Deploying & Configuring

Objective

Packaging and shipping a Spark application to a cluster

Cluster deploy modes

3 ways of configuring Spark applications

Spark App Execution

```
/spark/bin/spark-submit \
  --class par1foundation.TestApp \
  --master spark://(dockerID):7077 \
  --deploy-mode client \
  --verbose \
  --supervise \
  spark_playground.jar data/movies.json data/goodMovies
```

Some command line arguments to pass:

--executor-memory: allocate a certain amount of RAM/executor (we'll learn later how to choose values)

--driver-memory: allocate a certain amount of RAM for the driver

--jars: add additional JVM libraries for Spark to have access to

--packages: add additional libraries as Maven coordinates

--conf (configName) (configValue): other configurations to the Spark application (including JVM options)

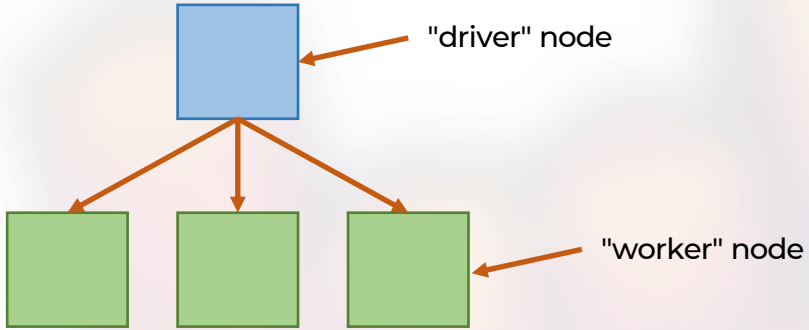
--help: show all options

The Anatomy of a Cluster

Spark cluster manager

- one node manages the state of the cluster
- the others do the work
- communicate via driver/worker processes

Standalone, YARN, Mesos, Kubernetes

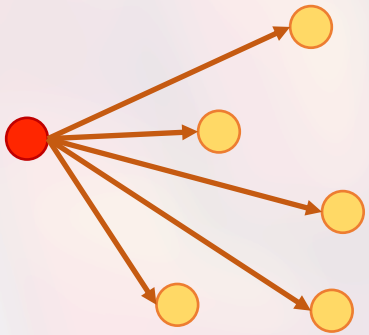


Spark driver

- manages the state of the stages/tasks of the application
- interfaces with the cluster manager

Spark executors

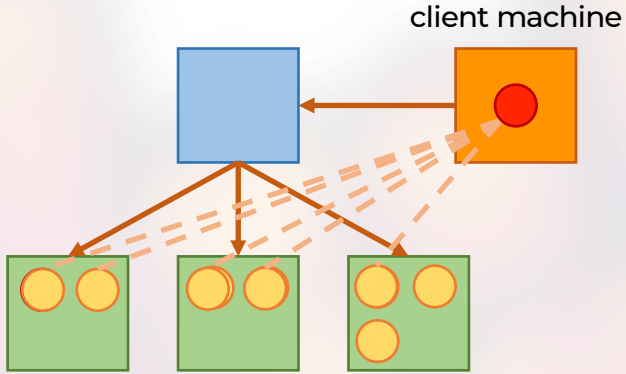
- run the tasks assigned by the Spark driver
- report their state and results to the driver



The Anatomy of a Cluster

Execution mode

- cluster
- client
- local



Cluster mode

- the Spark driver is launched on a worker node
- the cluster manager is responsible for Spark processes

Client mode

- the Spark driver is on the client machine
- the client is responsible for the Spark processes and state management

Local mode

- the entire application runs on the same machine

Spark Cluster mode

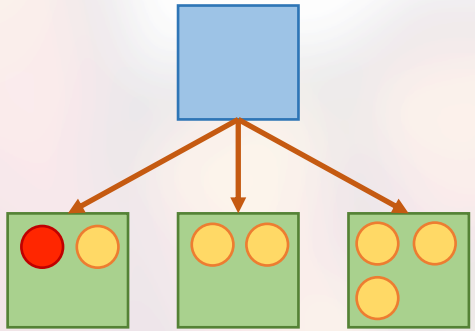
Driver is a dedicated JVM container on the cluster

Pros

- (usually) more memory availability for the driver
- faster communication between driver and executors
- faster perf overall

Cons

- failure of node with the driver means application failed
- fewer resources allocated to the executors



Spark Client mode

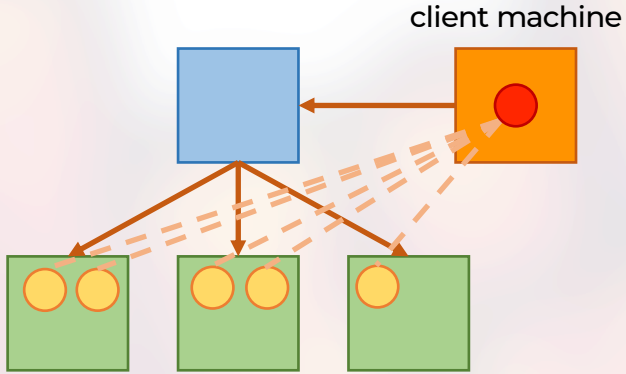
Driver is created on the machine which submits the job

Pros

- more resources to the executors
- node failure doesn't crash the application
- results are immediately available on the machine

Cons

- (usually) fewer resources available to the driver
- communication overhead between the driver and the executors
- (very likely) slower perf



Spark rocks