Data Science
Big Data

2023

<lapti>

- No tempo dos pioneiros, se usavam bois para puxar cargas pesadas, e quando um boi não conseguia mover uma tora, não se tentava criar um boi maior. Nós não deveríamos estar tentando criar computadores maiores, mas sim, mais sistemas de computadores.
  - Almirante Grace Hopper

Evolução de Bancos de Dados

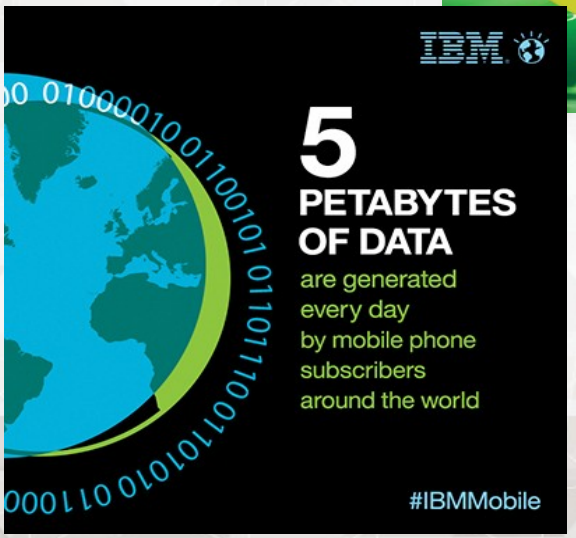
- Internet mudou tudo!
  - Internet of Things também
  - Dados demais!!!
  - Dados não estruturados
    - Mails, textos, tweets, logs, etc.
      - DBs não foram programados exatamente para isso
  - Complexidade de DW e BI aumentando
    - Quanto mais dados históricos, melhor
      - Porque manter só 24 horas de dados?
      - Porque não um ano inteiro? E dois?

Dados demais!

- Geração de dados por todas as figuras de uma sociedade
  - E seus dispositivos
- Mais de 15 Pb sobre operações comerciais, financeiras, clientes, fornecedores, etc
  - E por ano...

Dados demais!

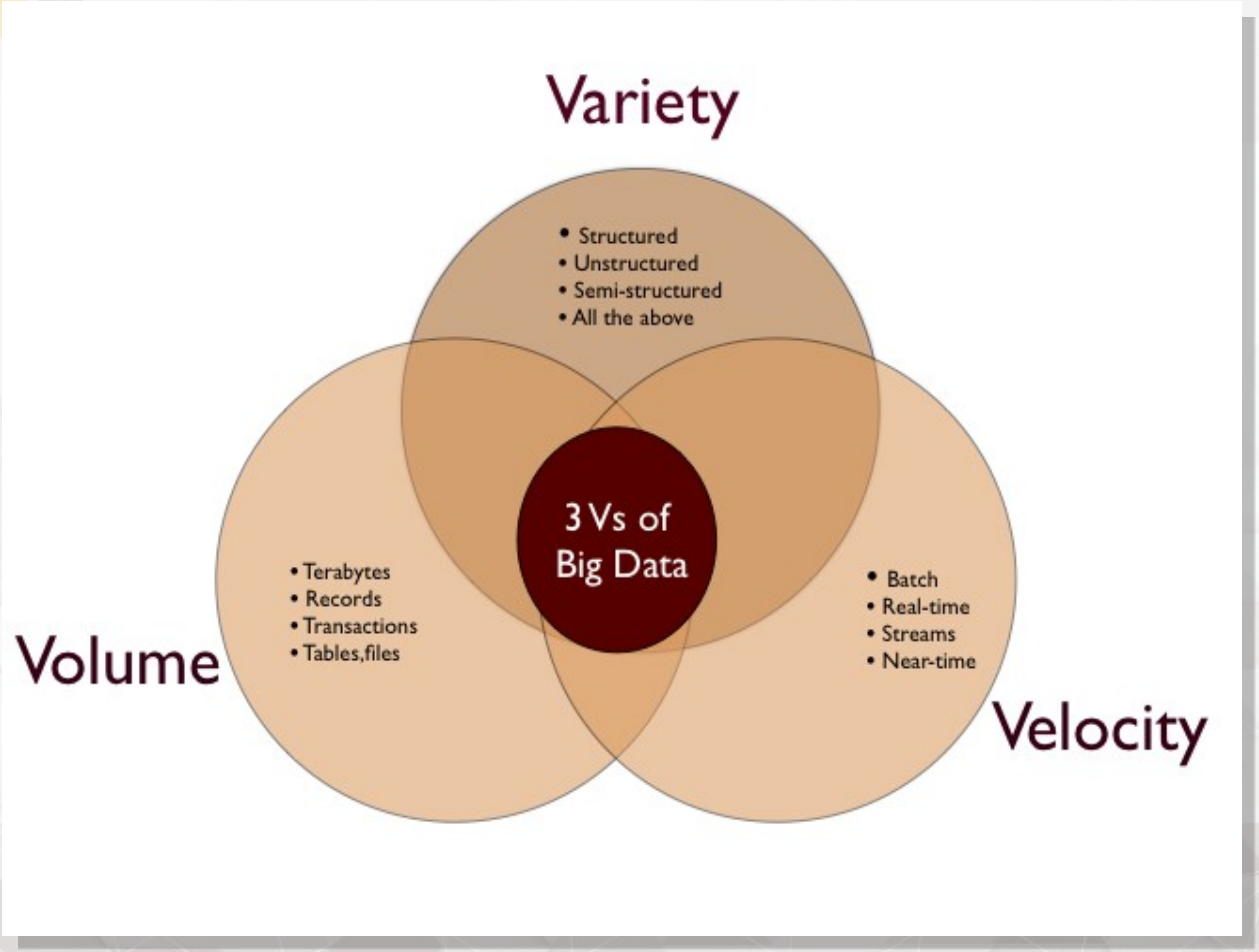
- Bolsa de Nova Iorque gera um Tb de transações por dia
- Facebook administra aproximadamente 10 bilhões de fotos, em mais de 1 Pb de armazenamento
- Internet Archive armazena em torno de 2 Pb de dados e cresce 20 Tb por mês
- O LHC (Large Hadron Collider) produz aproximadamente 15 Pb de dados por ano



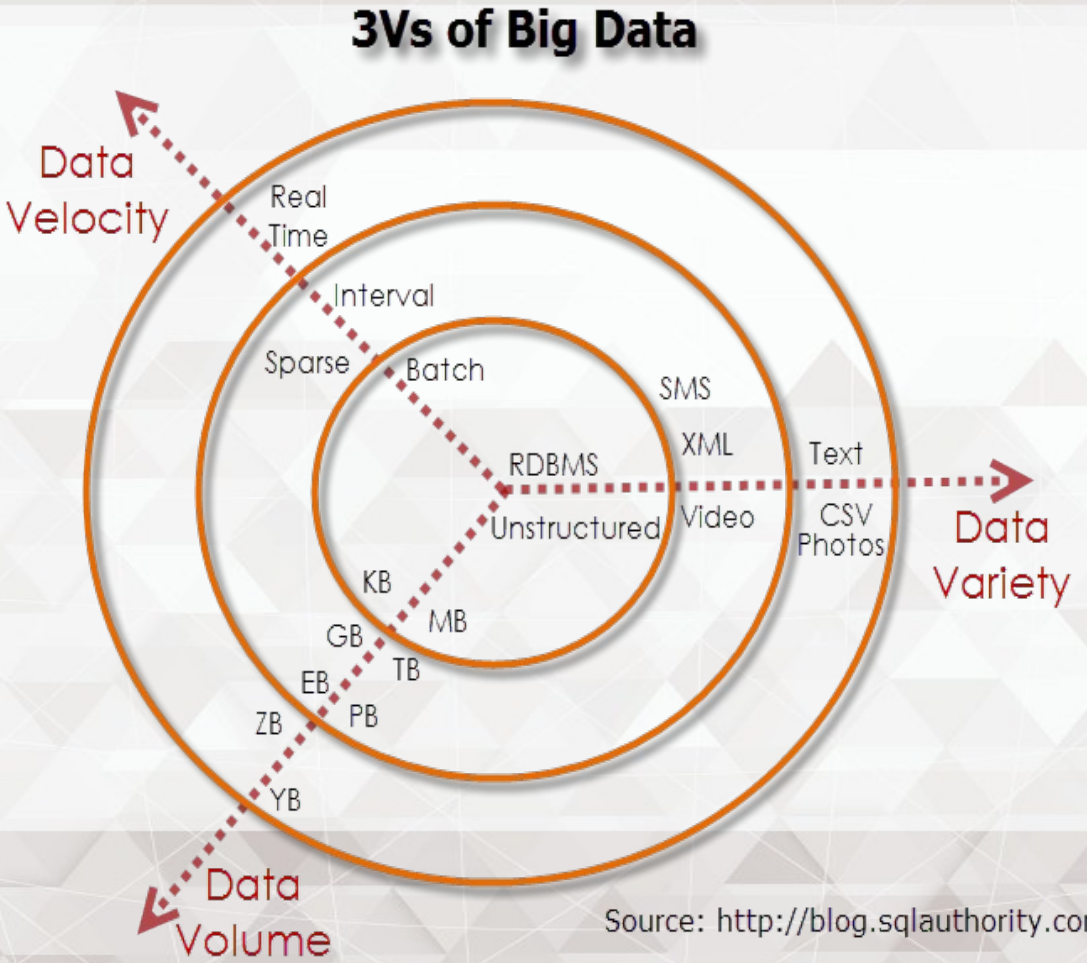
Dados demais!

