RDD

Igor Yakushin
ivy2@uchicago.edu

RDD

- RDD - Resilient Distributed Dataset
- RDD - collection of records partitioned across the nodes and can be operated on in parallel
- RDD can be created from files in various supported formats or by transforming other RDDs
- One can ask Spark to cache RDD in memory or disk for fast reuse
- RDDs automatically recover from node failure
- RDD supports two types of operations:
  - Transformations - create a new dataset from an existing one. Lazy evaluation - evaluated only when required by action.
  - Actions - return a value to the driver program after running a computation on the dataset.

Transformations

Examples of RDD transformations:

- map(func) - transform each element by a applying a function
- filter(func) - select records satisfying boolean function
- sample(withReplacement, fraction, seed) - Sample a fraction fraction of the data, with or without replacement, using a given random number generator seed.
- union(otherDataset), intersection(otherDataset)
- distinct([numTasks]))
- groupByKey([numTasks])
- sortByKey([ascending], [numTasks])
- pipe(command, [envVars])

Actions

Examples of RDD actions:

- reduce(func) - Aggregate the elements of the dataset using a function func (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel.
- collect() - Return all the elements of the dataset as an array at the driver program.
- count()
- take(n) - Return first n elements of the results
- countByKey() - Only available on RDDs of type (K, V). Returns a hashmap of (K, Int) pairs with the count of each key.
- foreach(func) - Run a function func on each element of the dataset. This is usually done for side effects such as updating an accumulator variable (see below) or interacting with external storage systems.