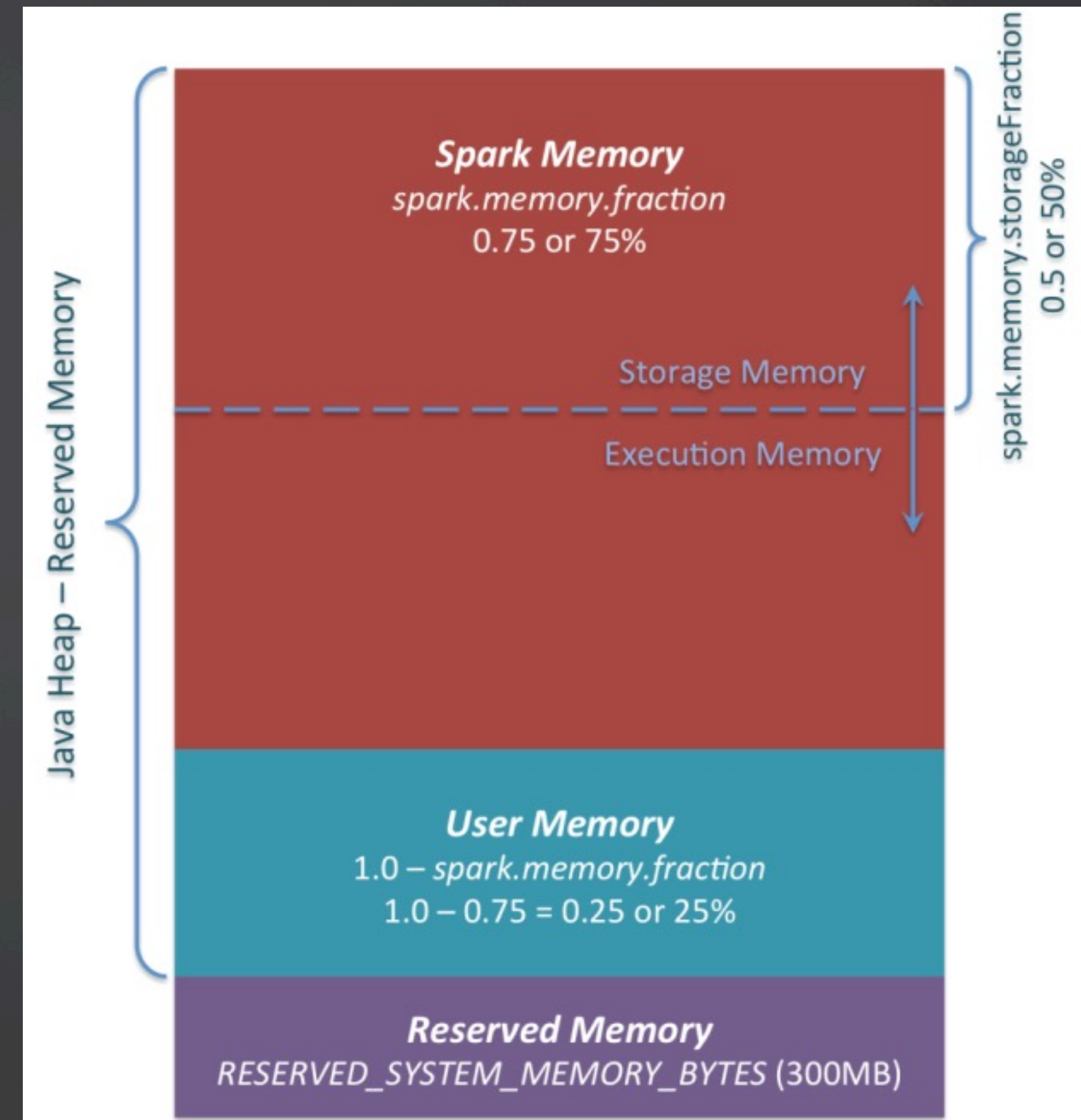
4 Spark UI

重点内容回顾



统一内存管理

- Reserved Memory: 存储 Spark 内部对象
- User Memory: 存储用户自定义的数据结构
- Execution Memory: 存储分布式任务执行, 如 Shuffle、Aggregate 等操作
- Storage Memory: 存储 RDD 缓存和广播变量



