Spark企业级大数据项目实战 第4课

DATAGURU专业数据分析社区

法律声明

【声明】本视频和幻灯片为炼数成金网络课程的教学资料，所有资料只能在课程内使用，不得在课程以外范围散播，违者将可能被追究法律和经济责任。

课程详情访问炼数成金培训网站

http://edu.dataguru.cn

炼数成金逆向收费式网络课程

- Dataguru（炼数成金）是专业数据分析网站，提供教育，媒体，内容，社区，出版，数据分析业务等服务。我们的课程采用新兴的互联网教育形式，独创地发展了逆向收费式网络培训课程模式。既继承传统教育重学习氛围，重竞争压力的特点，同时又发挥互联网的威力打破时空限制，把天南地北志同道合的朋友组织在一起交流学习，使到原先孤立的学习个体组合成有组织的探索力量。并且把原先动辄成千上万的学习成本，直线下降至百元范围，造福大众。我们的目标是：低成本传播高价值知识，构架中国第一的网上知识流转阵地。
- 关于逆向收费式网络的详情，请看我们的培训网站 http://edu.dataguru.cn

本课内容

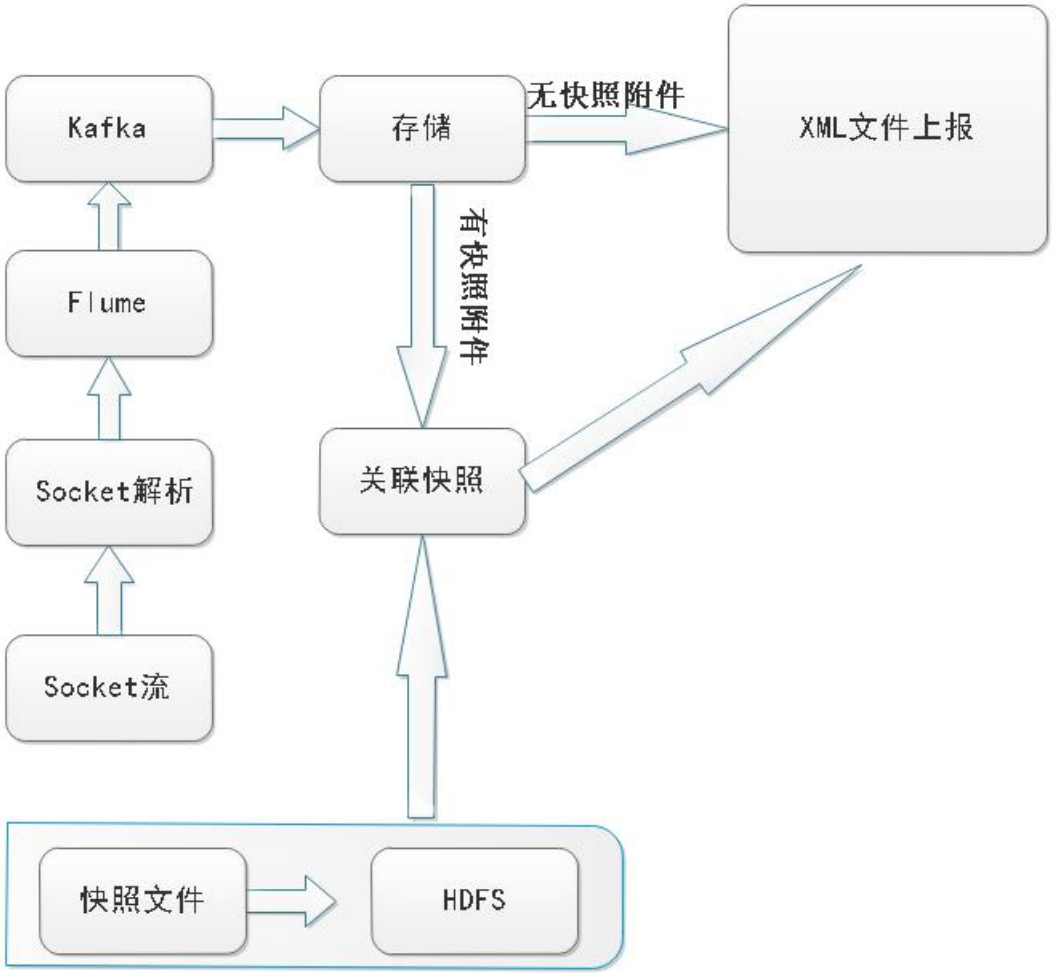
- Spark Streaming消费数据反写Kafka
- 实际生产使用存在问题分析解决
- 消息传递语义（Exactly-once、At-least-once、At-most-once）
- Spark Streaming实现Exactly-once的案例

Spark Streaming消费数据反写Kafka

存储选择使用Kafka

1. 创建读取Kafka的数据流
2. 清洗数据
3. 数据反写Kafka

问题：
1. 性能问题
2. Kafka生产者对象序列化问题
...



