Spark中函数addFile和addJar函数介绍

我们在使用Spark的时候有时候需要将一些数据分发到计算节点中。一种方法是将这些文件上传到HDFS上，然后计算节点从HDFS上获取这些数据。当然我们也可以使用addFile函数来分发这些文件。

addFile

addFile方法可以接收本地文件（或者HDFS上的文件），甚至是文件夹（如果是文件夹，必须是HDFS路径），然后Spark的Driver和Exector可以通过SparkFiles.get()方法来获取文件的绝对路径（Get the absolute path of a file added through SparkContext.addFile()）,addFile的函数原型如下：

```
def addFile(path: String): Unit
def addFile(path: String, recursive: Boolean): Unit
```

addFile把添加的本地文件传送给所有的Worker，这样能够保证在每个Worker上正确访问到文件。另外，Worker会把文件放在临时目录下。因此，比较适合用于文件比较小，计算比较复杂的场景。如果文件比较大，网络传送的消耗时间也会增长。

path：可以是local、hdfs（任何hadoop支持的文件系统）、HTTP、HTTPS、FTP等。local方式时，在windows下使用绝对路径时需要加个“/”，如“d:/iteblog.data”得写成“/d:/iteblog.data”或“file:///d:/iteblog.data”。

recursive：如果path是一个目录，那么我们可以设置recursive为true，这样Spark会递归地分发这个路径下面的所有文件到计算节点的临时目录。

通过SparkFiles.get(path:String)获取添加的文件路径。

```
var path = "/user/iteblog/ip.txt"
sc.addFile(path)
val rdd = sc.textFile(SparkFiles.get(path))
```

上面的实例展示了如何在Driver中获取分发出去的文件，我们还可以在Exector获取到分发的文件：

```
var path = "/user/iteblog/ip.txt"
```

```
sc.addFile(path)
val rdd = sc.parallelize((0 to 10))
rdd.foreach{ index =>
   val path = SparkFiles.get(path)
   ......
}
```

如果我们添加的是压缩文件，比如.tar.gz、.tgz或者.tar，Spark会调用Linux的解压缩命令tar去解压缩这些文件。

addJar

addJar添加在这个SparkContext实例运行的作业所依赖的jar。，其函数原型如下：

```
def addJar(path: String)
```

path：可以是本地文件（local file）、HDFS文件（其他所有的Hadoop支持的文件系统也可以）、HTTP、 HTTPS 或者是FTP URI文件等等。

其实Spark内部通过spark.jars参数以及spark.yarn.dist.jars函数传进去的Jar都是通过这个函数分发到Task的。