- 90% dos dados do mundo foram criados nos últimos dois anos
- Empresas perceberam tesouro escondido em documentos e dados não estruturados
  - E crescem em volume exponencial
- Mas...
  - Como armazenar tanta informação? (Pb, Hb)
  - Como acessar tanta informação de forma rápida?
  - Como tratar informações em formatos tão heterogêneos
    - Era tão mais fácil se tudo estivesse em tabelas...
  - Como tornar isso escalável, tolerante a falhas e flexível?

Os “V”s do Big Data



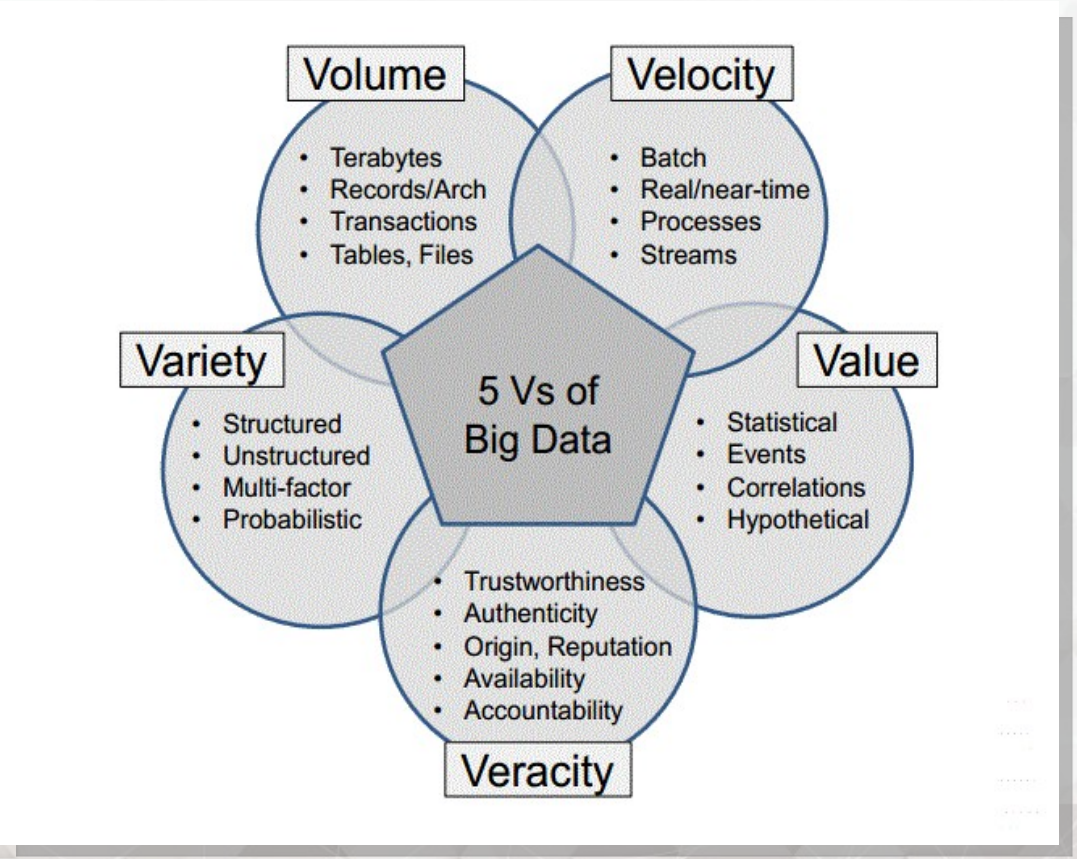
Os “V”s do Big Data



Source: http://blog.sqlauthority.com

Os “V”s do Big Data

- Mais dois Vs foram adicionados em seguida
  - Veracidade
  - Valor
- Ambiente corporativo
- E ainda tem Variabilidade...



Big Data?

- O termo Big Data é bem amplo e não existe um consenso comum em sua definição
- Entretanto, pode ser resumidamente definido como o processamento (eficiente e escalável) analítico de grande volumes de dados complexos produzidos por (várias) aplicações
- É ideal:
  - Ao analisar dados semiestruturados e não estruturados de uma variedade de fontes
  - Quando todos os dados ou quase todos devem ser analisados
  - Para analises interativas e exploratórias
  - Big Data releva as formalidades e restrições do Data Warehouse
  - Preserva a fidelidade dos dados

Big Data?

- Armazenamento e processamento de dados em larga escala
  - Pb, Hb
- Dados estruturados e não estruturados
  - Fogem aos conceitos estabelecidos de tratamento de registros e campos
    - Se estruturados, tabelas com bilhões de registros, milhões de colunas
- Em geral, tarefas que consumiriam um tempo proibitivo em tecnologias convencionais
  - Clusters de bancos de dados tem crescimento limitado
    - Essa tecnologia não foi prevista para isso

Big Data?

- Ok, se não podemos usar clusters de bancos de dados, usamos o que?
  - Porque não dividir o armazenamento e processamento em um número enorme de máquinas?
  - Se os dados são enormes, dividimos em porções tratáveis, distribuímos entre todas as máquinas
  - Criamos um programa que trate de cada porção, dividimos também entre as máquinas
  - Criamos outro programa que controle a execução e combine os resultados parciais
  - Simples, não?

Big Data?

- Processamento paralelo e clusterização são conceitos até antigos em computação
  - Porque vocês acham aprendemos a programar em threads? Só para fazer jogos com mais de um personagem??
    - Por mais que isso também seja importante, claro...
- Mas é muito complexo exigir de um programador todas as competências listadas anteriormente
  - Além do tempo de programação muito grande

Big Data e Google

- Google sempre teve problemas com exagero de dados
  - Precisou desenvolver tecnologia, já que não existia
- Distribuição de informações em um cluster
  - Google File System
    - The Google File System (2003)
- Processamento distribuído em um cluster
  - Map Reduce (sobre GFS)
    - MapReduce: Simplified Data Processing on Large Clusters (2004)
- Acesso NoSQL sobre dados em grande volume
  - BigTable (sobre GFS)
    - BigTable: A Distributed Storage System for Structured Data (2006)

Hadoop

- Implementação open-source do GFS e MapReduce
  - Baseado nos artigos originais
  - HDFS e Hadoop MapReduce
    - http://hadoop.apache.org

- Desenvolvido inicialmente pelo Yahoo
- Desempenho otimizado para OLAP
- Crescimento explosivo
  - Usado mesmo em versões muito incipientes
- Deu origem a diversos projetos complementares

Hadoop - Histórico

- Criado por Doug Cutting
  - Também criador do Apache Lucene
- Objetivo: construir um indexador de páginas para um web search engine
  - Hardware caro, programação complexa
  - Projeto Nutch
    - Iniciado em 2002
    - Logo se percebeu que a tarefa era muito complexa e que não iria escalar para os bilhões de páginas da web
    - A partir do artigo do GFS da Google, implementaram uma versão aberta do GFS, chamada NDFS

Hadoop – Histórico

- Projeto Nutch
  - Em 2004, Google publica artigo do MapReduce
  - Em 2005, Nutch já tem implementação funcional de MapReduce
    - Todos os principais algoritmos do Nutch foram portados para NDFS e MapReduce
  - Em 2006, foi criado o projeto Hadoop, como projeto separado do Lucene, abarcando o Nutch
    - Ao mesmo tempo, Doug Cutting foi contratado pelo Yahoo!
      - Que forneceu equipe e recursos para manter o Hadoop
  - Em 2008, Yahoo! já mantinha um cluster de 10.000 cores usando Hadoop

