Engenharia de Dados

Introdução ao ecossistema Apache Hadoop

DCA0132 – Engenharia de Dados

Prof. Carlos M. D. Viegas

Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte

Introdução ao ecossistema Apache Hadoop

- Problemática:
  - O **volume de dados** e a **velocidade** com que crescem tornaram-se tão grandes que o seu **processamento** se torna **inviável** em um **único computador**, por mais poderoso que seja!
    - Não é escalável
    - Consumo de energia torna-se um fator limitante
    - Limite físico de espaço para tantos discos rígidos
  - Possível solução: Utilizar a computação paralela por meio de múltiplos computadores
    - Mas é complexo!
    - Divisão e escalonamento de tarefas
    - Balanceamento de carga
    - Sincronismo entre tarefas (muito complexo!)
    - Largura de banda é limitada
      - Transferência de volumes de dados entre computadores
        - Funciona para pequenos volumes de dados, mas não para grandes como no big data
  - Então... o Apache Hadoop surge como uma solução para remover a complexidade da computação paralela!

Introdução ao ecossistema Apache Hadoop

- Hadoop é para problemas grandes cujos sistemas tradicionais não são capazes de processar!
- Mas afinal, o que é Hadoop?
  - Framework de código aberto criado em 2005 por Doug Cutting e Mike Carafella
    - Desenvolvido em linguagem Java
  - Projetado para armazenar e processar grandes volumes de dados em larga escala
    - Computação distribuída
- Aplicações comuns do Hadoop:
  - Processamento de texto em larga escala
  - Aprendizado de máquina e mineração de dados
  - Análise de dados em larga escala de redes sociais

Introdução ao ecossistema Apache Hadoop

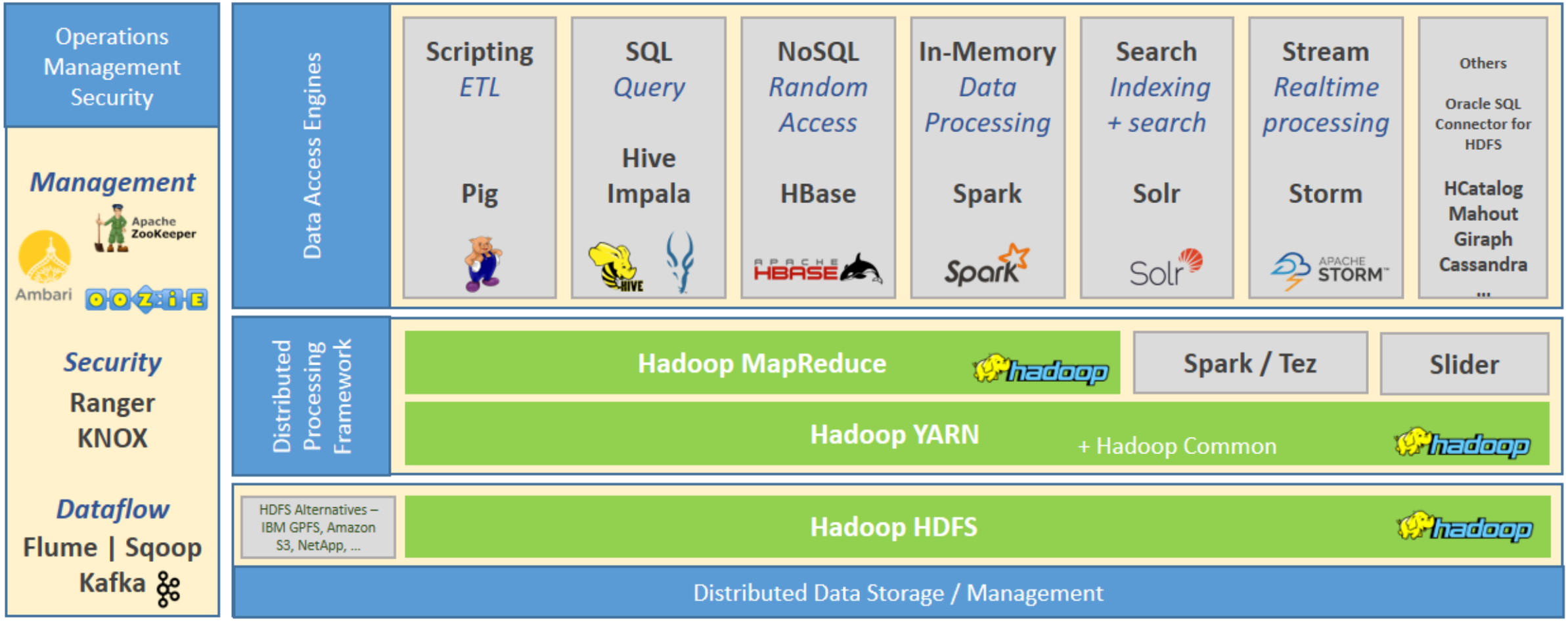
- Fatores para o sucesso do Hadoop:
  - Gratuito!
  - Permite utilizar computadores de baixo custo
  - Não exige alterações na infraestrutura da rede (rede comum)
  - Tolerância a falhas
  - Facilidade de uso
  - Escalável

