Parallel applications in the cloud

Diana Naranjo Pomalaya

Computação Paralela e Distribuida

Agenda

- Introduction
- MapReduce
- Solutions
  - Haloop
  - iMapReduce
  - Pig

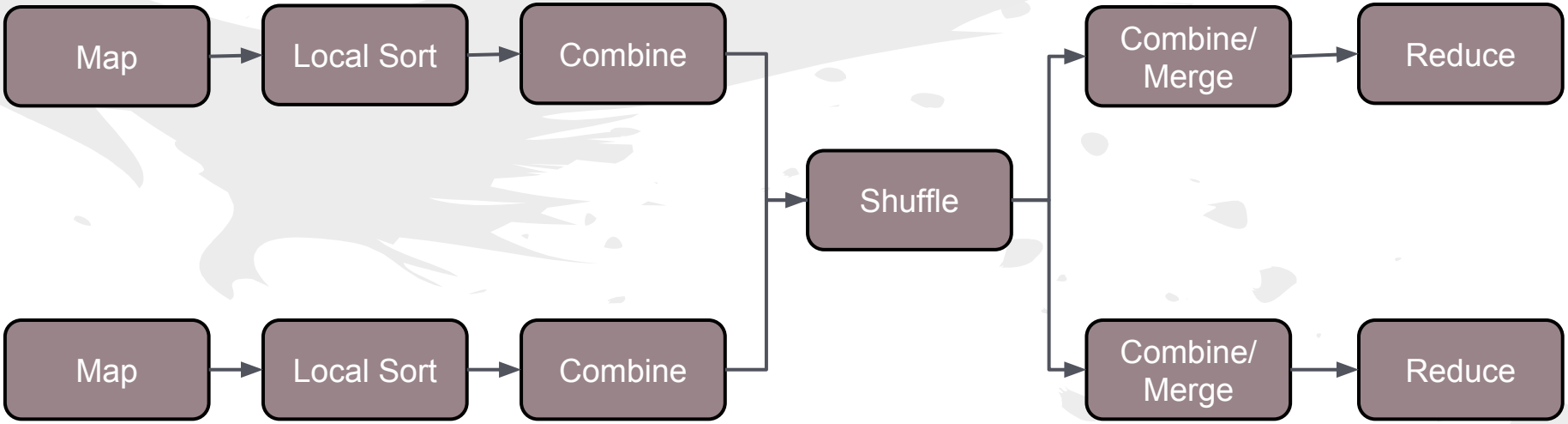
Global Data Center Traffic

Source: Cisco Global Cloud Index, 2013–2018

Data-intensive applications

- Industry
  - Web-data analysis
  - Click-stream analysis
  - Network-monitoring log analysis
- Sciences
  - Massive-scale simulations data analysis
  - Sensor deployments
  - High-throughput lab equipment

MapReduce


Map
Local Sort
Combine
Map
Local Sort
Combine
Shuffle
Combine/
Merge
Reduce
Combine/
Merge
Reduce

MapReduce

- Easy-to-use programming model (2 functions)
- Scalability
- Fault-tolerance
- Load balancing
- Data locality-based optimization
- Designed for Batch-oriented computations (N-step dataflows)
- Low-level abstraction (combined data sets, primitive operations)

