云数据管理 1 大作业
Proposal

2019-09-29

郝天翔 张洋

大作业选题

实现一个简单版本的 Spark。

目前对 Spark 的理解

Resilient Distributed Dataset

RDD 是分布与各节点上的数据构成的数据集，是 Spark 背后的模型。

```
// from collection
val intRDD =
  spark.sparkContext.parallelize(List(1, 2, 3, 4, 5, 6, 7, 8, 9))

// from text file on local FS
val strRDD =
  spark.sparkContext.textFile("/home/zhang/Temp/ss.log")

// or any data source offering a Hadoop InputFormat (e.g. HDFS)
// ...
```

RDD 上的操作

Spark 提供了一系列 API，可以对 RDD 进行各种操作。

- **Transformation**: `map`, `filter`, `sample`
- **Action**: `reduce`, `collect`, `count`

```
// calculate 1*1 + 2*2 + 3*3 + ...
val squares = intRDD.map(x => x * x)
val sums = squares.reduce((a, b) => a + b)

// count lines containing "shadow"
val lineCount = strRDD.filter(x => x.contains("shadow")).count()

println(sums)
println(lineCount)
```

一个简单的例子

使用 Spark 可以从高层定义数据的处理逻辑。

```
val dataset =
  spark.sparkContext.textFile("/home/zhang/Temp/ss.log")
val words =
  dataset
    .map(x => x.toLowerCase)
    .flatMap(x => x split "[^a-z]")
    .filter(x => x.length > 4)
val top10 =
  words
    .map(w => (w, 1))
    .reduceByKey((a, b) => a + b)
    .sortBy[Int]({ case (_, c) => c }, ascending = false)
    .take(10)
top10 foreach { case (w, c) => println("% 10d %s".format(c, w)) }
```

一个简单的例子（续）

然后 Spark 会自动将计算任务分解到各节点上。

```
$ sbt package
$ spark-submit target/scala-2.11/foo.jar
  --class "SimpleApp"
  --master "local[4]"

   2341221 hyperion
   2340827 sslocal
    385310 connecting
     32414 google
     10724 mtalk
      9028 error
      9028 errno
      8884 gstatic
      7954 warning
      7814 timed
```

初步预期目标

我们计划仿照 Spark 的工作方式，实现一个至少支持以下特性的类 Spark 库：

- 简单的 transformations 和 actions 操作
- 简单的需要进行 shuffle 的操作
- 节点之间可进行通信和协作
- 主节点自动划分任务
- 惰性计算

初步预期目标（续）

如果时间充足，可能还会实现以下特性：

- 支持 HDFS

  （初期底层存储为 OS 提供的文件系统）
- 实现更多的 transformations 和 actions
- 内存不够时使用磁盘进行交换

  （初期假定数据可以放在所有节点的内存中）
- 尝试在特定目标上运行效率接近或超过 Spark