Introdução ao ecossistema Apache Hadoop

- Quem usa Hadoop?

Introdução ao ecossistema Apache Hadoop

- Ecossistema Apache Hadoop



Introdução ao ecossistema Apache Hadoop

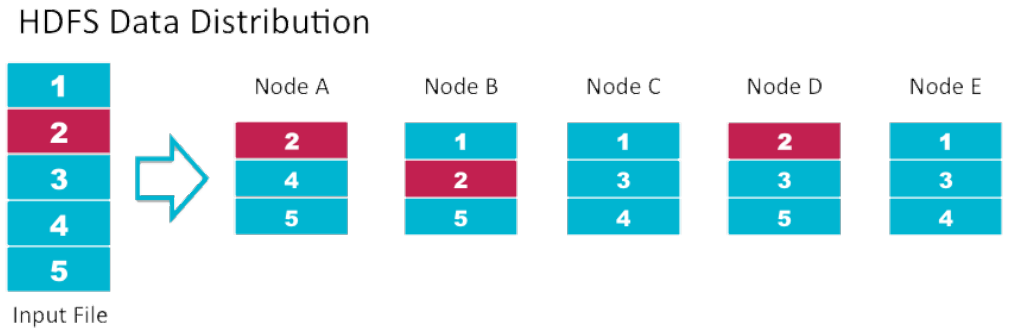
- Componentes base (core) do Apache Hadoop
  - HDFS (Hadoop Distributed File System)
    - Armazenamento distribuído
  - MapReduce
    - Computação distribuída

