Стек технологий Apache Hadoop. Распределённая файловая система HDFS

Сергей Рябов

Цели

- Осветить наиболее значимые технологии стека Apache Hadoop для распределённой обработки данных:
  - **MapReduce**
  - **HDFS**
  - **Hbase**
  - **ZooKeeper**
  - **Pig**
  - **Hive**
  - **Avro**
- Рассмотреть архитектуру распределённой файловой системы HDFS

Архитектурные принципы

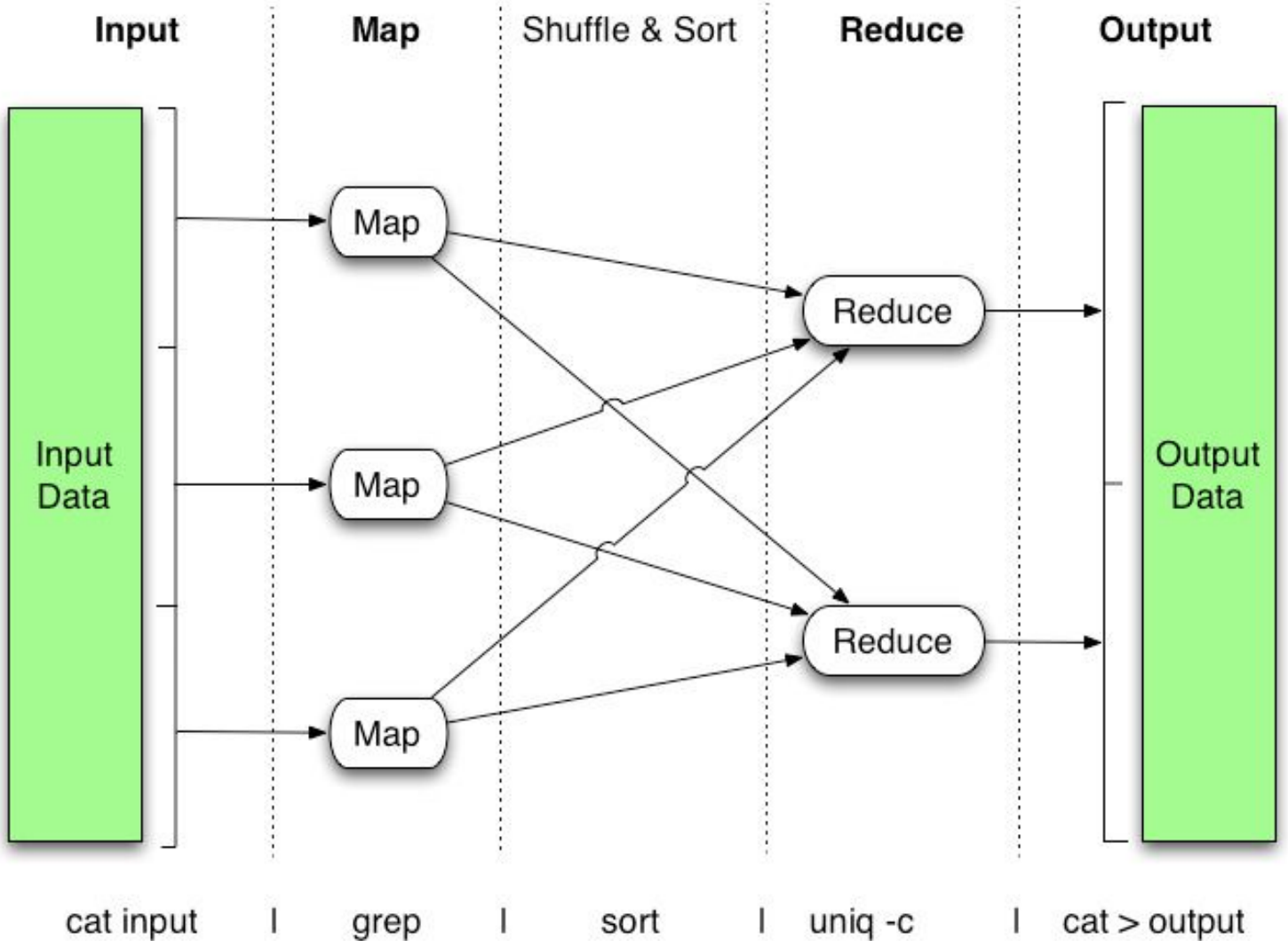
- Линейная масштабируемость
- Надёжность и доступность
- Ненадёжное (commodity) оборудование
- Перемещение данных дороже перемещения программ
- Высокая производительность

