

Análise de Dados com Base em Processamento Massivo em Paralelo

Aula 6: Processamento Paralelo e Distribuído

Cristina Dutra de Aguiar
ICMC/USP
cdac@icmc.usp.br



Agenda

- Ambientes Computacionais
- Modelo MapReduce
- Apache Spark RDD

Ambientes Computacionais

- Características
 - Grande capacidade de armazenamento e processamento
 - Suporte para a manipulação de *big data*
- Tipos
 - *Cluster* de computadores
 - Ambiente de computação em nuvem


Cluster de Computadores

- Popularizado em meados da década de 90
- Modelo composto por uma coleção de computadores
 - Chamados de nós
 - Dispostos de forma paralela e distribuída
 - Interconectados por redes de alta velocidade

Características

- Processamento
 - **Recebimento** das tarefas
 - **Divisão** das tarefas entre os nós
 - **Execução simultânea** das tarefas nos nós
- Transparência
 - *Cluster* deve ser visto como um **único computador**
- Computadores
 - **Hardware** não precisa ser exatamente igual em cada nó
 - Podem ser **dedicados** ou não

Software

- Sistema operacional
 - Todos os nós devem utilizar o mesmo sistema operacional
 - Objetivos
 - Diminuir a complexidade de configuração e manutenção
 - Facilitar monitoramento, distribuição de tarefas e controle de recursos
- *Middleware* 
 - Sistema que permite o gerenciamento do *cluster*
 - Diretamente relacionado ao sistema operacional
 - Oferece uma interface para configuração

instalado em uma
máquina chamada
nó mestre
(nó controlador)

Tipos de Cluster

- Voltados a atender aos requisitos específicos das aplicações
 - *Cluster* de alto desempenho
 - *Cluster* de alta disponibilidade
 - *Cluster* para balanceamento de carga
- Combinação de tipos de *cluster*
 - Suprir às demandas das aplicações
 - Exemplo
 - *cluster* de alta disponibilidade para funcionamento 24 x 7
 - *cluster* de balanceamento de carga para garantir eventual aumento de requisições em períodos de pico

Cluster de Alto Desempenho

- Voltado à computação intensiva
 - Realiza o **processamento coletivo** de uma única **tarefa computacional complexa**
- Os nós devem, idealmente
 - Ser majoritariamente **homogêneos** em termos de arquitetura de processadores e de sistema operacional
 - **Compartilhar** uma rede dedicada
 - Estar **acoplados** de forma próxima

Cluster de Alta Disponibilidade

- O sistema deve **funcionar** continuamente
 - Mesmo em caso de eventuais **falhas**
- Manutenção de nós redundantes
 - Quando um nó falha, ele é **substituído** por outro nó sem prejuízo
- Recursos que podem ser utilizados
 - Ferramentas de monitoramento dos nós para investigar falhas
 - Replicação (redundância) de sistemas
 - Computadores para substituição imediata de máquinas com problemas
 - Uso de geradores para garantir o funcionamento do sistema

Cluster para Balanceamento de Carga

- **Divisão** o mais **uniforme** possível das tarefas entre os nós
 - Cada nó deve receber e atender a uma tarefa
 - Nós não devem, necessariamente, dividir uma tarefa com outros nós
- Recursos que podem ser utilizados
 - Ferramentas de monitoramento dos nós para analisar a carga
 - Direcionamento de tarefas para nós que possuam menor quantidade de tarefas

Cluster Beowulf

- Padrão de *cluster* disponibilizado pela NASA em 1994
- Características
 - A conexão entre os nós pode ser feita por redes Ethernet
 - Não é necessário o uso de *hardware* específico ou potente
 - Sistema operacional
 - Deve ser baseado em código aberto
 - Deve conter as ferramentas necessárias para a configuração do *cluster*