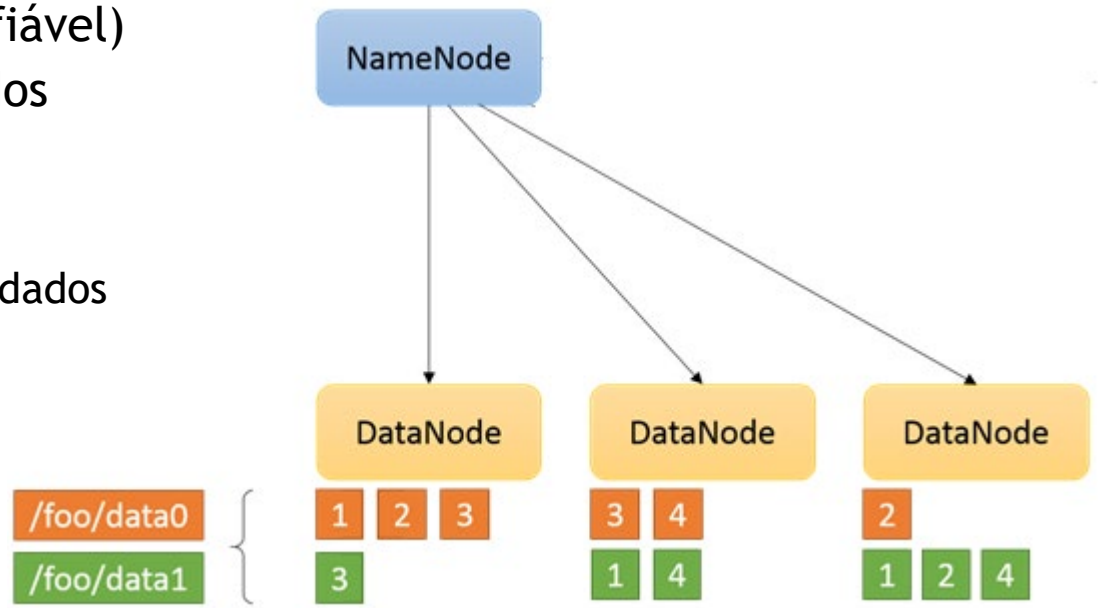
=

+

Introdução ao ecossistema Apache Hadoop

- HDFS (Hadoop Distributed File System)
  - Armazenamento distribuído
  - Otimizado para grandes arquivos
  - Princípio WORM (*Write Once, Read Many Times*)
  - Formam clusters de dados
    - Criam várias réplicas de dados e os espalham nos nós do cluster
    - Máquinas podem falhar: tolerante a falhas (confiável)
    - Recuperação automática: dados são redistribuídos
  - Um cluster HDFS possui basicamente dois nós:
    - Mestre (*NameNode*)
      - Mantém e gerencia informações sobre blocos de dados
    - Escravo (*DataNode*)
      - Armazenam os dados em blocos
    - Pode ainda existir um segundo Mestre
      - *Secondary NameNode*



