

Análise de dados de alta dimensão utilizando Apache Spark com R

Trabalho de Conclusão de Curso II

Daniel dos Santos

20 de Setembro de 2021

Universidade Federal Fluminense

1. Objetivos
2. Big Data
3. Sobre o Spark
4. MapReduce
5. Spark vs Hadoop MapReduce
6. Tolerância à Falhas
7. Ecossistema Spark
8. SparkR
9. Aplicação
10. Conclusões
11. Referências

Objetivos

Objetivos

1. Primeira etapa:

- Apresentar o Spark;
- Guiar o leitor a como utilizar o Spark junto a linguagem R;

2. Segunda etapa:

- Aplicar os conceitos, funções e ferramentas mostradas em um banco de dados.

Big Data

Desafios do Big Data

1. Grande volume de dados;

Desafios do Big Data

1. Grande volume de dados;
2. Dados em tempo real (*streaming*);

Desafios do Big Data

1. Grande volume de dados;
2. Dados em tempo real (*streaming*);
3. Variedade nas fontes de dados;

Desafios do Big Data

1. Grande volume de dados;
2. Dados em tempo real (*streaming*);
3. Variedade nas fontes de dados;
4. Trazer valor aos dados.

Desafios do Big Data

Douglas Laney e os "V's" do Big data.

1. Grande volume de dados;
2. Dados em tempo real (*streaming*);
3. Variedade nas fontes de dados;
4. Trazer valor aos dados.

Desafios do Big Data

Douglas Laney e os "V's" do Big data.

1. Grande volume de dados; **Volume**
2. Dados em tempo real (*streaming*); **Velocidade**
3. Variedade nas fontes de dados; **Variedade**
4. Trazer valor aos dados. **Valor**

Sobre o Spark

Sobre o Spark

1. Criado para trazer melhorias ao **MapReduce**. Com o intuito de melhorar e facilitar análise de dados de grande dimensão;

Sobre o Spark

1. Criado para trazer melhorias ao **MapReduce**. Com o intuito de melhorar e facilitar análise de dados de grande dimensão;
2. Desenvolvido em 2009 em Berkley, Universidade da Califórnia;

Sobre o Spark

1. Criado para trazer melhorias ao **MapReduce**. Com o intuito de melhorar e facilitar análise de dados de grande dimensão;
2. Desenvolvido em 2009 em Berkley, Universidade da Califórnia;
3. Doado à Fundação Apache em 2010;

Sobre o Spark

1. Criado para trazer melhorias ao **MapReduce**. Com o intuito de melhorar e facilitar análise de dados de grande dimensão;
2. Desenvolvido em 2009 em Berkley, Universidade da Califórnia;
3. Doado à Fundação Apache em 2010;
4. Utilizado em diversas instituições como, Yahoo!, IBM, Huawei, Alibaba, Tencent, etc.



1. *Framework* de código aberto;



1. *Framework* de código aberto;
2. Escalável;



1. *Framework* de código aberto;
2. Escalável;
3. Tolerante à falhas;



1. *Framework* de código aberto;
2. Escalável;
3. Tolerante à falhas;
4. Disponível em diversas linguagens, em especial o R.

MapReduce

MapReduce

MapReduce é um paradigma computacional que visa processar informações em duas etapas, nomeadas de **Map** e **Reduce**.

- **Map**: Processar um conjunto de valores e transformá-los em valores intermediários;

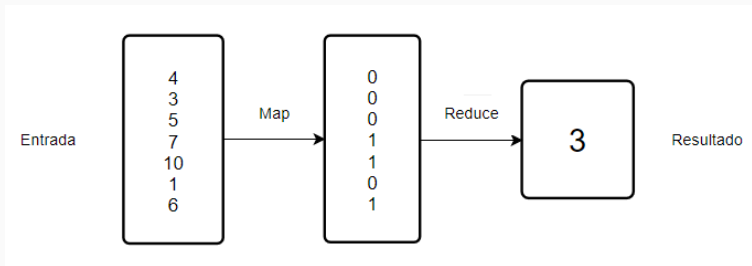
MapReduce

MapReduce é um paradigma computacional que visa processar informações em duas etapas, nomeadas de **Map** e **Reduce**.

