Big Data Technologies

Agenda

- Spark Core

Spark Terminologies

- RDD -- Resilient Distributed Dataset
  - Dataset -- Set of data -- Any format from any location (local fs, HDFS, S3, ...)
  - Distributed -- Divided into partitions -- Processed on multiple workers
  - Resilient -- If any worker fails, the partition will be recomputed (info taken from Lineage)
  - Characteristics: Immutable, Lazily evaluated.
- DAG -- Execution plan submitted for execution on Spark cluster
  - Triggered by action operation on RDD
- Application -- Set of Jobs
  - For each job one DAG is prepared and executed on cluster.
  - Execution is initiated by SparkContext.
- SparkContext -- In-charge of Spark application
  - Single SparkContext for the application
  - RDD Creation, DAG preparation & submit, Track DAG execution, Broadcast vars & Accumulators.

    ```
    conf = SparkConf().setAppName("demo01")
    sc = SparkContext(conf = conf)
    ```

- Job -- Triggered by action
  - Per job a DAG is created and executed.
- Stages -- Logical part of execution
  - DAG (execution plan) is divided into stages due to shuffle operations

  - Number of stages = Number of shuffle operations + 1
- Tasks -- Processing of Partition
  - Processing of Partition within a Stage
  - Executes in a worker node by a thread -- task
  - Tasks are created in executor process
- Executor -- Process for application execution
  - Process created on worker nodes for executing an application.
  - All tasks (threads) for an application are created in its executor.
  - RDD partitions processed in executor memory.
- Driver -- Process that creates SparkContext
  - SparkContext resides in Driver process memory.
  - If deploy-mode is client (given with spark-submit command), then Driver is SparkSubmit process created in client machine (from where job is submitted).
    - client terminal> spark-submit --master spark://master:7077 demo01.py
      - Deploy-mode is client by default.
  - If deploy-mode is cluster (given with spark-submit command), then Driver is created in an executor process on one of the worker node (in the cluster).
    - client terminal> spark-submit --master spark://master:7077 --deploy-mode cluster demo01.py
  - If driver fails after submitting job, all executors will terminate and application fails. However, application can be submitted using --supervise option, to restart the driver in case of failure.
    - client terminal> spark-submit --master spark://master:7077 --supervise --deploy-mode cluster demo01.py
- Worker -- Represents worker node/machine in the cluster
  - Worker is a process that handles (initialization/tracking) execution of applications (executors) on worker node.
  - Started from master node with cmd> start-workers.sh
- Master -- Represents master node/machine in the cluster.
  - Master is a process that manages spark cluster (cluster resources/track workers).
  - Started from master node with cmd> start-master.sh
  - Typically, backup/reserve master is maintained to handle failure of master.

Spark Cluster Managers

- Standalone -- No separate cluster manager is used -- Built-in cluster manager of Spark.

  - Job is submitted with url spark://
    - e.g. spark-submit --master spark://master-ip:7077 app.py
- YARN -- Cluster Management -- ResourceManager (like Master) + NodeManager (like Worker)
  - Job is submitted with url "yarn"
  - Usually used when Spark installed as eco-system of Hadoop e.g. AWS EMR.
- Mesos -- Cluster Manager by Spark team.
  - e.g. spark-submit --master mesos://master-ip:port app.py
- Kubernetes -- Orchstration (Horizontal scaling -- on-demand)
  - e.g. spark-submit --master k8s://https://master-ip:port app.py