An Introduction to Spark's Architecture

An Introduction to Spark's Architecture

Recap

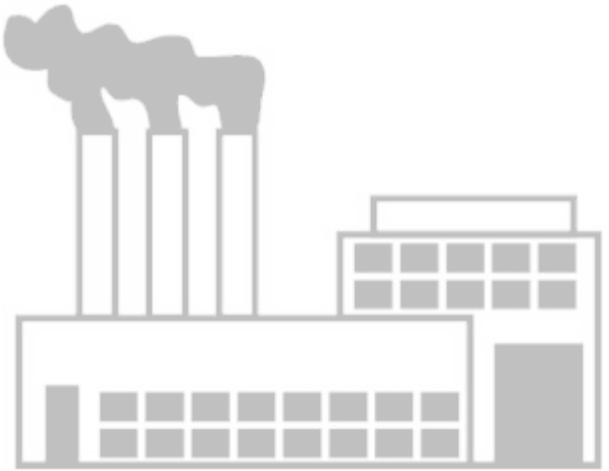
(Key Concept: Distributed Computing)

An Introduction to Spark's Architecture

Spark Architecture Concepts: Recap

Clusters and Nodes

- A Cluster is a group of nodes
- Nodes are the individual machines within a cluster (generally a VM)
- With Databricks, the driver (a JVM) and each executor (each a JVM) all run in their own nodes



Spark Architecture Concepts: Recap

Driver

- Runs the Spark application
- Assigns tasks to slots in an executor
- Coordinates the work between tasks
- Receives the results, if any

Spark Architecture Concepts: Recap

Executors

- Provides an environment in which tasks can be run
- Leverages the JVM to execute many threads

Spark Architecture Concepts: Recap

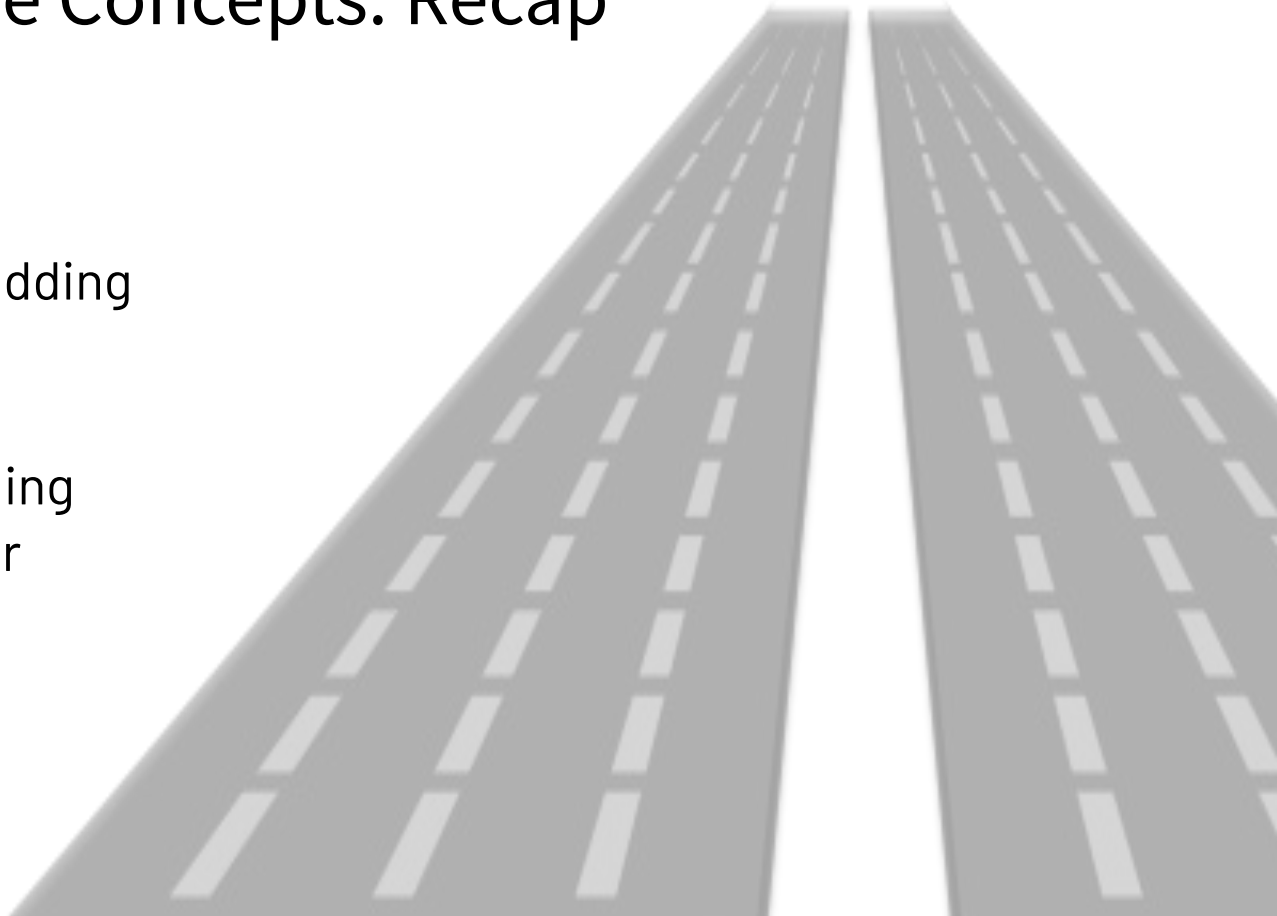
Slots/Cores/Threads

- The lowest unit of parallelization
- Generally interchangeable terms,
  but "slot" is the most accurate term
- Executes a set of transformations against
  a partition as directed to by the driver

Spark Architecture Concepts: Recap

Parallelization

- Scale horizontally by adding more executors
- Scale vertically by adding cores to each executor



An Introduction to Spark's Architecture

Spark Architecture Concepts: Recap

Partitions

- A ~128 MB chunk of the larger dataset
- Each task processes one and only one partition
- The size and record splits are decided by the driver
- The initial size is partially adjustable with various configuration options

Spark Architecture Concepts: Recap

Applications, Jobs, Stages, and Tasks

- The hierarchy into which work is subdivided
- One Spark action results in one or more jobs
- The number of stages depends on the operations submitted with the application
- Tasks are the smallest unit of work

Spark Architecture Concepts: Recap

General Notes

- Executors share machine level resources
- Tasks share executor (JVM) level resources
- Rarely are significant performance improvements made by tweaking Spark configuration settings