**20200625学习报告**

1813068郭冠琛

**学习内容**

* MapReduce原理

JobClient.runJob（）提交作业，每秒轮询作业进度（submitjob），通过jobtracker获得状态

jobtracker将作业放入内部队列同时初始化，由作业调度器调度

tasktracker定期发送心跳方法调用jobtracker表明其是否存活，指明其是否准备好运行新的任务（从属于job），jobtracker分配任务给tasktracker

tasktracker：本地化作业jar文件-本地化运行所需文件-新建taskrunner实例运行任务

作业进度：map-已处理完输入的百分比；reduce：与shuffle有关（拷贝排序reduce各占1/3）

执行成功：收到最后一个任务完成通知后将作业状态设置为完成，然后jobclient查询后告诉用户同时从runjob方法返回，最后jobtracker和tasktracker清空作业状态（如中间输出）

任务失败：用户代码运行时异常——子进程JVM向tasktracker发送错误报告并退出，tasktracker释放一个槽并将此次尝试标记为failed 子JVM退出：tasktracker标记为failed 任务挂起超时也会被标记，子JVM被自动杀死

任务失败时将重新调度，但尝试次数达到上限后整个作业失败

* shuffle

map：输出写到缓冲区，缓冲区达到某个阈值则启动后台进程将内容溢写到磁盘（每次都新建一个溢写文件），同时map继续输出直到缓冲区满（阻塞）任务完成前所有溢写文件被合并成一个已排序和分区的文件（指定combiner且溢写次数达到某个值时就在输出文件被写之前执行combiner）

reduce：可并行复制输出

* 推测式执行

一个任务执行比预期慢时会启动另一个任务作为备份

同时启动两个备份任务会造成竞争

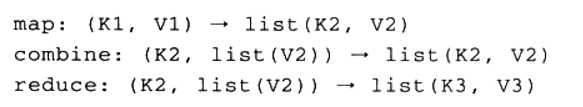
只有在所有任务都启动后才会启动备份任务，同时只针对已运行一段时间且比作业中其他任务进度慢的任务

* JVM重用

tasktracker虽然可以一次运行多个任务，但都运行在独立JVM中。

JVM启动耗时；作业任务间状态共享

* 输入输出形式



默认mapper：将输入的键值对按原样输出

* textinputformat

默认类型，键为该行字节偏移量

keyvaluetextinputformat

每行包括一个键值对，键和值之间以指定符号进行分割

nlineinputformat

类似textinputformat，但不同的是每个mapper会收到指定数量的键值对

二进制格式

multipleinput.addinputpath（conf，输入路径，输入格式.class，处理类.class）

dbinputformat

**需要解决的问题**

tasktracker失败

shuffle中的reduce

错误记录

输入分片