统一内存管理

Execution 和 Storage 的抢占规则

- 计算任务抢占的 Storage Memory，需要等任务执行完毕后释放
- RDD 缓存任务抢占的 Execution Memory，要立即释放归还

实时计算语义

Exactly-Once 指在流计算引擎中, 算子给下游的结果是 Exactly-Once 的（即给下游的结果有且仅有一个, 且不重复、不少算）。

目录

1 重点内容回顾

2 作业讲解

3 Structured Streaming

4 Spark UI

作业讲解

作业一：使用 RDD API 实现带词频的倒排索引

1. 读取文件集合，构建各文件和对应的单词集合
2. 构建每个单词到文件名
3. 按照单词聚合成到倒排索引

作业讲解

```
val linesWithFileNames = text.asInstanceOf[NewHadoopRDD[LongWritable, Text]]
    .mapPartitionsWithInputSplit((inputSplit, iterator) => {
        val file = inputSplit.asInstanceOf[FileSplit]
        iterator.map(tup => (file.getPath.toString.split(regex = "/").last, tup._2))
    })

val tempIndex = linesWithFileNames.flatMap {
    case (fileName, text) => text.toString.split(regex = "\r\n")
        .flatMap(line => line.split(regex = " "))
        .map { word => (word, fileName) }
}

val invertedIndex = tempIndex.groupByKey()

val group = invertedIndex.map {
    case (word, tup) =>
        val fileCountMap = scala.collection.mutable.HashMap[String, Int]()
        for (fileName <- tup) {
            val count = fileCountMap.getOrElseUpdate(fileName, 0) + 1
            fileCountMap.put(fileName, count)
        }
        (word, fileCountMap)
}.sortByKey().map(word => s"${word._1}:${word._2}")

group.repartition(numPartitions = 1).saveAsTextFile(outputPath)
```


作业讲解

作业二：Distcp 的 Spark 实现

DistCp（分布式复制）是一个用于大型集群间/集群内复制的工具

- 使用 MapReduce 来实现分布式、错误处、恢复、报告等特性
- 将文件和目录列表作为 Map 任务的输入，每个 Map 任务将复制源列表中指定的文件的一个分区

作业讲解

```
def checkDir(sparkSession: SparkSession, sourcePath: Path, targetPath: Path,
             fileList: ArrayBuffer[(Path, Path)], options: SparkDistCPOptions): Unit = {

  val fs = FileSystem.get(sparkSession.sparkContext.hadoopConfiguration)
  fs.listStatus(sourcePath)
    .foreach(currPath => {
      if (currPath.isDirectory) {
        val subPath = currPath.getPath.toString.split(sourcePath.toString)(1)
        val nextTargetPath = new Path(targetPath + subPath)
        try {
          fs.mkdirs(nextTargetPath)
        } catch {
          case ex: Exception => if (!options.ignoreFailures) throw ex else logWarning(ex.getMessage)
        }
        checkDir(sparkSession, currPath.getPath, nextTargetPath, fileList, options)
      } else {
        fileList.append((currPath.getPath, targetPath))
      }
    })
}
```


作业讲解

```
def copy(sparkSession: SparkSession, fileList: ArrayBuffer[(Path, Path)], options: SparkDistCPOptions): Unit = {  
    val sc = sparkSession.sparkContext  
    val maxConcurrenceTask = Some(options.maxConcurrenceTask).getOrElse(5)  
    val rdd = sc.makeRDD(fileList, maxConcurrenceTask)  
  
    rdd.mapPartitions(ite => {  
        val hadoopConf = new Configuration()  
        ite.foreach(tup => {  
            try {  
                FileUtil.copy(tup._1.getFileSystem(hadoopConf), tup._1, tup._2.getFileSystem(hadoopConf),  
                    tup._2, deleteSource = false, hadoopConf)  
            } catch {  
                case ex: Exception => if (!options.ignoreFailures) throw ex else logWarning(ex.getMessage)  
            }  
        })  
        ite  
    }).collect()  
}
```

目录

1 重点内容回顾

2 作业讲解

3 Structured Streaming

4 Spark UI

Triggers




触发器定义了流式数据处理的时间，代表多长时间产生一次结果。

触发类型	执行模型	描述
未指定	micro batch	框架决定切换粒度。上一个微批处理完成，产生下一个微批
固定间隔		按照固定时间，划分数据流
一次		一次处理所有数据流
固定检查点间隔	continuous processing	连续处理数据流

Triggers

```
/**
 * Policy used to indicate how often results should be produced by a [[StreamingQuery]].
 *
 * * @since 2.0.0
 */
@Evolving
public class Trigger {
```

