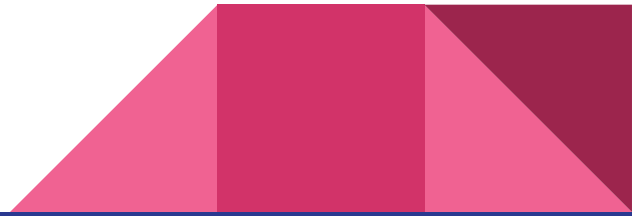
Arcabouços para Big Data e modelo de Dataflow

Fernanda de Camargo Magano

25 de maio, 2018

Motivação

- Cada ferramenta defende seus **pontos fortes**
- Seja melhor modelo de dados, programação ou execução
- Todas usam modelo de **Dataflow**
- As várias ferramentas têm mesma expressividade
- Usam níveis diferentes de abstração



Motivação

- Difícil ter uma imagem do **modelo de programação** que roda abaixo das ferramentas
- Difícil enxergar a expressividade delas
- Vamos entender
  - visão geral das APIs
  - relação com computação paralela

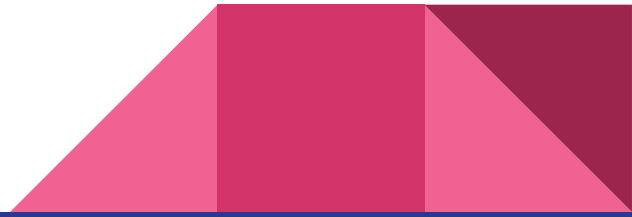
Modelo de Dataflow

- É o que melhor descreve todos os níveis de abstração
- Representação por grafo dirigido de atores
- Modela kernels independentes (paralelizáveis)
- Grafo de **dependência verdadeira** de dados
- Execução do kernel acionada pelos dados

Grafo de dependências (exemplo)

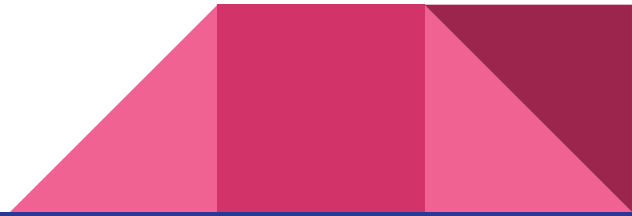
1. a = b + c
2. (if a > 10) jump to L1
3. d = b * e
4. e = d +1
5. L1: d = e/2

Cinco estados: S1, S2, S3, S4, S5



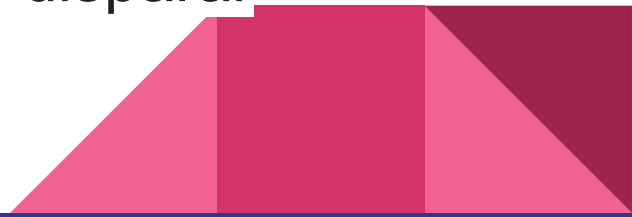
Modelo de Dataflow

- Expressivo
- Descreve lotes, micro-lotes e modelos de streaming
- As várias ferramentas de Big Data **compartilham** quase os mesmos **conceitos** básicos
- **Diferem** principalmente em escolhas de **implementação**.



Definições - atores e kernels

- **Ator**: unidade funcional de computação sobre tokens
- Funcional = sem efeitos colaterais = stateless
- **Kernel** de um ator: função que mapeia os tokens de entrada para tokens de saída
- O processamento consiste de disparos de atores
- Código do ator: normalmente uma instrução de máquina
- **Regra de disparo:** quando o ator deve disparar



Definições - atores

- Extensão dos atores mencionados anteriormente
- Com estado (**stateful**)
- Podem ser considerados como objetos com métodos
- Emulados com atores sem estado (**stateless**)
- Usando canal extra