MapReduce

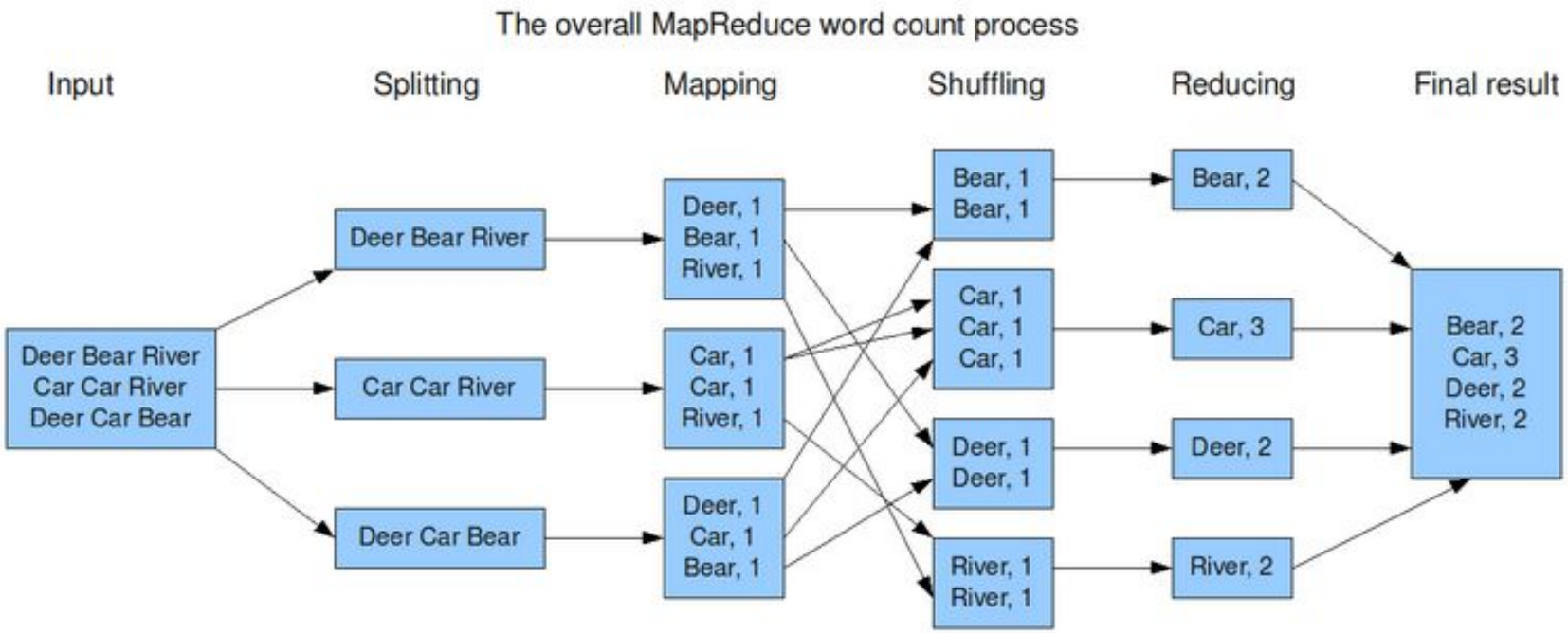
- Фреймворк для распределённых вычислений
- MapReduce job – 2 этапа
  - Map: {<inpK,inpV>} -> {<intK,intV>}
  - Reduce: {<intK,intV>} -> {<outK,outV>}
- Map – предварительная обработка

  Reduce – агрегация
- Shuffle – сортировка и слияние, невидимый для пользователя переход от Map к Reduce

MapReduce


Input
Map
Shuffle & Sort
Reduce
Output
Input Data
Map
Map
Map
Reduce
Reduce
Output Data
cat input
|
grep
|
sort
|
uniq -c
|
cat > output

