

Análise de Dados com Base em Processamento Massivo em Paralelo

Lista de Exercícios: Processamento Paralelo e Distribuído

Profa. Dra. Cristina Dutra de Aguiar

Observação:

Esta lista contém exercícios classificados como essenciais e complementares. A indicação da classificação de cada exercício é feita junto de sua definição. Recomenda-se fortemente que a lista de exercícios seja respondida antes de se consultar as respostas dos exercícios.

1. (Essencial) Transformações são operações que transformam um RDD (*Resilient and Distributed Datasets*) em outro RDD. No Spark, as transformações não são executadas imediatamente sobre os dados do RDD. Elas são anexadas ao grafo acíclico direcionado de transformações, o qual é executado apenas quando uma ação sobre o RDD em questão for solicitada. Essa característica é conhecida como *lazy-evaluation*.

Como resultado, operações podem ser executadas em conjunto, possibilitando otimizações. Por exemplo, os métodos *map()* e *reduceByKey()* são transformações, e eles somente são executados quando ocorrer uma ação, como quando o método *collect()* for chamado. Portanto, *map()*, *reduceByKey()* e *collect()* podem ser executados em conjunto.

No exemplo do contador de palavras, comumente utiliza-se o que é conhecido como *map-side reducer*, indicando que internamente o *Spark* usa uma estrutura de *HashMap* e faz o método de *reduceByKey()* localmente antes de enviar o resultado para a etapa de *reducing* propriamente dita. Isso é ilustrado por meio da seta azul na Figura 1.

Por outro lado, o *framework* Hadoop não provê operações com a característica de *lazy-evaluation*. Portanto, operações não são executadas em conjunto. Refaça a Figura 1 de forma que o exemplo do contador de palavras não considere essa característica.

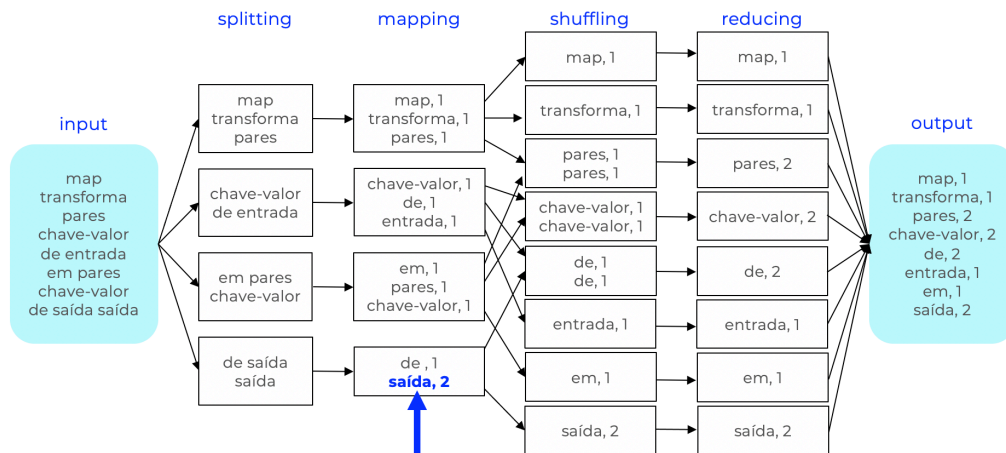


Figura 1: Exemplo de contador de palavras

2. (Essencial) Considere o código Python a seguir, o qual tem como objetivo contar quantas vezes uma determinada palavra aparece em um arquivo texto com 4 linhas de log.

```
error_lines = ["""
(2021-11-01 06:58:43) ERROR - Event with job id 1abc Failed,
(2021-11-01 06:58:47) ERROR - Event with job id 2abc Failed,
(2021-11-01 06:58:50) ERROR - Event with job id 3abc Failed,
(2021-11-01 06:59:43) INFO - Number of table lines is 102345.
"""]

output = spark. \
    parallelize(error_lines). \
    flatMap(lambda element: element.split(" ")). \
    filter(lambda element: True if element == 'ERROR' else False). \
    count()
```

Explique qual é a função dos métodos *flatMap()*, *filter()*, *count()* e *collect()*. Explique também qual o resultado da variável *output*.



3. (Essencial) Considere o cenário descrito a seguir.