As camadas do Dataflow

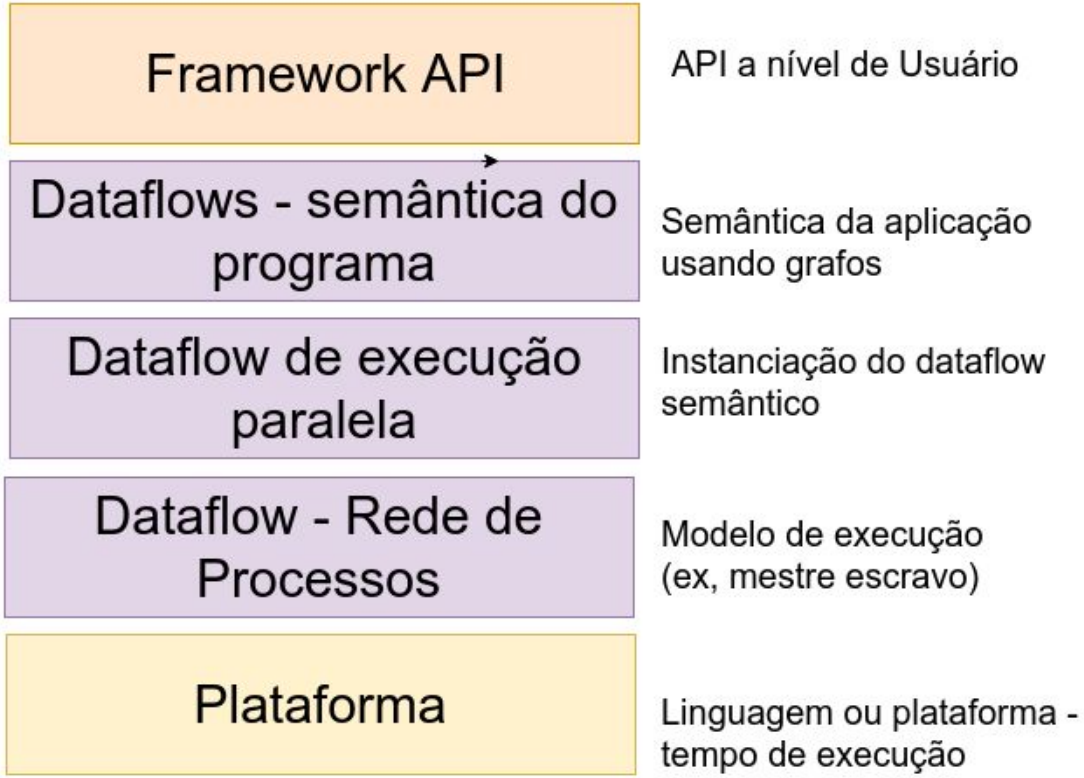
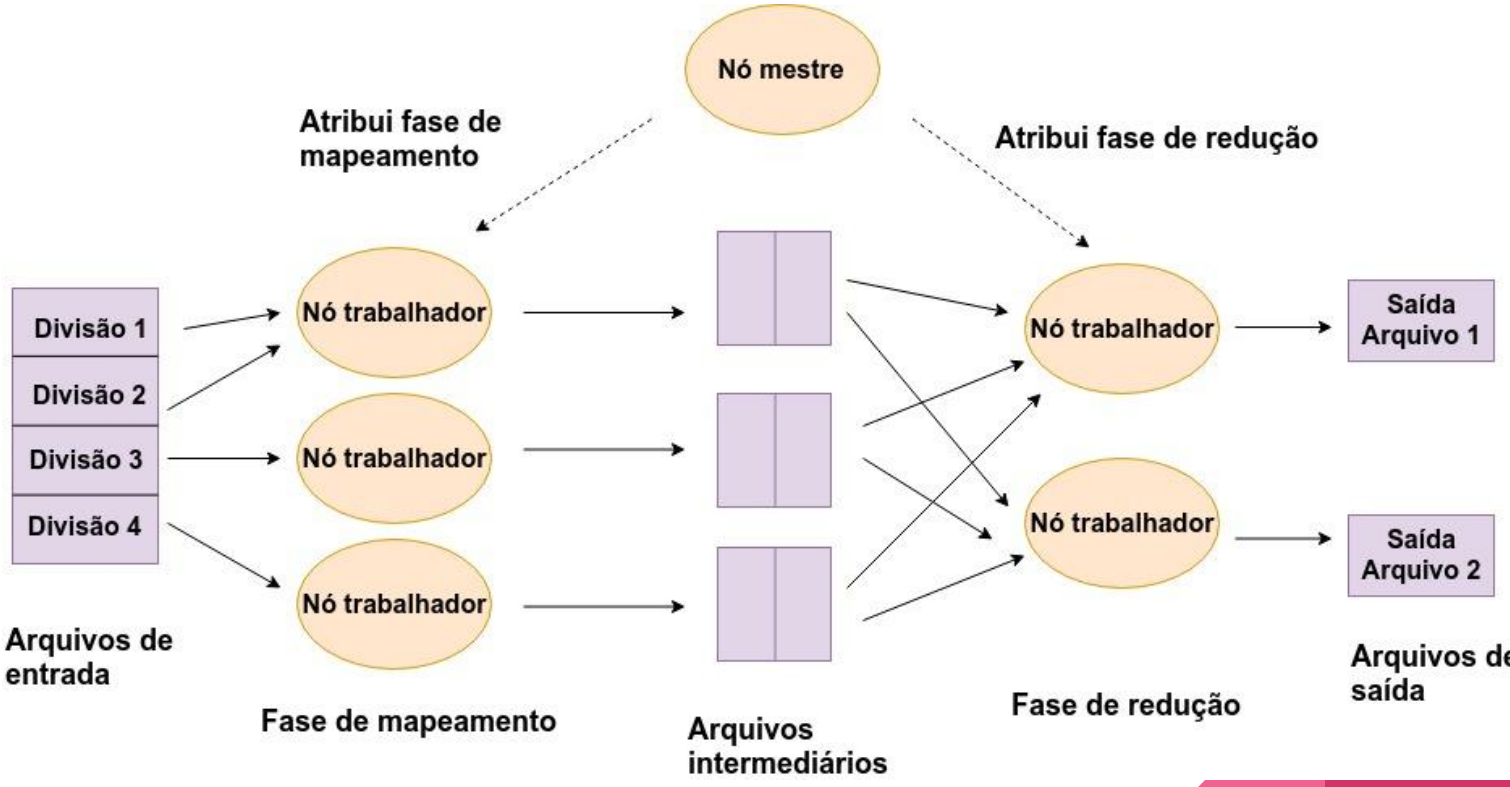