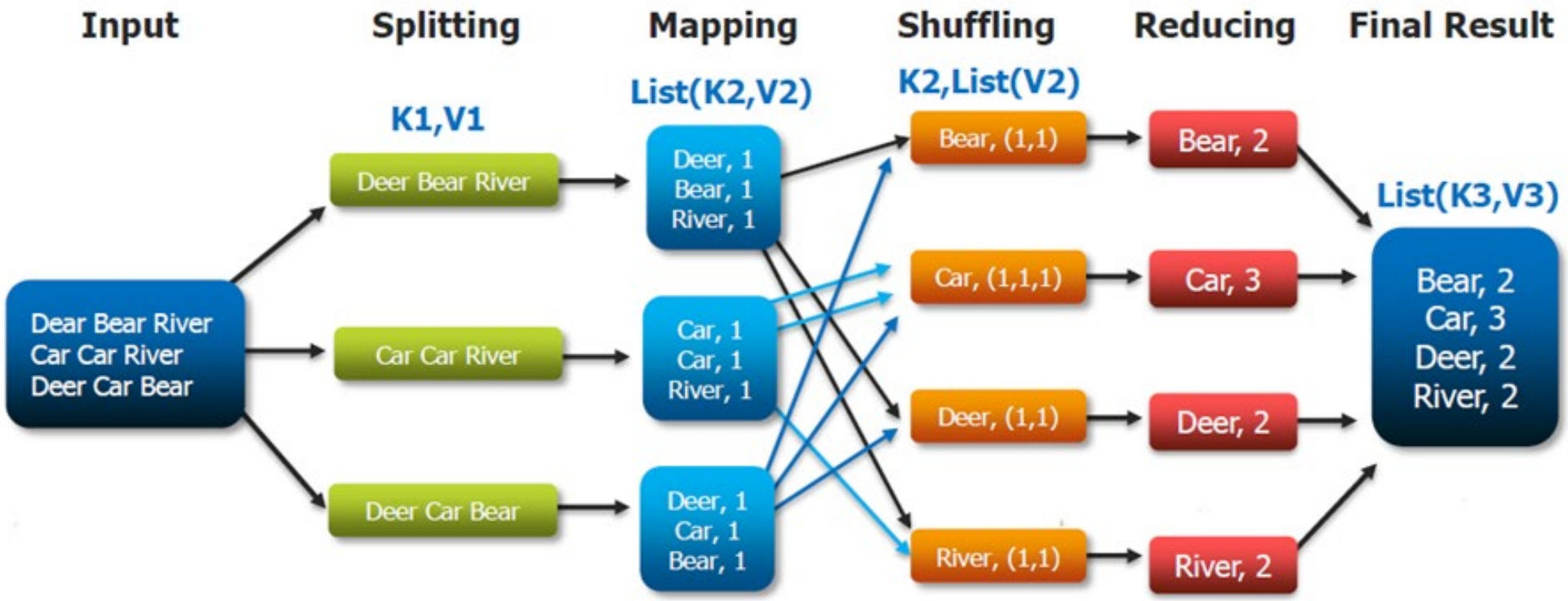


Introdução ao ecossistema Apache Hadoop

- MapReduce
  - Computação distribuída
  - Programação para processamento de grandes conjuntos de dados
    - Pode ser programado em várias linguagens
  - Processa de forma paralela e distribuída os dados armazenados no HDFS
    - São criadas tarefas para processamento em lote
  - Lê os dados como pares chave/valor:
    - `map(K1, V1)`
    - `map(K2, V2)`
  - Basicamente funciona fazendo mapeamento e redução
    - Mapeia os dados
    - E os reduz (classificando)
    - Vários processos são disparados para realizar essas funções
      - *Job trackers*: gerencia as tarefas do MapReduce
      - *Task Trackers*: monitora individualmente cada tarefa de mapeamento e redução

