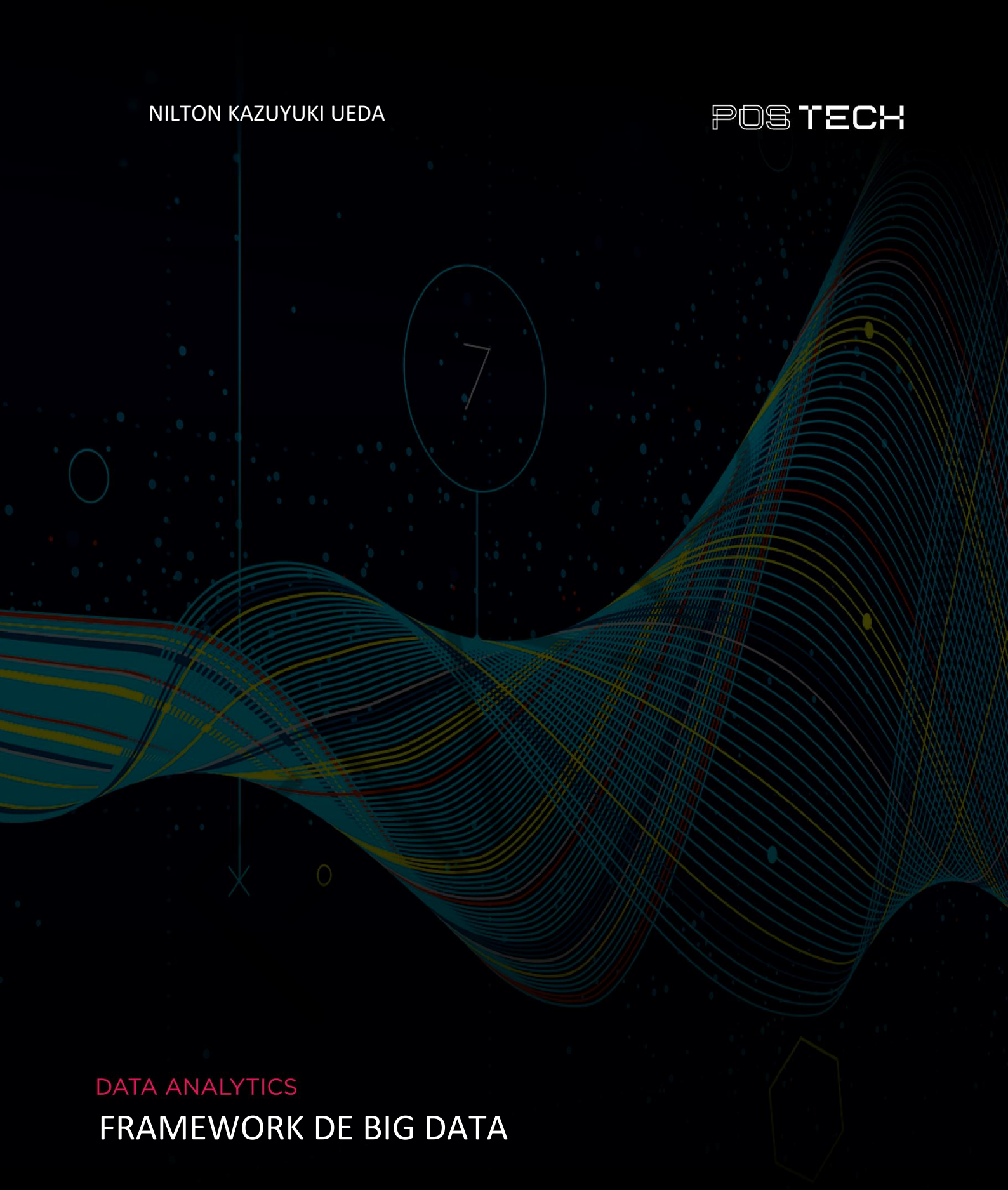
NILTON KAZUYUKI UEDA



DATA ANALYTICS

FRAMEWORK DE BIG DATA

AULA 01

SUMÁRIO

O QUE VEM POR AÍ?