- **Map**: Processar um conjunto de valores e transformá-los em valores intermediários;
- **Reduce**: Resumir os valores transformados, de forma que gerem o resultado esperado.

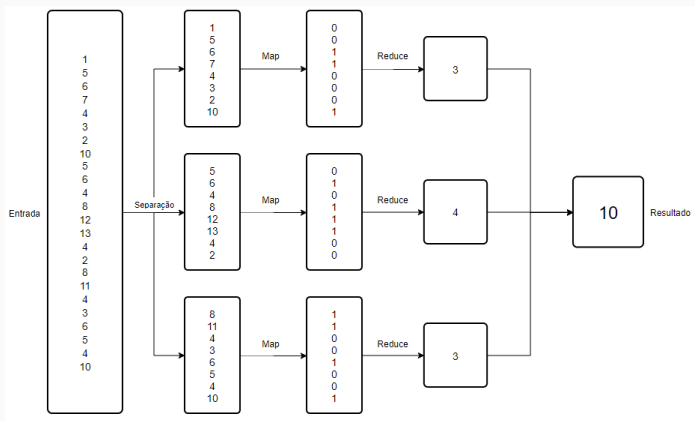
MapReduce

Suponha-se que deseja-se calcular a quantidade de valores maiores do que 5 em um conjunto de dados, o diagrama abaixo exemplifica a utilização do **MapReduce**.



MapReduce Paralelizado

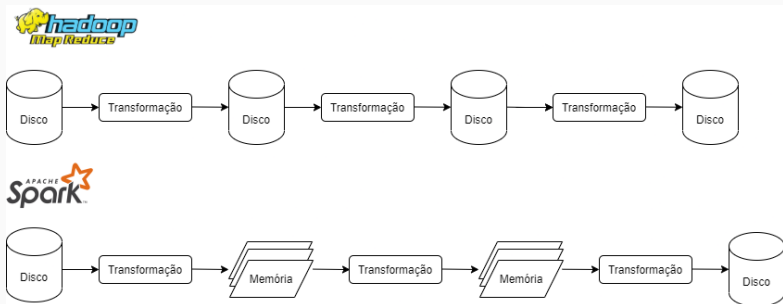
Este paradigma é facilmente escalável, já que o processamento dos dados pode ser feito em paralelo, assim o **Hadoop MapReduce** foi capaz de paralelizar uma série de ações.



Spark vs Hadoop MapReduce

Spark vs Hadoop MapReduce

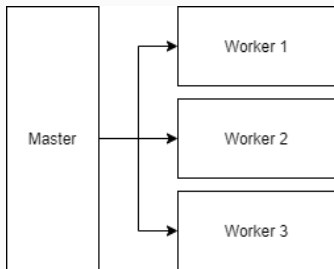
Através da implementação dos **RDDs**, o Spark é capaz de realizar as transformações necessárias sem salvar em disco. O que se mostrou uma vantagem em relação ao Hadoop MapReduce.



Tolerância à Falhas

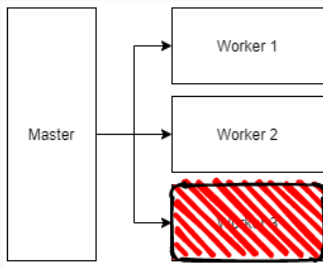
Tolerância à Falhas

Tolerância à falhas é a característica que define a capacidade de um sistema se recuperar e não perder informações, mesmo após a ocorrência de problemas no fluxo.



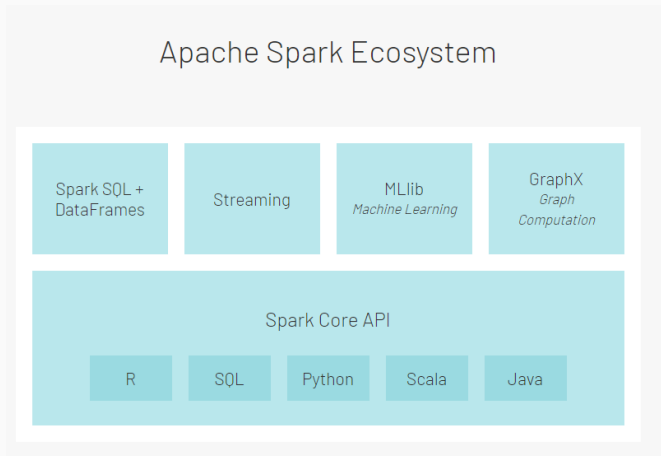
Tolerância à Falhas

Tolerância à falhas é a característica que define a capacidade de um sistema se recuperar e não perder informações, mesmo após a ocorrência de problemas no fluxo.