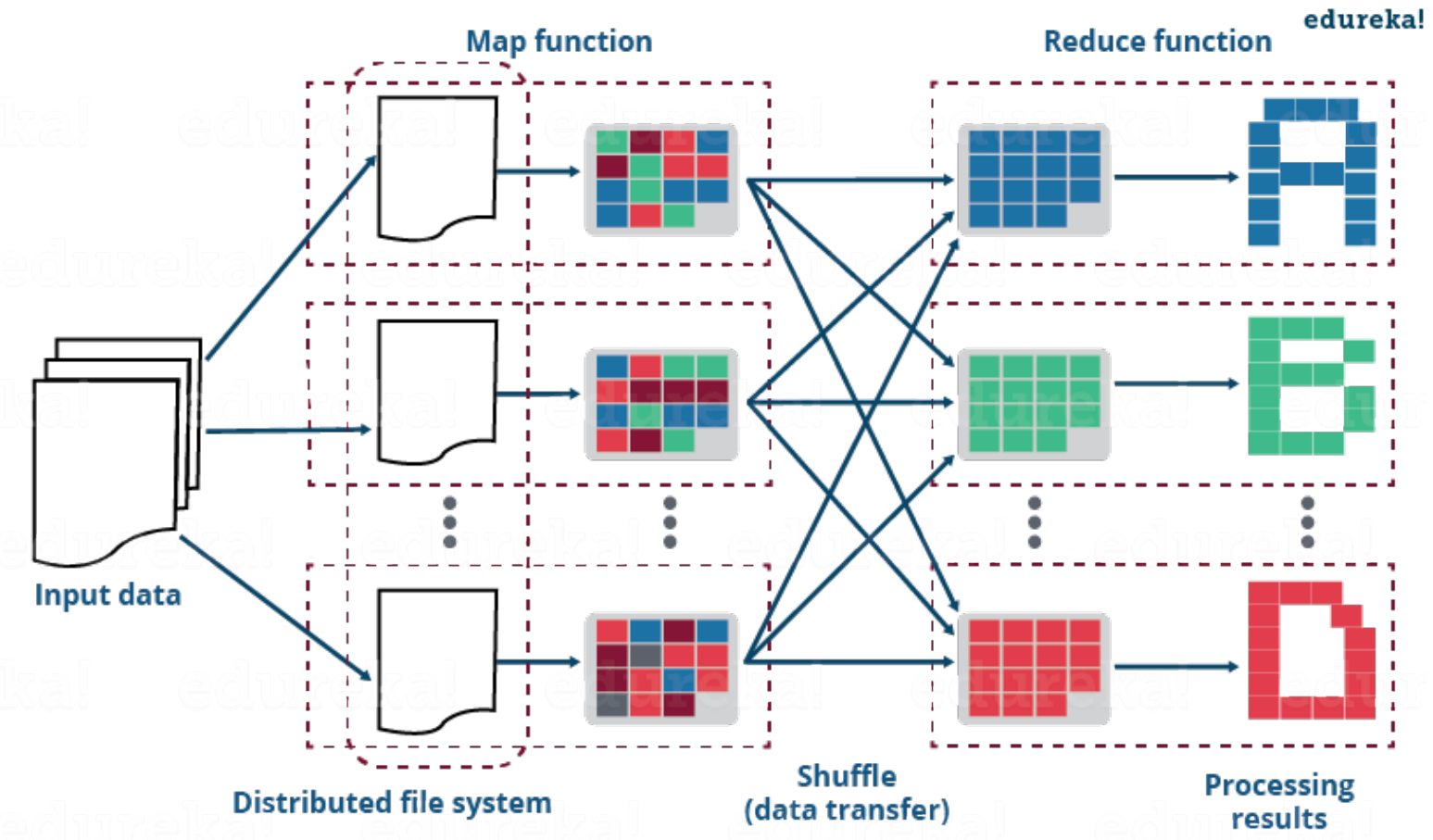
Introdução ao ecossistema Apache Hadoop

- MapReduce
  - Exemplo ilustrativo de funcionamento
    - Deve ser criado um script ou um esquema de aprendizado de máquina, de acordo com problema