Nos últimos anos, a quantidade de dados gerada e armazenada por empresas e organizações tem crescido exponencialmente. Para lidar com esse enorme volume de informações surge o Apache Spark, uma poderosa estrutura de processamento distribuído que revolucionou a maneira como lidamos com análise e processamento de dados em escala.

Uma aula de introdução ao Apache Spark é uma jornada fascinante rumo ao mundo do processamento de dados em tempo real. Durante ela, você terá a oportunidade de mergulhar nos conceitos fundamentais do Spark e entender como ele pode ser aplicado para resolver problemas complexos de maneira eficiente.

No decorrer da aula, você descobrirá como o Spark supera as limitações de outras ferramentas de processamento de dados, aproveitando o poder do processamento distribuído e da memória em massa. Você aprenderá sobre a arquitetura do Spark, incluindo o modelo de programação baseado em RDDs (Resilient Distributed Datasets), que permite a execução de operações em paralelo e a tolerância a falhas.

Além disso, exploraremos as diversas funcionalidades do Spark, desde a manipulação e transformação de dados até o processamento de streaming e aprendizado de máquina. Durante as atividades práticas, você terá a chance de experimentar com a linguagem Scala ou Python, que são as principais linguagens de programação utilizadas com o Spark, e aplicar seus conhecimentos em casos de uso reais.

Prepare-se para adentrar o emocionante mundo do Apache Spark, no qual você poderá desbravar as fronteiras do processamento de dados em escala, transformando dados brutos em informações valiosas para impulsionar decisões inteligentes e inovação. Esta aula de introdução é o ponto de partida ideal para explorar as possibilidades infinitas do Spark e embarcar em uma jornada de aprendizado contínuo em análise de Big Data.

HANDS ON

Durante essa aula, teremos a oportunidade de aplicar os conceitos fundamentais do Spark em atividades práticas. Vamos aprender sobre sua arquitetura e o modelo de programação baseado em RDDs (Resilient Distributed Datasets), que nos permite executar operações em paralelo e lidar com falhas de maneira eficiente. Além disso, vamos explorar as diversas funcionalidades do Spark, como manipulação e transformação de dados, processamento de streaming e até mesmo uso de aprendizado de máquina.

SAIBA MAIS

O QUE É O SPARK?

O Spark é um poderoso mecanismo de processamento de código aberto construído em torno de velocidade, facilidade de utilização e análises sofisticadas. Ele foi originalmente desenvolvido na Universidade de Berkeley em 2009 e é 100% open source, sendo hospedado no Apache Software Foundation independente de fornecedor.

Potências da Internet como Netflix, Yahoo e eBay o implementaram em grande escala com o objetivo de processar coletivamente múltiplos petabytes de dados em clusters de mais de 8.000 nós. Ele rapidamente se tornou a maior comunidade open source em dados grandes, com mais de 1000 colaboradores de mais de 250 organizações.

O projeto veio para resolver problemas de performance e processamento paralelo, criando assim um poderoso ambiente de execução em memória nunca visto anteriormente. Rapidamente adotado, se transformou em base para diversas aplicações como as de Big Data, Machine Learning, Streaming de Dados e SQL, além de processamento massivo de dados que antes eram inviáveis de serem consumidos.

HADOOP E SPARK

Para tornar a comparação justa, aqui vamos contrastar o Spark com o Hadoop MapReduce, uma vez que ambos são responsáveis pelo processamento de dados. De fato, a principal diferença entre eles está na abordagem do processamento: o Spark pode fazer isso na memória, enquanto o Hadoop MapReduce precisa ler e gravar em um disco. Como resultado, a velocidade de processamento difere significativamente. O Spark pode ser até 100 vezes mais rápido. No entanto, o volume de dados processados também difere: o Hadoop MapReduce é capaz de trabalhar com conjuntos de dados muito maiores do que o Spark.

Agora, vamos dar uma olhada nas tarefas para as quais cada estrutura é adequada.

O Hadoop MapReduce é bom para:

- Processamento linear de grandes conjuntos de dados.

O Hadoop MapReduce permite o processamento paralelo de grandes quantidades de dados. Ele divide um grande fragmento em partes menores para serem processadas separadamente em diferentes nós de dados e reúne automaticamente os resultados nos vários nós para retornar um único resultado. Caso o conjunto de dados resultante seja maior que a RAM disponível, o Hadoop MapReduce pode superar o Spark.