MapReduce


The overall MapReduce word count process
Input
Splitting
Mapping
Shuffling
Reducing
Final result
Deer Bear River
Car Car River
Deer Car Bear
Deer Bear River
Car Car River
Deer Car Bear
Deer, 1
Bear, 1
River, 1
Car, 1
Car, 1
River, 1
Deer, 1
Car, 1
Bear, 1
Bear, 1
Bear, 1
Car, 1
Car, 1
Car, 1
Deer, 1
Deer, 1
River, 1
River, 1
Bear, 2
Car, 3
Deer, 2
River, 2
Bear, 2
Car, 3
Deer, 2
River, 2

HDFS

- Иерархия каталогов и файлов
- Файлы поделены на блоки (128 MB)
- Метаданные отделены от данных
- NameNode хранит все метаданные в ОП
- DataNode хранит реплики блоков в виде файлов на диске
- Блоки дублируются на 3 DataNode

HBase

- Распределённое ключ-значение хранилище на базе HDFS
- Таблицы:
  - Строки с уникальными ключами
  - Произвольное количество колонок
  - Колонки сгруппированы в группы колонок
- Таблицы разбиты на «регионы»
  - Горизонтально по строкам
  - Вертикально по группам колонок

ZooKeeper

- Распределённая служба координации распределённых задач
  - Выборы лидера
  - Распределённые блокировки
  - Координация и уведомления о событиях

Pig

- Платформа для анализа больших наборов данных
- Pig Latin – SQL-подобный язык
  - Простота кодирования
  - Возможности оптимизации
  - Расширяемость
- Pig-программы преобразуются в набор MapReduce заданий (jobs)

Hive

- Служит тем же целям, что и Pig
- Таблицы
  - Типизированные колонки (int, float, string, date, boolean)
  - Поддержка списков и отображений
  - Реально данные хранятся в плоских файлах
- Хранит метаданные о Hive-таблицах в RDB
  - Схемы таблиц
  - Расположение в HDFS