Cluster Beowulf: Nó Mestre e Nós Escravos

Front-End

Nó Mestre

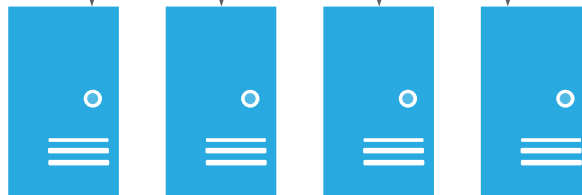


Nó Mestre (um ou mais)
Controla os nós escravos



Switch

Back-End



Nós Escravos

Funcionalidades

Distribuição das tarefas

Monitoramento do desempenho

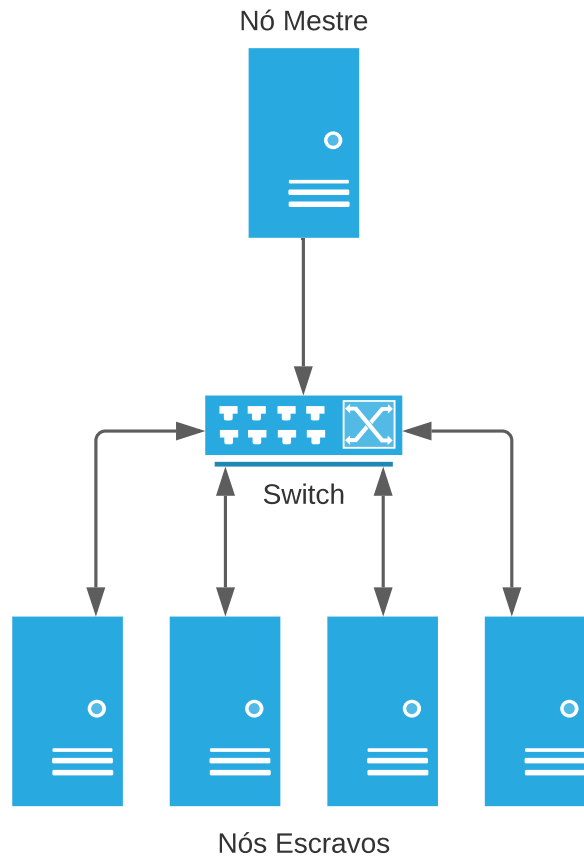
Gerenciamento de possíveis falhas

Realização do balanceamento de carga

Cluster Beowulf: Nó Mestre e Nós Escravos

Front-End

Back-End



Funcionalidades

Execução paralela das tarefas
Dedicação exclusiva ao *cluster*

Nós escravos

Dificuldades de Cluster de Computadores

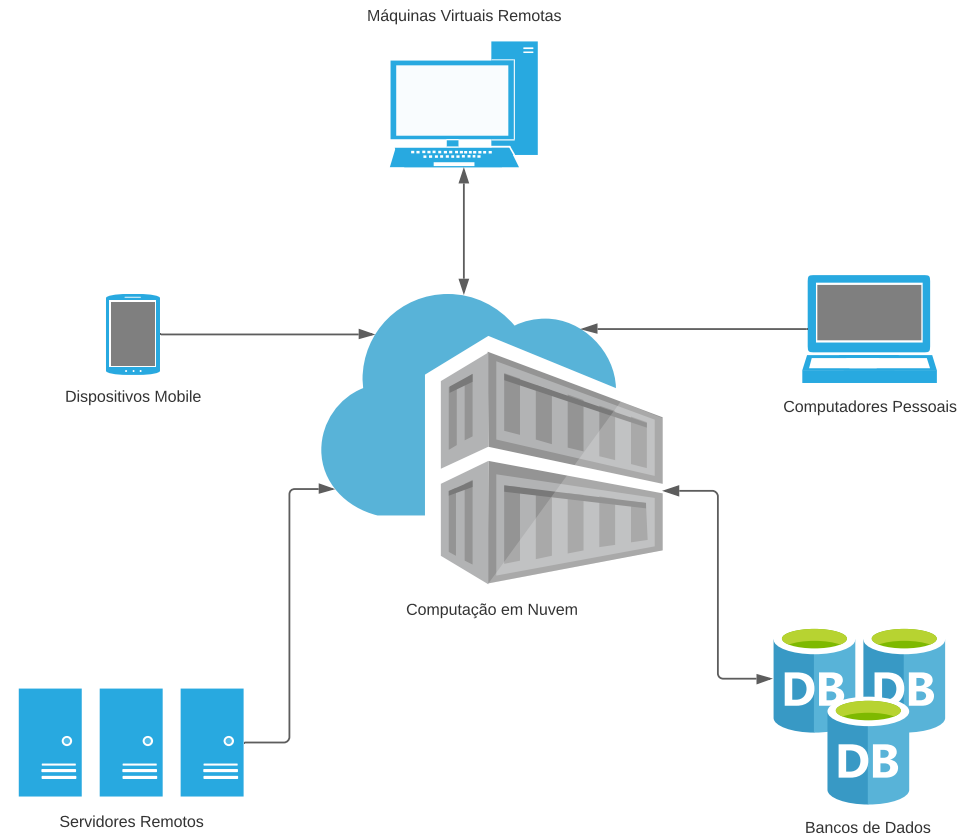
- Usuários
 - Enxergam o *cluster* como uma coleção de computadores independentes
- Manipulação do *cluster* pode ser uma tarefa complexa
 - Manipulação dos componentes
 - Divisão de tarefas entre os nós
 - Gerenciamento da comunicação entre os nós

Computação em Nuvem

- Modelo que possibilita acesso a
 - Recursos computacionais compartilhados e interligados via rede
- Exemplos de recursos
 - Redes, servidores, equipamentos de armazenamento
 - Aplicações, serviços
- Nuvem
 - Abstração que oculta a complexidade de infraestrutura

Abstração Nuvem

Metáfora para
Internet ou infraestrutura
de comunicação entre
ambientes computacionais



Definição Segundo NIST

Computação em nuvem é um modelo que permite **acesso ubíquo, conveniente e sob demanda** via rede a um conjunto **compartilhado e configurável** de recursos **computacionais** que pode ser **rapidamente fornecido e liberado** com esforços mínimos de gerenciamento ou interação com o provedor de serviços

Composição Segundo NIST

- Tecnologias chave
 - Redes de longa distância rápidas
 - Computadores servidores poderosos e/ou baratos
 - Virtualização de alto desempenho
- Ambiente de nuvem
 - Cinco características essenciais
 - Três modelos de serviços
 - Quatro modelos de implantação

Composição Segundo NIST

- Tecnologias chave
 - Redes de longa distância rápidas
 - Computadores servidores poderosos e/ou baratos
 - Virtualização de alto desempenho
- Ambiente de nuvem
 - Cinco características essenciais
 - Três modelos de serviços
 - Quatro modelos de implantação

Características Essenciais

- Serviço sob demanda
 - Recursos são acessados de forma direta e **sob demanda**
 - Alocação e liberação de recursos ocorre **sem interação** entre o usuário e o provedor
 - **Usuários têm interação mínima com o provedor**
- Serviço sob demanda
 - Recursos são acessados de forma direta e **sob demanda**
 - Alocação e liberação de recursos ocorre **sem interação** entre o usuário e o provedor
 - **Usuários têm interação mínima com o provedor**

Características Essenciais

- Compartilhamento de recursos
 - Recursos são agrupados e **compartilhados** entre diversos usuários
 - **Usuários não precisam ter conhecimento acerca da localização dos recursos**
- Rápida elasticidade
 - Recursos são **rapidamente alocados e liberados** a qualquer momento
 - Aplicação pode demandar **qualquer quantidade** de recursos
 - **Usuários têm a sensação de capacidade de armazenamento e processamento infinita**

Características Essenciais

- Serviço mensurável
 - Baseado no modelo *pay-as-you-go*
 - Identificação de quais recursos foram utilizados
 - Cobrança apenas desses recursos utilizados
 - Usuários pagam apenas os serviços que usam e não pagam pelos recursos ociosos

Infraestrutura da Nuvem

- Coleção de *hardware* e *software*
 - Oferece suporte para as cinco características essenciais

software

Camada de Abstração

exemplos

sistema operacional, sistema gerenciador de banco de dados

hardware

Camada Física

exemplos

rede, servidores, armazenamento

Modelos de Serviços

nível de abstração

alto



baixo

Camada de Abstração

Camada Física



Software as a Service



Platform as a Service



Infrastructure as a Service

Software as a Service (SaaS)

- Recursos fornecidos são os [programas aplicativos](#)
- Acesso
 - Diferentes dispositivos cliente
 - Uso de uma interface de navegador ou uma interface de programa
- Usuários
 - Podem apenas configurar seus aplicativos específicos

Platform as a Service (PaaS)

- Recurso fornecido é a [plataforma para a execução dos aplicativos](#)
 - Implantação de aplicativos criados ou adquiridos
- Aplicativos
 - Desenvolvidos usando linguagens de programação, bibliotecas, serviços e ferramentas presentes no provedor
 - Devem ser executados nos recursos da nuvem
- Usuários
 - Têm controle sobre os aplicativos implantados
 - Podem definir configurações para o ambiente de hospedagem de aplicativos

Infrastructure as a Service (IaaS)