- Solução econômica, se não houver resultados imediatos.

O Hadoop considera o MapReduce uma boa solução se a velocidade de processamento não for crítica. Por exemplo: se o processamento de dados puder ser realizado durante a noite, faz sentido considerar o uso do MapReduce do Hadoop.

O Spark é bom para:

- Processamento rápido de dados.

O processamento na memória torna o Spark mais rápido que o Hadoop MapReduce — até 100 vezes para dados na RAM e até 10 vezes para dados armazenados.

- Processamento iterativo.

Se a tarefa for processar dados repetidamente, o Spark elimina o Hadoop MapReduce. Os RDDs (Distributed Datasets) resilientes do Spark permitem várias operações de mapa na memória, enquanto o Hadoop MapReduce tem que gravar resultados provisórios em um disco.

- Processamento quase em tempo real.

Se uma empresa precisar de insights imediatos, deverá optar pelo Spark e pelo processamento na memória.

- Processamento gráfico.

O modelo computacional do Spark é bom para cálculos iterativos típicos no processamento de gráficos. Além disso, o Apache Spark tem o GraphX - uma API para computação gráfica.

- Aprendizado de máquina.

O Spark possui MLlib — uma biblioteca de aprendizado de máquina integrada, enquanto o Hadoop precisa de um terceiro para fornecê-lo. O MLlib possui algoritmos prontos que também são executados na memória.

- Juntando conjuntos de dados.

Devido à sua velocidade, o Spark pode criar todas as combinações mais rapidamente, embora o Hadoop possa ser melhor se for necessário juntar conjuntos de dados muito grandes que requeiram muito embaralhamento e classificação.

CARACTERÍSTICAS DO SPARK

Como pudemos observar nos tópicos anteriores, essas são as principais características do Apache Spark:

- Modelo de Computação Distribuída: o Spark permite a computação distribuída, o que significa que ele pode processar grandes volumes de dados dividindo-os em várias partes menores e executando essas partes em diferentes nós de um cluster.
- Motor de Processamento em Memória: o Spark possui um motor de processamento em memória que armazena dados em memória principal, reduzindo a necessidade de acessar o disco para leitura e gravação de dados. Isso proporciona um desempenho significativamente mais rápido em comparação com sistemas que dependem exclusivamente do acesso ao disco.
- RDD (Resilient Distributed Datasets): o RDD é a estrutura de dados fundamental no Spark. Ele é uma coleção distribuída e tolerante a falhas de objetos imutáveis que podem ser processados em paralelo. Os RDDs oferecem uma abstração de programação flexível e eficiente para manipulação de dados distribuídos.
- Transformações e Ações: o Spark fornece um conjunto rico de transformações e ações para manipular RDDs. As transformações são operações que criam um novo RDD a partir de um RDD existente (por

exemplo map, filter, reduceByKey), enquanto as ações são operações que retornam resultados ou gravam dados (por exemplo count, collect, save).

- Suporte a várias linguagens de programação: o Spark oferece suporte a várias linguagens de programação, incluindo Scala, Java, Python e R. Isso permite que especialistas em desenvolvimento usem a linguagem de programação de sua escolha para escrever aplicativos Spark.
- Bibliotecas Integradas: o Spark vem com bibliotecas integradas para processamento de dados estruturados (Spark SQL), processamento de fluxo de dados em tempo real (Spark Streaming), aprendizado de máquina (Spark MLlib) e processamento de gráficos (GraphX). Essas bibliotecas estendem as capacidades do Spark para diferentes casos de uso.
- Suporte à integração com outros sistemas: o Spark pode se integrar com outros sistemas de armazenamento e processamento de dados, como Hadoop, HBase, Hive, Cassandra e muitos outros. Isso permite que os(as) usuários(as) acessem e processem dados de várias fontes usando o Spark.
- Otimizações de Desempenho: o Spark incorpora várias otimizações de desempenho para melhorar a eficiência e o tempo de resposta das operações de processamento. Isso inclui otimizações de execução de consultas, planejamento automático de tarefas, particionamento de dados, coleta de dados em paralelo e cache em memória.
- Modo Cluster e Modo Local: o Spark pode ser executado em modo cluster, distribuindo o processamento em vários nós de um cluster, ou em modo local, permitindo que especialistas em desenvolvimento executem e testem aplicativos Spark em uma única máquina.