Choose Subclass of **Trigger** (3 classes found)

-  ContinuousTrigger (org.apache.spark.sql.execution.streaming) spark-sql_2.12
-  OneTimeTrigger\$ (org.apache.spark.sql.execution.streaming) spark-sql_2.12
-  ProcessingTimeTrigger (org.apache.spark.sql.execution.streaming) spark-sql_2.12

时间语义

Event Time（事件时间）

- 事件实际发生的时间，以附加在数据流中事件的时间戳为依据。

Processing Time（处理事件）

- 当前流处理算子所在机器上的本地时钟时间，即消息到达流处理系统的时间。

为什么选 Event Time?

- 举例：处于弱网、无网的应用产生的数据，网络恢复后上传本地的数据。使用 Processing Time 计算可能是错的（取决于业务场景）。

Window 操作

Window 操作

- 基于事件时间或是处理时间，以固定间隔划定时间窗口，然后以窗口为粒度处理消息。

Tumbling Window（滚动窗口）

- 将事件分配到长度固定且互不重叠的桶中。不重不漏。

Sliding Window（滑动窗口）

- 将事件分配到大小固定且允许相互重叠的桶中。
- 两个参数：窗口大小、窗口间隔。

Watermark

需要等多久才能确定已经收到了所有发生在某个特定时间点之前的事件？

1. 客户端无法联网
2. 客户端到服务器的网络有阻塞
3. 客户端发送数据限速

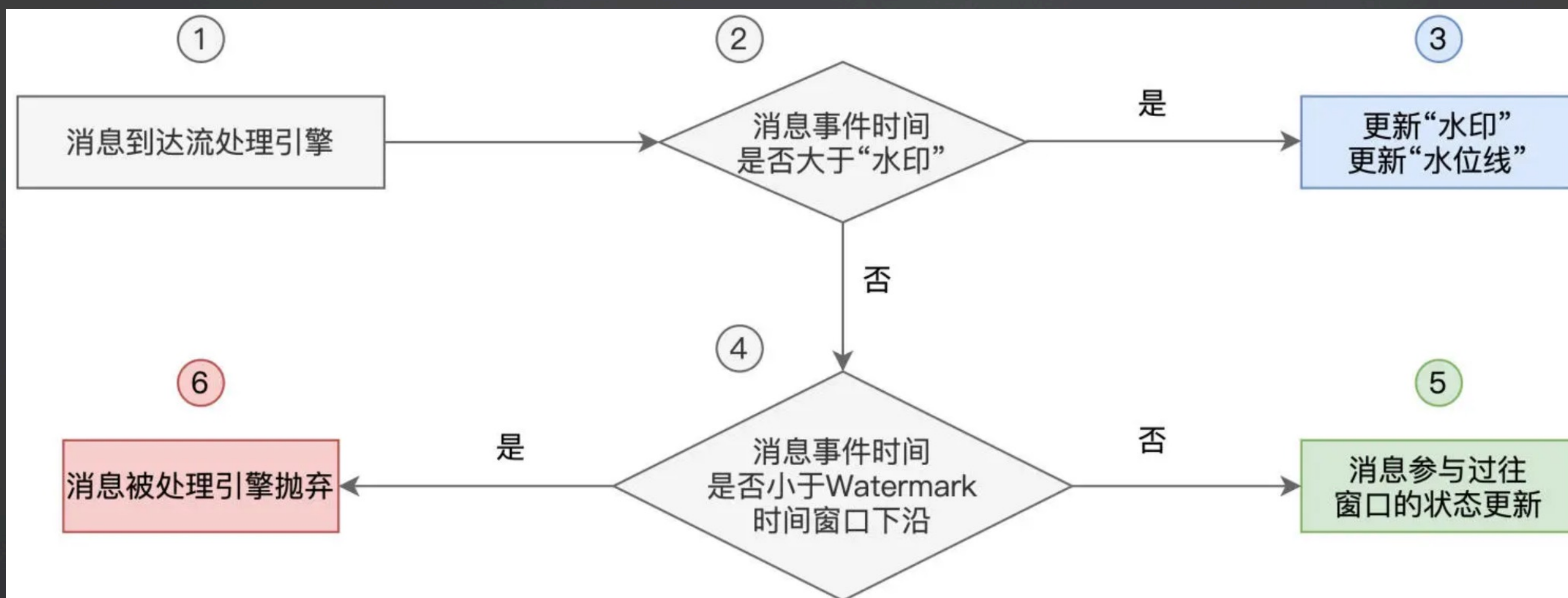
watermark（水位线），表示不会再有延迟事件到来的某个时间点。

当算子接收到时间为 T 的水位线，可以认为不会再收到任何时间戳小于等于 T 的事件了，此时可以触发窗口计算（action）。

水位线是结果的准确性和延迟之前的取舍。

Watermark

- 水印相当于系统当前接收到的所有消息中最大的事件时间。
- 水位线指的是水印对应的事件时间，减去用户设置的容忍值。



目录

1 重点内容回顾

2 作业讲解

3 Structured Streaming

4 Spark UI

Spark UI

- 通过 Yarn UI 跳转

Cluster

About

Nodes

Node Labels

Applications

NEW

NEW SAVING

SUBMITTED

ACCEPTED

RUNNING

FINISHED

FAILED

KILLED

Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Rese
2097	0	1	2096	1	3 GB	74.50 GB	0 B	1	64	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
8	0	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:32, vCores:1>	<memory:9536, vCores:8>	0

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking UI
