Aula1: Introdução ao Apache Spark

Disciplina: Tópicos de Big Data
Professor: Henrique Mota

1. O Problema: A Biblioteca Gigante

Imaginem que vocês são responsáveis por uma biblioteca com milhões de livros. Um dia, recebem uma tarefa simples: contar quantas vezes a palavra "conhecimento" aparece em *todos os livros da seção de filosofia*.
Como vocês fariam isso?

- **Abordagem 1 (Uma só pessoa):** Você, sozinho, pega o primeiro livro, lê página por página, e anota cada vez que a palavra aparece. Depois, pega o segundo livro, o terceiro, e assim por diante. Isso levaria... talvez anos? É um processo **lento e ineficiente**.
- **Abordagem 2 (Trabalho em Equipe):** E se você pudesse chamar 100 ajudantes? Você poderia dar uma pilha de livros para cada um e pedir: "Contem a palavra 'conhecimento' nos seus livros e me tragam apenas o total". Depois, você só precisaria somar os 100 resultados. Bem mais rápido, certo?

Essa segunda abordagem é a essência do **processamento distribuído**, e é exatamente aqui que o Apache Spark brilha.

2. O que é o Apache Spark?

O Apache Spark é uma plataforma de computação em cluster de código aberto.
Vamos traduzir isso:

- **Plataforma de computação:** É uma ferramenta para processar dados.
- **Em cluster:** Ela não usa apenas um computador, mas vários (os "ajudantes" da nossa biblioteca), que trabalham juntos como se fossem um só supercomputador.
- **Código aberto:** É gratuito e mantido por uma comunidade global.

A grande vantagem do Spark é sua **velocidade**. Ele faz a maior parte do processamento na memória RAM dos computadores do cluster, que é muito mais rápida do que ler e escrever em discos rígidos tradicionais.
Para entender como ele funciona, precisamos conhecer sua estrutura principal.

3. A Estrutura Fundamental do Spark: RDDs

O coração do Spark é o **RDD (Resilient Distributed Dataset)**, ou Conjunto de Dados Distribuído Resiliente.
Vamos quebrar o nome:

- **Conjunto de Dados (Dataset):** É a nossa coleção de dados. Pense nos livros da biblioteca. No mundo digital, pode ser um arquivo de texto, uma planilha, uma tabela de banco de dados, etc.
- **Distribuído (Distributed):** O Spark pega esse conjunto de dados e o divide em pedaços menores, espalhando-os pelos vários computadores (nós) do cluster. Cada "ajudante" da biblioteca fica com uma parte dos livros.

- **Resiliente (Resilient):** Se um dos seus ajudantes (um computador do cluster) ficar doente e for para casa (falhar), o Spark é inteligente o suficiente para pegar a pilha de livros dele e entregar para outro ajudante continuar o trabalho. Ele se recupera de falhas automaticamente.

Em resumo, um RDD é uma coleção de dados dividida e distribuída entre várias máquinas, com capacidade de se recuperar de falhas.

4. As Operações do Spark: Transformações e Ações

Com os dados distribuídos em RDDs, o Spark realiza dois tipos de operações:

a) Transformações (Transformations)

São operações que criam um *novo* RDD a partir de um existente. Elas são "preguiçosas" (*lazy*), o que significa que o Spark não as executa imediatamente. Ele apenas anota o que precisa ser feito, como se estivesse criando um plano de trabalho.

- **Exemplo:** `map()`
  - **O que faz?** Aplica uma função a cada elemento do RDD.
  - **Analogia:** Você diz aos seus ajudantes: "Para cada livro, criem um novo documento contendo apenas as frases com a palavra 'conhecimento'". Eles ainda não fazem nada, só entendem a instrução.
- **Exemplo:** `filter()`
  - **O que faz?** Cria um novo RDD contendo apenas os elementos que satisfazem uma condição.
  - **Analogia:** Você diz: "Peguem apenas os livros escritos depois do ano 2000".

b) Ações (Actions)

São operações que disparam a execução de todas as transformações planejadas e retornam um resultado final para o "chefe" (o programa principal). É aqui que o trabalho acontece de verdade!

- **Exemplo:** `count()`
  - **O que faz?** Conta o número de elementos no RDD.
  - **Analogia:** Você grita: "Ok, pessoal, executem o plano e me digam quantos livros sobraram!". Os ajudantes filtram os livros, extraem as frases e, no final, você recebe o número total.
- **Exemplo:** `collect()`
  - **O que faz?** Traz todos os elementos do RDD para o programa principal.
  - **Cuidado:** Use apenas com dados pequenos, ou você pode sobrecarregar a máquina principal (imagine 100 ajudantes tentando te entregar milhões de livros ao mesmo tempo!).

5. Exemplo Prático: Contando Palavras (O "Hello, World!" do Big Data)

Vamos ver como nosso problema da biblioteca seria resolvido em código Spark (usando Python, que é uma das linguagens mais comuns para interagir com o Spark).

Python

```python
from pyspark.sql import SparkSession

# definir a palavra alvo
target = 'Alma'
# criar sessão Spark
spark =
SparkSession.builder.appName("count_word").getOrCreate()

# ler o arquivo de texto e criar RDD
rdd = spark.sparkContext.textFile('livros.txt')

# transformação: Quebrar cada linha em palavras
rdd = rdd.flatMap(lambda linha: linha.split(" "))
# transformação: Filtrar apenas a palavra que nos interessa
rdd = rdd.filter(lambda palavra: palavra == target)
# ação: Contar quantas vezes a palavra aparece
count = rdd.filter(lambda w: w == target).count()

print(f"A palavra '{target}' apareceu {count} vezes no arquivo
'livros.txt'.")

# parar a sessão Spark
spark.stop()
```

O que aconteceu aqui?

1. O Spark distribuiu o arquivo `livros.txt` entre as máquinas.
2. Ele anotou que precisava quebrar as linhas em palavras (`flatMap`).
3. Anotou também que precisava filtrar e manter apenas a palavra "conhecimento" (`filter`).
4. Quando chamamos `count()`, ele executou todo esse plano em paralelo nas diferentes máquinas e somou os resultados para nos dar o total.

Conclusão da Aula

Hoje, aprendemos que:

1. **Apache Spark** é uma ferramenta para processamento de grandes volumes de dados de forma rápida e distribuída.
2. Sua principal estrutura de dados é o **RDD**, uma coleção de dados distribuída e tolerante a falhas.
3. O Spark trabalha com **Transformações** (o plano de trabalho, que é preguiçoso) e **Ações** (que executam o plano e retornam um resultado).
4. Com exemplos simples como contar palavras, podemos ver o poder de dividir um grande problema em tarefas menores e executá-las em paralelo.

Nas próximas aulas, vamos explorar estruturas mais modernas como DataFrames (que se parecem com planilhas e são mais otimizados que RDDs para muitas tarefas), o módulo de SQL do Spark e como ele se integra a outras ferramentas do ecossistema de Big Data. Alguma pergunta?