Imagem baseada na figura do artigo [1]

Ferramentas de Big Data

Características/ Ferramentas	Modelo de processamento
Apache Flink	Stream com suporte a batch
Apache Storm	Stream (micro-batch com o Trident)
Apache Spark	Micro lotes (mesmo no Spark Streaming o resultado é dado em batches)

MapReduce


Nó mestre
Atribui fase de mapeamento
Atribui fase de redução
Divisão 1
Divisão 2
Divisão 3
Divisão 4
Nó trabalhador
Nó trabalhador
Nó trabalhador
Nó trabalhador
Nó trabalhador
Saída Arquivo 1
Saída Arquivo 2
Arquivos de entrada
Fase de mapeamento
Arquivos intermediários
Fase de redução
Arquivos de saída

APIs para usuário

- Duas categorias: **declarativa** e **topológica**
- Spark e Flink são declarativas
- Storm é topológico

APIs declarativas

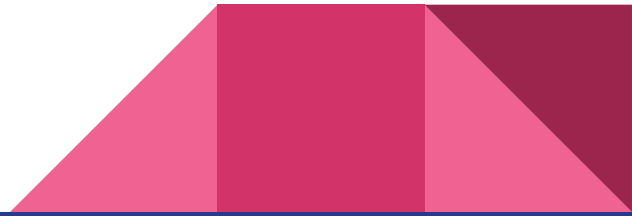
- Coleções de dados e operações
- Processamento declarativo em **lote**: expressos como métodos em objetos representando coleções
- São expressos em termos de uma **álgebra** (lembrem-se de álgebra relacional)
- Os dados podem ser finitos ou não e expressos por tuplas
- Processamento de **stream**: estado e janelas

APIs declarativas

Código do Word Count comparando Flink e Spark

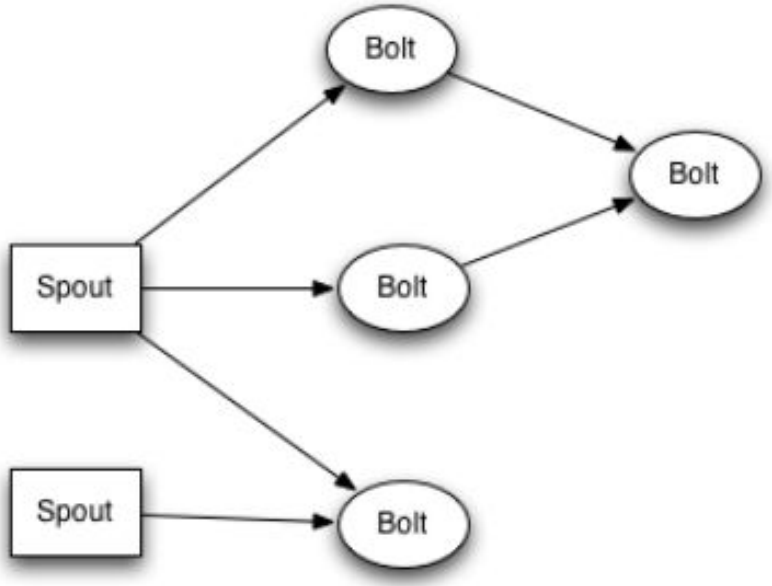
API - Processamento topológico

- Programas expressos por grafos
- Conexão **explícita** entre os nós
- É especificado o código executado pelos nós
- Exemplo: Apache Storm
- Storm utiliza spouts, bolts e topologias.



API - Processamento topológico

- Topologia: grafo de transformações de stream
- Cada nó é um spout ou bolt.