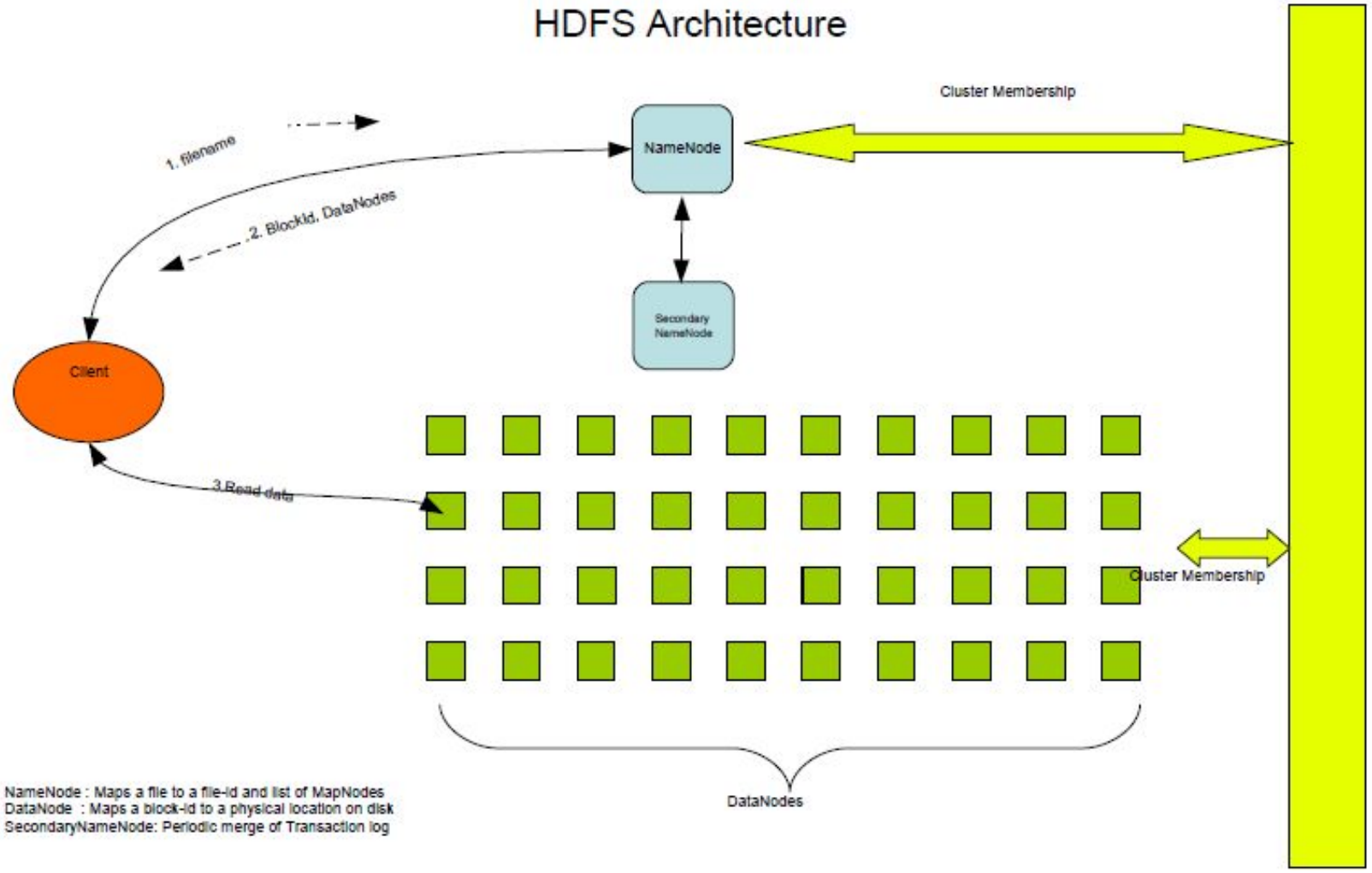
Avro

- Система сериализации данных
- Предоставляет:
  - Компактный бинарный формат
  - Удалённые вызовы процедур (RPC)
  - Простая интеграция с динамическими языками
- Чтение/запись с использованием схем

HDFS. Поставленные цели

- Очень большой объём распределённых данных
  - 10K узлов, 100M файлов, 10ПБ данных
- Ненадёжное (commodity) оборудование
  - Репликация данных
  - Обнаружение и восстановление после сбоев
- Оптимизация для пакетной обработки
  - Вычисление перемещается к данным
  - Большая совокупная пропускная способность