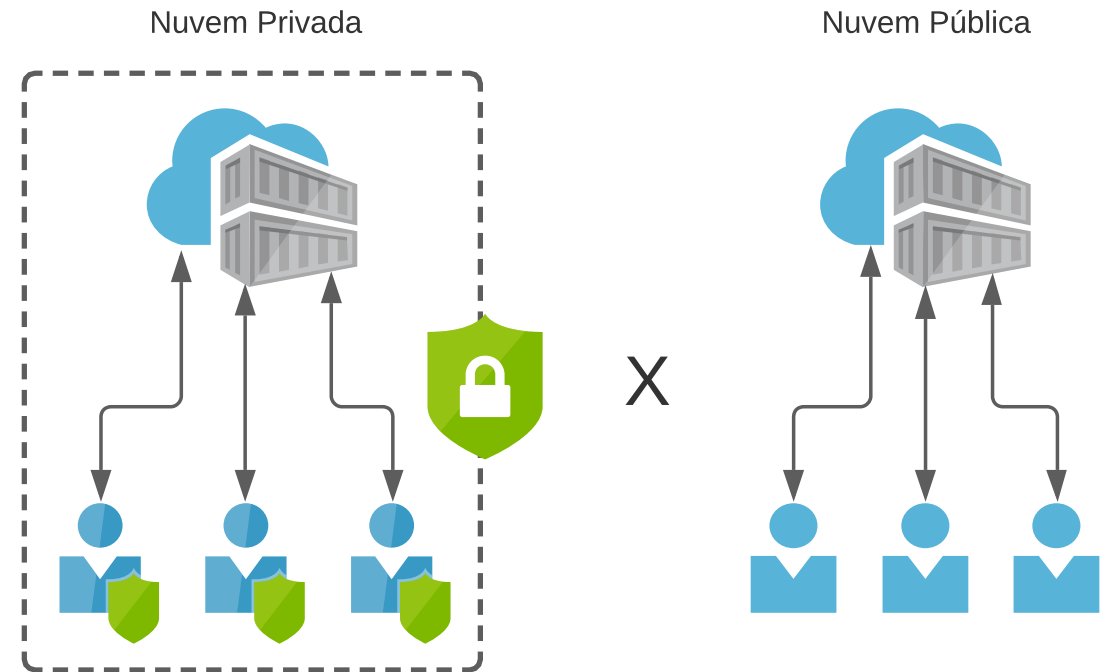
- Recursos fornecidos são relacionados à **infraestrutura**
 - Processamento, armazenamento, redes
 - Outros recursos necessários para implantar e executar qualquer *software*
 - Uso de máquinas virtuais acessadas de forma transparente
- Usuários
 - Têm controle sobre os sistemas operacionais, armazenamento e aplicativos implantados
 - Podem ter controle limitado sobre componentes de rede selecionados

Modelos de Implantação

- Diferenciam-se pelas **restrições de acesso**

- Modelos

- Nuvem privada
- Nuvem comunitária
- Nuvem pública
- Nuvem híbrida



Cluster de Computadores e Computação em Nuvem

	Cluster de Computadores	Computação em Nuvem
Funcionamento dos Componentes	responsabilidade do usuário	responsabilidade do provedor de serviços
Custos de Funcionamento	todos os custos envolvidos, inclusive os referentes aos recursos ociosos	somente os recursos usados (serviço mensurável)

Cluster de Computadores e Computação em Nuvem

	Cluster de Computadores	Computação em Nuvem
Adição ou Remoção de Componentes	responsabilidade do usuário	responsabilidade do provedor de serviços (serviço sob demanda)
Visão do Usuário	transparente (vários computadores conectados para suprir uma necessidade)	ubíqua (serviço criado para atender uma necessidade, sem conhecer detalhes de funcionamento)

Agenda

- Ambientes Computacionais
- Modelo MapReduce
- Apache Spark RDD

MapReduce

- Introduzido pela Google em 2004
- Modelo de programação funcional
- Voltado ao processamento massivo de dados
 - Projetado para *clusters* formados por computadores comuns
 - Foco em processamento paralelo e distribuído
 - Garante alta disponibilidade e tolerância a falhas

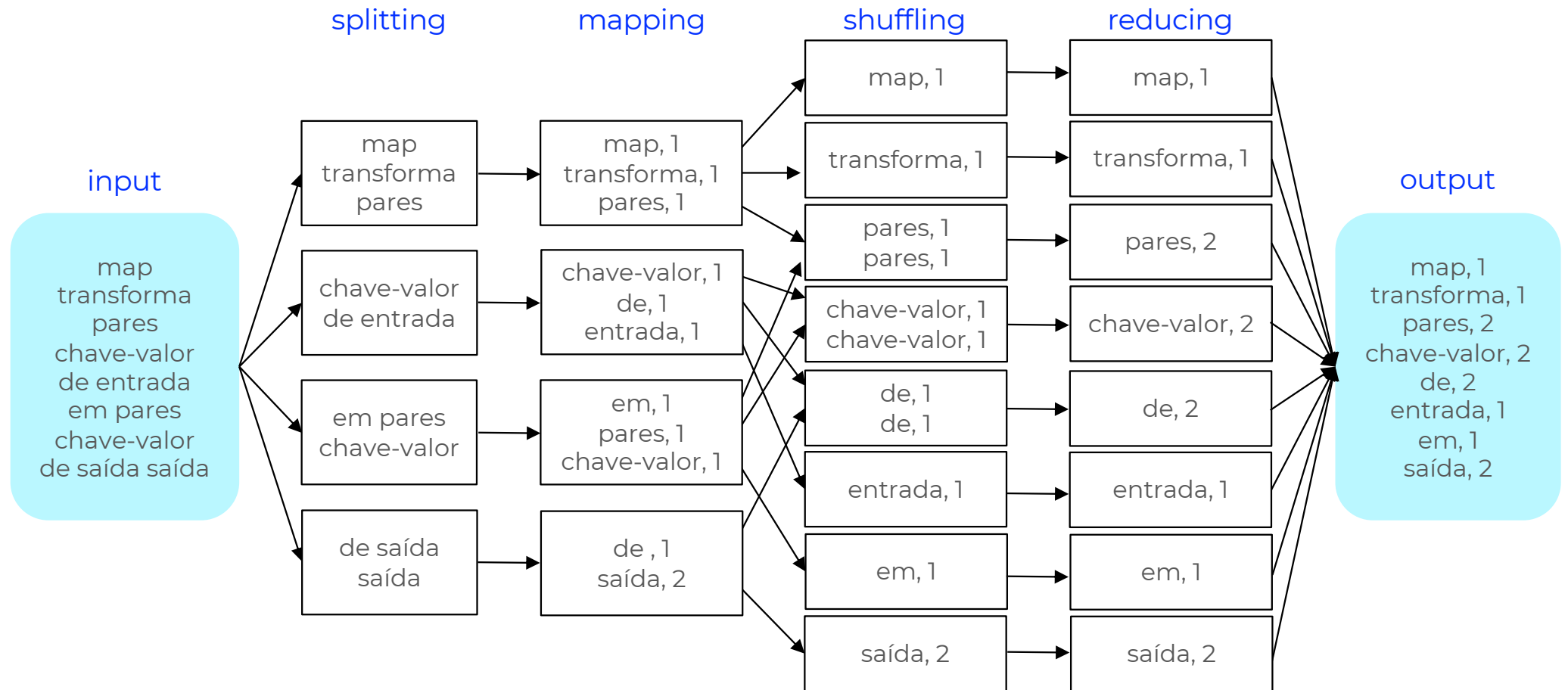
Abstração

- Esconde a complexidade inerente ao paralelismo e à distribuição
 - Armazenamento dos dados
 - Particionamento (distribuição e replicação) dos dados de entrada
 - Balanceamento de carga
 - Escalonamento da execução das tarefas nos nós do *cluster*
 - Manipulação de falhas
 - Gerenciamento da comunicação entre os nós do *cluster*
- Possibilita
 - Manipulação de gigantescos volumes de dados