Ecosystema Spark

Ecosistema Spark



O **Spark Core** é o seu núcleo, a parte mais importante do ecossistema, que permitiu que outras componentes fossem construídas sobre ele.

Permite a utilização da linguagem **SQL** para realizar consultas a bancos de dados no Spark.

Spark Streaming

Esta extensão do **Spark Core**, permite o processamento de dados em streaming, ou seja, em tempo real, oferecendo escalabilidade, alto rendimento e tolerância à falhas.



Spark Streaming

Esta extensão do **Spark Core**, permite o processamento de dados em streaming, ou seja, em tempo real, oferecendo escalabilidade, alto rendimento e tolerância à falhas.



Volume; Variedade; Velocidade.

Spark MLlib ou *Spark Machine Learning Library* tem como objetivo trazer toda praticidade e velocidade do Spark para a aplicação de aprendizado de máquina de forma escalável.

Alguns algoritmos presentes no **Spark MLlib**:

1. **Classificação**: Regressão logística, máquina de vetores de suporte, naive Bayes, ...
2. **Regressão**: Regressão linear generalizada, ...
3. **Clusterização**: Modelo de Mistura de Gaussianas, K-Means, ...
4. **Modelagem de tópicos**: Alocação Latente de Dirichlet, ...

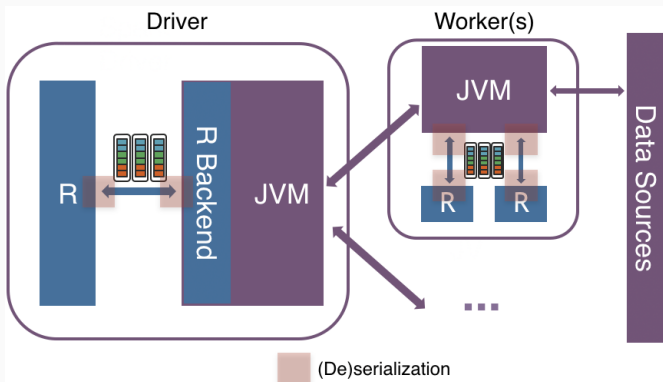
Spark GraphX é a componente que possui diversas ferramentas para o processamento e análise de grafos complexos.

SparkR

Através do pacote **SparkR** é possível trazer diversas ferramentas do Spark para dentro da linguagem R.

SparkR

O **SparkR** faz com que o **R** se comunique com o Spark através de uma **API**.



ENEM 2016, 166 colunas e cerca de 8 milhões de linhas (aproximadamente 6GB).

Objetivo: Executar uma *query* que conta a quantidade de pessoas por gênero.

Exemplos no SparkR - ENEM

```
1 SPARK_HOME = "D:/Spark"
2 Sys.setenv(SPARK_HOME=SPARK_HOME)
3 library(SparkR,
4         lib.loc=c(file.path(Sys.getenv("SPARK_HOME"), "R", "lib")))
5
6 sparkR.session(master = "local[3]",
7               sparkConfig = list(spark.driver.memory = "2g"))
8
9 data = read.df(
10   path = "../data/microdados_enem_2016.csv",
11   source = "csv",
12   delimiter = ";",
13   header = TRUE
14 )
15
16 createOrReplaceTempView(data, "enem")
17 collect(
18   sql(
19     "
20     SELECT
21       TP_SEXO,
22       COUNT(*) AS n
23     FROM
24       enem
25     GROUP BY
26       TP_SEXO
27     "
28   )
29 )
```

