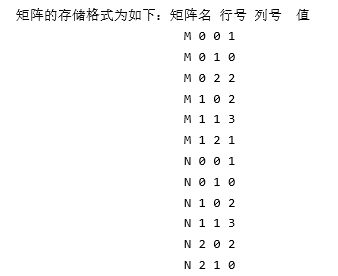
Hadoop

**Hadoop和Spark实现矩阵相乘**



Paste\_Image.png

**一、方法一**

需要两步完成, Mi,kNk,j  
1.Map阶段，对于矩阵 M，把列作为 key，对于矩阵 N 把行作为 key  
2.Reduce阶段，对于相同 key 的值，M 矩阵和 N 矩阵的值做笛卡尔积，输出 【（M 的行）+ （N 的列值）+ （MN相乘 value 值）】

public static class ReadMapper extends Mapper<Object, Text, Text, Text>{

public void map(Object key ,Text value,Context context) throws IOException, InterruptedException{

String str=value.toString(); String [] strs=str.split(" ");

if(strs[0].equals("M")&&strs.length==4){//如果是 M矩 阵，则以 j 作为 key

context.write(new Text(strs[2]),new Text(strs[0]+" "+strs[1]+" "+strs[3]));

} else if(strs[0].equals("N")&&strs.length==4){

context.write(new Text(strs[1]),new Text(strs[0]+" "+strs[2]+" "+strs[3]));

}

}

}

public static class WriteReducer extends Reducer<Text, Text, Text, Text>{

public void reduce(Text key,Iterable<Text> values,Context context) throws IOException, InterruptedException{

//定义连个 ArrayList 第一个存放 M,第二个存放 N

ArrayList<String> mTrix=new ArrayList<String>();

ArrayList<String> nTrix=new ArrayList<String>();

for(Text value : values){

if((value.toString()).contains("M")){

mTrix.add(value.toString());

} else {

nTrix.add(value.toString());

}

}

String[] mItems, nItems;

//进行合并计算

for(String m : mTrix){

mItems=m.split(" ");

for(String n : nTrix){

nItems=n.split(" ");

}

context.write(new Text(key+"+"+mItems[1]+"+"+nItems[1]+"+"+(Integer.parseInt(nIt ems[2])\*Integer.parseInt(mItems[2]))+"+"),new Text(""));

}

}

}

下面是关键，如果M 和N 是分别输入 的，上面的都可以不要。

4.map 阶段读取上一阶段 reduce 产生的 i，j，value 值  
5.把所有相同 key 的 value 值相加，输出即可

public static class ReadMapper1 extends Mapper<Object, Text, Text, Text>{

public void map(Object key ,Text value,Context context)

throws IOException, InterruptedException{

String str=value.toString();

String [] strs=str.split("\\+");

if(strs.length>=4){

context.write(new Text(strs[1]+" "+strs[2]),new Text(strs[3]));

}

}

}

public static class WriteReducer1 extends Reducer<Text, Text, Text, Text>{

public void reduce(Text key,Iterable<Text> values,Context context) throws IOException, InterruptedException{

int num=0;

for(Text value : values){

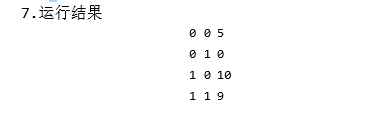
num+=Integer.parseInt(value.toString());

}

context.write(key,new Text(num+""));

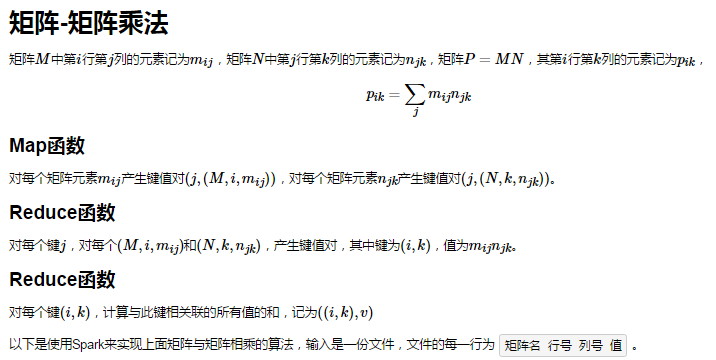
}

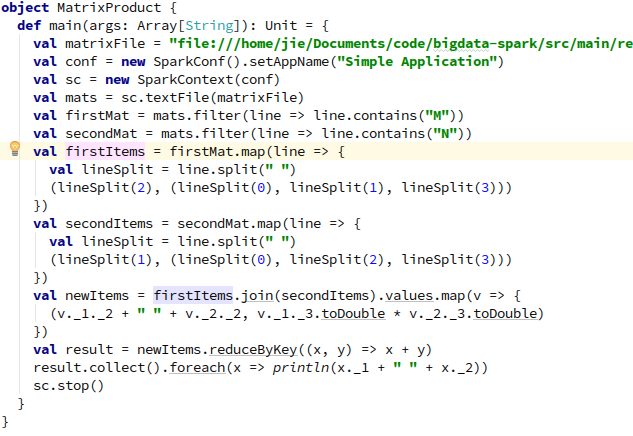
}



Paste\_Image.png

Spark **矩阵-矩阵乘法**





var rdd1 = sc.makeRDD(Array(("A","a1"),("C","c1"),("D","d1"),("F","f1"),("F","f2")),2)

var rdd2 = sc.makeRDD(Array(("A","a2"),("C","c2"),("C","c3"),("E","e1")),2)

scala> rdd1.join(rdd2).collect()

res5: Array[(String, (String, String))] = Array((A,(a1,a2)), (C,(c1,c2)), (C,(c1,c3)))

A(v.\_1 , v.\_2)

A( ,)