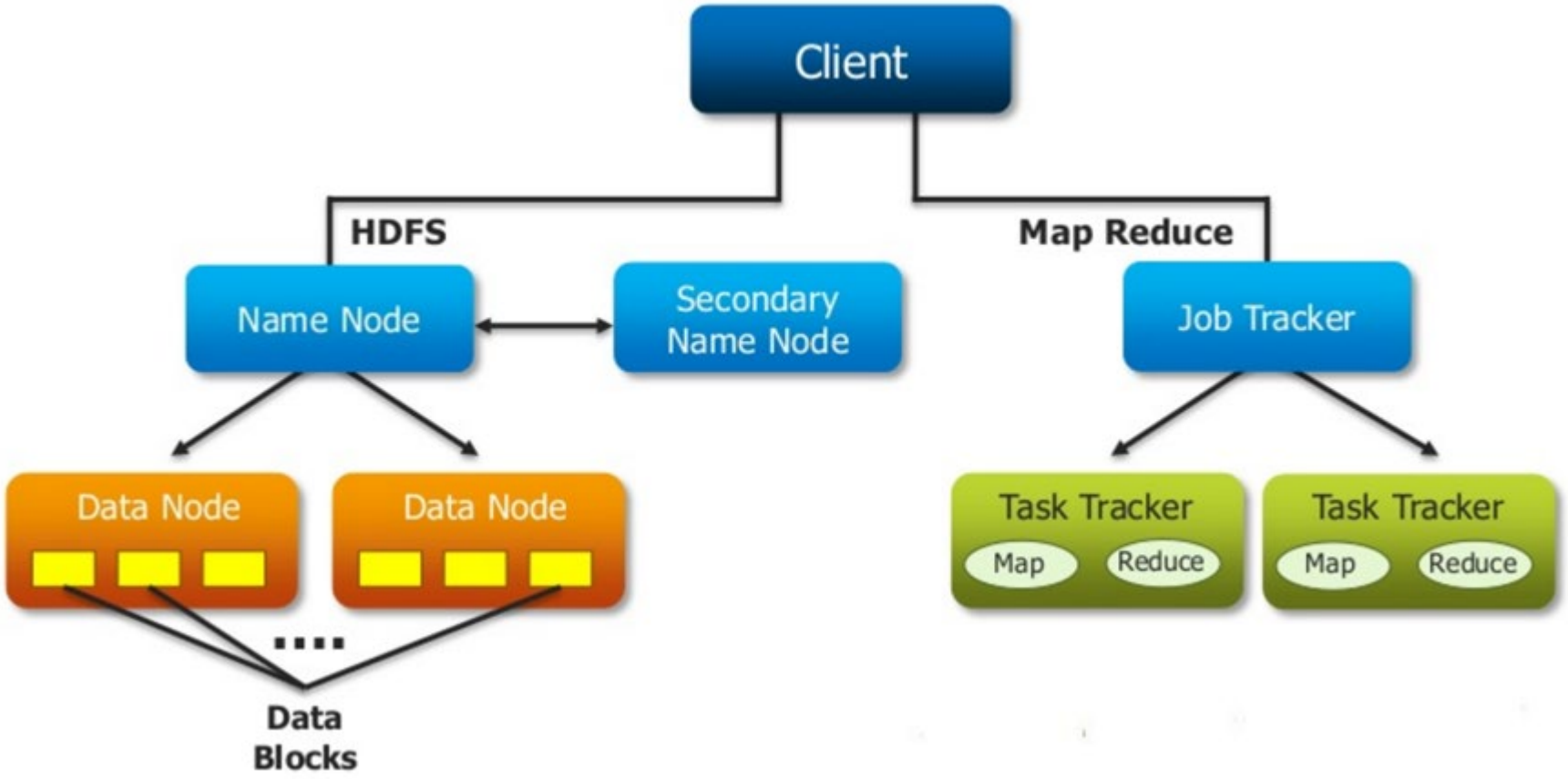
Introdução ao ecossistema Apache Hadoop

- MapReduce
  - Exemplo ilustrativo de funcionamento



Introdução ao ecossistema Apache Hadoop

- Arquitetura Hadoop com HDFS + MapReduce



Programando com MapReduce

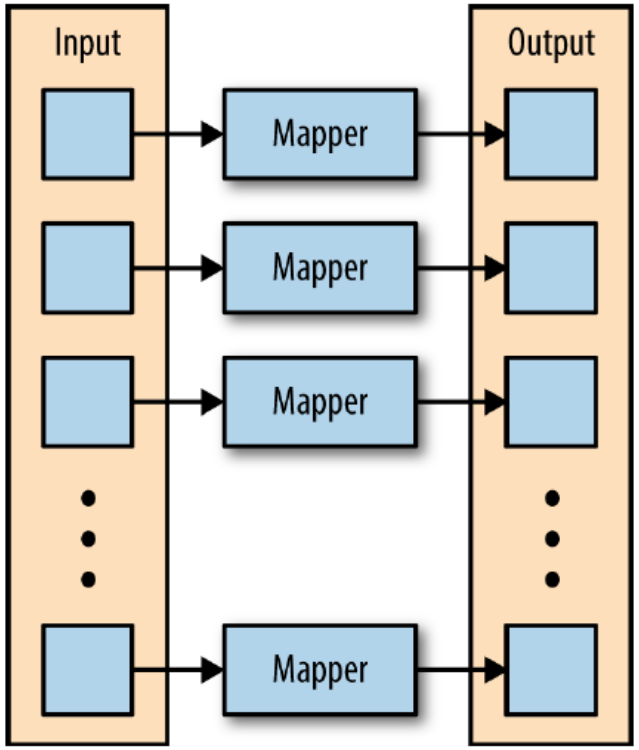
- Como vimos, o MapReduce é utilizado para o processamento dos dados
  - Utilizamos um exemplo pré-existente para contar o número de ocorrências das palavras em um determinado conjunto de dados