Exemplos no SparkR - ENEM

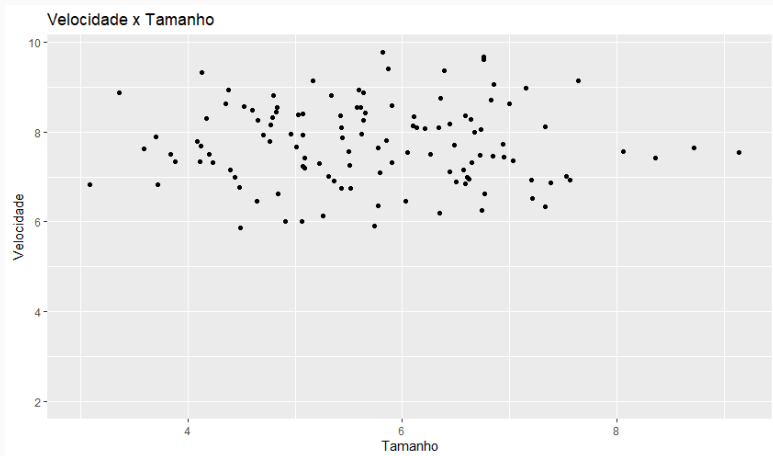
Gênero	Contagem
Masculino	3644728
Feminino	4982639

Exemplo do SparkR

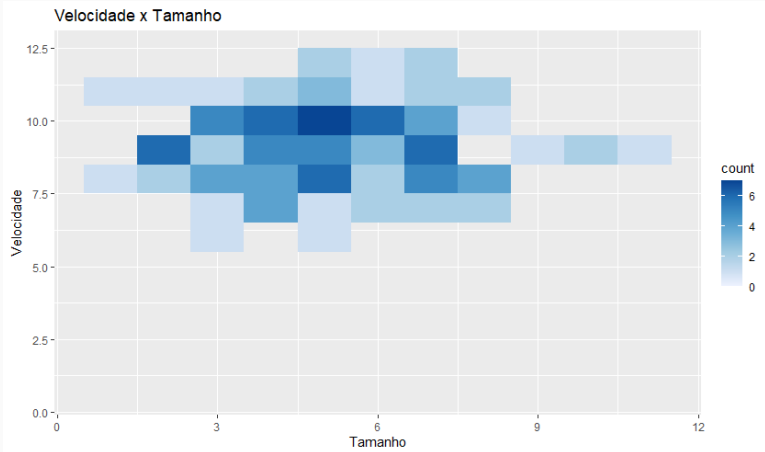
Dados simulados com tamanho e velocidades de **partículas** .

Objetivo: Verificar a correlação de tamanho e velocidade das partículas através de uma análise gráfica.