Funções Base

- Map
 - Processa dados de entrada na forma de **pares chave-valor**
 - Transforma esses dados em saídas intermediárias na forma de **pares chave-valor**
 - Reduce
 - Processa as saídas intermediárias na forma de **pares chave-valor**
 - Agrupa os valores associados a uma mesma chave em um **resultado único**
 - Produz **pares chave-valor**
- Cada *job* MapReduce executa as funções map e reduce em sequência

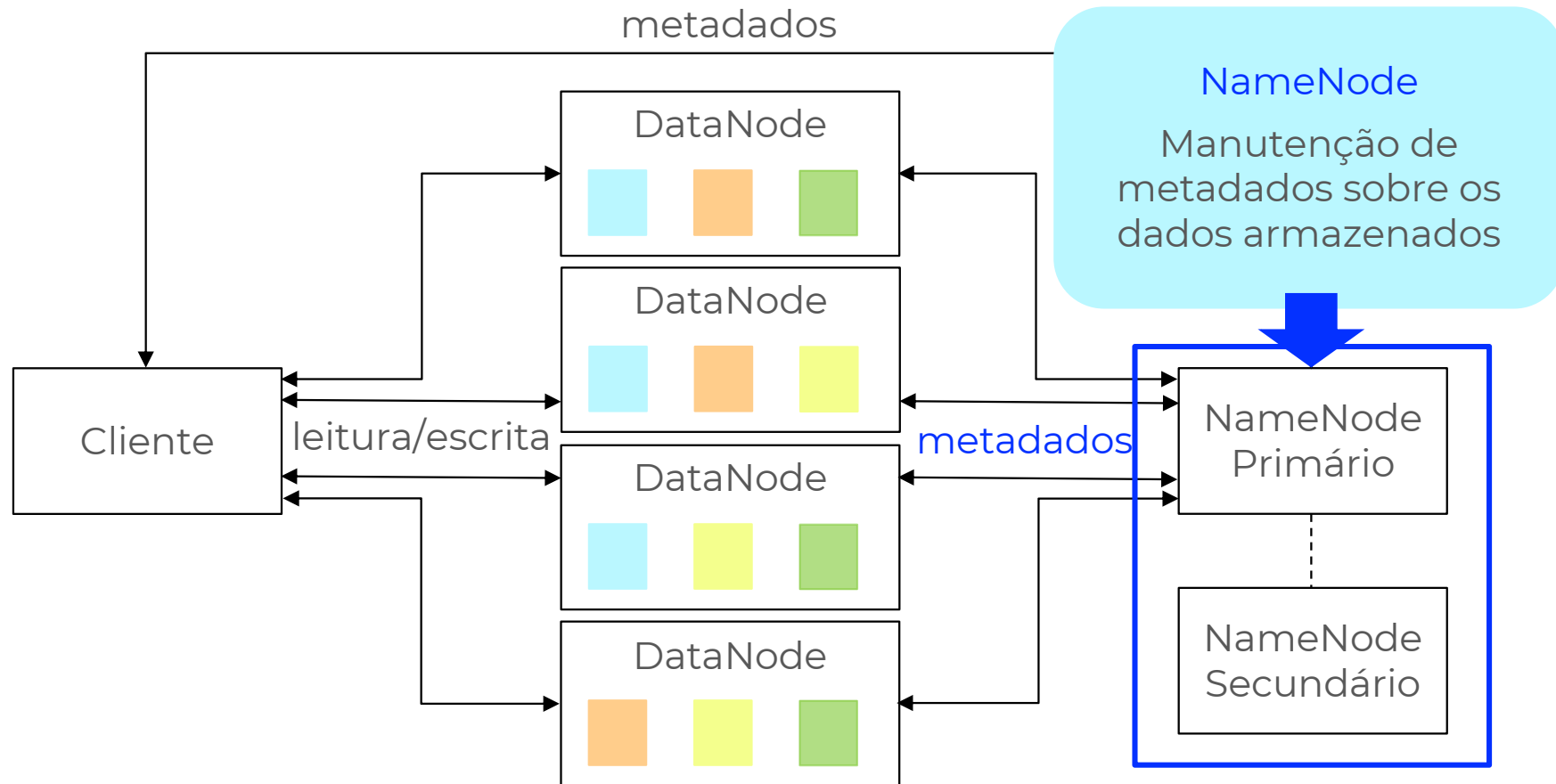
Contador de Palavras



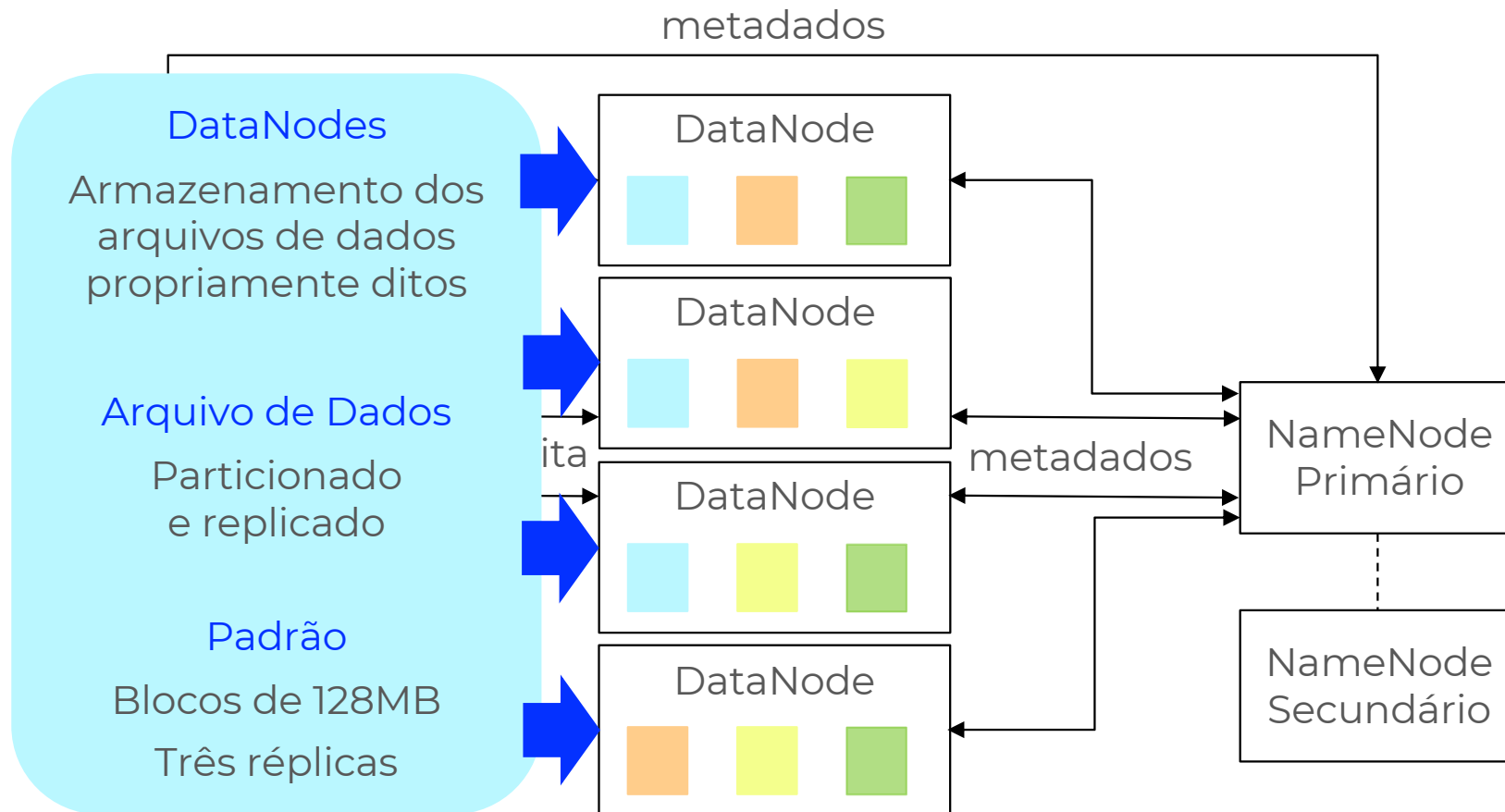
HDFS (Hadoop Distributed File System)

