MapReduce Task的用户自定义程序适配

MapReduce的Task 执行流程如下：



Task的执行相关类图如下：



1. JobImpl初始化，生成MapTask

*private void createMapTasks(JobImpl job, long inputLength,*

*TaskSplitMetaInfo[] splits) {*

*for (int i=0; i < job.numMapTasks; ++i) {*

*TaskImpl task =*

*new MapTaskImpl(job.jobId, i,*

*job.eventHandler,*

*job.remoteJobConfFile,*

*job.conf, splits[i],*

*job.taskAttemptListener,*

*job.jobToken, job.jobCredentials,*

*job.clock,*

*job.applicationAttemptId.getAttemptId(),*

*job.metrics, job.appContext);*

*job.addTask(task);*

*}*

*}*

1. MapTask通过YarnChild JVM启动，其启动如下：

*$JAVA\_HOME/bin/java*

*org.apache.hadoop.mapred.YarnChild*

*10.139.4.107 //host*

*41211 //port ，以上是MRAppMaster Address信息*

*attempt\_1533738450228\_0006\_m\_000000\_02 // attemptID*

*2 //JVMID*

启动YarnChild进程后，从MRAppMaster获取Task信息，获取过程通过RPC调用，源码如下：

*public JvmTask getTask(JvmContext context) throws IOException {*

*JVMId jvmId = context.jvmId;*

*WrappedJvmID wJvmID = new WrappedJvmID(jvmId.getJobId(), jvmId.isMap,*

*...*

*org.apache.hadoop.mapred.Task task =*

*jvmIDToActiveAttemptMap.remove(wJvmID);*

*launchedJVMs.remove(wJvmID);*

*LOG.info("JVM with ID: " + jvmId + " given task: " + task.getTaskID());*

*task.setEncryptedSpillKey(encryptedSpillKey);*

*jvmTask = new JvmTask(task, false);*

*....*

*return jvmTask;*

*}*

从TaskAtemptListenerImpl#jvmIDToActiveAttemptMap中获取Task

1. jvmIDToActiveAttemptMap初始化，其用于存放Task数据，在Container创建过程中，生成对应Task，并存放到MRAppMaster中，其执行如下：

*taskAttempt.remoteTask = taskAttempt.createRemoteTask();  
taskAttempt.jvmID =  
 new WrappedJvmID(taskAttempt.remoteTask.getTaskID().getJobID(),  
 taskAttempt.remoteTask.isMapTask(),  
 taskAttempt.container.getId().getContainerId());  
taskAttempt.taskAttemptListener.registerPendingTask(  
 taskAttempt.remoteTask, taskAttempt.jvmID);*

*MapTaskAttemptImpl#createRemoteTask*

*@Override*

*public Task createRemoteTask() {*

*MapTask mapTask =*

*new MapTask("", TypeConverter.fromYarn(getID()), partition,*

*splitInfo.getSplitIndex(), 1);*

*mapTask.setUser(conf.get(MRJobConfig.USER\_NAME));*

*mapTask.setConf(conf);*

*return mapTask;*

*}*

1. MapTask的执行

MapTask#runNewMapper的执行如下：

*void runNewMapper(final JobConf job,*

*final TaskSplitIndex splitIndex,*

*final TaskUmbilicalProtocol umbilical,*

*TaskReporter reporter*

*) {*

*...*

*//获取Map执行的Context*

*mapContext =   
 new MapContextImpl<INKEY, INVALUE, OUTKEY, OUTVALUE>(job, getTaskID(),   
 input, output,   
 committer,   
 reporter, split);*

*// 获取Mapper类，并实例化*

*org.apache.hadoop.mapreduce.Mapper<INKEY,INVALUE,OUTKEY,OUTVALUE> mapper =*

*(org.apache.hadoop.mapreduce.Mapper<INKEY,INVALUE,OUTKEY,OUTVALUE>)*

*ReflectionUtils.newInstance(taskContext.getMapperClass(), job);*

*...*

*try {*

*input.initialize(split, mapperContext);*

*mapper.run(mapperContext);*

*...*

*}*

Mapper的run方法如下：

*public void run(Context context) throws IOException, InterruptedException {  
 setup(context);  
 try {  
 while (context.nextKeyValue()) {  
 map(context.getCurrentKey(), context.getCurrentValue(), context);  
 }  
 }....  
}*

run方法中通过while循环获取所有数据。数据封装在Context，类图如下：



对于MapContextImpl，数据源为InputSplit，从HDFS中获取数据。