Exemplos no SparkR - Partículas

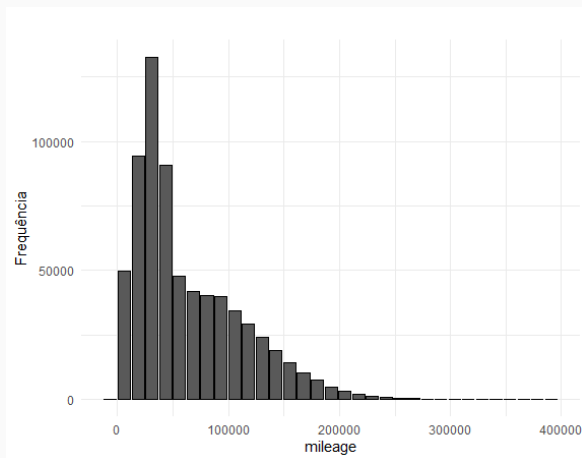


Exemplos no SparkR - Partículas

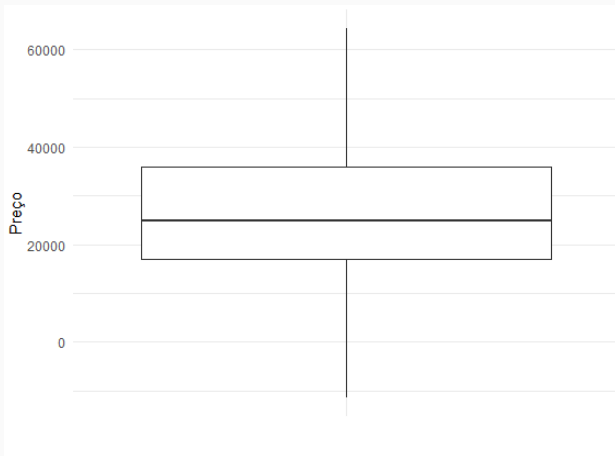


Aplicação

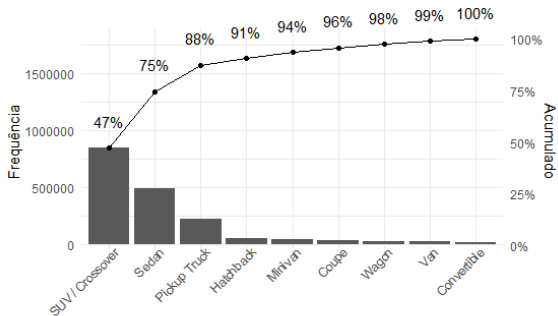
Previsão de preços de carros usados; 9,7GB, 3 milhões de linhas e 66 colunas.



Aplicação



Aplicação



- Árvore de decisão: R^2 0.60;

- Árvore de decisão: R^2 0.60;
- Tempo: Aproximadamente 2 horas e 30 minutos

Conclusões

Spark

O Spark se mostra uma ferramenta muito vasta e útil para análise de grandes volumes de dados, devido as suas características.

1. Escalável;
2. Eficiente;
3. Integrado com diversas linguagens e fontes de dados.

SparkR

O SparkR é uma API muito poderosa. Porém, muito inferior as *APIs* de outras linguagens.

1. Carência de funcionalidades;

SparkR

O SparkR é uma API muito poderosa. Porém, muito inferior as *APIs* de outras linguagens.

1. Carência de funcionalidades;
2. Documentação ruim;

SparkR

O SparkR é uma API muito poderosa. Porém, muito inferior as *APIs* de outras linguagens.

1. Carência de funcionalidades;
2. Documentação ruim;
3. Falta de padronização no nome das funções;

SparkR

O SparkR é uma API muito poderosa. Porém, muito inferior as *APIs* de outras linguagens.

1. Carência de funcionalidades;
2. Documentação ruim;
3. Falta de padronização no nome das funções;
4. Comunidade fraca.

Dúvidas?








Image de uso livre, Pixbay

Dúvidas?






Repositório: `https://github.com/Daniel-EST/spark-tcc`





Referências





Referências i

-  Apache, *Welcome! - the apache software foundation*, Apache.
-  Jeffrey Dean and Sanjay Ghemawat, *Mapreduce: Simplified data processing on large clusters*, USENIX Association OSDI '04: 6th Symposium on Operating Systems Design and Implementation.
-  The Apache Software Foundation, *Documentation for package 'sparkr' version 3.0.1*.
-  _____, *Mapreduce tutorial*.
-  _____, *Sparkr (r on spark)*.

Referências ii

-  GTA/UFJR, *Os 5 v's do big data*.
-  IBM, *What is mapreduce?*
-  Edgar Ruiz Javier Luraschi, Kevin Kuo, *Mastering spark with r: The complete guide to large-scale analysis and modeling*, no. ISBN-10 : 149204637X, O'Reilly Media, 2019.
-  Wagner Kolberg, *Simulação e estudo da plataforma hadoop mapreduce em ambientes heterogêneos*, Universidade Federal do Rio Grande do Sul, 2010.
-  Oracle, *Big data definido*.

-  Ripon Patgiri and Arif Ahmed, *Big data: The v's of the game changer paradigm*, 12 2016.
-  R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2014.
-  RStudio, *sparklyr: R interface for apache spark*.
-  Phil Simon, *Too big to ignore: The business case for big data*, Wiley, 2015.

-  Xiaojun Chen Patrick Xiaogang Peng Salman Salloum, Ruslan Dautov and Joshua Zhexue Huang, *Big data analytics on apache spark*, Springer International Publishing Switzerland 2016, 2016.
-  Hadley Wickham, Romain François, Lionel Henry, and Kirill Müller, *dplyr: A grammar of data manipulation*, 2021, R package version 1.0.7.
-  Hadley Wickham, *ggplot2: Elegant graphics for data analysis*, Springer-Verlag New York, 2016.
-  et al Xiangrui Meng, *Mllib: Machine learning in apache spark*, Tech. report, Databricks, 2016.



Matei Zaharia, N. M. Mosharaf Chowdhury, Michael Franklin, Scott Shenker, and Ion Stoica, *Spark: Cluster computing with working sets*, Tech. Report UCB/EECS-2010-53, EECS Department, University of California, Berkeley, May 2010.