- Sistema de arquivos distribuídos
 - Baseado no Google File System (GoogleFS)
- Características
 - Nó mestre
 - Controle dos outros nós
 - Servidores de dados
 - Armazenamento dos “pedaços” dos arquivos

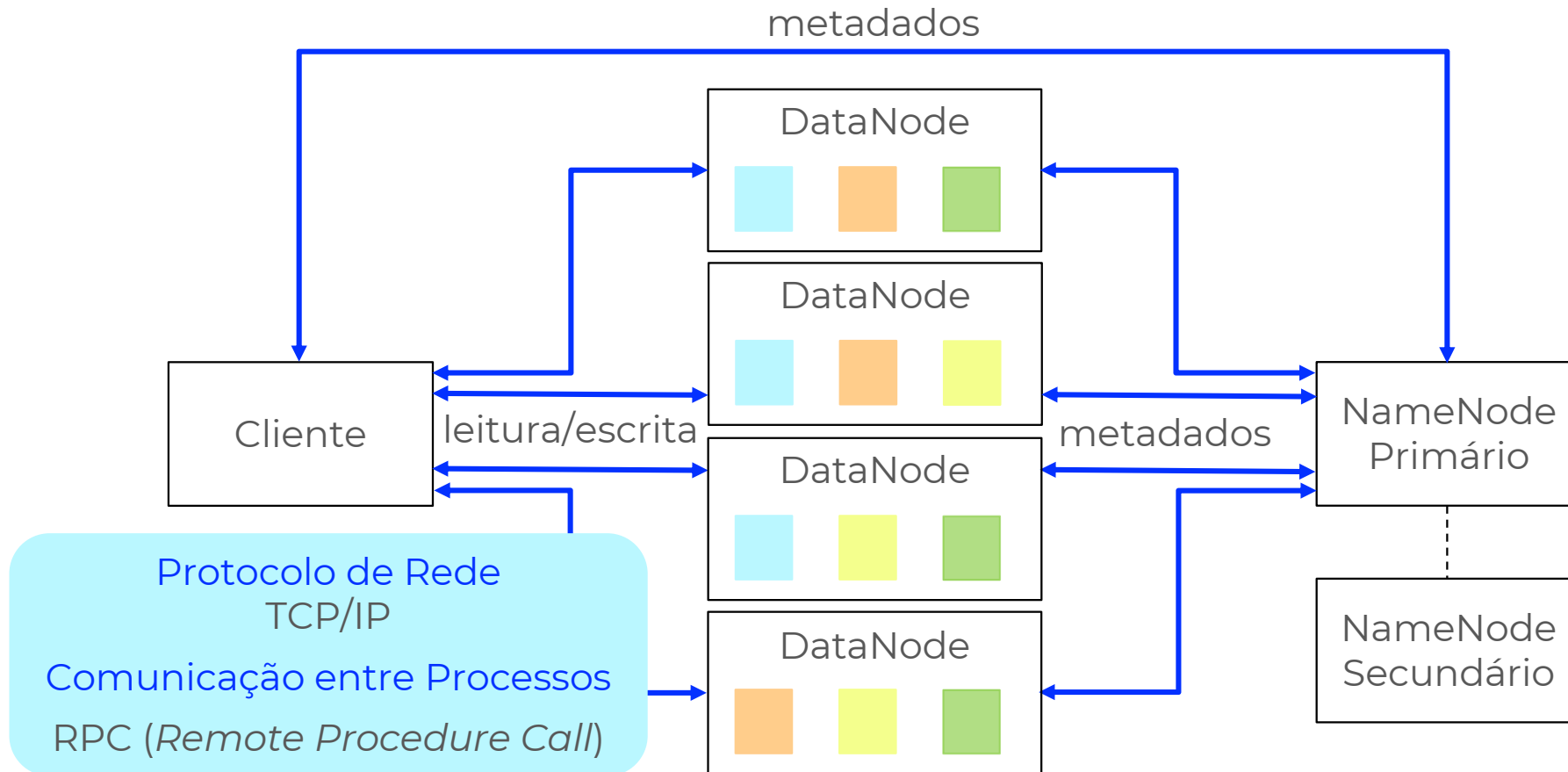
Arquitetura do HDFS



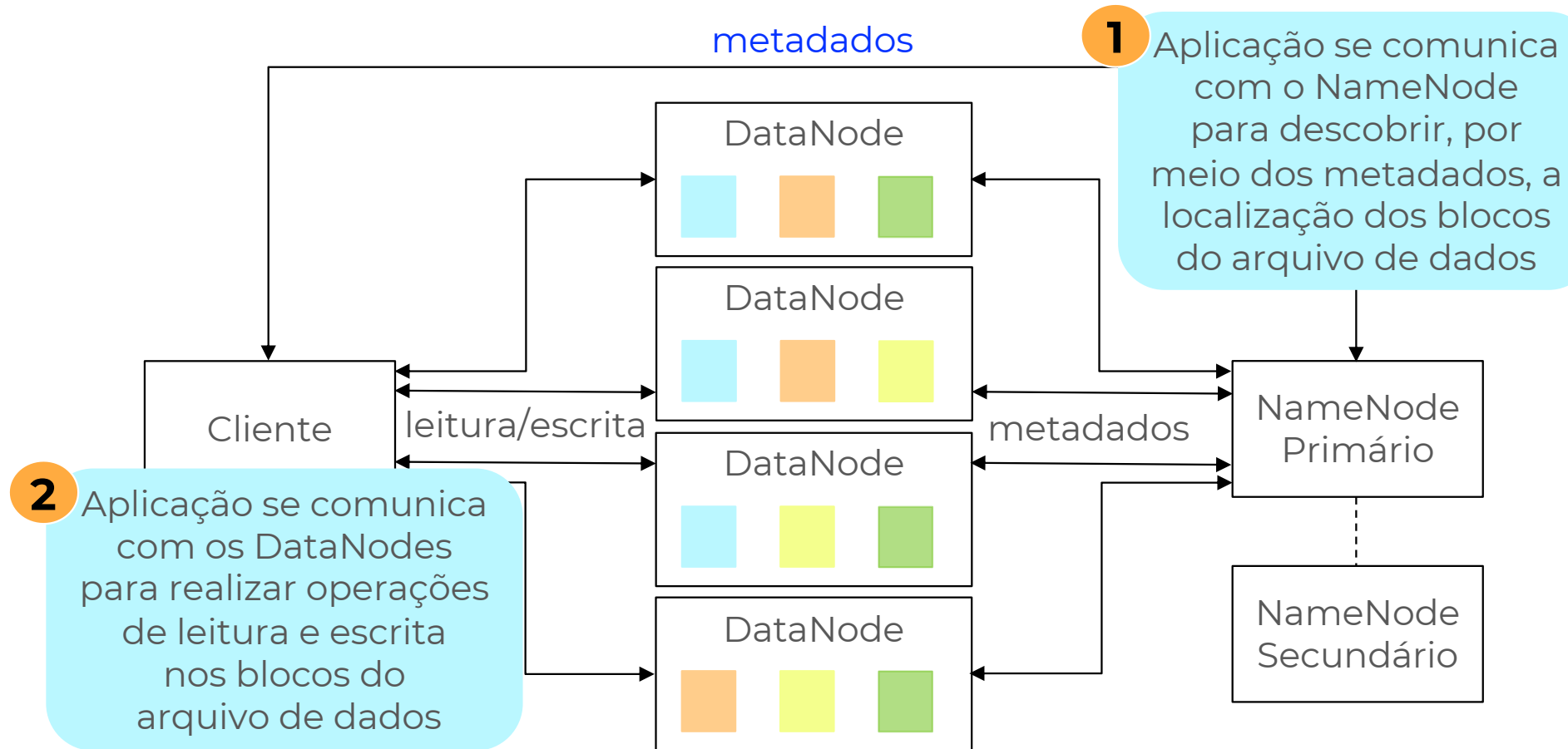
Arquitetura do HDFS



Comunicação



Comunicação



Implementações do MapReduce

- Objetivo
 - **Abstrair** e **facilitar** o uso dos conceitos subjacentes
- *Frameworks* de processamento paralelo e distribuído
 - **Hadoop**
 - **Spark**



Apache **Hadoop**



Apache **Spark**

Apache Hadoop

- Escrito na linguagem de programação [Java](#)
- Evolução
 - [Originalmente](#) projetado para
 - Executar sobre o [HDFS](#)
 - Incorporar os conceitos relacionados a [MapReduce](#)