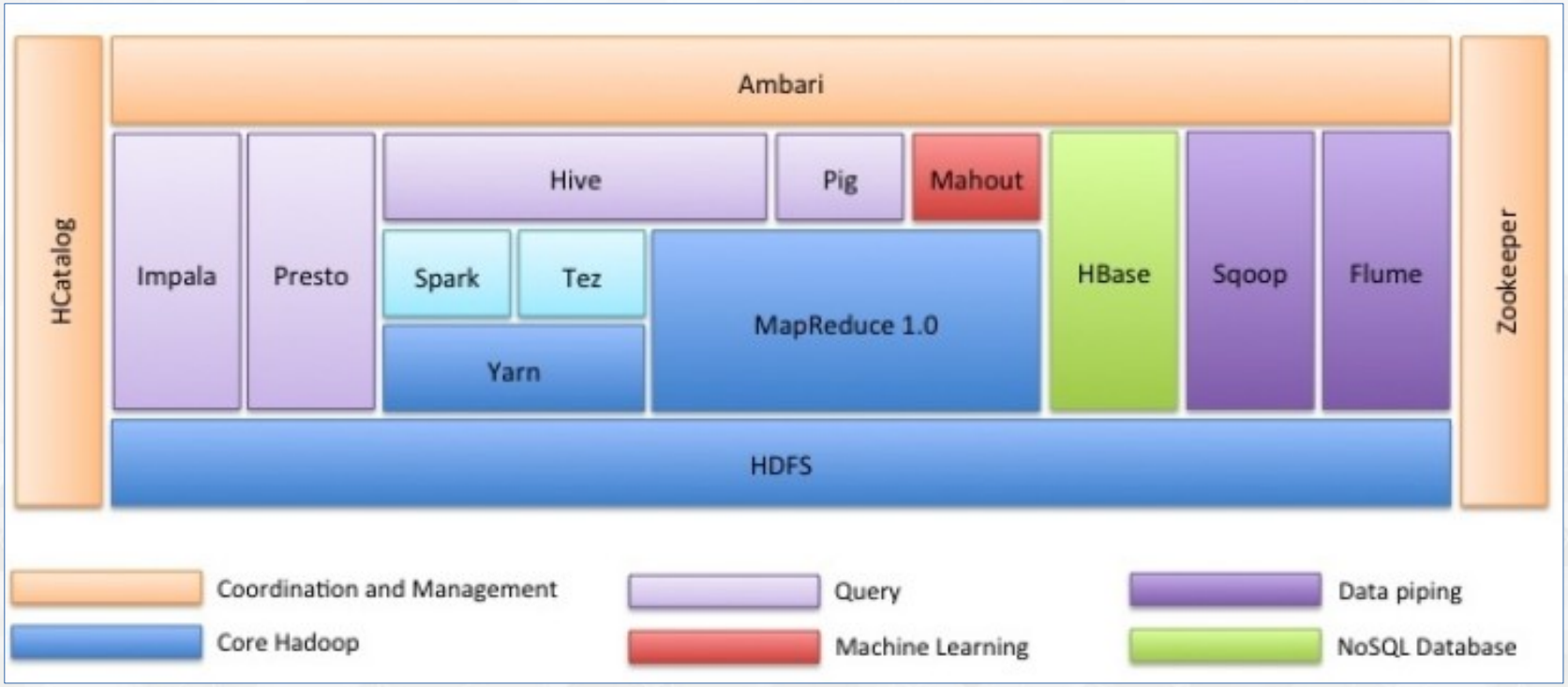
Hadoop - Histórico

- 2008, Hadoop passa a ser um projeto top-level no Apache
  - Usado nessa época por Yahoo, Last.fm, Facebook, New York Times e outros
- Em abril de 2008, Hadoop quebrou o recorde mundial para classificação de dados
  - Classificar 1 Tb de dados, usando um cluster de 910 nós
    - 209 segundos
    - Em novembro, o cluster do Google fez em 68 segundos
    - Em 2009, cluster do Yahoo fez em 62 segundos
- Adoção corporativa massiva desde então
  - Amazon, EMC, IBM, Microsoft, Oracle, Cloudera, MapR, etc

Hadoop

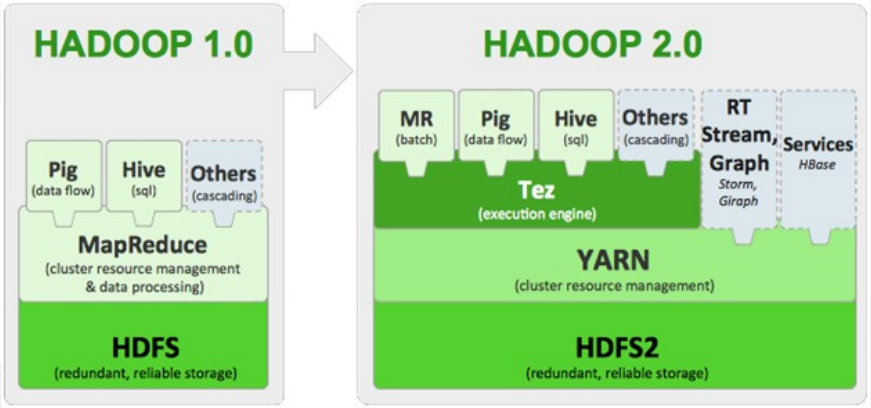
- Plataforma de armazenamento e programação distribuída
  - HDFS + MapReduce
- Responsabilidades separadas
  - Armazenamento (HDFS: NameNode, DataNode)
  - Processamento (TaskTracker / NodeManager)
  - Coordenação (JobTracker / ResourceManager)
- Tolerância a falhas
  - Replicação do sistema de arquivos (default 3)
  - Tentativas de tasks

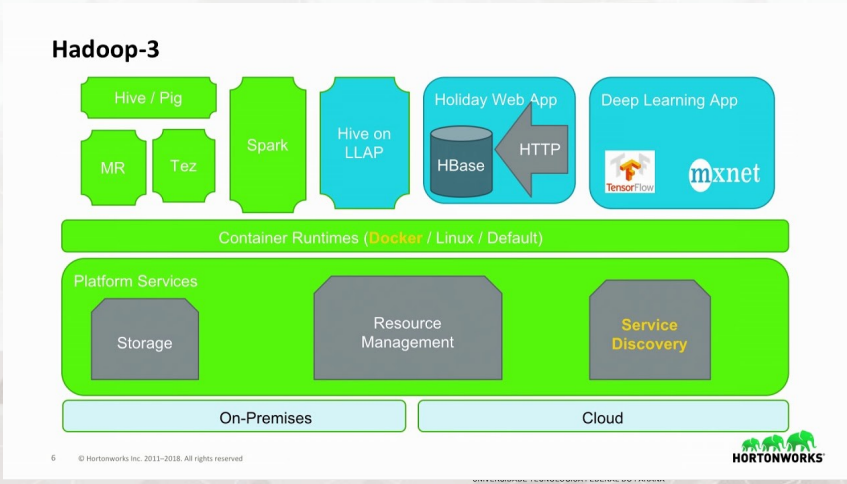
Hadoop Stack



Releases

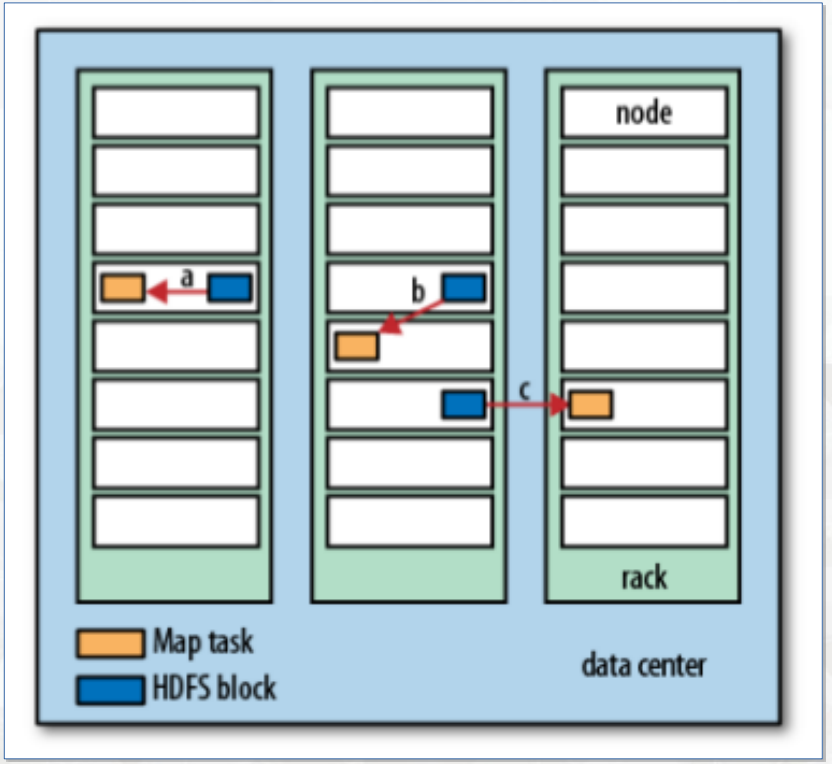
- 1.x
  - HDFS + MapReduce
- 2.x
  - Aumento em tamanho possível de clusters
    - Havia alguma instabilidade em clusters com mais de 2000 nós…
  - Novo gerenciador de recursos
    - YARN (Yet Another Resource Negotiator)
- 3.x
  - Suporte a GPUs
  - Múltiplos gerenciadores (Namenodes e namespaces)
  - Alteração em replicação HDFS (redução de overhead para 50%) e disk balancing
  - Gerenciamento de containers no cluster





