Spark Transformation和Action算子速查表

Transformation算子

Transformation算子	作用
map(func)	返回一个新的分布式数据集，其中每个元素都是由源RDD中每一个元素经过func函数转换得到的
filter(func)	返回一个新的数据集，其中包含的元素来自源RDD中元素经过func函数过滤后的结果（func函数返回true的结果）
flatMap(func)	类似于map, 但是每个元素可以映射到0到n个输出元素(func函数必须返回的是一个Seq而不是单个元素)
mapPartitions(func)	类似于map, 但是它是基于RDD的每个Partition(或者数据block)独立运行，所以如果RDD包含元素类型为T，则func函数必须是Iterator => Iterator 的映射函数
mapPartitionsWithIndex(func)	类似于mapPartitions，只是func多了一个整型的分区索引值，因此如果RDD包含元素类型为T，则func必须是Iterator => Iterator的映射函数
mapWith(func)(func)	mapWith是map的另外一个变种，map只需要一个输入函数，而mapWith有两个输入函数, 第一个函数是把RDD的partition index（index从0开始）作为输入，输出为新类型A；第二个函数f是把二元组(T, A)作为输入（其中T为原RDD中的元素，A为第一个函数的输出），输出类型为U
flatMapWith(func)(func)	与mapWith很类似，都是接收两个函数，一个函数把partitionIndex作为输入，输出是一个新类型A；另外一个函数是以二元组（T,A）作为输入，输出为一个序列，这些序列里面的元素组成了新的RD
mapValues(func)	mapValues顾名思义就是输入函数应用于RDD中Kev-Value的Value，原RDD中的Key保持不变，与新的Value一起组成新的RDD中的元素。因此，该函数只适用于元素为KV对的RDD。
flatMapValues(func)	flatMapValues类似于mapValues，不同的在于flatMapValues应用于元素为KV对的RDD中Value。每个一元素的Value被输入函数映射为一系列的值，然后这些值再与原RDD中的Key组成一系列新的KV对。
sample(withReplacement, fraction, seed)	采样部分(比例取决于fraction)数据，同时可以指定是否使用回置采样(withReplacement)，以及随机数种子(seed)
union(otherDataset)	返回源数据集和参数数据集(otherDataset)的并集
intersection(otherDataset)	返回源数据集和参数数据集(otherDataset)的交集
distinct([numTasks])	返回对源数据集做元素去重后的新的数据集
groupByKey([numTasks])	必须应用于键值对的元素类型，如源RDD包含(K,V)对，则该算子返回一个新的数据集包含(K, Iterator)对

Transformation算子	作用
reduceByKey(func, [numTasks])	如果源RDD包含元素类型为(K,V)对，则该算子也返回包含(K, V)对的RDD， 只不过每个Key对应的Value是经过func函数聚合后的结果，而func函数本身是一个(V, V) => V的映射函数。 另外，和groupByKey类似，可以通过可选参数numTasks指定reduce任务的个数
aggregateByKey(zeroValue)(seqOp, combOp, [numTasks])	如果源RDD包含(K, V)对，则返回的新RDD包含(K, V)对，其中每个Key对应的Value都是由combOp函数和一个”0”值zeroValue聚合得到。允许聚合后Value类型和输入Value类型不同，避免了不必要的开销。和groupByKey类似，可以通过可选参数numTasks指定reducer任务的个数
sortByKey([ascending], [numTasks])	如果源RDD包含元素类型(K, V)对，其中K可以排序，则返回新的RDD包含(K, V)对，并按照K进行排序(由ascending参数决定是升序还是降序)
sortBy(func,[ascending], [numTasks])	类似于sortByKey，只不过sortByKey只能够按照key去排序，sortBy会更加灵活，既可以按照key，也可以按照value排序。
join(otherDataset, [numTasks])	如果源RDD包含元素类型(K, V)且参数RDD(otherDataset)包含元素类型(K, W), 则返回的新RDD中将包含内联后Key对应的(K, (V, W)对。外关联(Outer joins)操作请参考leftOuterJoin、rightOuterJoin以及fullOuterJoin算子)
cogroup(otherDataset, [numTasks])	如果源RDD包含元素类型(K, V)且参数RDD(otherDataset)包含元素类型(K, W)，则返回的新的RDD中包含(K, (Iterable, Iterable))。该算子还有个别名：groupWith
cartesian(otherDataset)	如果源RDD包含元素类型T且参数RDD(otherDataset)包含元素类型U，则返回的新RDD包含前二者的笛卡尔积，其元素类型为(T, U)对
pipe(command, [envVars])	以shell命令行管道处理RDD的每个分区，如：Perl或者bash脚本。RDD中每个元素都将依次写入进程的标准输入(stdin)，然后按行输出到标准输出(stdout)，每一行输出字符串即成为一个新的RDD元素
coalesce(numPartitions)	将RDD的分区数减少到numPartitons。当以后大数据集被过滤成小数据集后，减少分区，可以提升效率
repartition(numPartitions)	将RDD数据重新混洗(reshuffle)并随机分步到新的分区中，使数据分布更均衡，新的分区个数取决于numPartitions。该算子总是需要通过网络混洗所有数据。
repartitionAndSortWithPartitions(partitioner)	根据Partitioner(spark自带有HashPartioner和RangePartitoner等)重新分区RDD，并且在每个结果分区中按Key做排序。这是一个组合算子，功能上等价于先reparation再在每个分区内排序，但这个算子内部做了优化(将排序过程下推到混洗同时进行)，因此性能更好