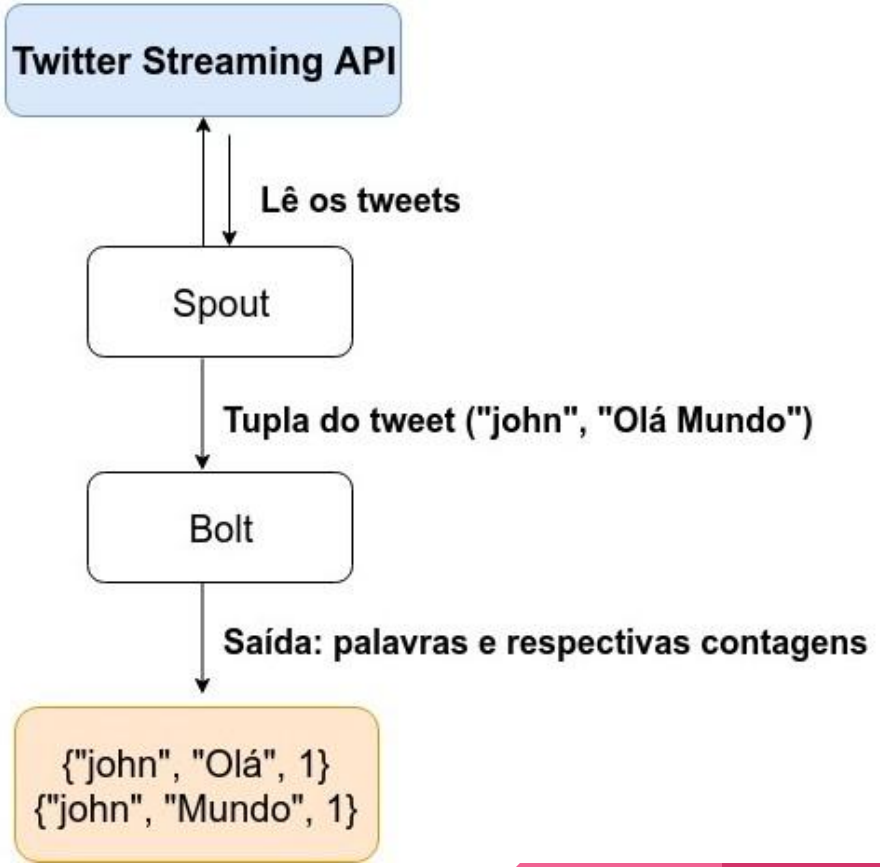


Fonte:

site do Storm

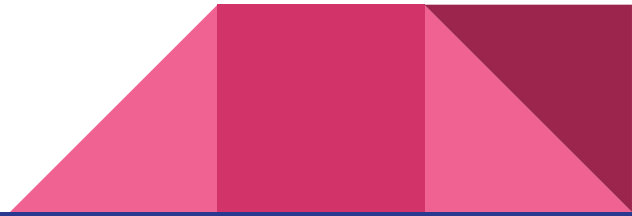
API - Processamento topológico

Pausa para código do Word Count - Apache Storm



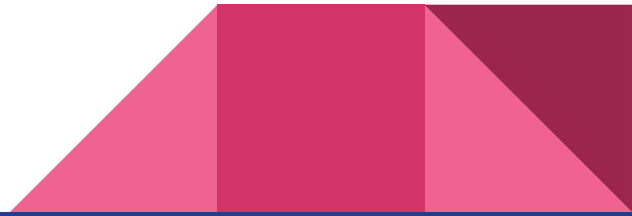
Processamento paralelo

- A forma mais direta: atores independentes podem rodar em paralelo
- Alguns atores podem ser replicados (full data parallelism)
- Para atores dependentes: paralelismo por pipeline



Execução (scheduling-based)

- Spark, Flink e Storm usam padrão mestre-escravo
- **Master:** controle total da execução, comunicação, gerenciamento de falhas, alocação de recursos
- **Workers:** recebem tarefas para executar
  - representam atores do dataflow executado pelo mestre
  - não têm informação sobre o dataflow
  - escalonados pelo mestre



Execução (scheduling-based)

- No ***Storm*** e ***Flink***:
  - distribuição de dados gerenciada de forma descentralizada
  - delegada para cada executor/worker

- No ***Spark Streaming***:
  - o mestre distribui os dados
  - micro-lotes são armazenados no buffer do worker

Referências

[1] MISALE, Claudia et al. A comparison of big data frameworks on a layered dataflow model. Parallel Processing Letters, v. 27, n. 01, p. 1740003, 2017.

[2] http://storm.apache.org/releases/current/Tutorial.html