Solutions

Haloop

Loop-aware scheduler

Caching mechanisms

iMapReduce

Persistent tasks

Input data loaded once

Asynchronous execution

Pig

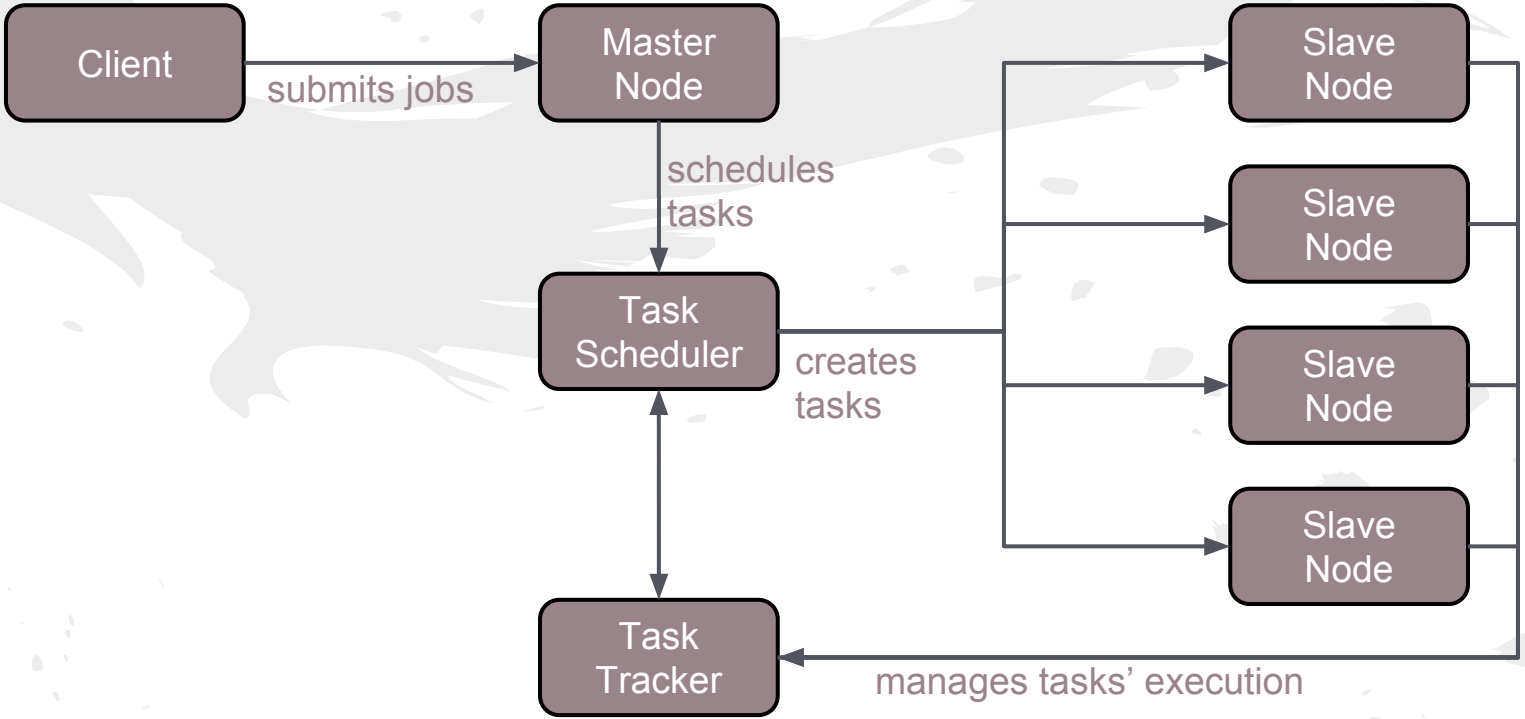
High-level data manipulation

Hadoop execution

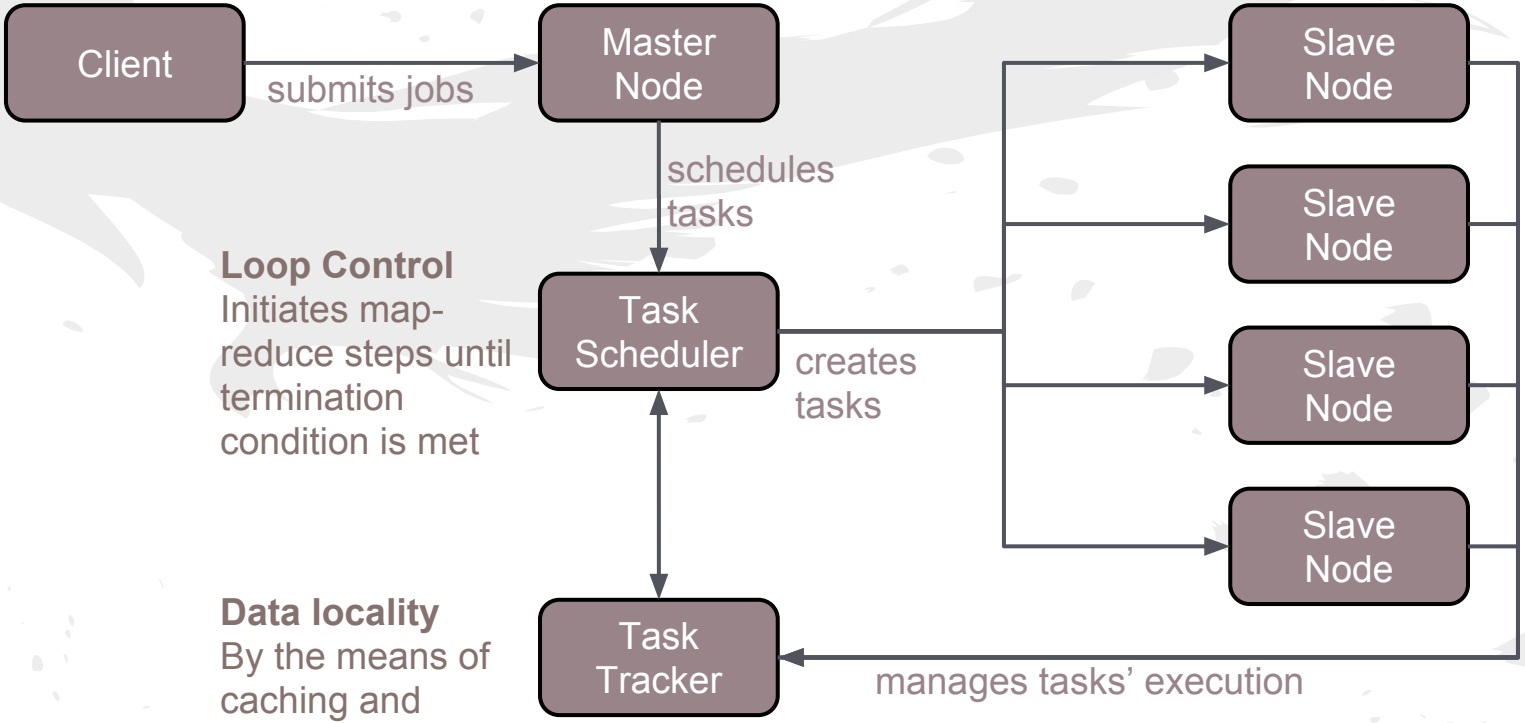
Haloop

- Hadoop based framework
- Supports iterative programs
- Loop-aware scheduler and Caching Mechanisms

Hadoop


Client
submits jobs
Master Node
schedules tasks
Task Scheduler
creates tasks
Slave Node
Slave Node
Slave Node
Slave Node
Task Tracker
manages tasks' execution

Haloop


Client
submits jobs
Master Node
schedules tasks
Loop Control
Initiates map-reduce steps until termination condition is met
Task Scheduler
creates tasks
Slave Node
Slave Node
Slave Node
Slave Node
Data locality
By the means of caching and indexing
Task Tracker
manages tasks' execution

Haloop - Loop control

- Goal: place on same physical machine map/reduce tasks that occur in different iterations but access same data
- How:
  - Keep track of data partitions processed by each task on each physical machine
  - Map new tasks to slave nodes that have alreasy processed that data partition
  - If node full, then re-assign to other node

Haloop - Caching and indexing

- Reducer input cache: useful for repeated joins against large invariant data (wastes less time in shuffling)
- Reducer output cache: reduce cost of fixpoint termination condition avaliation
- Mapper input cache: useful in k-means similar applications (input data does not vary)
- Cache reloading: if node is full, copy all required data to new assigned node

iMapReduce

- Based on Hadoop
- Framework for iterative algorithms
- Concept of persistent tasks, input data loaded to persistent tasks once and facilitates asynchronous execution of tasks within iteration