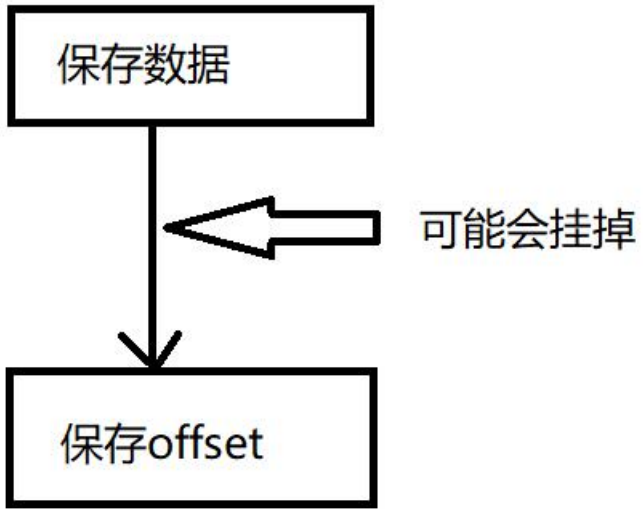
实际生产使用存在问题分析

1. kafka的保存的offset过期， 如何解决

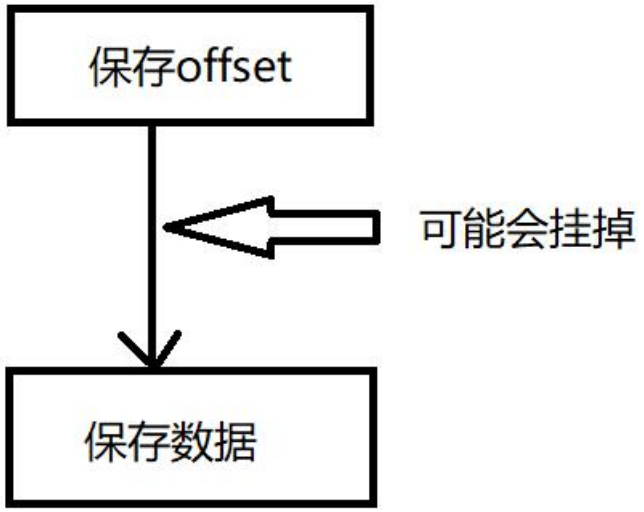
2. 数据峰值期间如何限速

```
    at org.apache.spark.streaming.scheduler.JobScheduler$JobHandler$$anonfun$run$1.apply(JobScheduler.scala:224)
    at org.apache.spark.streaming.scheduler.JobScheduler$JobHandler$$anonfun$run$1.apply(JobScheduler.scala:224)
    at scala.util.DynamicVariable.withValue(DynamicVariable.scala:57)
    at org.apache.spark.streaming.scheduler.JobScheduler$JobHandler.run(JobScheduler.scala:223) <2 internal calls>
    at java.lang.Thread.run(Thread.java:748)
Caused by: kafka.common.OffsetOutOfRangeException <4 internal calls>
    at java.lang.Class.newInstance(Class.java:442)
    at kafka.common.ErrorMapping$.exceptionFor(ErrorMapping.scala:102)
    at org.apache.spark.streaming.kafka.KafkaRDD$KafkaRDDIterator.handleFetchErr(KafkaRDD.scala:184)
    at org.apache.spark.streaming.kafka.KafkaRDD$KafkaRDDIterator.fetchBatch(KafkaRDD.scala:193)
    at org.apache.spark.streaming.kafka.KafkaRDD$KafkaRDDIterator.getNext(KafkaRDD.scala:208)
    at org.apache.spark.util.NextIterator.hasNext(NextIterator.scala:73)
    at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:327)
    at scala.collection.Iterator$class.foreach(Iterator.scala:727)
    at scala.collection.AbstractIterator.foreach(Iterator.scala:1157)
    at org.apache.spark.rdd.RDD$$anonfun$foreach$1$$anonfun$apply$32.apply(RDD.scala:912)
    at org.apache.spark.rdd.RDD$$anonfun$foreach$1$$anonfun$apply$32.apply(RDD.scala:912)
    at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1869)
    at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1869)
    at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:66)
    at org.apache.spark.scheduler.Task.run(Task.scala:89)
    at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:214)
    ... 3 more
Exception in thread "main" org.apache.spark.SparkException: Job aborted due to stage failure: Task 1 in stage 0.0 failed 1
    at java.lang.Class.newInstance(Class.java:442)
```

消息传递语义



At-least-once
失败重启， 数据会重复



At-most-once
失败重启，数据丢失

消息传递语义（Exactly-once）

1. 幂等写入（ idempotent writes）

需要设置好唯一主键等
比如每次往一个目录覆盖写数据
主键不容易获取

2. 事务控制

保存数据和offset在同一个事务里面
需要事务存储的支持

3. 自己实现Exactly-once

offset和数据绑定保存等

Exactly-once 案例实现

Thanks

FAQ时间