MapReduce的类型和格式

**这节介绍各种类型从简单文本到结构化的二进制对象如何在MapReduce中使用**

## 7.1 MapReduce的类型

在Mapper和Reducr的接口中的context类对象是用于输出键值对的。因此他们通过输出类型参数化。

在程序中要求Mapper的输出类型和Reducer的输入类型相同，但是java编译器不强制要求。

如果使用combine函数，他的形式是和Reduce函数相同，其实也是reducer的一种实现。**不同之处在于他的输出类型是中间的键值对，可以输入到Reduce函数中。**

partition函数对中间结果的键值对进行处理，并返回一个分区索引**，实际上分区由键单独决定。**

**输入数据的类型由输入格式进行设置。** 比如，对应TextInputFormat的键类型为LongWritable，值类型为Text。其他类型通过调用job类的方法显式的设置。如果没有显式的设置，中间类型默认为最终输出类型。

原来，JAVA的泛型机制有很多限制：类型擦除导致运行过程中类型信息并非一直可见。

### 7.1.1 默认的MapReduce作业

如果不指定Mapper和Reducer就运行MapReduce程序会怎么样？

这边讨论的是书上范例7-1：

默认的输入格式是TextInputFormat，键类型是LongWritable，值类型是Text

默认的partitioner是HashPartitiioner,它对每个记录的键进行哈希操作以决定该记录应该属于哪个分区。每个分区对应一个Reducer。

默认情况下，只有一个reducer，因此只有一个分区，这个时候partitioner操作就变的无关紧要。当有很多reducer的时候就变的很重要。如果基于键的散列函数足够好，那记录就会均匀的分到若干reduce任务中。

你可能注意到没有设置map任务的数量，原因是**该数量等于输入文件被划分的分块数。**

### 7.1.2 默认的Streaming 作业

## 7.2 输入格式

从一般的文本文件到数据库，hadoop可以处理很多不同类型的数据格式。

### 7.2.1 输入分区与记录

**输入分片**就是一个由单个map操作来处理的输入块。

输入分片在java中被表示为**InputSplit**接口。

InputSplit 包含一个字节为单位的长度和一组存储位置（主机名）；注意分片并不包含数据，而是**指向数据的引用**。

MapReduce应用程序员不必直接处理InputSplit,因为它是由InputFormat创建的。

运行作业的客户端通过调用getSplit（）计算分片，map任务将输入分片给InputFormat的getRecordReader（）方法来获得分片的RecordReader。RecordReader就像是记录上的迭代器，map用一个RecordReader来生成记录的键值对，在传递给map函数。运行完setup（）之后，查看mapper的run（）方法，重复调用的Context的nextKeyValue()委托给RecordReader的同名函数实现为mapper产生key和value对象。

由于效率的原因，RecordReader 程序每次调用getCurrentKey()和getCurrentValue（）时**返回相同的键值对象，只是对象的内容被Reader的nextKeyValue（）方法改变了。**如果在map函数之后有对键值的引用时，就会出现问题**，如果需要引用，请保存对象的一份副本。**

1. **FileInputFormat类**

两个功能：用于指出输入文件位置；输入文件生成分片的实现代码段。

把分片分割成记录由其子类实现。

1. **FileInputFormat类的输入路径**

addInputPath() addInputPaths()方法可以将一个或多个路径加入路径列表，可以分别多次调用。setInputPath（）一次设定完整的路径列表（替换前面的设置）

当一个被指定为输入路径的目录，其内容是不被递归处理的。如果包含子目录，也会被解释为文件从而产生错误。处理一个问题的方法：使用一个glob或者一个过滤器根据命名模式限定选择目录中文件，二是通过配置设置输入目录进行递归地读取。

setInputPathFilter（）方法设置一个过滤器。FileInputFormat默认一个过滤器来排除隐藏文件（以“.”和“\_”开头的文件）

路径和过滤器也可以通过配置属性来设置，这对Streaming和pipes应用很有用。他们都使用 -input 选项来设置输入路径

Mapred.input.dir 逗号分隔的路径 默认值无

Mapred.input.path.Filter.classe 输入PathFilter类名，默认值无

1. **FileInputFormat类的输入分片**

最小的分片大小是1字节，不过某些格式可以使分片大小有一个更小值（例如，顺序文件中流中每次插入一个同步入口）。应用程序可以强制设置一个最小的输入分片大小，设置一个比HDFS块更大的值，如果数据存储在HDFS上，那么这样做是没有好处的。

最大的分片大小默认是有JAVA long 类型表示的。**只有将其值设置小于块大小才有效果**

默认情况下： mininumSize < blockSize < maxinumSize

1. **小文件和CombineFIleInputFormat**

相比大量的小文件，hadoop更适合处理少量的大文件。一个原因是FileInputFormat生成的分块是一个文件或者是文件的一部分。如果文件很小（比HDFS块小很多），并且文件数量很多，那么每次map任务只能处理很少的数据。

CombineFIleInputFormat可以缓解这个问题，它是针对小文件设计的。FileInputFormat产生分片，而CombineFIleInputFormat把多个文件打包到一个分区中。

减少大量小文件的方法：使用SequenceFile将这些小文件合并成一个或多个大文件。

CombineFIleInputFormat在处理大文件的时候也有好处。将CombineFIleInputFormat的最大分片大小设成块数的较小的整数倍，使每个mapper能处理多个块。

由于combineFileInputFormat 是一个抽象类，在使用时需要一些额外的工作。

1. **避免切分**

方法一：增加最小分片大小，直至大于要处理的文件；

方法二：使用FileInputFormat的具体子类，并且重载isSplitable（）方法，将返回值设置为false.