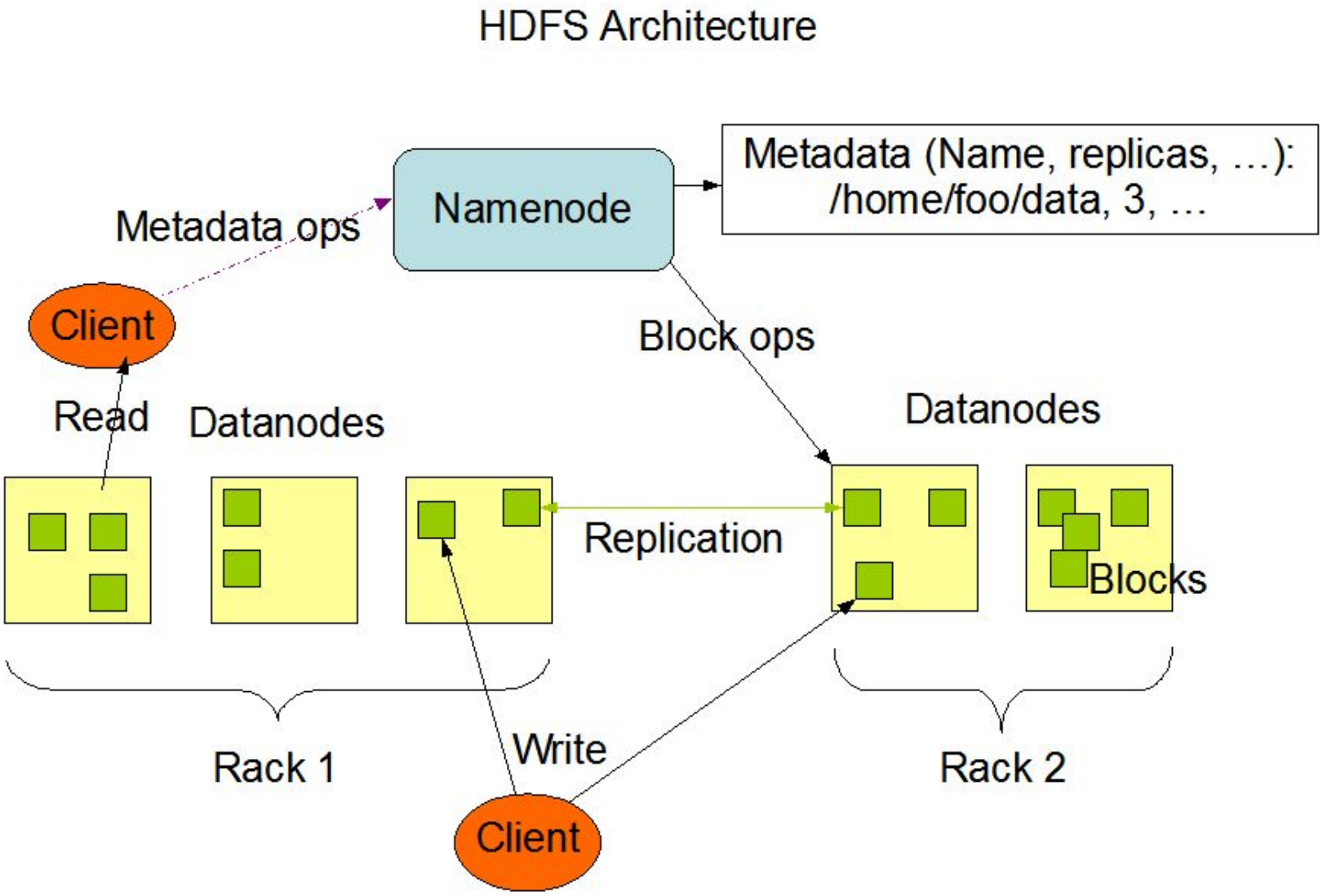
HDFS. Архитектура



HDFS. Архитектура

- Общее пространство имён для всего кластера
- Согласованность данных
  - Write-once-read-many модель доступа
  - Append-запись всё ещё нестабильна
- Файлы разбиваются на блоки
  - Обычно по 128МБ
  - Каждый блок дублируется на несколько узлов
- «Умный» клиент
  - Может узнать местоположение блоков
  - Доступ к данным непосредственно через DataNode

HDFS. Архитектура



HDFS. NameNode

- Управляет пространством имён
  - Связывает имя файла с набором блоков
  - Связывает блок с набором DN
- Контролирует процессы репликации
- Единственная точка отказа
- Лог транзакций (journal) хранится в нескольких местах
  - Локальный каталог
  - Каталог в удалённой ФС (NFS/CIFS)

HDFS. NameNode. Метаданные

- Метаданные для всего кластера хранятся в ОП
- Типы метаданных
  - Списки файлов
  - Списки блоков для каждого файла
  - Списки DN для каждого блока
  - Атрибуты файлов (время создания, количество реплик и т.д.)

HDFS. DataNode

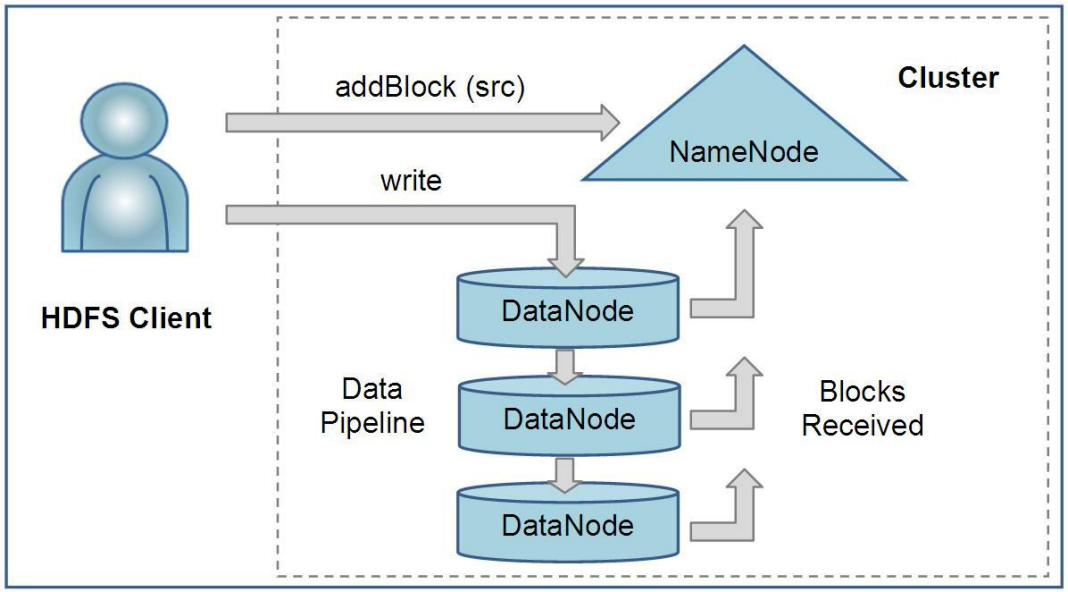
- Сервер блоков
  - Хранит данные в локальной ФС
  - Хранит метаданные блоков (CRC)
  - Предоставляет данные и метаданные клиентам
- Периодически (3 секунды) посылает статусное сообщение (heartbeat) NN
  - Список всех существующих блоков
  - Объём занятого/свободного места
  - Количество активных обменов данными
- Конвейерная работа с данными
  - Передача данных заданным DN