Essas são apenas algumas das principais características técnicas do Apache Spark. Ele é um projeto em constante evolução e novos recursos e aprimoramentos são adicionados regularmente à medida que a comunidade de especialistas em desenvolvimento contribui para o projeto.

O ECOSSISTEMA DO SPARK

Além da API do Spark, existem bibliotecas adicionais que fazem parte do seu ecossistema e fornecem capacidades adicionais para as áreas de análise de Big Data e aprendizado de máquina. De acordo com o Leonardo Afonso Almorim (2021), estas bibliotecas incluem:

- Spark Streaming:

O Spark Streaming pode ser usado para processar dados de streaming em tempo real baseado na computação de microbatch. Para isso se utiliza o DStream, que é basicamente uma série de RDD para processar os dados em tempo real.

- Spark SQL:

O Spark SQL fornece a capacidade de expor os conjuntos de dados Spark através de uma API JDBC. Isso permite executar consultas no estilo SQL sobre esses dados usando ferramentas tradicionais de BI e visualização. Além disso, também permite que usuários usem ETL para extrair seus dados em diferentes formatos (como JSON, Parquet ou um banco de dados), transformá-los e expô-los para consultas ad-hoc.

- Spark MLlib:

MLlib é a biblioteca de aprendizado de máquina do Spark que consiste em algoritmos de aprendizagem, incluindo classificação, regressão, clustering, filtragem colaborativa e redução de dimensionalidade.

- Spark GraphX:

GraphX é uma nova API do Spark para grafos e computação paralela. Em alto nível, o GraphX estende o Spark RDD para grafos. Para apoiar a computação de grafos, o GraphX expõe um conjunto de operadores fundamentais (como por exemplo subgrafos e vértices adjacentes), bem como uma variante optimizada do Pregel. Além disso, o GraphX inclui uma crescente coleção de algoritmos para simplificar tarefas de análise de grafos.

COLOCANDO A MÃO NA MASSA COM APACHE SPARK

Vamos escolher com qual IDE iremos trabalhar. No caso, escolheremos o Google Colab, por ser a interface mais acessível e de fácil escolha.

Para obter o acesso ao Google Colab, você precisa ter uma conta cadastrada no Gmail.

- Passo 1: acesse o Google Colab.

  Abra um navegador da web e vá para o site do Google Colab.

- Passo 2: faça login na sua conta do Google.

Se você já estiver logado(a) na sua conta do Google, o Colab será aberto diretamente.

Caso contrário, faça login na sua conta do Google para acessá-lo.

- Passo 3: Crie um novo notebook

No Google Colab, clique em "Novo notebook" ou vá para "Arquivo" > "Novo notebook Python".

- Passo 4: Nomeie o seu notebook

Após criar um novo notebook, você pode escolher um nome para ele, digitando-o na parte superior da página.

Agora vamos inicializar o Spark: primeiramente, será necessário realizar a setagem de algumas variáveis e também a instalação de algumas bibliotecas para uso do Apache Spark. Para isso, execute o código a seguir:

```
!sudo apt update
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
#Check this site for the latest download link
https://www.apache.org/dyn/closer.lua/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz
!wget -q https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz
!tar xf spark-3.2.1-bin-hadoop3.2.tgz
!pip install -q findspark
!pip install pyspark
!pip install py4j

import os
import sys
# os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
```

```
# os.environ["SPARK_HOME"] = "/content/spark-3.2.1-bin-hadoop3.2"


import findspark
findspark.init()
findspark.find()

import pyspark

from pyspark.sql import DataFrame, SparkSession
from typing import List
import pyspark.sql.types as T
import pyspark.sql.functions as F

spark= SparkSession \
       .builder \
       .appName("Our First Spark Example") \
       .getOrCreate()

spark
```

Após a execução deste código, será possível visualizar três principais variáveis do Spark, sendo:

- Version.
- Master.
- AppName.