HDFS

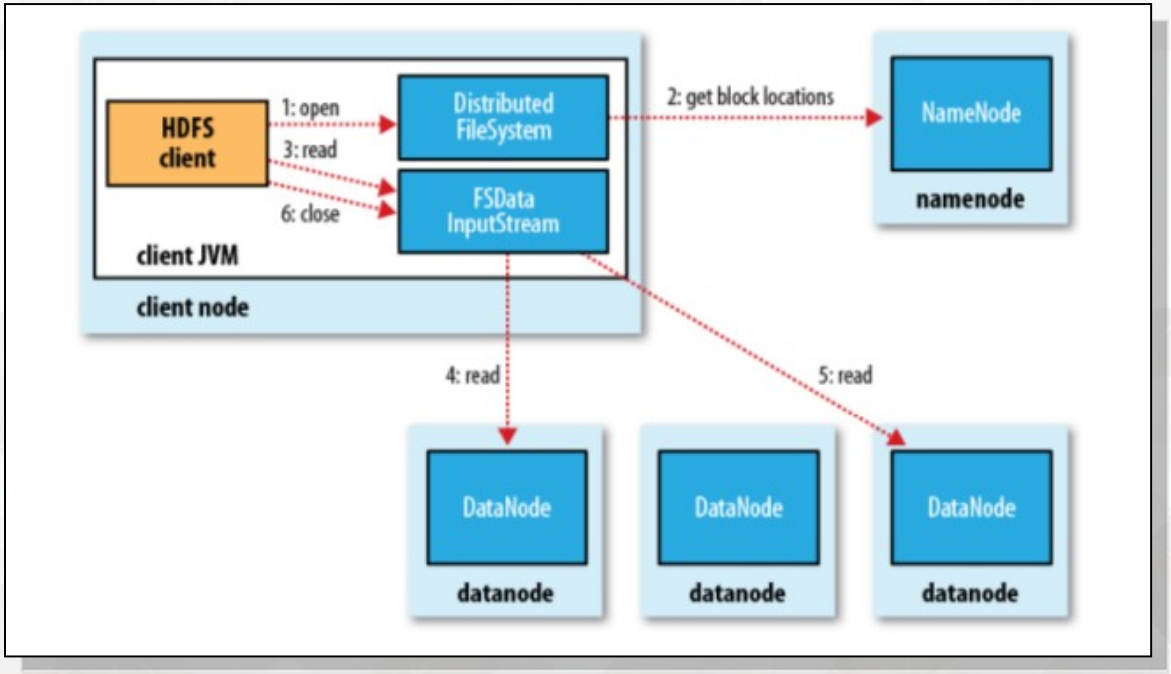
- Tolerância a falhas
- Replicação
- Reconhecimento de racks
  - Reconhecimento da estrutura da rede
  - Combinado com MapReduce e YARN

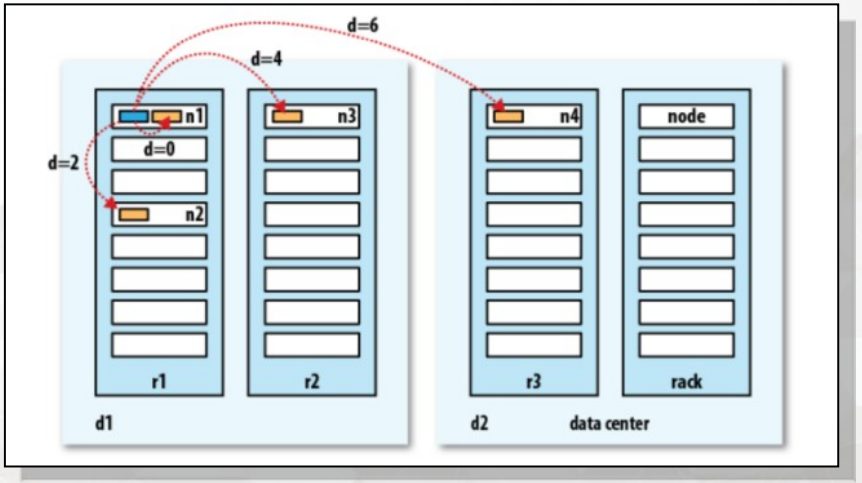


Motivações

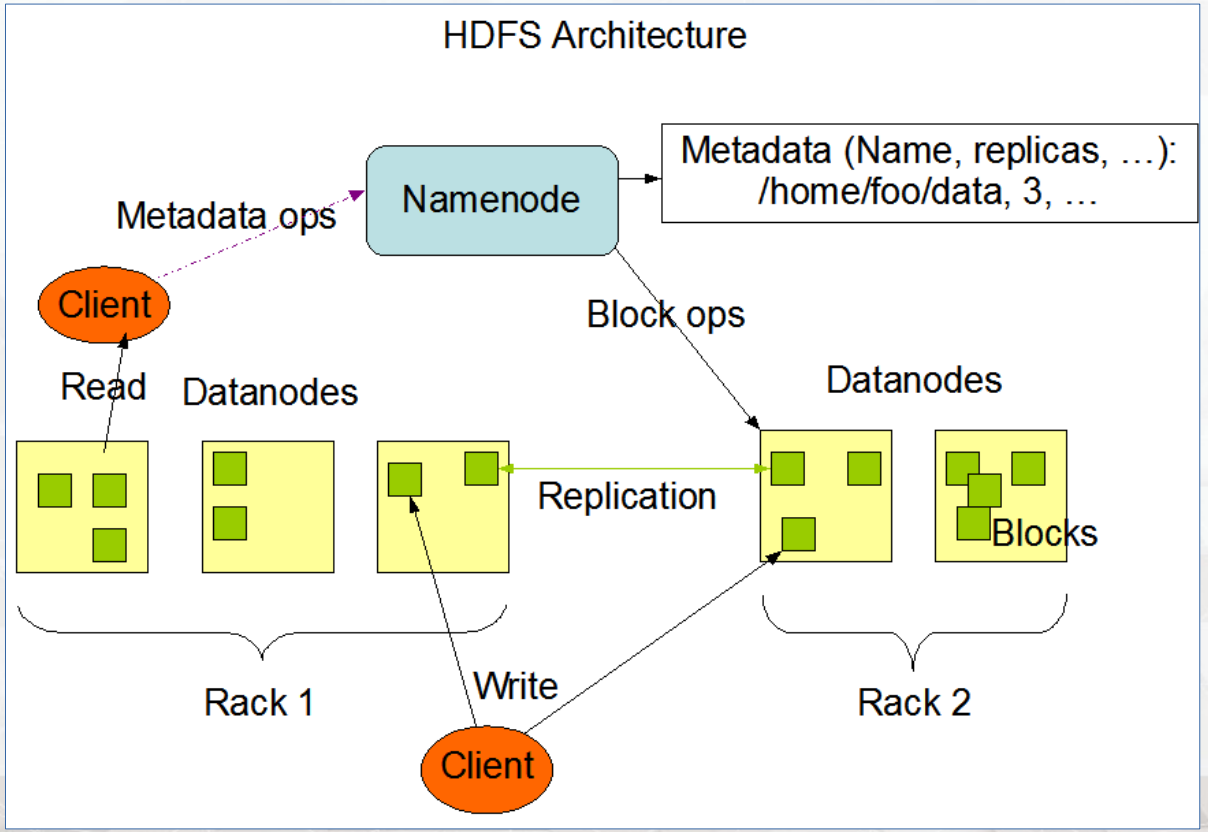
- Capacidade de discos cresce em ritmo menor do que velocidade de leitura
  - Discos encontrados em computadores desktops comuns
    - 1990: 1,3 Gb, 4.4 Mb/s
    - 2010: 1 Tb, 100 Mb/s
  - Tempo para ler o disco todo aumentou desproporcionalmente
    - Ler de vários discos ao mesmo tempo é uma solução óbvia
      - Mas precisa de uma infra-estrutura de distribuição

HDFS


HDFS client
1: open
3: read
6: close
Distributed FileSystem
FSData InputStream
client JVM
client node
2: get block locations
NameNode
namenode
4: read
5: read
DataNode
datanode
DataNode
datanode
DataNode
datanode