Transformation算子	作用
randomSplit(Array[Double],Long)	该函数根据weights权重，将一个RDD切分成多个RDD。该权重参数为一个Double数组第二个参数为random的种子，基本可忽略
glom()	该函数是将RDD中每一个分区中类型为T的元素转换成Array[T]，这样每一个分区就只有一个数组元素。
zip(otherDataset)	用于将两个RDD组合成Key/Value形式的RDD,这里默认两个RDD的partition数量以及元素数量都相同，否则会抛出异常。
partitionBy(Partitioner)	该函数根据partitioner函数生成新的ShuffleRDD，将原RDD重新分区。
leftOuterJoin(otherDataset)	leftOuterJoin类似于SQL中的左外关联left outer join，返回结果以前面的RDD为主，关联不上的记录为空
rightOuterJoin(otherDataset)	rightOuterJoin类似于SQL中的有外关联right outer join，返回结果以参数中的RDD为主，关联不上的记录为空

Action算子

Action算子	作用
reduce(func)	将RDD中元素按func函数进行聚合，func函数是一个(T, T) => T 的映射函数，其中T为源RDD的元素类型，并且func需要满足交换律和结合律以便支持并行计算
collect()	将数据集中所有元素以数组形式返回驱动器(driver)程序。通常用于在RDD进行了filter或其他过滤后，将足够小的数据子集返回到驱动器内存中，否则会OOM
count()	返回数据集中元素个数
first()	返回数据中首个元素（类似于take(1)）
take(n)	返回数据集中前n个元素
takeSample(withReplacement, num, [seed])	返回数据集的随机采样子集，最多包含num个元素，withReplacement表示是否使用回置采样，最后一个参数为可选参数seed，随机数生成器的种子
takeOrdered(n, [ordering])	按元素排序（可以通过ordering自定义排序规则）后，返回前n个元素
saveAsTextFile(path)	将数据集中元素保存到指定目录下的文本文件中（或者多个文本文件），支持本地文件系统、HDFS或者其他Hadoop支持的文件系统。保存过程中，Spark会调用每个元素的toString方法，将结果保存成文件中的一行
saveAsSequenceFile(path)	将数据集中元素保存到指定目录下的Hadoop Sequence文件中，支持本地文件系统、HDFS或者其他任何Hadoop支持的文件系统。适用于实现了Writeable接口的键值对RDD。在Scala中，同样也适用于能够被隐式转换为Writeable的类型
saveAsObjectFile(path)	将RDD元素以Java序列化的格式保存成文件，保存结果的文件可以使用SparkContext.objectFile来读取
countByKey()	只能适用于包含键值对(K, V)的RDD，并返回一个哈希表，包含(K, Int)对，表示每个Key的个数。
foreach(func)	foreach用于遍历RDD,将函数f应用于每一个元素。但要注意，如果对RDD执行foreach，只会在Executor端有效，而并不是Driver端必须应用于键值对的元素类型，如源RDD包含(K,V)对，则该算子返回一个新的数据集包含(K, Iterator)对

Action算子	作用
foreachPartition(func)	foreachPartition和foreach类似，只不过是对每一个分区使用函数，性能比foreach要高，推荐使用。