HDFS. CheckpointNode

- Периодически создаёт новый checkpoint образ из checkpoint и journal, загруженных с NN
- Загружает новый checkpoint на NN. Существующий journal урезается

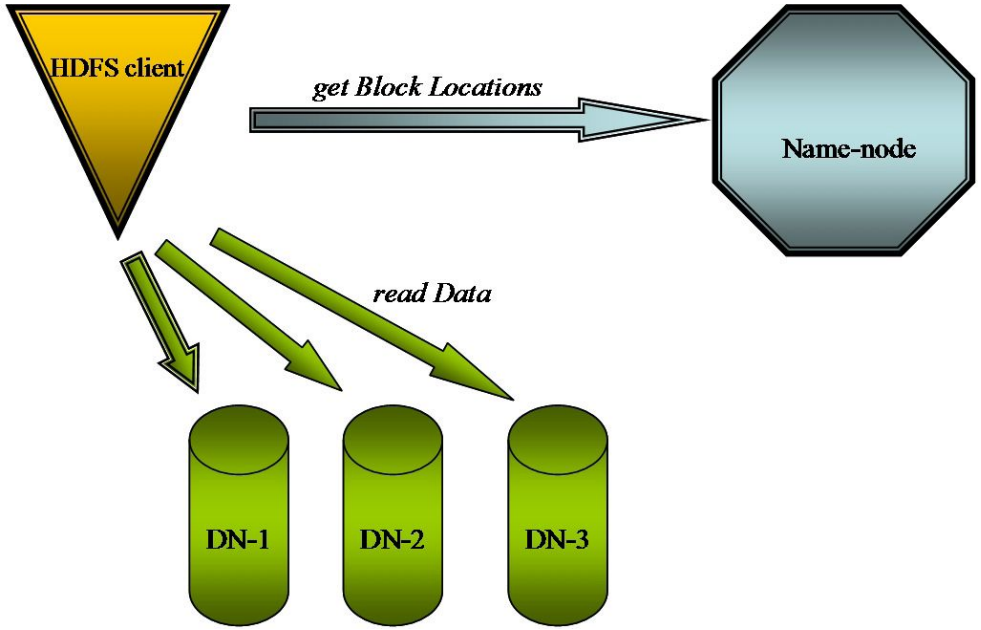
HDFS. Запись

- Клиент запрашивает у NN список DN-кандидатов на запись
- Начинает конвейерную запись с ближайшего узла



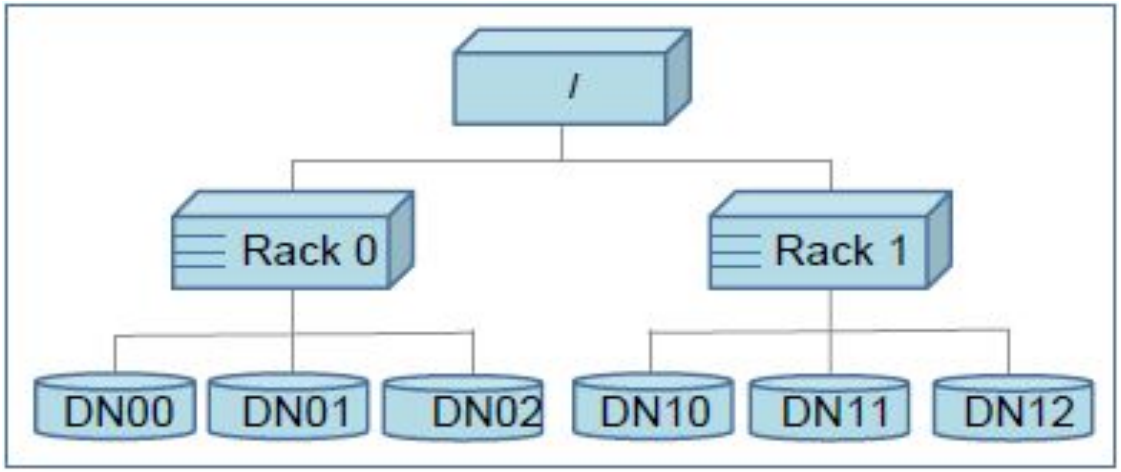
HDFS. Чтение

- Клиент запрашивает местоположение реплик блока у NN
- Начинает чтение с ближайшего узла, содержащего реплику блока