A empresa BI Solutions está desenvolvendo uma ferramenta voltada à análise de dados com o objetivo de utilizá-la na classificação de sentimentos do mercado financeiro. Considere que uma equipe formada por engenheiras(os) de dados já realizou todo o processo ETL (*extract, transform, load*) de *tweets* históricos juntamente com o preço de ativos pré-determinados. Considere também que os dados estão armazenados em um *cluster* que contém o sistema de arquivos distribuídos HDFS (*Hadoop Distributed File System*). No momento, a equipe de analistas de *big data* deve realizar análises para o estudo de viabilidade do desenvolvimento de modelos de aprendizado de máquina baseados nos dados já capturados, com o objetivo de entender se ainda é necessária a realização de uma nova etapa para aquisição e coleta de dados de novas fontes.

Os dados de *tweets* estão armazenados conforme a seguinte estrutura de tabela:

tweet_pk	date	tweet_text
10000	01-10-2021	o preco do bitcoin vai subir
10001	01-10-2021	bitcoin esta decolando e vai para a lua
10002	02-11-2021	tenha bitcoin nao pelos valores e sim por valores
10003	03-11-2021	a binance suspendeu todos os saques de bitcoin
10004	03-11-2021	ethereum vai substituir o bitcoin
10005	03-11-2021	btc to the moon

Onde:

- **tweet_pk**: representa uma chave única para cada *tweet*;
- **date**: representa a data em que o *tweet* foi postado;
- **tweet_text**: representa o texto do *tweet* postado.

A primeira análise a ser realizada pela equipe de analistas de *big data* tem como objetivo responder o seguinte questionamento: “Qual o tamanho médio dos *tweets* capturados?”

Considere que você foi contratado para fazer parte da equipe e que recebeu como atividade a seguinte tarefa: “Realize uma modelagem do dado usando como base o modelo MapReduce, definindo o formato dos pares chave-valor e como as etapas de *map* e *reduce* funcionarão”. A tarefa deve responder de forma correta ao questionamento recebido pela equipe.



4. (Essencial) Considere a Figura 2, a qual representa um *cluster* de computadores que segue a arquitetura do sistema de arquivos distribuídos HDFS (*Hadoop Distributed File System*). Ao lado de cada quadrado, existe uma numeração que representa o componente dentro daquele quadrado. Responda às questões a seguir, utilizando como base essa numeração.

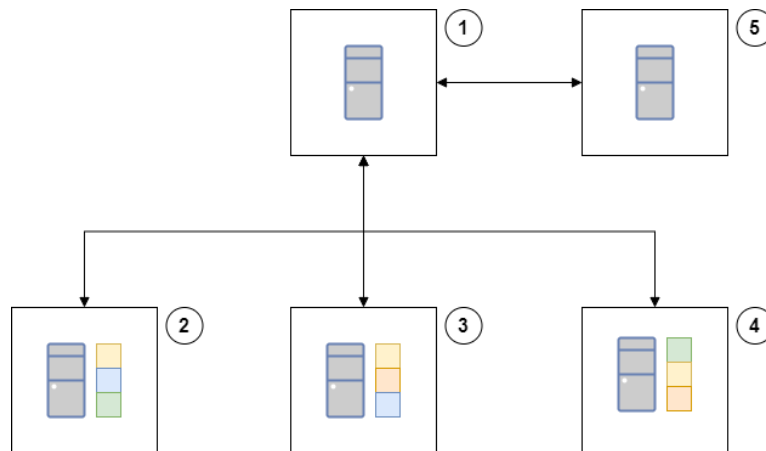


Figura 2: Representação de um *cluster* de computadores que segue a arquitetura do HDFS.

- (a) Qual componente representa o NameNode primário?
 - (b) Quais componentes representam os DataNodes?
 - (c) Qual componente representa o NameNode secundário?
 - (d) Quais são os componentes que são considerados nós escravos?
 - (e) Quais são os componentes responsáveis pela manutenção dos metadados sobre os dados armazenados?
 - (f) Quais são os componentes responsáveis por fornecer os dados solicitados nas operações de leitura e escrita?
5. (Complementar) Descreva um cenário que contenha um problema/case/necessidade de negócio. Em seguida, proponha uma solução em um ambiente computacional baseado em processamento paralelo e distribuído. Devem ser justificadas as escolhas realizadas, como o tipo de *cluster* empregado ou a descrição das características principais do ambiente de computação em nuvem (ex.: tipo de nuvem, modelos de serviços, modelos de implantação).