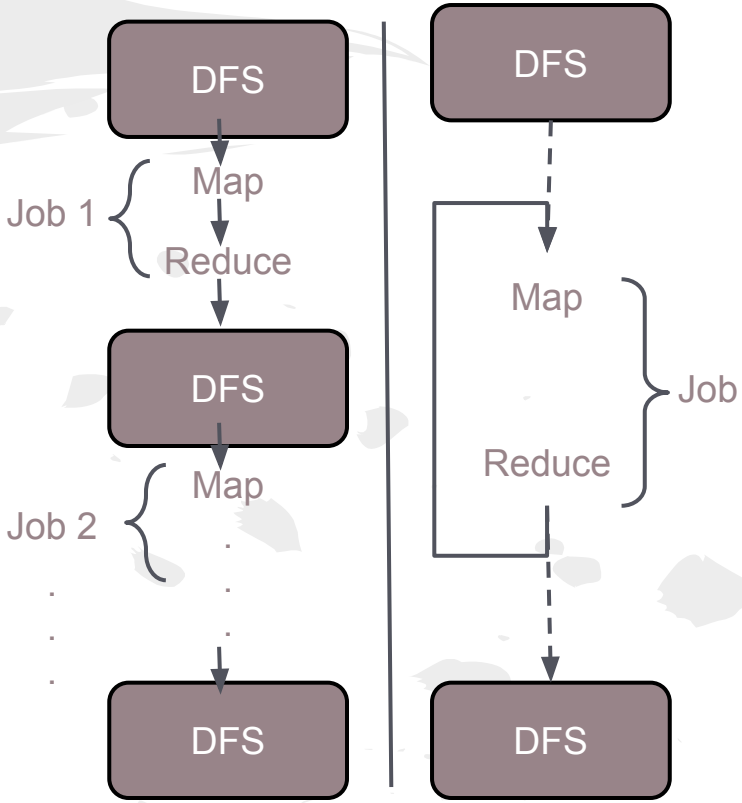
iMapReduce - Restrictions

- Map and reduce operations use same key (one-to-one mapping)
- Each iteration contains only one MapReduce job

} Graph-based iterative algorithms

iMapReduce - Persistent tasks

- Tasks keep alive during whole iteration process (dormat as data is parsed/processed)
- Depends on available task slots (problem with balancing load - strangles/leaders nodes)



iMapReduce - Data management

- Input data becomes: static data (invariant) and state data (variant)
- State data is passed from reduce to map tasks through socket connections
- Static data is partitioned with the same hash function used to shuffle state data
- Map and reduce tasks (one-to-one due to key restriction) are scheduled to same worker

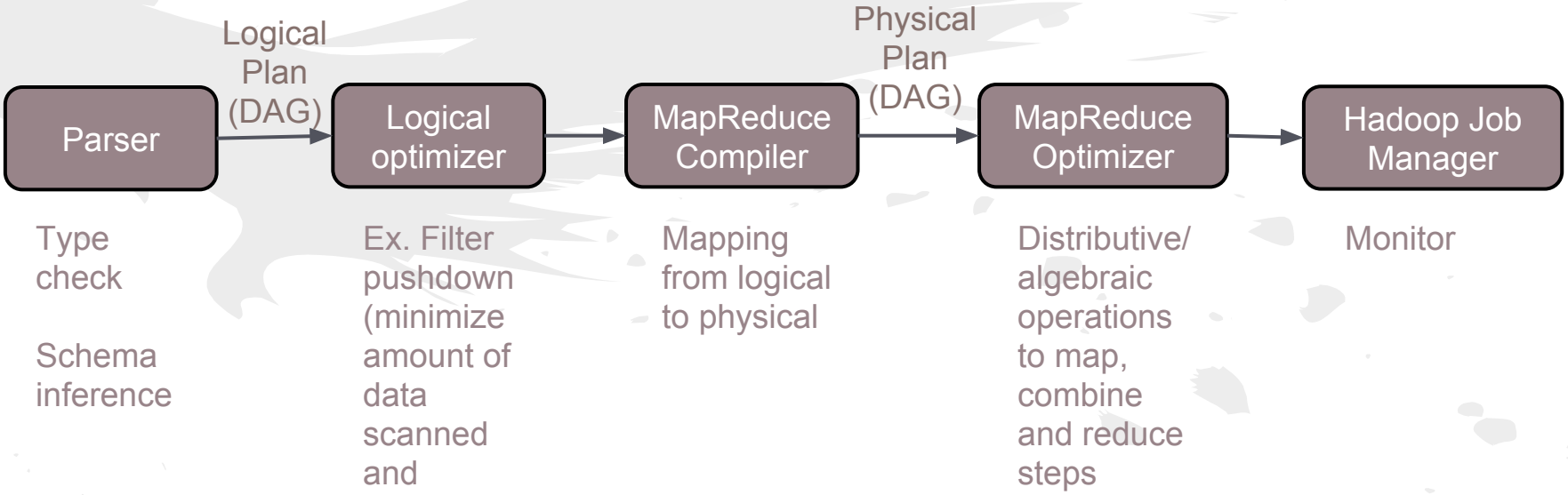
iMapReduce - Asynchronous execution

- Map tasks can start execution as soon as its state data arrives
- No need to wait for other map tasks
- Fault-tolerance problem: use buffer to save results from reduce tasks (return to last iteration)

