**Spark Tuning**

|  |  |
| --- | --- |
| **审核人** |  |
| **重要性** | 中 |
| **紧迫性** | 中 |
| **拟制人** | 张包峰 |
| **提交日期** |  |

**张包峰**

(版权所有,翻版必究)

**修改记录**

|  |  |  |
| --- | --- | --- |
| **更新时间** | **变更内容** | **变更理由** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

[**张包峰** 1](#_Toc393567549)

[1.1 文件读取 4](#_Toc393567550)

[1.2 本地调试 4](#_Toc393567551)

[1.3 RDD API 4](#_Toc393567552)

[1.4 Standalone模式 6](#_Toc393567553)

## 文件读取

sc.textFile读取HDFS目录下多个文件，默认根据blocksize，切分partition数目。如果都是小文件，那么RDD的partition数目和文件数相等。

使用coalesce(numPartitions)来减小分区数

使用repartition(numPartitions)来增大分区数，

本质上是coalesce(numPartitions, shuffle=true)

如果文件数目很多，可以适当调大spark.cores.max

Hdfs的path String是支持glob pattern的(详见FileInputFormat和Path类)。

## 本地调试

Window下是可以调试的，以local的模式在代码里运行。

需要把assembly包依赖进来。

另外，设置好SCALA\_HOME和SPARK\_HOME，把assembly包放到

$SPARK\_HOME/assembly/target/scala-2.10

目录下，在cmd里执行

$SPARK\_HOME/bin/spark-shell.cmd –-master local

也是可以在windows环境下执行repl的。

## RDD API

结合FlumeJava，体会哪些是最基础的原语，衍生出了哪些新原语。

Count来自。。

Join来自。。。

Top来自。。。

**coalecse说明**

repartition是带shuffle的coalesce，用于增大分区数

coalesce一般用于减小分区数，减小分区的时候是不需要shuffle的

coalesce在实现时候，

如果是带shuffle的，先通过RDD.mapPartitionsWithIndex方法，把partition进行新的分布，在coalesce里是为已有分区分配一个新的随机位置。借助这个新的分区分配结果和HashPartitioner，new一个ShuffledRDD，传给CoalesceRDD。

不带shuffle的情况下，直接传给CoalesceRDD。

这块需要再仔细看

|  |  |  |  |
| --- | --- | --- | --- |
| transforms | RDD子类 | 计算方式 | 细节 |
| map() | MappedRDD | 迭代每个分区，做一次map(f) |  |
| flatMap() | FlatMappedRDD | 迭代每个分区，做一次flatMap(f) |  |
| filter() | FilteredRDD | 迭代每个分区，做一次filter(f) |  |
| distinct() |  | 先map成(key, null)，然后执行一次reduceByKey() |  |
| coalesce() | CoalesceRDD  MapPartitionsRDD |  | 带shuffle或不带 |
| sample() | PartitionwiseSampledRDD | 指定不同的Sampler来做抽样 |  |
| randomSplit() |  |  | 特殊的sample |
| union() | UnionRDD |  |  |
| intersection(rdd) | CoGroupedRDD | 把两个rdd都map成(v, null)进行一次cogroup，最后filter一下 | PairRDDFunction里面 |
| glom() | GlommedRDD | 每个分区的数据都toArray |  |
| cartesian() | CartesianRDD | 双重循环两个rdd的迭代器，然后yield每组结果 |  |
| groupBy() |  |  | PairRDDFunction里面groupByKey |
| pipe() | PipedRDD |  |  |
| zip() | ZippedRDD | 遍历两个rdd的迭代器，进行zip()操作 | 两个RDD的分区数目要相同 |
|  |  |  |  |
| Actions |  |  |  |
| foreach() |  | sc.runJob |  |
| collect() |  | sc.runJob | 跑完进行一次Array.concat |
| sybtract() | SubtractedRDD |  | PairRDDFunction里面 |
| reduce() |  | sc.runJob |  |
| fold() |  | sc.runJob |  |
| aggregate() |  |  |  |
| count() |  |  |  |
| take() |  |  |  |
| top() |  |  |  |
| max() / min() |  |  |  |
| saveAsXXXX() |  |  |  |

PairRDDFunction里面的处理逻辑需要具体看

## Standalone模式