 - [Evoluiu](#) para uma [plataforma](#) que integra um número variado de máquinas de armazenamento e processamento
- Necessidade de incluir [YARN](#) como um novo componente

Visão Geral da Arquitetura do Hadoop

HDFS

sistema de arquivos
para gerenciar o
armazenamento dos
dados

Yarn

plataforma para
gerenciamento

MapReduce

modelo de
programação

Apache Spark

- Escrito na linguagem de programação [Scala](#)
- Características
 - Executa sobre o [HDFS](#)
 - Incorpora e estende os conceitos relacionados a [MapReduce](#)
 - Baseado no uso de conjuntos de [dados distribuídos e resilientes \(RDDs\)](#)
 - Possibilita o agendamento de tarefas na forma de [grafos acíclicos e direcionados](#)

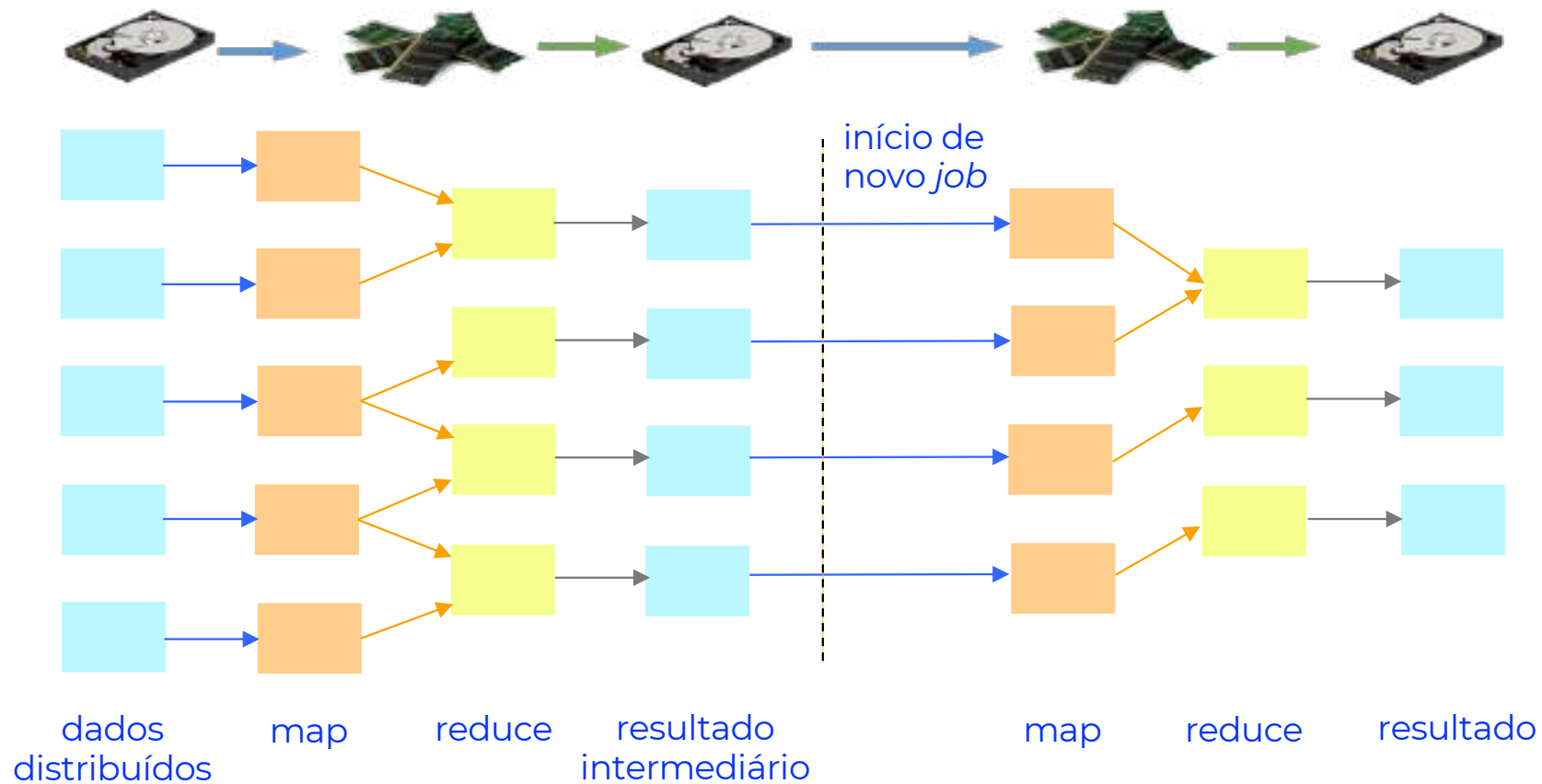
Resilient Distributed Datasets (RDDs)