HDFS. Расположение реплик

- Первая реплика помещается на локальном узле
- Вторая реплика – на узел удалённой стойки
- Третья – на другой узёл той же удалённой стойки
- Остальные размещаются случайно
  - DN содержит не более одной реплики блока
  - Стойка содержит не более двух реплик блока



HDFS. Balancer

- Процент используемого дискового пространства на всех DN должен быть одинаков
  - Обычно запускается при добавлении новой DN
  - Не мешает основной работе HDFS
  - При сильной загрузке сети трафик урезается до минимума (1 Мбит/с)

HDFS. Block Scanner

- Каждая DN периодически запускает BS
- BS проверяет, что контрольные суммы соответствуют блокам данных
- Если BS находит повреждённый блок, он оповещает об этом NN
- NN помечает реплику как испорченную и начинает процесс репликации для блока
- По окончании повреждённая реплика готова к удалению

HDFS. Интерфейс пользователя

- Команды пользователя HDFS
  - `hadoop fs –mkdir /foodir`
  - `hadoop fs –cat /foodir/barfile.txt`
  - `hadoop fs –ls /foodir`
- Команды администратора HDFS
  - `hadoop dfsadmin -report`
  - `hadoop dfsadmin –safemode enter`
- Веб-интерфейс
  - http://namenode:port/dfshealth.jsp

HDFS. Веб-интерфейс

NameNode '.local:8020'

Started:	Wed Dec 15 03:22:57 EST 2010
Version:	0.20.2+320, r9b72d268a0b590b4fd7d13aca17c1c453f8bc957
Compiled:	Mon Jun 28 19:13:09 EDT 2010 by root
Upgrades:	There are no upgrades in progress.

Browse the filesystem
Namenode Logs

Cluster Summary

367 files and directories, 211 blocks = 578 total. Heap Size is 47.38 MB / 888.94 MB (5%)

Configured Capacity	:	287.17 GB
DFS Used	:	7.35 GB
Non DFS Used	:	122.27 GB
DFS Remaining	:	157.55 GB
DFS Used%	:	2.56 %
DFS Remaining%	:	54.86 %
Live Nodes	:	4
Dead Nodes	:	0

NameNode Storage:

Storage Directory	Type	State
/var/lib/hadoop-0.20/cache/hadoop/dfs/name	IMAGE_AND_EDITS	Active

Cloudera's Distribution for Hadoop, 2010.

HDFS. Использование в Yahoo!

- 3500 узлов
  - 2 процессора Xeon@2.5GHz (по 4 ядра)
  - Red Hat Enterprise Linux Server Release 5.1
  - Sun Java JDK 1.6.0_13-b03
  - 4 SATA диска (1 TB каждый)
  - 16GB RAM
  - 1-gigabit Ethernet
- NamaNode с 64 GB RAM
- 3.3 PB данных (9.8 PB с репликами)
- 1-2 узла выходят из строя каждый день

HDFS. Benchmarks

NameNode benchmark. Несколько локальных клиентских потоков выполняют одну и ту же операцию.

Operation	Throughput (ops/s)
Open file for read	126 100
Create file	5600
Rename file	8300
Delete file	20 700
DataNode Heartbeat	300 000
Blocks report (blocks/s)	639 700

Bytes (TB)	Nodes	Maps	Reduces	Time	HDFS I/0 Bytes/s	
					Aggregate (GB)	Per Node (MB)
1	1460	8000	2700	62 s	32	22.1
1000	3658	80 000	20 000	58 500 s	34.2	9.35

Gray Sort benchmark. Сортировка 1 ТБ и 1 ПБ данных. Записи по 100 байт. При сортировке ТБ количество реплик было сокращено до одной, при сортировке ПБ - до двух.

Спасибо за внимание

Вопросы?