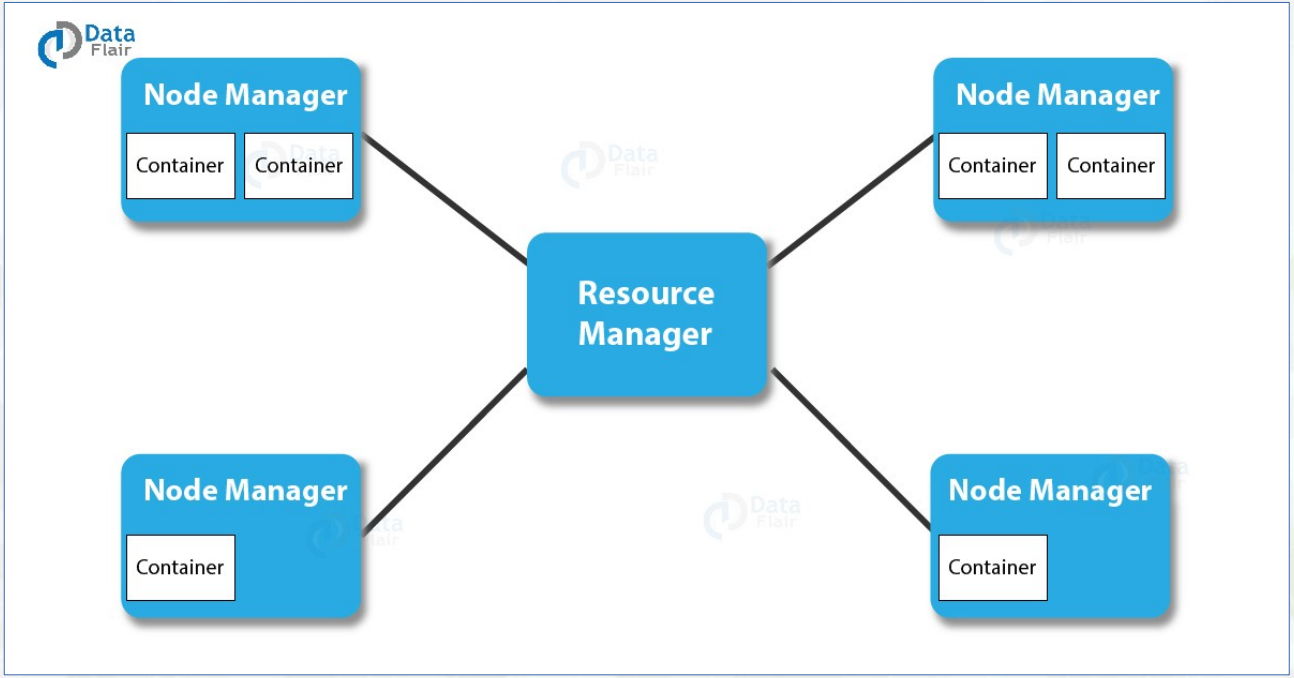

d=6
d=4
d=2
n1
d=0
n2
n3
n4
node
r1
r2
r3
rack
d1
d2
data center

HDFS

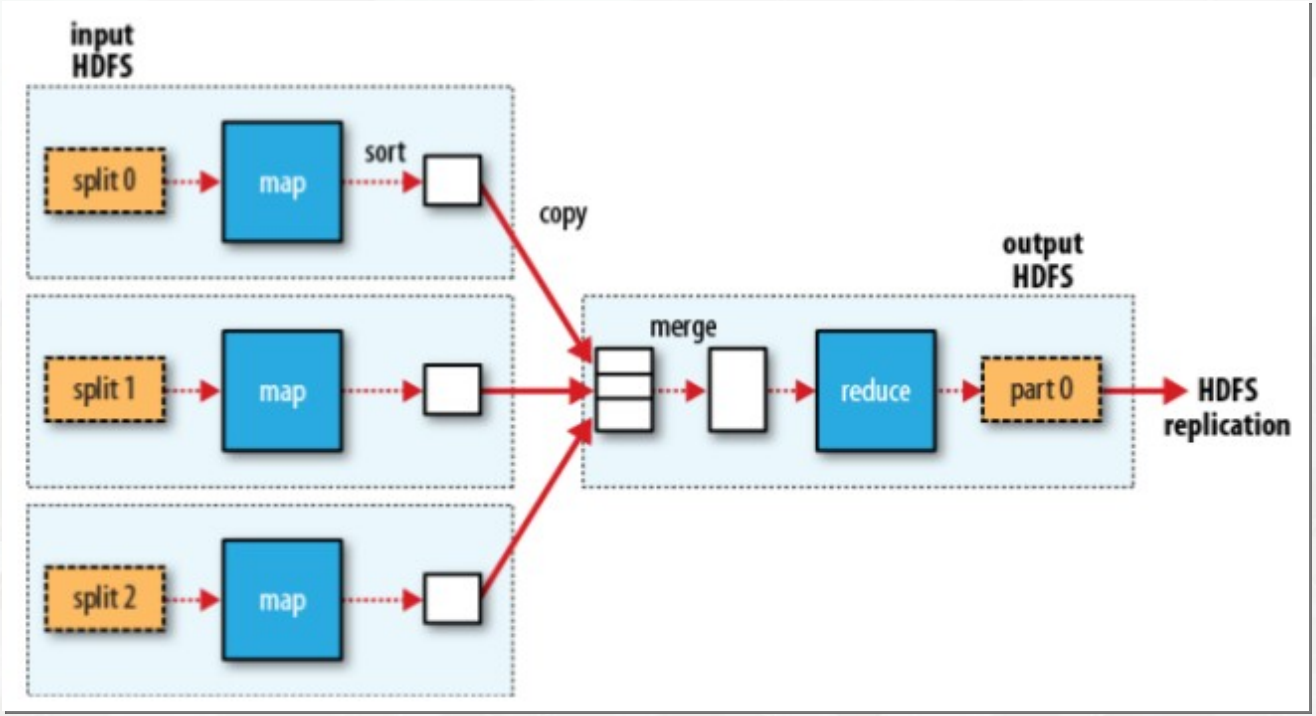

HDFS Architecture
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Metadata ops
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