1. **Mapper中的文件信息**

可以通过调用在mapper的context对象上getInputSplit()方法来实现。查看输入分片的属性：输入文件的路径，分区开始处的字节偏移量，分片长度

1. **把整个文件作为一个记录处理**

### 文本输入

Hadoop非常擅长处理分结构化的数据，本节讨论hadoop提供的用于处理文本的不同的InputFormat类。

1. **TextInputFormat**

这是默认的InputFormat，在分片知道行号是可能的，但是在文件中是不可能的。

有些行可能是会跨文件块存储，也就是map会执行一些远程的读写。

1. **关于KeyValueTextInputFormat**

TextInputFormat的键是每行在文件中的字节偏移量，通常是没有用的，而KeyValueTextInputFormat的键是**排在制表符之前的Text序列**。

1. **关于NlineInputFormat**

如果要接受固定行数的输入，就要使用NlineInputFormat,和TextInputFormat一样，键是字节偏移量，值是文本本身。

通过设置mapreduce.input.lineinputformat.linespermap属性设置每个map接受的行数。

键值和TextInputFormat生成的一样，不同的是输入分片的构造方法。

例子：

1. 有时候程序对少量数据做一些扩展的计算任务，然后输出，比如仿真。
2. 用hadoop从多个数据源加载数据，创建一个“种子”输入文件，记录所有的数据源。然后每个mapper分到一个数据源，将数据源加载数据到HDFS中。
3. **关于XML**

如果xml文件不大，交给一个mapper处理。如果是一个大型的xml文档，由多个输入分片组成，可以通过字符串匹配和正则表达式来查找记录的开始和结束标签，来解决MapReduce的分割问题。Hadoop提供了StreamXMLRecordReader类，reader的配置方法是通过作业配置属性来设置Reader开始标签和结束标签。

### 7.2.3二进制输入

**1.关于SequenceFIleINputFormat类**

Hadoop顺序文件格式存储二进制的键值对的序列。由于他们**是可分割**的，所以很符合MapReduce数据的格式要求。并且支**持压缩**，可以使用一些**序列化的技术存储任意类型。**

**2.关于SequenceFileAsTextInputFormat类**

它将顺序文件的键值转换成Text对象。转换是通过toString()方法调用的。

**3.关于SequenceFileAsBinaryInputFormat类**

它获取顺序文件的键值作为二进制对象，他们被封装成BytesWriteable对象。

### 7.2.4 多个输入

一个MapReduce程序可能包含很多输入文件，所有文件都有同一个InputFormat和同一个Mapper解释。所以可能出现数据格式不同的情况，对于不同的数据集进行join连接操作。

这些问题使用**multipleInputs**类来妥善处理。**它允许为每条输入路径指定InputFormat和Mapper。**

### 7.2.5 数据库输入（输出）

DBInputFormat这种输入格式用于使用JDBC从关系型数据库中读取数据。因为它没有共享能力，加载大数量集的数据时会出错，所以**它适合加载小量的数据集**。

将中等规模的数据传输到HDFS中，使用multipleInputs，对于的输出格式DBOutputFormat.

在关系型数据库和HDFS之间转移数据的另一种方法使用Sqoop。

## 输出格式

### 7.3.1文本输出

默认的输出是TextOutputFormat；他的键值可以是任意类型。可以使用NullWritable来省略输出的键 值（或者两者都省略，相当于NullOutputFormat）

### 二进制输出

1. **关于SequenceFileOutputFormat**

他的输出是顺序文件，是个很好的输出格式，容易压缩。

1. **关于SequenceFileAsBinaryOutputFormat**

它与SequenceFileAsBinaryInputFormat相对应，将键值写入一个SequenceFile容器中。

1. **关于MapFileOutputFormat**

它把Mapfile作为输出，MapFile中的键必须顺序添加。所以确保r**educe输出的键已经排序。**

**Reduce输入的键一定是有序的，**但是输出的键由reducer控制，没有硬性的要求输出有序**。使用MapFileOutputFormat，必须限制reduce的输出是有序。**

### 多个输出

每个reducer输出一个文件，文件名由分区号命名。有时候需要对输出的文件名控制或者reduce输出多个文件。MapReduce 提供了multipleOutputFormat；

1. **范例 数据分割**

考虑一个需求：按照气象站区分气象数据，这需要运行一个作业，作业的输出是每个气象站一个文件。

一种方法是：每个气象站对于一个reducer。第一步，写一个partitioner，把同一个气象站的数据放到同一个分区下。第二步，把作业的reducer的个数设置为气象站的个数。

这么做有两个缺点：第一，需要在运行前知道分区数和气象站的个数。第二，一般来说让应用程序严格限定分区数并不好，因为会导致分区数少或分区不均。

最好的就是让集群为作业决定分区数。

1. **关于multipleOutput类**

multpleOutputFormat类可以将数据写入多个文件。

### 延迟输出

FileOutputFormat的子类会产生输出文件，即便文件时空的。为了避免这种情况，LazyOutputFormat是一种封装的输出格式，可以保证分区的第一条数据输出时才真正创建文件。JobConf和相关的输出格式作为参数来调用setOutputFormatClass().

Streaming 和Pipes 支持 -LazyOutput

### 7.3.5 数据库输出