application_1645699879292_2259	student3	org.apache.spark.examples.SparkPi	SPARK	default	0	Sat Apr 16 12:07:49 +0800 2022	Sat Apr 16 12:07:49 +0800 2022	N/A	RUNNING	UNDEFINED	1	1	3072	0	0	4.0	4.0	<div></div>	ApplicationMaster

- 通过 Spark History Server UI 跳转

 3.2.0

History Server

Event log directory: hdfs://emr-header-1.cluster-285604:9000/spark-history

Last updated: 2022-04-16 12:08:48

Client local time zone: Asia/Shanghai

Show 20 entries

Search:

Version	App ID	App Name	Attempt ID	Started	Completed	Duration	Spark User	Last Updated	Event Log
3.2.0	application_1645699879292_2259	Spark Pi	1	2022-04-16 12:07:53	2022-04-16 12:08:03	10 s	student3	2022-04-16 12:08:03	Download

Jobs

Jobs 页展示涉及的Actions 动作，以及与数据读取、移动有关的动作，每一个 Action 都对应着一个 Job。

Spark Jobs (?)

User: student3

Total Uptime: 27 s

Scheduling Mode: FIFO

Completed Jobs: 20

Event Timeline

☐ Enable zooming

Executors

Added

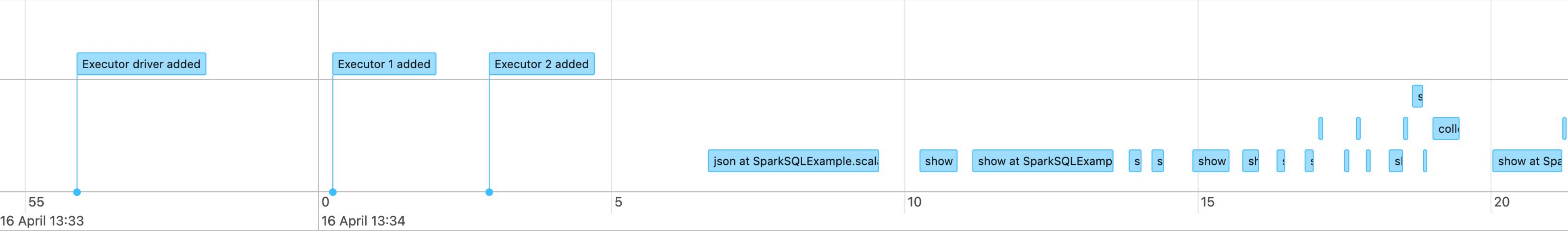
Removed

Jobs

Succeeded

Failed

Running



Completed Jobs (20)

Page: 1

1 Pages. Jump to 1. Show 100 items in a page. Go

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
19	show at SparkSQLExample.scala:263 show at SparkSQLExample.scala:263	2022/04/16 13:34:21	65 ms	1/1	<div>1/1</div>
18	show at SparkSQLExample.scala:263 show at SparkSQLExample.scala:263	2022/04/16 13:34:20	1 s	1/1	<div>1/1</div>

Job详情页

隶属于当前 Job 的所有 Stages。

Details for Job 19

Status: SUCCEEDED

Submitted: 2022/04/16 13:34:21

Duration: 65 ms

Associated SQL Query: 18

Completed Stages: 1

▶ Event Timeline

▶ DAG Visualization

▼ Completed Stages (1)