Gerenciador de Tarefas Distribuídas

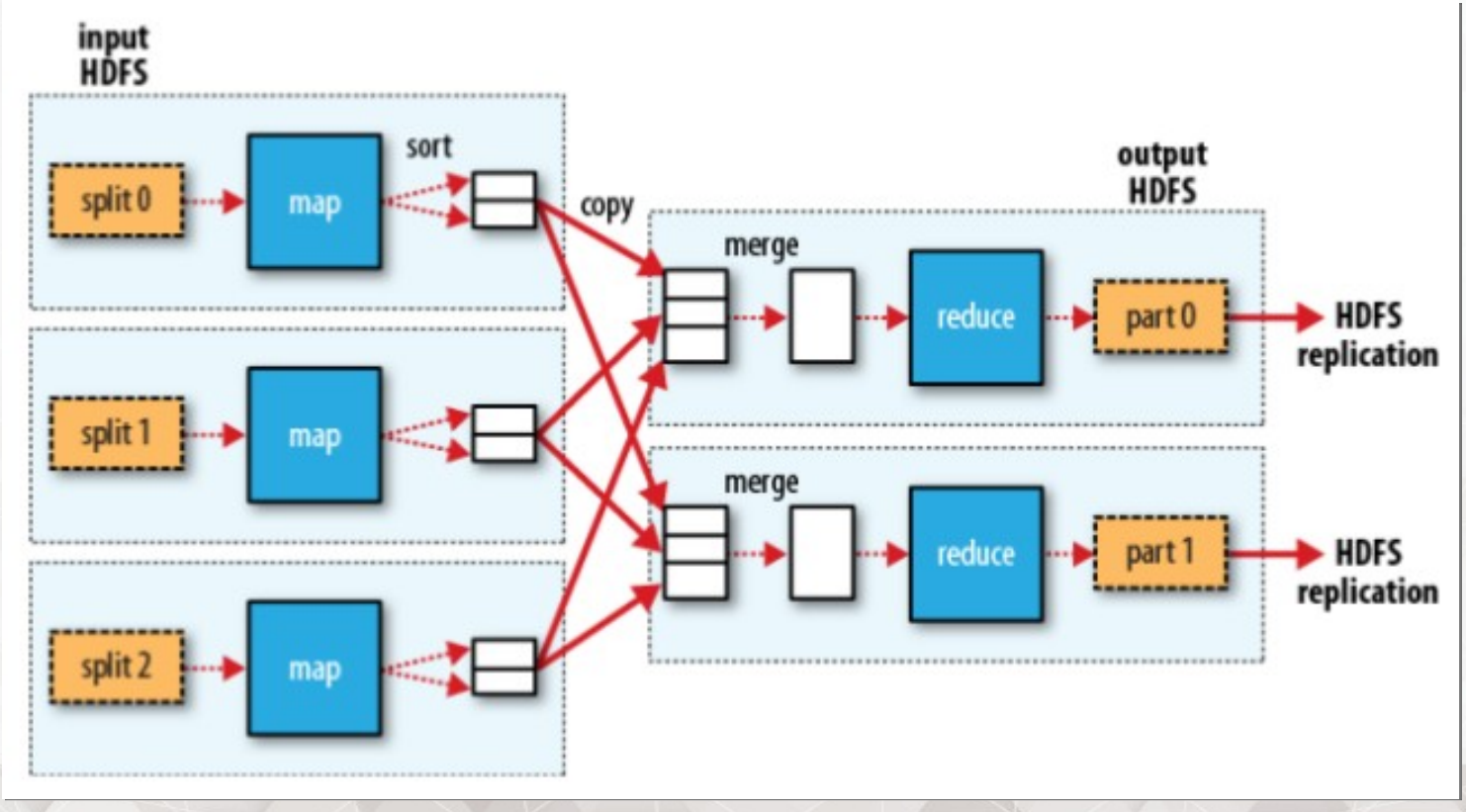
- API MapReduce
- YARN



API MapReduce



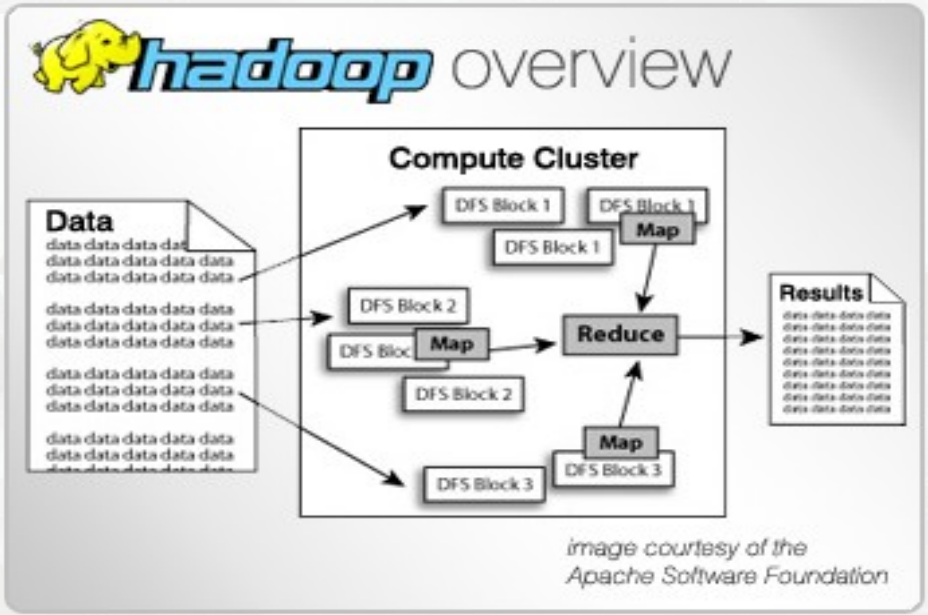
API MapReduce

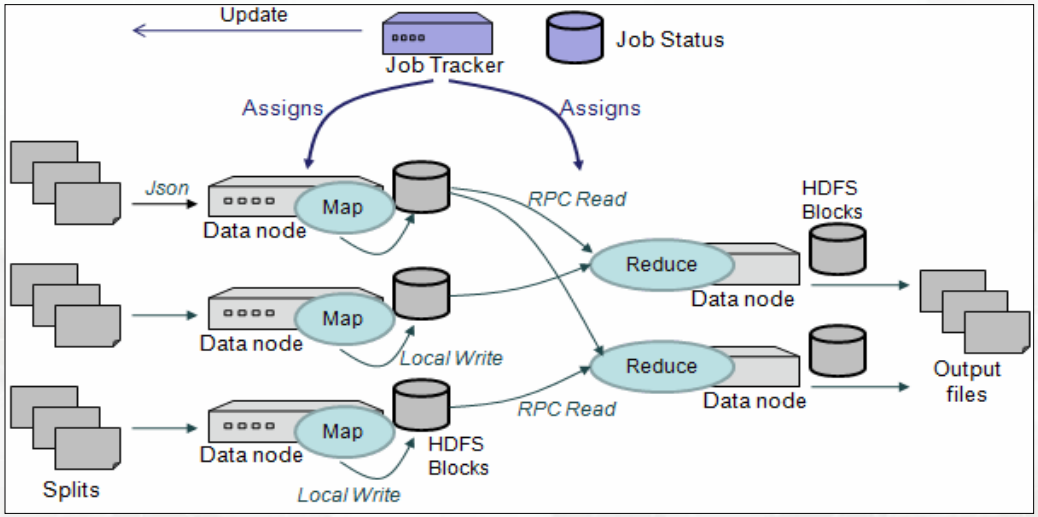


API MapReduce

- Componentes
  - Mappers
  - Reducers
  - Combiners / Partitioners
  - Classes wrapers para tipos
- Versões da API
  - API até versão 0.20 e nova API
    - Criada para permitir evolução com menor impacto
      - Context Objects
      - Incompatível a nível de tipo com a API antiga
    - Nova API é compatível com várias versões do Hadoop

Map Reduce


hadoop overview
Compute Cluster
Data
DFS Block 1
DFS Block 1
Map
DFS Block 1
DFS Block 2
DFS Bloc
Map
Reduce
DFS Block 2
Results
Map
DFS Block 3
DFS Block 3
image courtesy of the
Apache Software Foundation


Update
Job Tracker
Job Status
Assigns
Assigns
Json
Map
Data node
RPC Read
HDFS
Blocks
Reduce
Data node
Map
Data node
Local Write
Reduce
Data node
Output
files
Map
Data node
HDFS
Blocks
RPC Read
Splits
Local Write

Map Reduce

- Abstração é inspirada em 'Map' e 'Reduce‘
  - Primitivas presentes em Lisp e outras linguagens funcionais.
- Google:
  - “Percebemos que a maioria dos nossos cálculos envolvia a aplicação de uma operação de Map para cada "registro" lógico em nossa entrada, a fim de calcular um conjunto de pares intermediários de chave/valor e, em seguida , a aplicação de uma operação de Reduce a todos os valores que partilhavam a mesma chave, a fim de combinar os dados derivados apropriadamente.”

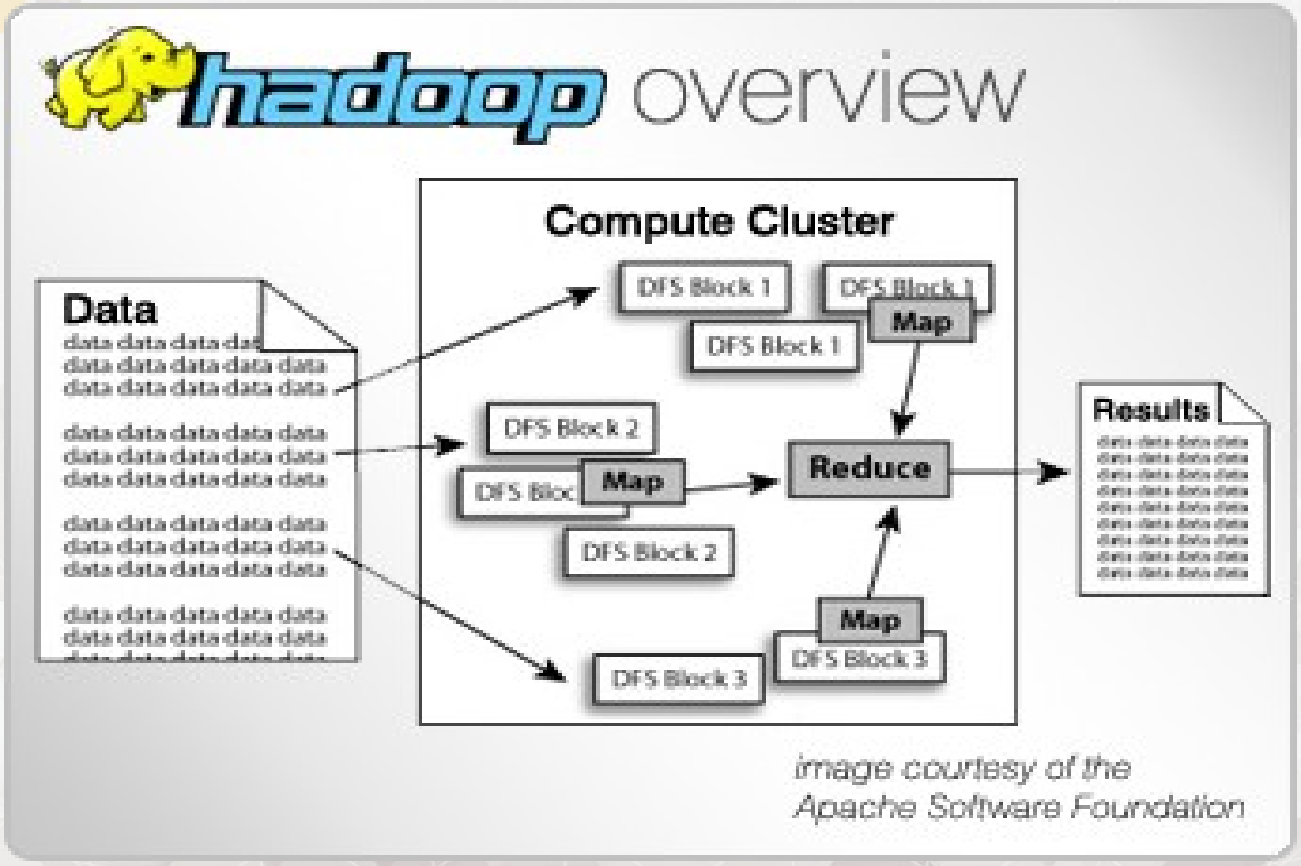
Map Reduce

- Divide o processamento em duas etapas:
  - (1) Map, que mapeia e distribui os dados em diversos nós de processamento e armazenamento
  - (2) Reduce, que agrega e processa os resultados parciais para gerar um resultado final (ou intermediário para outro processo MapReduce)
- Provavelmente uma das maiores vantagens deste paradigma é a sua simplicidade
  - Manipulação dos dados é feita pelo uso de duas funções básicas
    - Map (função de mapeamento) e Reduce (função de redução).

