Big Data

Apache Spark - Pair RDDs

Prof. Jean Paul Barddal

Agenda

1 Pair RDDs

2 Transformações e Operações

3 Agregação em PairRDDs

4 Exercícios

Agenda

1 Pair RDDs

2 Transformações e Operações

3 Agregação em PairRDDs

4 Exercícios

Motivação

- Muitos casos demandam processamentos no formato (chave, valor)
- Examplos:
  - (CPF, Nome)
  - (Big Data, [Prof. Jean Paul, Felisberto, Ronaldo])
- Muitas vezes, precisamos realizar processamentos por chave (group by)
- Para facilitar o processamento deste tipo de dados, Spark possui uma estrutura chamada Pair RDD

PairRDD

- RDD cujos elementos estão no formato <chave, valor>
- Criação pode ser feita:
- A partir de uma lista de chaves e valores (tuplas)
- Com a transformação de um RDD tradicional em um PairRDD

Tupla

- No Python, PairRDDs usa tuplas disponíveis na linguagem:

```
1 t = ('pug', 2018)
2 key = t[0]
3 value = t[1]
```

PairRDD a partir de Tuplas

```
1 l = [('Jean', 1992), ('Fabricio', 1972), ('Lucas', 1994)]
2 prdd = sc.paralellize(l)
```

Criando PairRDDs a partir de RDDs

- Usando a função de map (mapToPair)
- Parâmetro: uma PairFunction que converte um valor único em uma tupla

```
1 def build_tuple(s):
2   vals = s.split(',')
3   return (vals[0], vals[1])
4
5 l = ['Jean,1992', 'Fabricio,1972', 'Lucas,1994']
6 rdd = sc.parallelize(l)
7 pairRDD = rdd.map(build_tuple)
```

Agenda

1 Pair RDDs

2 Transformações e Operações

3 Agregação em PairRDDs

4 Exercícios

Transformações e Operações em PairRDDs

- Suporta todas as transformações e operações como em RDDs tradicionais
- Contudo, funções são aplicadas a tuplas ao invés dos tipos básicos

filter

- Similar ao uso normal em RDDs
- Filtragem pode ocorrer sobre a tupla completa (chaves e/ou valores)

mapValues

- Inspirado na função de map
- Spark assume que a chave não vai ser alterada, portanto, essa função só permite alterar os valores

Atividade

- Vamos codificar o seguinte:
  - Carregamento dos dados dos aeroportos
  - Conversão do RDD em um PairRDD com <Cidade, País>
  - Filtrar aeroportos nos EUA
  - Converter todos os valores para caixa alta
  - Salvar os resultados em arquivo

Solução possível

```
1  def build_tuple(s):
2    vals = s.split(',')
3    return (vals[1], vals[3])
4  # data load
5  rdd = sc.textFile('airports.csv')
6  # conversion into pair rdd
7  prdd = rdd.map(build_tuple)
8  # filtering
9  prdd_usa = prdd.filter(lambda x: x[1] == 'United States')
10 # upper case conversion
11 prdd_usa_uc = prdd_usa.mapValues(lambda x: x.upper())
12 # data output
13 prdd_usa_uc.saveAsTextFile('usa_upper.txt')
```

Agenda

1 Pair RDDs

2 Transformações e Operações

3 Agregação em PairRDDs

4 Exercícios

Agregação: reduceByKey

- reduceByKey realiza uma operação de reduce em paralelo e por chave no PairRDD
- Cada reduce gera um valor único por chave
- Importante: reduceByKey não é uma operação, pois o resultado ainda é um PairRDD

Examplo - reduceByKey

- Vamos reimplementar o word count!
- Idéia: se o número de chaves (palavras) é muito grande, o código que fizemos anteriormente pode falhar, dado que countByValue envia todos os dados para o Driver Program
- Usando reduceByKey os dados ficam distribuídos, dado que ainda são um PairRDD

groupByKey

- Outra forma de agrupar dados por chave
- Se as chaves estiverem corretas, groupByKey irá agrupá-las
- Em uma estrutura com chave K e valores V, o resultado será PairRDD<K, List de V>
- Exemplo: agrupando os aeroportos por país

Exemplo - groupByKey

```
1  def build(s):
2    vals = s.split(',')
3    return (vals[3], vals[1])
4
5  # data load
6  rdd = sc.textFile('airports.csv')
7  # prdd conversion
8  prdd = rdd.map(build)
9  # grouping by key
10 grouped = prdd.groupByKey()
11 # analyzing the output
12 grouped.mapValues(list).collect()
```

sortBy e sortByKey

- Ordenação em PairRDDs
- Uma vez que um PairRDD esteja ordenado, todas as funções subsequentes manterão a ordem definida anteriormente
- Ordem crescente ou decrescente
- sortByKey(boolean ascending)

Agenda

1 Pair RDDs

2 Transformações e Operações

3 Agregação em PairRDDs

4 Exercícios

Exercício #1

- Usando o arquivo RealEstate.csv, calcule o preço médio das casas por número de quartos
- Formato do arquivo:
  - MLS
  - Location
  - Price (USD)
  - Number of rooms
  - Number of bathrooms
  - House size, in square feet
  - House price, per square foot
  - Sale type
- Apresentar os resultados ordenados de acordo com o número de quartos

Exercício #2

- Usando o arquivo bible.txt, codifique uma rotina usando Pair RDDs que calcule a ocorrência de cada palavra
- Apresente os resultados em ordem decrescente de acordo com o número de ocorrências
- Exemplo:
- (apple, 200)
- (shoes, 193)
- (bag, 176)
- ...