Page: 1

1 Pages. Jump to 1. Show 100 items in a page. Go

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
20	show at SparkSQLExample.scala:263 +details	2022/04/16 13:34:21	42 ms	1/1	16.0 B			

Stages

应用中涉及的所有 Stages，这些 Stages 分属于不同的作业。

APACHE  3.2.0

JobsStagesStorageEnvironmentExecutorsSQL

Spark SQL basic example application UI

Stages for All Jobs

Completed Stages: 20
Skipped Stages: 1

▼ Completed Stages (20)

Page: 11 Pages. Jump to 1. Show 100 items in a page. Go

Stage Id ▼	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
20	show at SparkSQLExample.scala:263	+details	2022/04/16 13:34:21	42 ms	1/1	16.0 B			
19	show at SparkSQLExample.scala:263	+details	2022/04/16 13:34:20	1 s	1/1	32.0 B			
18	collect at SparkSQLExample.scala:228	+details	2022/04/16 13:34:19	0.4 s	2/2	48.0 B			
17	show at SparkSQLExample.scala:215	+details	2022/04/16 13:34:18	45 ms	1/1	16.0 B			

Stage 详情页

Stage DAG

- 当前 Stage 的 DAG。

Event Timeline

- 记录着分布式任务调度与执行的过程中，不同计算环节主要的时间花销。

Summary Metrics（汇总指标）

- 所有 Tasks 执行细节的统计汇总。

Task Metrics（任务指标）

1. Locality level: 本地行级别
2. Logs: 执行日志
3. Errors: 执行错误细节

Storage

Storage 详情页，记录着每一个分布式缓存的细节，包括缓存级别、已缓存的分区数、缓存比例、内存大小与磁盘大小。

缓存级别（Storage_Level）：存储位置、是否序列化存储、是否将缓存数据进行备份

Storage						
▼ RDDs						
ID	RDD Name	Storage Level	Cached Partitions	Fraction Cached	Size in Memory	Size on Disk
1	rdd	Memory Serialized 1x Replicated	5	100%	236.0 B	0.0 B
4	LocalTableScan [count#7, name#8]	Disk Serialized 1x Replicated	3	100%	0.0 B	2.1 KiB

Environment

环境变量与配置项信息

- Runtime Information
- **Spark Properties**
- Resource Profiles
- Hadoop Properties
- System Properties
- Classpath Entries

Executors

- Summary: 所有 Executors 度量指标的简单加和。
- Executors: 每一个 Executor 的详情。

Executors

[Show Additional Metrics](#)

Summary

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Excluded
Active(3)	0	0.0 B / 2.8 GiB	0.0 B	2	0	0	22	22	9 s (0.6 s)	995 B	186 B	186 B	0
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0.0 ms (0.0 ms)	0.0 B	0.0 B	0.0 B	0
Total(3)	0	0.0 B / 2.8 GiB	0.0 B	2	0	0	22	22	9 s (0.6 s)	995 B	186 B	186 B	0

Executors

Show20entries

Search:

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Logs
driver	emr-worker-2.cluster-285604:36275	Active	0	0.0 B / 912.3 MiB	0.0 B	0	0	0	0	0	0.0 ms (0.0 ms)	0.0 B	0.0 B	0.0 B	stdout stderr
1	emr-worker-2.cluster-285604:36265	Active	0	0.0 B / 964.8 MiB	0.0 B	1	0	0	14	14	5 s (0.3 s)	680 B	186 B	186 B	stdout stderr

SQL

以 Actions 为单位，记录着每个 Action 对应的 Spark SQL 执行计划。

APACHE
Spark 3.2.0

JobsStagesStorageEnvironmentExecutors**SQL**

Spark SQL basic example application UI

SQL

Completed Queries: 19

▼ Completed Queries (19)

Page: 1 Pages. Jump to . Show items in a page.

ID ▼	Description		Submitted	Duration	Job IDs
18	show at SparkSQLExample.scala:263	+details	2022/04/16 13:34:19	1 s	[18] [19]
17	createOrReplaceTempView at SparkSQLExample.scala:256	+details	2022/04/16 13:34:19	11 ms	
16	collect at SparkSQLExample.scala:228	+details	2022/04/16 13:34:18	0.7 s	[17]

QA

THANKS