Map Reduce

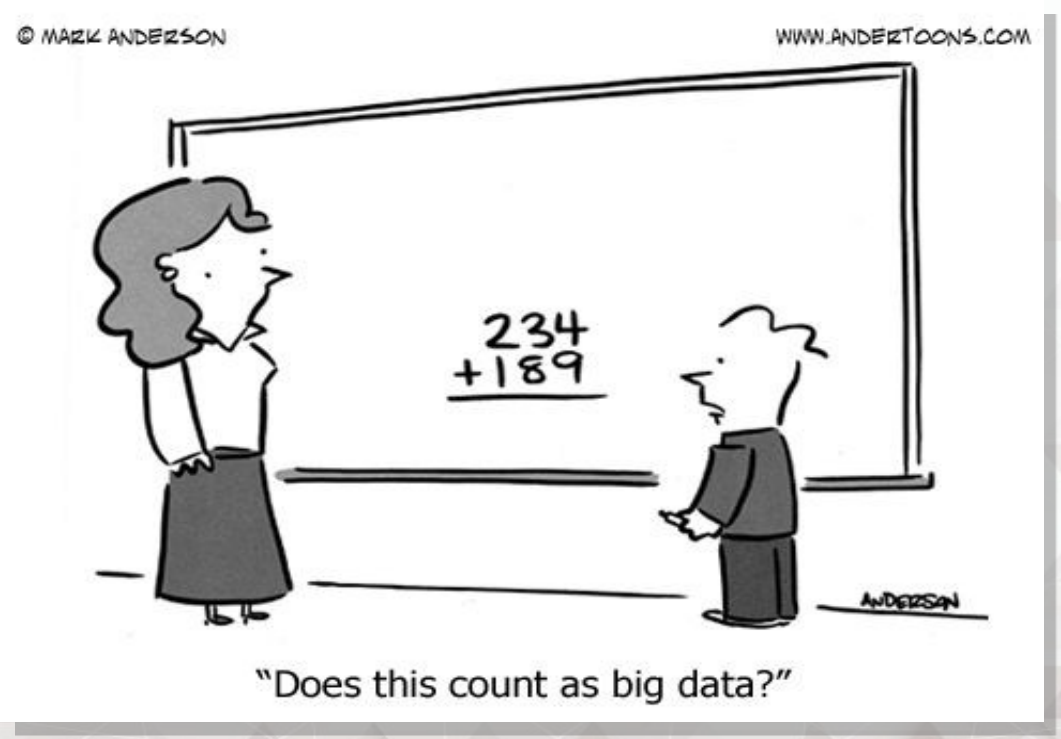
- Maiores contribuições dessa nova abordagem
  - Interface poderosa e simples
  - Permite a paralelização automática
  - Distribuição da computação em grande escala
  - Alta performance em grandes aglomerados de máquinas
- Processamento toma um conjunto de pares de entradas de chave/valor
  - Produz um conjunto de pares de saídas chave/valor
  - Implementador do MapReduce expressa o cálculo em duas funções
    - Map e Reduce.

Map Reduce

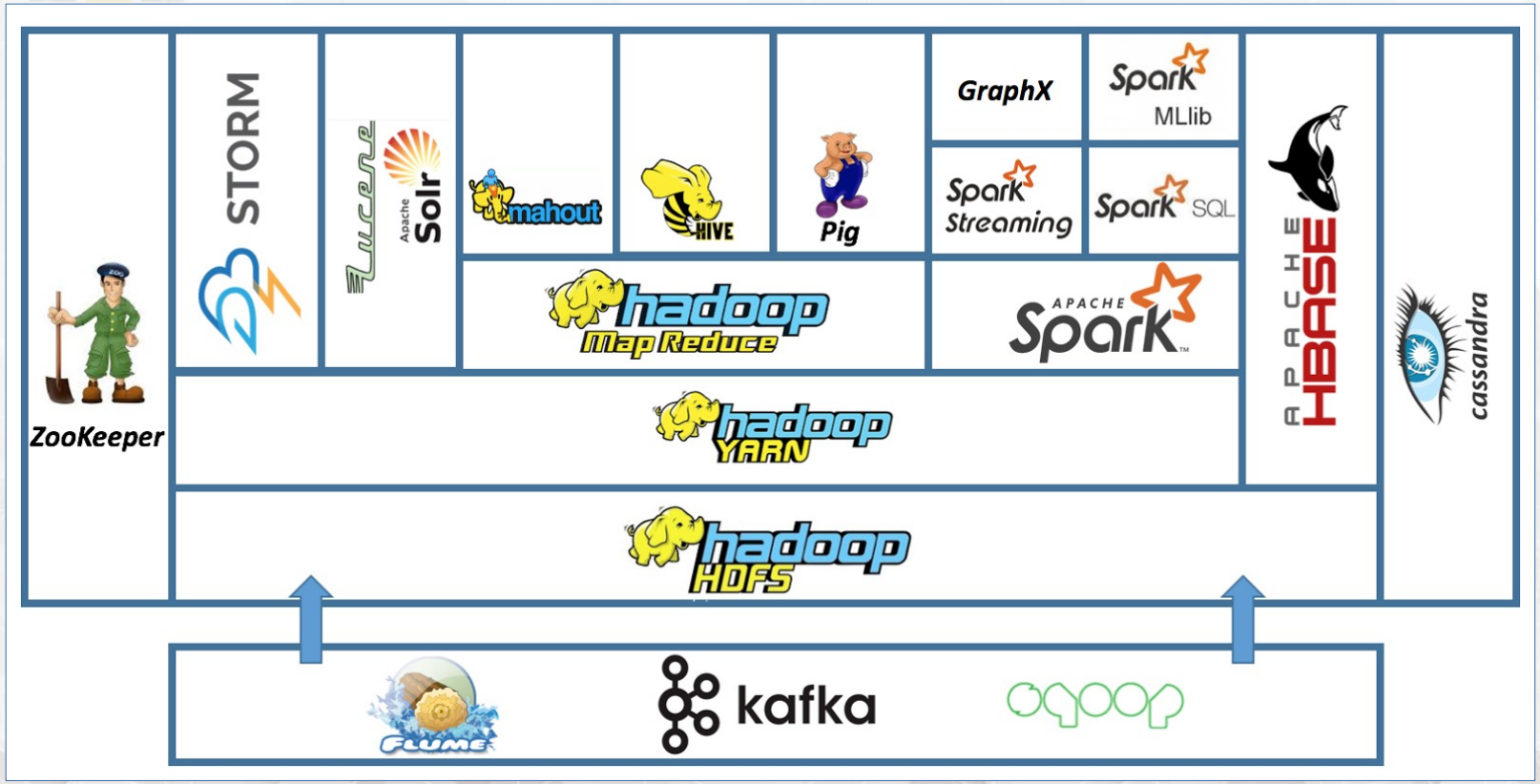

hadoop overview
Compute Cluster
Data
data data data data
data data data data data
data data data data data
data data data data data
data data data data data
data data data data data
data data data data data
data data data data data
data data data data data
DFS Block 1
DFS Block 1
DFS Block 1
Map
DFS Block 2
DFS Bloc
Map
Reduce
DFS Block 2
Map
DFS Block 3
DFS Block 3
Results
image courtesy of the
Apache Software Foundation

Map Reduce

- Programação pesada...



Ecossistema Hadoop



Ecossistema Hadoop

- PIG
  - Programar as classes Mapper e Reducer exige conhecimentos razoáveis em Java
  - PIG cria sobre Hadoop uma camada que aceita comandos em linguagem de fluxo de dados
    - PIG Latin
    - Apreciado por profissionais de DW
  - Cria jobs MapReduce
  - Última versão em 2017
    - Incluindo Pig on Spark

Apache Pig

Ecossistema Hadoop

- Hive
  - Mesma questão de dificuldade (e tempo) em criar jobs MapReduce
  - Desenvolvido pelo Facebook
  - Baseado em dialeto SQL
    - Acesso familiar a desenvolvedores
    - Inclui driver JDBC
  - Desempenho escalável (como Hadoop)
  - Novas versões com processamento dinâmico de consultas
  - Versão 2.3.x e 3.1.x com releases em 2020

Ecossistema Hadoop

- HBase (baseado em Google BigTable)
  - Base NoSQL baseada em HDFS
    - Baseado em colunas (famílias de colunas) e regiões
  - Latência menor em setup de cada consulta
  - Apropriado para OLTP, além de OLAP

Panorama de Aplicações

- McKinsey Global Institute publicou recentemente um relatório sobre as oportunidades de negócios e do governo ao usar BigData.
  - “Big Data: The Next Frontier for Innovation, Competition and Productivity”
    - "Nós estimamos que um revendedor com o apoio de BigData tem o potencial de aumentar a sua margem operacional em mais de 60%”
    - “Big Data cria valor para as empresas descobrindo padrões e relacionamentos entre dados que antes estavam perdidos não apenas em data warehouses internos, mas na própria Web, em tuítes, comentários no Facebook e mesmo videos no YouTube.”
- Segundo a consultoria IDC, o mercado global de Big Data crescerá quase 40% ao ano entre 2010 e 2015
  - Saltando de US$ 3,2 bilhões para US$ 16,9 bilhões

Porque usar “Big Data” hoje em dia?

Porque usar “Big Data” hoje em dia?

- Grandes volumes de dados

Porque usar “Big Data” hoje em dia?