Agora, vamos importar um arquivo público de dados da COVID-19 em formato CSV:

```
import requests
path = "https://raw.githubusercontent.com/owid/covid-19-
data/master/public/data/owid-covid-data.csv"
req = requests.get(path)
url_content = req.content

csv_file_name = 'owid-covid-data.csv'
csv_file = open(csv_file_name, 'wb')

csv_file.write(url_content)
csv_file.close()
```

```
df = spark.read.csv('/content/'+csv_file_name, header=True,
inferSchema=True)
```

Com isso, está realizada a importação deste arquivo.

Vamos começar a realizar algumas explorações básicas dentro do arquivo público que acabamos de importar.

O comando printSchema() irá te apresentar um resumo básico dos campos dentro do conjunto de dados que acabou de ser importado com o uso do Spark para um dataframe:

```
#Viewing the dataframe schema
df.printSchema()
```

A função a seguir irá converter o tipo de dados da coluna “date” de string para date:

```
#Converting a date column
df.select(F.to_date(df.date).alias('date'))
```

O próximo comando irá retornar para você um resumo de todas as colunas com algumas agregações como COUNT, MEAN, STDDEV, MIN e MAX:

```
#Summary stats
df.describe().show()
```

Este comando irá retornar uma simples agregação de SUM, sendo agrupada pelo campo de “location”:

```
#Simple Group by Function
df.groupBy("location").sum("new_cases").orderBy(F.desc("sum(new_cases)"
)).show(truncate=False)
```

O QUE VIMOS NESTA AULA?

Nesta aula de introdução ao Apache Spark, exploramos essa poderosa estrutura de processamento distribuído, que tem revolucionado a análise e o processamento de dados em larga escala.

Durante a aula, mergulhamos nos conceitos fundamentais do Spark, aprendendo sobre sua arquitetura e o modelo de programação baseado em RDDs (Resilient Distributed Datasets).

Descobrimos como o Spark supera as limitações de outras ferramentas, aproveitando o processamento distribuído e a memória em massa. Também exploramos as funcionalidades deste framework, como manipulação de dados, processamento de streaming e aprendizado de máquina, por meio de atividades práticas com as linguagens Scala ou Python.

Essa aula é o ponto de partida para explorar as possibilidades do Spark e embarcar em uma jornada de aprendizado em análise de Big Data.

REFERÊNCIAS

AFONSO, Leonardo Amorim. Introdução ao Spark com Pyspark. Disponível em: https://sol.sbc.org.br/livros/index.php/sbc/catalog/download/80/346/605-1?inline=1. Acesso em: 11 jul 2023.

APACHE STARK. **Documentação Oficial Apache Spark**. Disponível em: https://spark.apache.org/documentation.html Acesso realizado em 28/05/2023. Acesso em: 20 jun. 2023.

GARCIA, M. **Spark, Saiba mais sobre esse poderoso framework**. 2022. Disponível em: https://cetax.com.br/conheca-mais-sobre-o-framework-apache-spark/. Acesso em: 20 jun. 2023.

GIMINO, A. **Spark vs. Hadoop MapReduce**: Qual estrutura de big data escolher. Disponível em: https://medium.com/mangue-data/spark-vs-hadoop-mapreduce-qual-estrutura-de-big-data-escolher-b8927de07f7e. Acesso em: 20 jun 2023.

PENCHIKALA, S. **Big Data com Apache Spark Parte 1 – Introdução**. 2015. Disponível em: https://www.infoq.com/br/articles/apache-spark-introduction/. Acesso em: 20 jun. 2023.

RELVA, C. **Apache Spark**. 2015. Disponível em: https://www.ime.usp.br/~gold/cursos/2015/MAC5742/reports/ApacheSpark.pdf. Acesso em: 20 jun. 2023.

WANG, J. **Introduction to Apache Spark**. 2020. Disponível em: https://towardsdatascience.com/1-introduction-to-apache-spark-299db7a4b68d. Acesso em: 20 jun. 2023.

PALAVRAS-CHAVE

Palavras-chave: Big Data, Spark, Dados, Processamento, MapReduce, Hadoop.