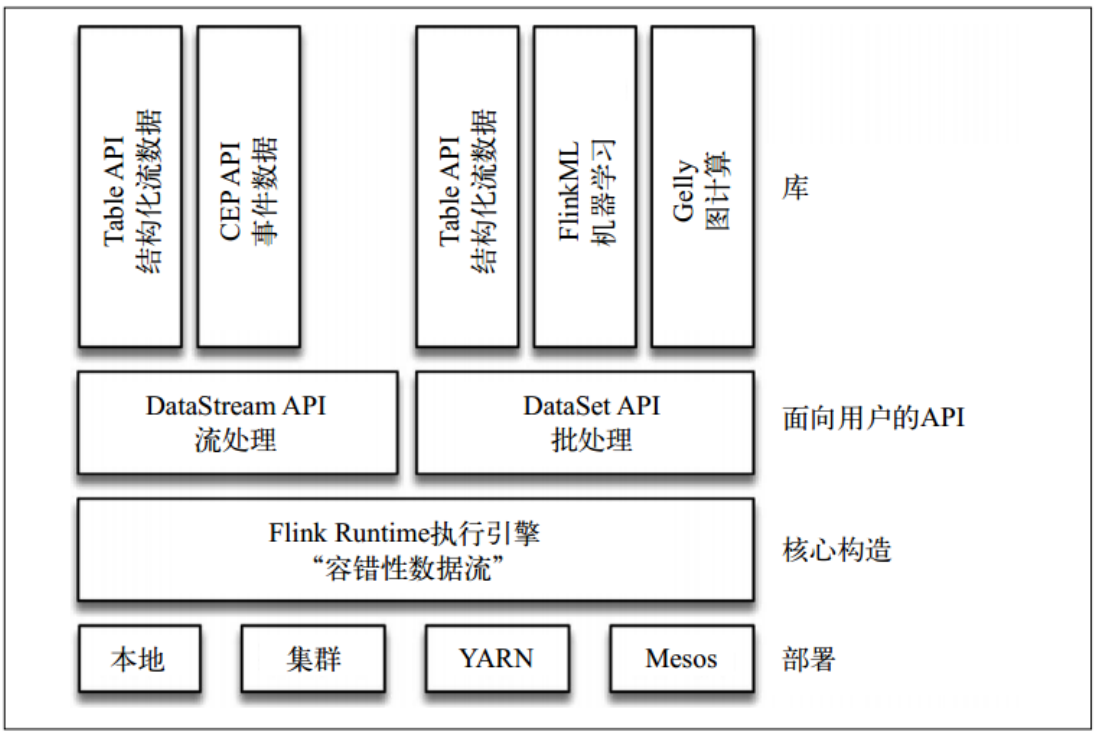
滴雨 cdh/hadoop 集成 flink

Flink 是新一代流处理引擎，Flink 的一个优势是，它拥有诸多重要的流式计算功能。其他项目为了实现这些功能，都不得不付出代价。比如， Storm 实现了低延迟，但是还做不到高吞吐，也不能在故障发生时准确地处理计算状态； Spark Streaming 通过采用微批处理方法实现了高吞吐和容错性， 但是牺牲了低延迟和实时处理能力，也不能使窗口与自然时间相匹配，并且表现力欠佳。



1.设集群环境是 cdh5.15,hadoop2.6，Scala2.11 版本的，下载 flink 版本是：

flink-1.7.2-bin-hadoop26-scala_2.11.tgz

https://archive.apache.org/dist/flink/flink-1.7.2/

2.将该压缩包拷贝到 master01,slave01,slave02 的 /myflink 目录

在所有节点：mkdir /myflink

现将该压缩包拷贝到 master01 的 /myflink 目录，然后用 scp 远程拷贝到其他节点上：

scp flink-1.7.2-bin-hadoop26-scala_2.11.tgz

root@slave01:/myflink

scp flink-1.7.2-bin-hadoop26-scala_2.11.tgz

root@slave02:/myflink

在 3 个节点执行如下命令解压：tar xzf flink-1.7.2-bin-hadoop26-scala_2.11.tgz

3.接着，在 3 个节点都修改 Flink 配置文件 flink-1.7.2/conf/flink-conf.yaml，最简单的修改如下：

[root@master01 conf]# vi flink-conf.yaml

taskmanager.numberOfTaskSlots: 4

jobmanager.rpc.address: master01（This setting

is only used in Standalone mode，Yarn/Mesos

automatically configure the host name based on the hostname of the node where the