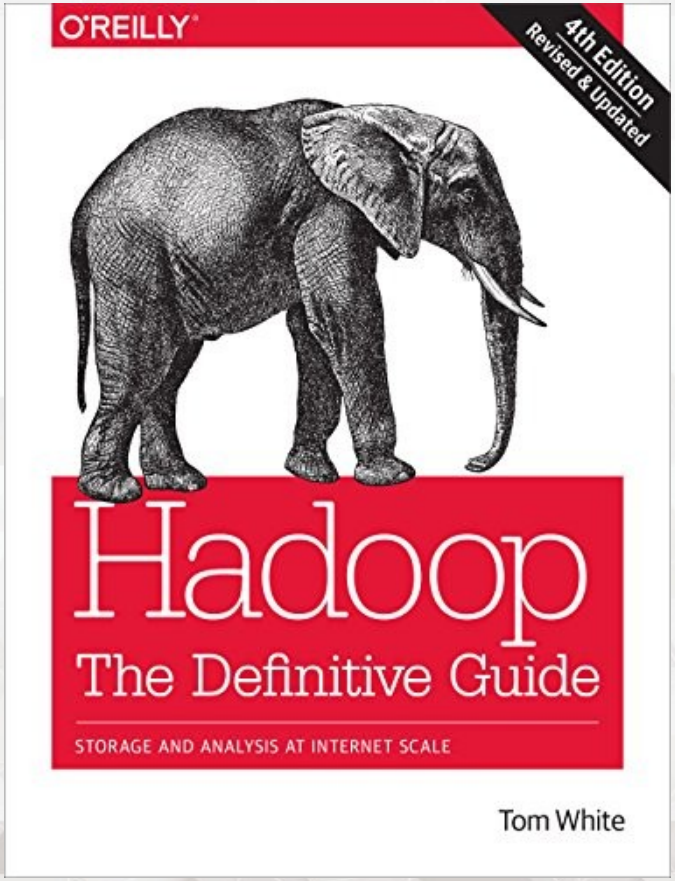
- Grandes volumes de dados
  - Mas GRANDE,
    absurdamente grande,
    OBSCENAMENTE grande...

Porque usar “Big Data” hoje em dia?

- Grandes volumes de dados
  - Mas GRANDE,
    absurdamente grande,
    OBSCENAMENTE grande…
- Dados não-estruturados

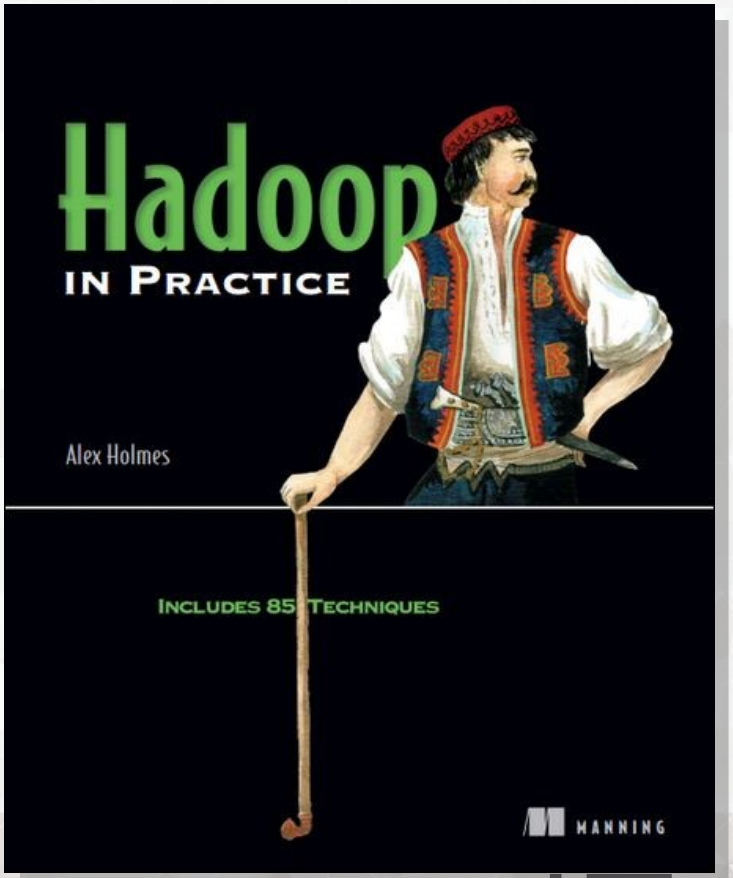
Bibliografia

- Hadoop: The Definitive Guide, 4rd ed
  - Tom White



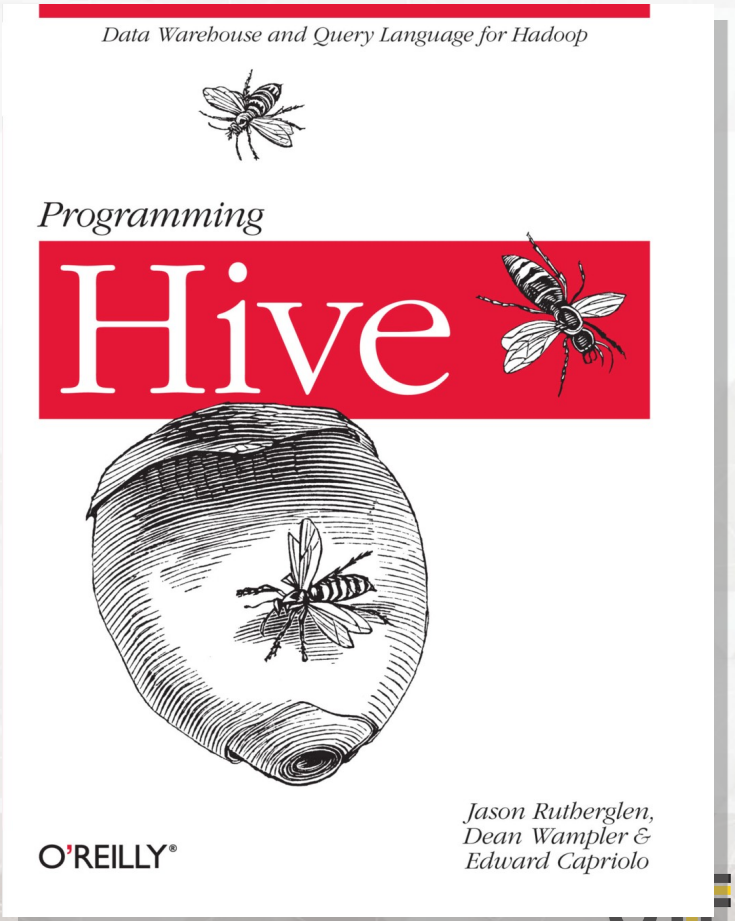
Bibliografia

- Hadoop in Practice
  - Alex Holmes



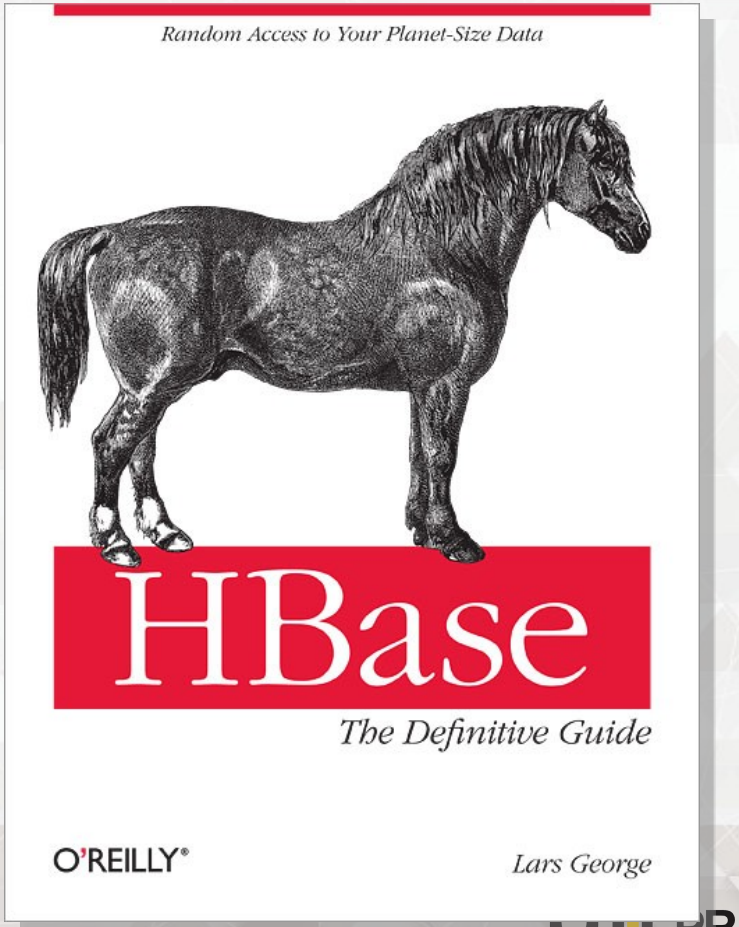
Bibliografia

- Programming Hive
  - Edward Capriolo



Bibliografia

- HBase: The Definitive Guide
  - Lars George



Obrigado

leandro@utfpr.edu.br
http://lapti.ct.utfpr.edu.br

<lapti>