Pig

- Provides constructs that allow high-level data manipulation
- Allows employment of user-provided executables
- Compiles data-flow programs (pig latin) into sets of MapReduce jobs and coordinates its execution (Hadoop)

Pig - Compilation and execution stages



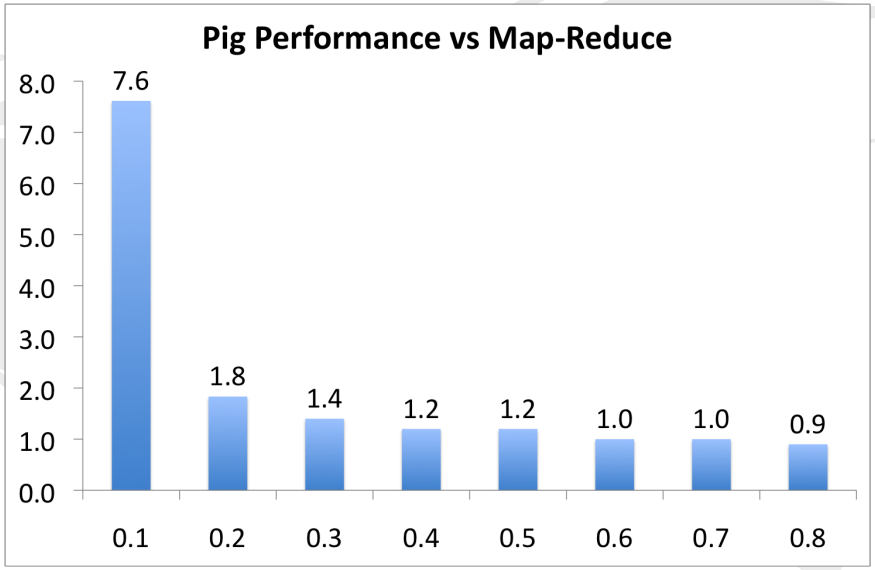
Pig - Memory Management

- Pig is implemented in JAVA
- Memory overflow situations when large bags of tuples are materialized between and inside operators
- Solution: List of bags ordered in descending order (estimated size), spill bags when threshold is reached

iMapReduce - Streaming

- User-defined functions are supported in JAVA and are synchronous
- Streaming executables allow other languages to be used (scripts/compiled binaries)
- Streaming executables are asynchronous (queues)

Pig - Performance



17 December, 2010

6 June, 2015: release 0.15.0 available

Source: India Hadoop Summit - Feb, 2011

PigPen

- MapReduce language that looks and behaves like Clojure.core
- Supports unit tests and iterative development
- Used in Netflix

References

- Cisco and/or its affiliates. Cisco global cloud index: Forecast and methodology 20132018 white paper, 2014.
- Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D. Ernst. Haloop: Efficient iterative data processing on large clusters.
- Yanfeng Zhang, Qixin Gao, Lixin Gao, and Cuirong Wang. C.: Imapreduce: a distributed computing framework for iterative computation. In In: Proceedings 8 of the 1st International Workshop on Data Intensive Computing in the Clouds (DataCloud, page 1112, 2011.
- Thilina Gunarathne, Bingjing Zhang, Tak lon Wu, and Judy Qiu. Portable parallel programming on cloud and hpc: Scientific applications of twister4azure," presented at the portable. In Parallel Programming on Cloud and HPC: Scientific Applications of Twister4Azure, 2011.
- Jaliya Ekanayake, Xiaohong Qiu, Thilina Gunarathne, Scott Beason, and Geoffrey Fox. High performance parallel computing with cloud and cloud technologies.

Questions?