- **Abstrações** para a manipulação de dados
 - Toda manipulação de dados deve ser feita sobre os RDDs
 - Coleções de blocos de dados distribuídos
 - Capazes de serem **reconstruídos** em caso de falhas
 - Permitem o armazenamento de resultados intermediários em **memória primária**
 - Importante quando se deseja reutilizar essas saídas em operações futuras
- **Resultado: Uso de RDDs diminui o número de acessos a disco**

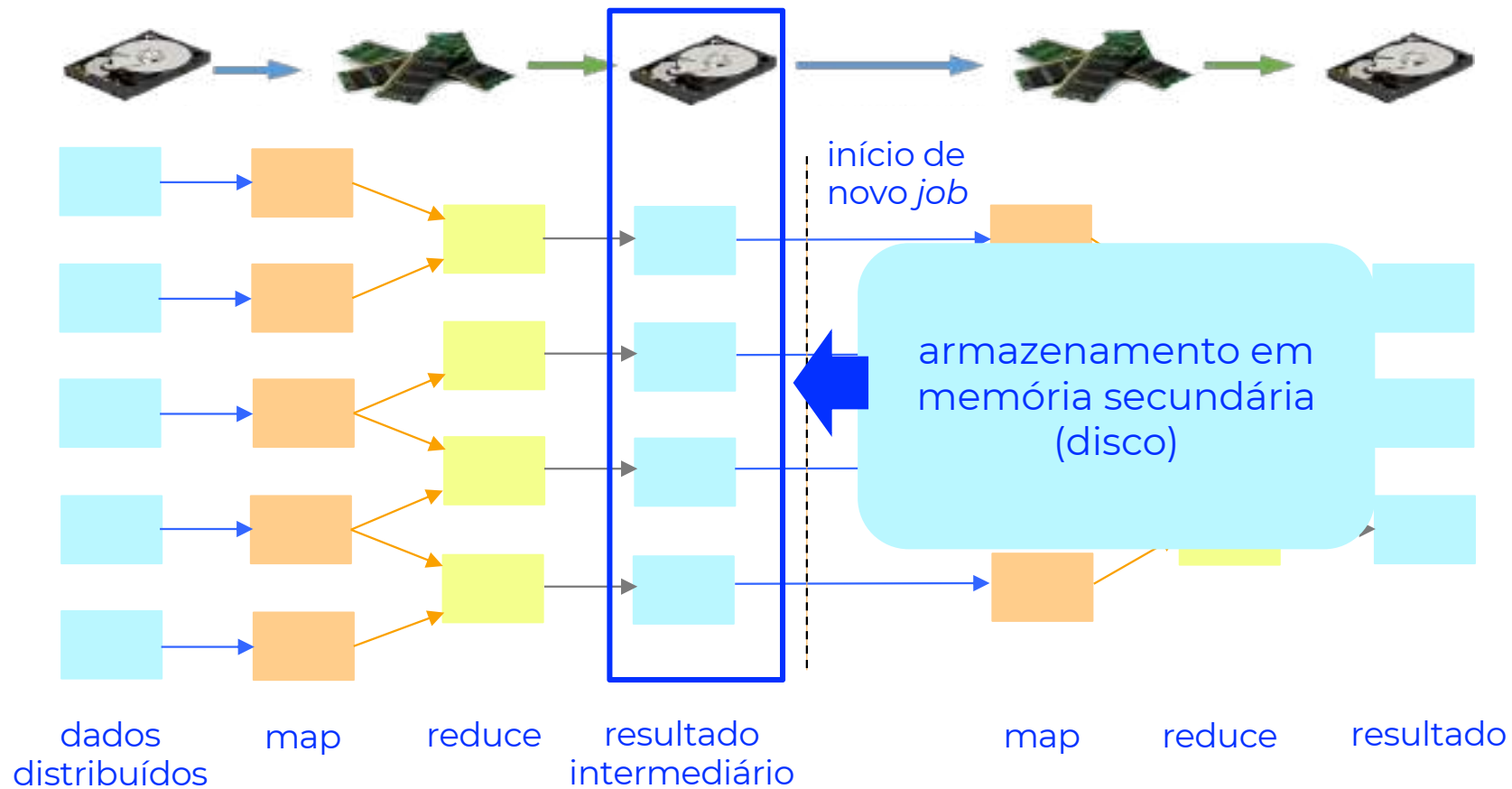
Grafos Acíclicos Direcionados

- Permitem o **agendamento** de estágios
 - Processamento de tarefas consiste de vários estágios
- Executam os estágios de forma **paralela**
 - Desde que não existam dependências entre os estágios
- **Resultado: Uso de grafos acíclicos direcionados melhora o desempenho computacional**

