Pair RDDs in PySpark

Big Data & Predictive Analysis Lanjut

Arif Laksito, M. Kom

Introduction to pair RDDs in PySPark

- Real life datasets are usually key/value pairs
- Each row is a key and maps to one or more values
- Pair RDD is a special data structure to work with this kind of datasets
- Pair RDD: Key is the identifier and value is data

Creating pair RDDs

- Two common ways to create pair RDDs
    1. From a list of key-value tuple
    2. From a regular RDD
- Get the data into key/value form for paired RDD

```
my_tuple = [('Sam',23),('Marry',34),('Peter',25),]

pairRDD_tuple = sc.parallelize(my_tuple)


my_list = ['Sam 24','Marry 34', 'Peter 25']

regularRDD = sc.parallelize(my_list)

pair_RDD_RDD = regularRDD.map(lambda s:(s.split(' ')[0], s.split(' ')[1]))
```

Creating pair RDDs

- Two common ways to create pair RDDs
    1. From a list of key-value tuple
    2. From a regular RDD
- Get the data into key/value form for paired RDD

```
my_tuple = [('Sam',23),('Marry',34),('Peter',25),]

pairRDD_tuple = sc.parallelize(my_tuple)


my_list = ['Sam 24','Marry 34', 'Peter 25']

regularRDD = sc.parallelize(my_list)

pair_RDD_RDD = regularRDD.map(lambda s:(s.split(' ')[0], s.split(' ')[1]))
```

Creating pair RDDs

- Two common ways to create pair RDDs
    1. From a list of key-value tuple
    2. From a regular RDD
- Get the data into key/value form for paired RDD

```
my_tuple = [('Sam',23),('Marry',34),('Peter',25),]

pairRDD_tuple = sc.parallelize(my_tuple)


my_list = ['Sam 24','Marry 34', 'Peter 25']

regularRDD = sc.parallelize(my_list)

pair_RDD_RDD = regularRDD.map(lambda s:(s.split(' ')[0], s.split(' ')[1]))
```

Transformations on pair RDDs

- All regular transformations work on pair RDD
- Have to pass functions that operate on key value pairs rather than on individual elements
- Examples of paired RDD Transformations
  - reduceByKey(func): Combine values with the same key
  - groupByKey(): Group values with the same key
  - sortByKey(): Return an RDD sorted by the key
  - join(): Join two pair RDDs based on their key

Let's code

PySpark Data Manipulation