JobManager runs.所以不用 standalone 模式的话就可以不修改，要用的话都指向 master01 把）Flink on Yarn 会覆盖下面几个参数，如果不希望改变配置文件中的参数，可以动态的通过-D 选项指定，如 -Dfs.overwrite-files=true -

Dtaskmanager.network.numberOfBuffers=16368

jobmanager.rpc.address：因为 JobManager 会经常分配到不同的机器上

taskmanager.tmp.dirs：使用 Yarn 提供的 tmp 目录

parallelism.default：如果有指定 slot 个数的情况下

4.在 3 个节点都修改 conf/master、slaves 文件

[root@master01 conf]# vi masters

master01:8081

[root@master01 conf]# vi slaves

slave01

slave02

5. 启动 Flink 集群，执行如下命令：

5.1 启动 standalone 模式：

[root@master01 flink-1.7.2]# bin/start-cluster.sh

访问：http://master01:8081/#/overview

停止 standalone 模式：

[root@master01 flink-1.7.2]# bin/stop-cluster.sh

5.2. 启动 Flink Yarn Session

5.2.1 在 YARN 上启动长时间运行的 Flink 集群:

[root@master01 flink-1.7.2]# ./bin/yarn-session.sh -n 8 -jm 1024 -tm 1024 -s 4 -nm FlinkOnYarnSession -d

2019-02-20 17:06:06,471 INFO org.apache.flink.yarn.AbstractYarnClusterDescriptor - Cluster specification:

ClusterSpecification{masterMemoryMB=1024,
taskManagerMemoryMB=1024, numberTaskManagers=8,
slotsPerTaskManager=4}

对参数说明：

-n,--container 指 YARN container 分配的个数(即 TaskManagers 的个数)

-jm,--jobManagerMemory 指 JobManager Containe 的内存大小，单位为 MB

-tm,--taskManagerMemory 指每个 TaskManagerContainer 的内存大小，单位为 MB

-s 指每个 TaskManager 的 slot 个数。

-nm flink appName

-d 如果不希望 Flink Yarn client 长期运行，Flink 提供了一种 detached YARN session，启动时候加上参数-d 或--detached。

执行上面命令来分配 8 个 TaskManager，每个都拥有 1GB 的内存和 4 个 slot，同时会请求启动 8+1 个容器，因为对于 ApplicationMaster 和 JobManager 还需要一个额外的容器。

yarn-session 下提交：

[root@master01 flink-1.7.2]./bin/flink run
/myflink/projects/spark2_kafka_stream.jar

访问：

yarn ui:

http://master01:8088/cluster/scheduler

flink ui:

2019-02-20 19:28:19,241

INFO org.apache.flink.runtime.rest.RestClient

- Rest client endpoint started.

Flink JobManager is now running on master01:42809 with leader id 00000000-0000-0000-0000-000000000000.

JobManager Web Interface: http://master01:42809(每次启动端口号都不一样)

关闭 yarn-session:

[root@master01 conf]# yarn application -kill

application_1550056788069_0008

5.2.2 yarn 的 single job

在 YARN 上直接运行单个 Flink 作业,在 yarn 上不启动 flink 集群，就不用一直占用 yarn 资源:

[root@master01 flink-1.7.2]# ./bin/flink run -m yarn-cluster -yn 4 -yjm 1024m -ytm 4096m

/myflink/projects/spark2_kafka_stream.jar

yarn ui:

http://master01:8088/cluster/scheduler

spark2_kafka_stream.jar-------scala code:

```
package com.test.flink_kafka


import org.apache.flink.api.java.ExecutionEnvironment
import org.apache.flink.api.java.aggregation.Aggregations


object Flink {
  def main(args: Array[String]): Unit = {
    val env = ExecutionEnvironment.getExecutionEnvironment;
    val stream = env.fromElements(1, 2, 3, 4, 3, 4, 3, 3, 5)
    //.map(x=>(x))
    //     .map { x => (x, 1) }
    //      .groupBy(0).aggregate(Aggregations.SUM, 1)
//.sum(1)
    stream.print()
    // 批处理不要 env.execute()
    //    env.execute()
  }
}
```

flink yarn-session 与 yarn 的 single job 的对比：

session 模式保留了 Standalone 的优势，可以节省申请启动 tm 的时间，适合短小 job 共用资源执行；per job 有更好的资源 quota 管理、隔离等，性能稳定性