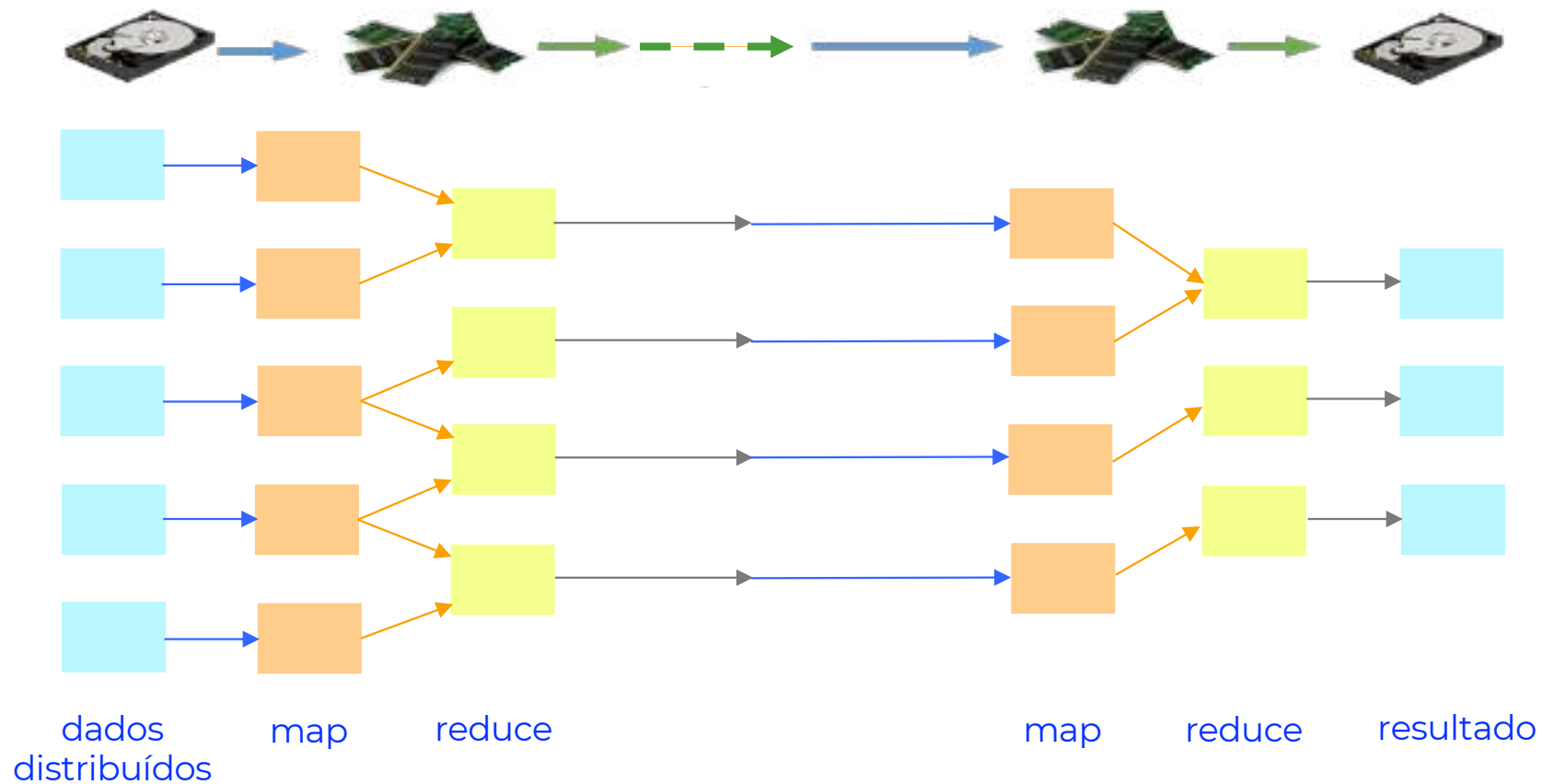
Fluxo de Dados no Hadoop



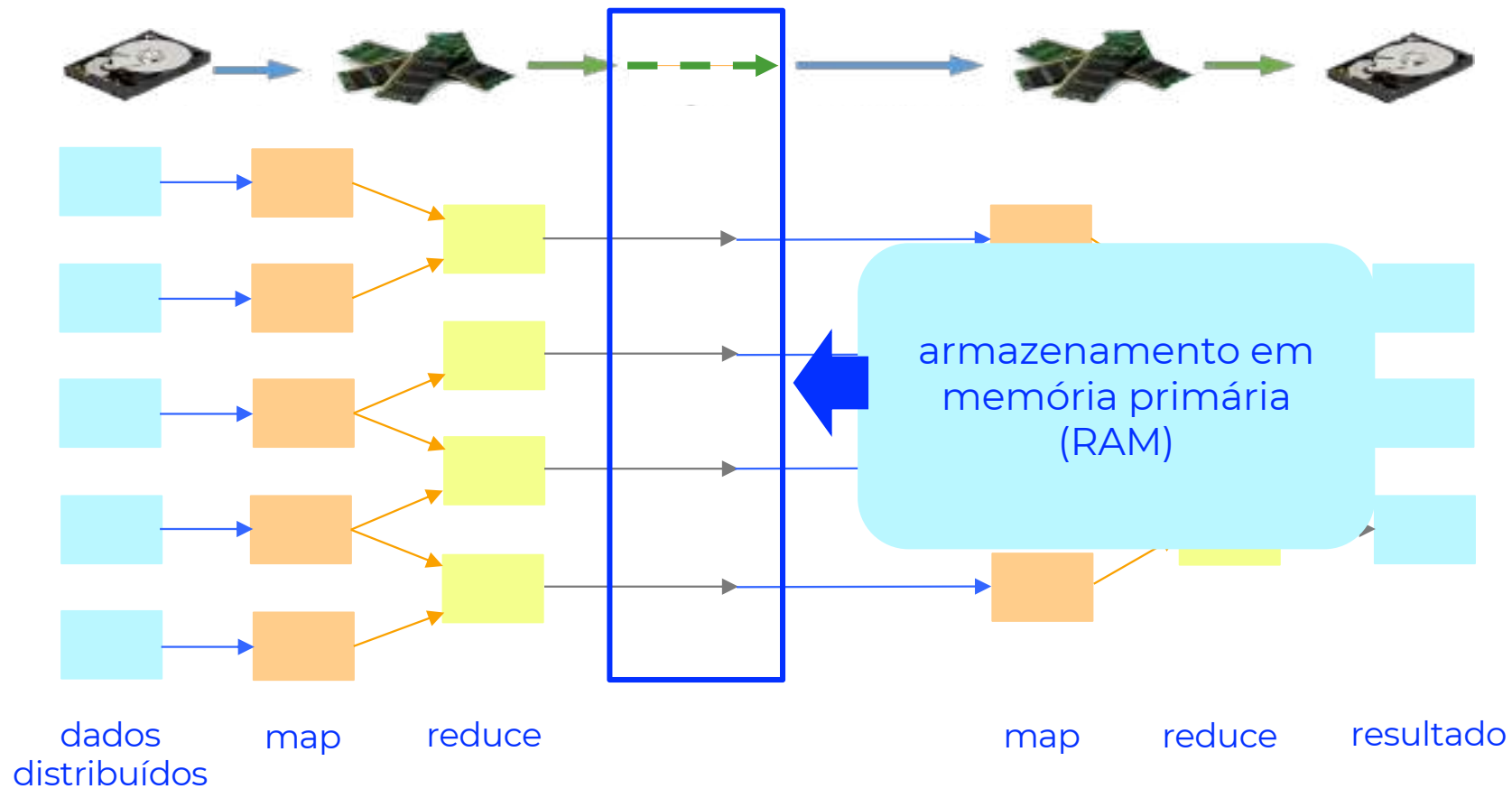
Fluxo de Dados no Hadoop



Fluxo de Dados no Spark



Fluxo de Dados no Spark



Ecosystem Hadoop (Tecnologias)

