Spark函数讲解：aggregate

我们先来看看aggregate函数的官方文档定义：

> Aggregate the elements of each partition, and then the results for all the partitions, using given combine functions and a neutral "zero value". This function can return a different result type, U, than the type of this RDD, T. Thus, we need one operation for merging a T into an U and one operation for merging two U's, as in scala.TraversableOnce. Both of these functions are allowed to modify and return their first argument instead of creating a new U to avoid memory allocation.

aggregate函数将每个分区里面的元素进行聚合，然后用combine函数将每个分区的结果和初始值(zeroValue)进行combine操作。这个函数最终返回的类型不需要和RDD中元素类型一致。

函数原型

```
def aggregate[U: ClassTag](zeroValue: U)(seqOp: (U, T) => U, combOp: (U, U) => U): U
```

实例

```
/**
 * User: 过往记忆
 * Date: 15-02-12
 * Time: 上午08:30
 * bolg:
 * 本文地址：/archives/1268
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */

scala> def seqOP(a:Int, b:Int) : Int = {
     | println("seqOp: " + a + "₩t" + b)
     | math.min(a,b)
     | }
seqOP: (a: Int, b: Int)Int

scala> def combOp(a:Int, b:Int): Int = {
     | println("combOp: " + a + "₩t" + b)
     | a + b
```

```
    | }
combOp: (a: Int, b: Int)Int

scala> val z = sc. parallelize ( List (1 ,2 ,3 ,4 ,5 ,6) , 2)
scala> z. aggregate(3)(seqOP, combOp)
seqOp: 3 1
seqOp: 3 4
seqOp: 1 2
seqOp: 3 5
seqOp: 1 3
seqOp: 3 6
combOp: 3 1
combOp: 4 3

res20: Int = 7

scala> def seqOp(a:String, b:String) : String = {
    | println("seqOp: " + a + "₩t" + b)
    | math.min(a.length , b.length ).toString
    | }
seqOp: (a: String, b: String)String

scala> def combOp(a:String, b:String) : String = {
    |  println("combOp: " + a + "₩t" + b)
    | a + b
    | }
combOp: (a: String, b: String)String

scala> val z = sc. parallelize ( List ("12" ,"23" ,"345" ,"4567") ,2)
scala> z. aggregate ("")(seqOp, combOp)
seqOp:  345
seqOp:  12
seqOp: 0 4567
seqOp: 0 23
combOp:  1
combOp: 1 1

res25: String = 11

scala> val z = sc. parallelize ( List ("12" ,"23" ,"345" ,"") ,2)
scala> z. aggregate ("")(seqOp, combOp)
seqOp:  12
seqOp:  345
seqOp: 0 23
seqOp: 0
combOp:  0
```

combOp: 0 1
res26: String = 01

注意

1、reduce函数和combine函数必须满足交换律(commutative)和结合律(associative)
2、从aggregate 函数的定义可知，combine函数的输出类型必须和输入的类型一致