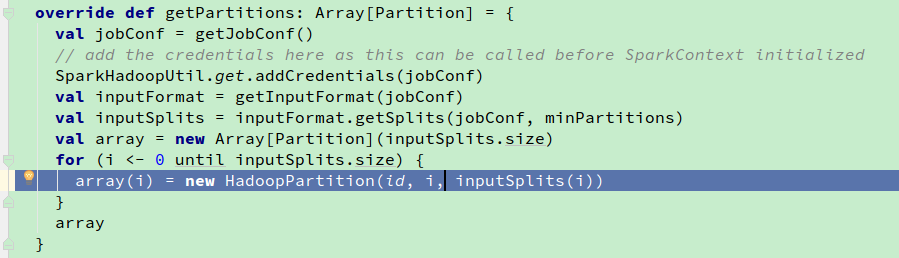
Spark HDFS Splits and Partition

**一:Spark中Partition的概念与Hadoop中的Splits的概念是类似的**

**都是计算层的逻辑概念，而Block是存储层面的概念。**

首先明确：多少个split 其实就对应了多少个Partition



**sc.textFile(path,minPartitions)**

注意对minPartitions的理解,并不代表最后partition个数

FileInputFormat.getSplits() 得到的splits个数才是最终个数

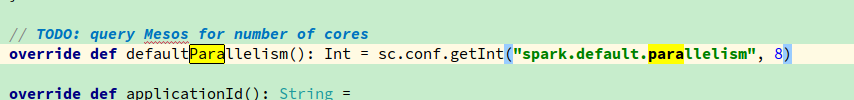
nimPartitions有默认值：

math.*min*(defaultParallelism, 2)

**针对不同部署模式，defaultParallelism 有不同的值：**

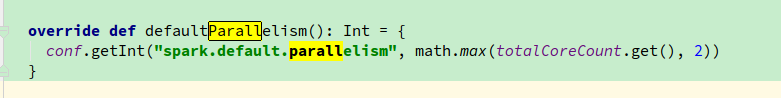
Local mode: number of cores on the local machine

Mesos fine grained mode: 8



Others: total number of cores on all executor nodes or 2, whichever is larger

YarnSchedulerBackend 继承自CoarseGrainedSchedulerBackend

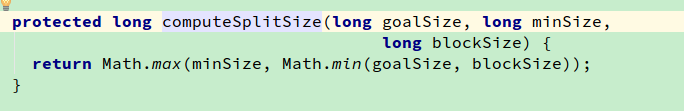


总结一下：一般来讲如何配置了defaultParallelism,则使用默认值，否则，各部署模式得到的值大于一般来讲会2，所以，math.min(defaultParallelism,2) 一般会取2，

也就是说使用textFile加载文件minPartitions 的默认值是2.

1. **下面深入分析Splits个数的计算：**

**首先确定split size:**



其中：numSplits:输入partitions的个数

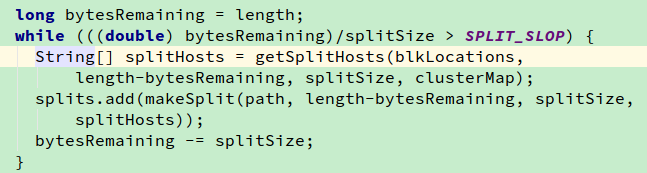
totalsize:所有输入文件的大小

goalSize = totalSize / (numSplits == 0 ? 1 : numSplits);

minSize: just a hadoop conf property

blockSize: this file block size

然后：判断文件是否可以split，如果是进入下面的while循环，否则，直接作为一个split

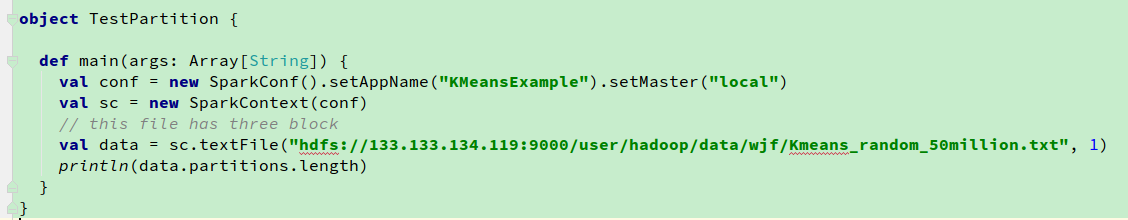


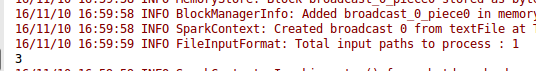
总结：如果想增大partition 大小，可以有两种方式：

1提供一个更小的minPartitions，从而得到更大的goalSize, 但是计算splitSize的时候，使用了min(goalSize,blockSize),所以该方式不可能得到太大的partition

2：直接配置更大的Split minSize，该方法没有限制。理论上可以任意大。

实验验证：

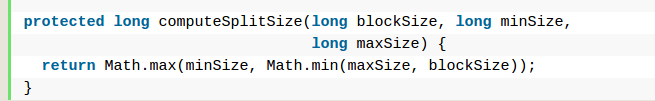




三．如果Partition 个数大于Block 个数，那么能否保证顺序，切分Block 还是如何处理？

~~多个输入文件，按照文件进行处理，不够一个Split 算一个Split~~

~~Split size 计算如下，这个是hadoop老板了~~



实验验证：

Situation1: with minPartition = 2

顺序是一致的

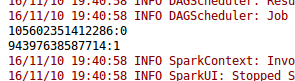


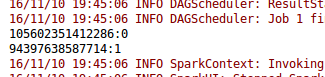




Situation 2: with minPartition = 1

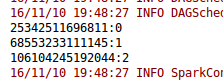
与sit1比较虽说都是两个Partition ,但是顺序并不一样。

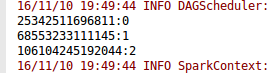


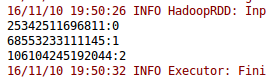


Situation 3:with minPartition = 3

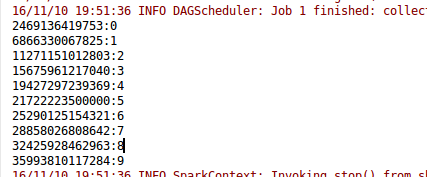
结果一致

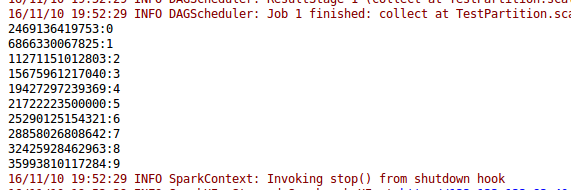


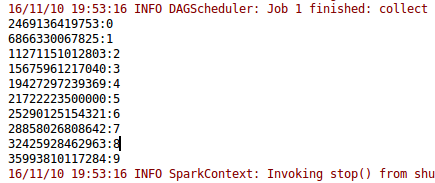




Situation 3: with minPartition = 10







结论：使用textFile读取 HDFS 文件：

* 只要设置了相同的minPartition,多次运行各个Partition结果是一致的。
* 对于minPartition 小于真实的Splits个数的情况，设置和不设置结果不一致。