- Entretanto, na maioria dos problemas do mundo real se faz necessário que o MapReduce processe dados de maneira diferente
  - Nestes casos, são criados algoritmos específicos

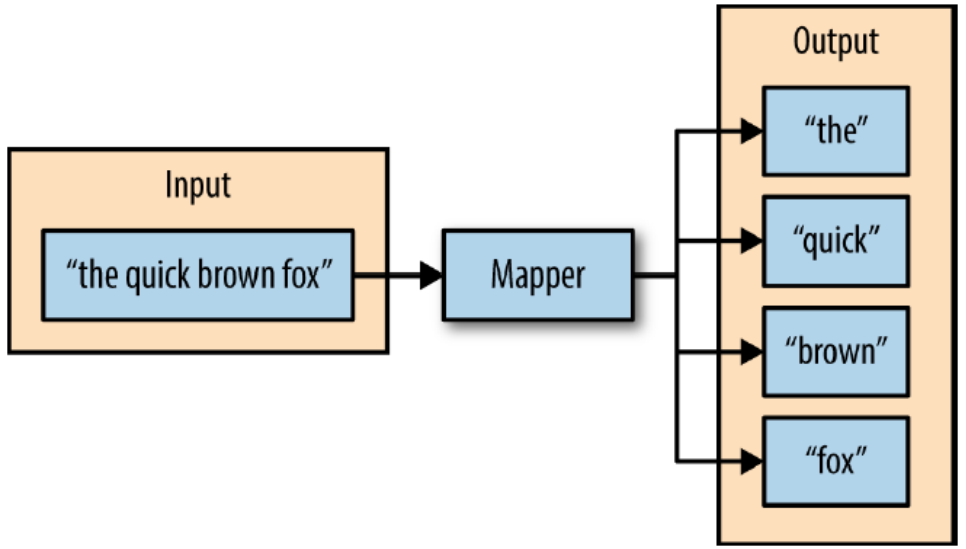
- Para a finalidade de desenvolvimento, é necessário compreender alguns detalhes sobre o funcionamento do MapReduce
  - Basicamente é composto por três fases:
    - Mapeamento (*map*)
    - Embaralhamento e ordenação/classificação (*shuffle and sort*)
    - Redução (*reduce*)

Programando com MapReduce

- Mapeamento (*map*)
  - Nesta fase, uma função (`mapper()`) processa uma série de pares chave-valor (sequencialmente e individualmente), e produz na saída também pares de chave-valor



Exemplo:

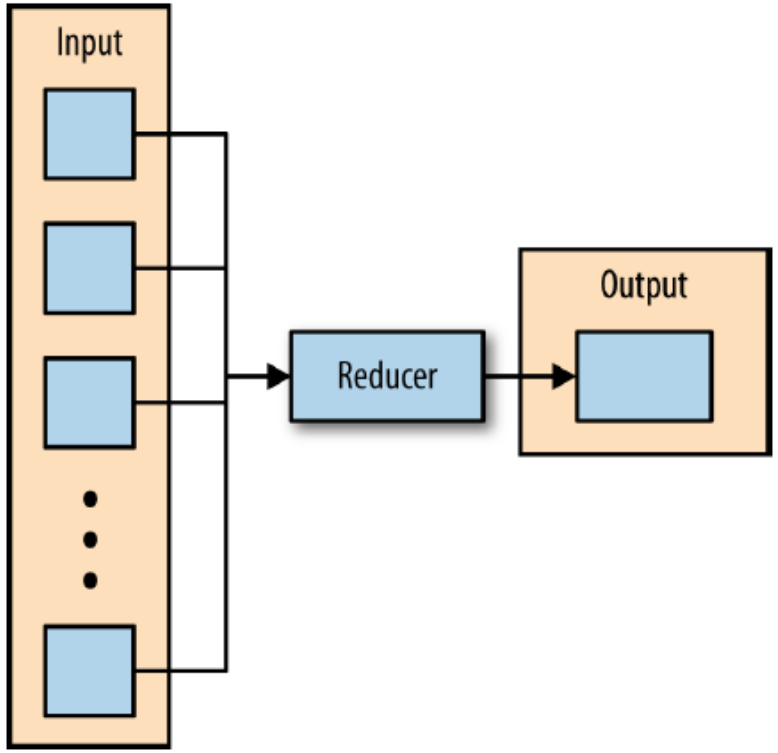


Programando com MapReduce

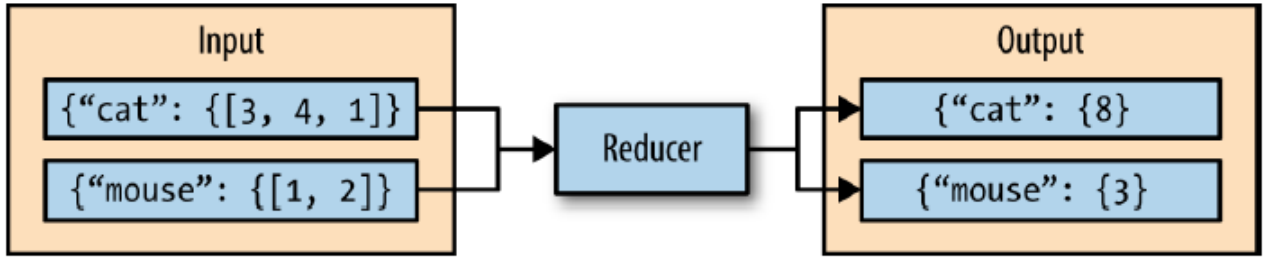
- Embaralhamento e ordenação/classificação (*shuffle and sort*)
  - Nesta fase, à medida que o mapeamento é finalizado, o resultado da saída é enviado para a função de redução (`reducer()`). Este processo é chamado de embaralhamento (*shuffle*).
  - Existe um particionador que controla o fluxo de pares chave-valor da função `mapper()` para `reducer()`
    - Garante que todos os valores com a mesma chave sejam enviados para o mesmo `reducer()`
    - Utiliza cálculo hash sobre o resultado da saída do `mapper()`
  - E na ordenação/classificação todos os pares chave-valor da saída são ordenados para serem enviados para a função `reducer()`

Programando com MapReduce

- Redução (*reduce*)
  - Nesta fase, existe um iterador de valores que é um conjunto de valores para cada chave (única) gerada da saída da fase de mapeamento. A função `reducer()` agrega os valores de cada chave e produz pares de saída chave-valor.



Exemplo:



Programando com MapReduce

- Uma vez compreendido o funcionamento base do MapReduce, vamos proceder à sua programação
  - Por padrão, o MapReduce é programado em linguagem Java
  - Entretanto, recorrendo à API Hadoop Streaming é possível programar as funções de mapeamento e redução em várias outras linguagens
    - Nos nossos trabalhos, vamos utilizar a linguagem Python

Programando com MapReduce

- Primeiro exemplo: contagem de palavras em Python
  - Baixar os arquivos `mapper.py` e `reducer.py` da página da disciplina: `https://goo.gl/A3MhFS`
  - Em um terminal, atribuir as devidas permissões: `chmod a+x reducer.py mapper.py`
  - Examinar o código de cada um desses arquivos em um editor de texto para compreender o seu funcionamento
  - A caráter de teste, executar localmente em um terminal ambos os arquivos:

```
echo 'isto isto eh um teste teste teste de eng de dados' | python mapper.py | sort -t 1 | python reducer.py
```