Платформа Apache Hadoop

Примеры использования

Summary

- Актуальность обработки большого объема данных
- Distributed File System, MapReduce
- Apache Hadoop
- Смежные технологии: Pig, Apache Hive
- Column Oriented Database

Объемы данных

- Facebook: 20Tb сжатых данных в день
- Нью-Йоркская биржа: 1Tb в день
- Большой андронный коллайдер: 40Tb в день
- ContextWeb (online advertising): 115Gb в день

DFS/MapReduce

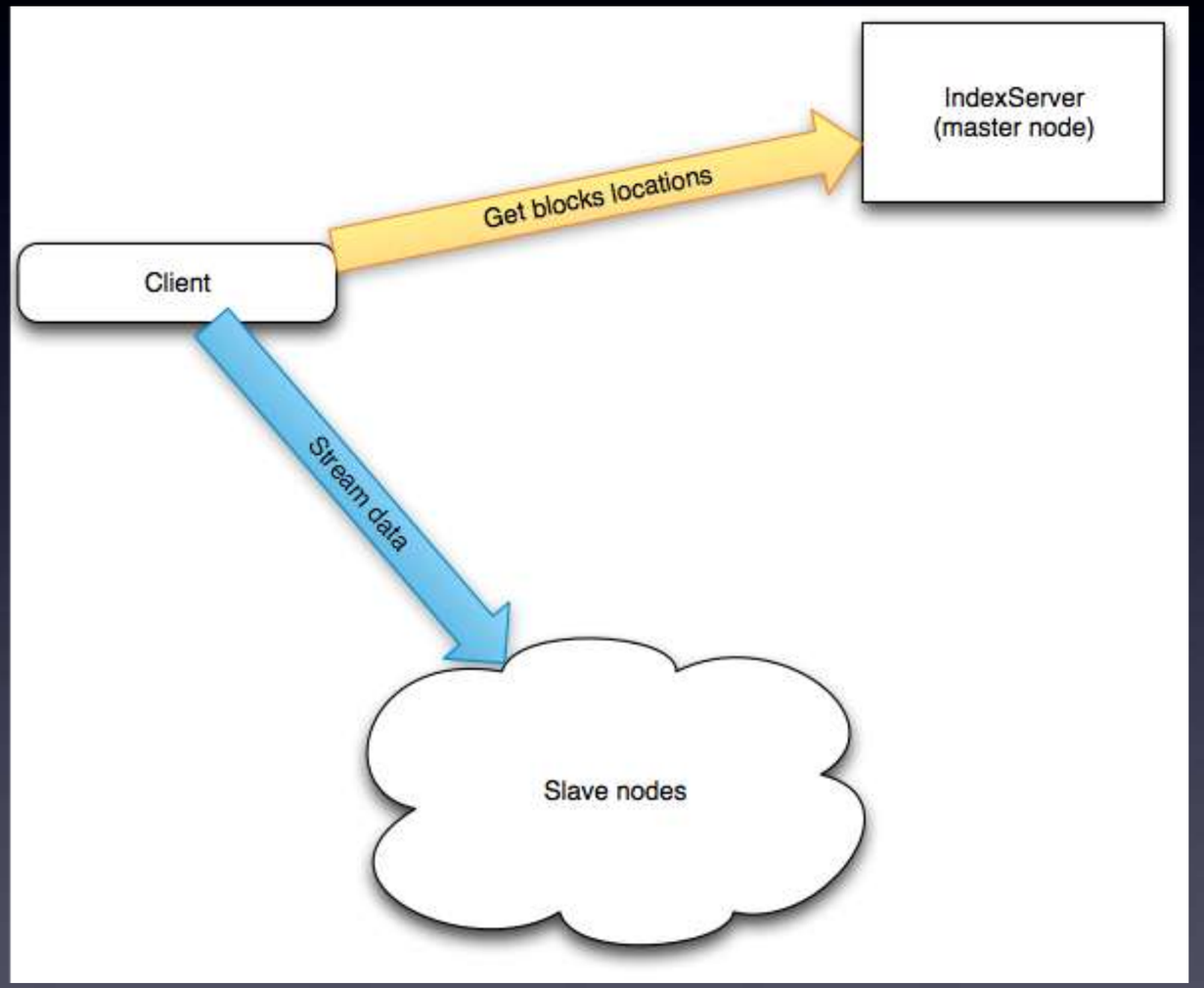
- Проблема 1: где хранить данные?
- Проблема 2: как обрабатывать?
- Октябрь 2003: появление Google File System
- Декабрь 2004: появление MapReduce

Distributed FS

- Требования к распределенной FS
- Хранить файлы любого размера
- Мягкое масштабирование
- Надежность

DFS: архитектура

- Данные разбиываются на блоки (64mb)
- Чтение идет напрямую из slave nodes
- Репликация



Конфигурация

- 40 nodes
- 4Tb/8Gb RAM/4x Xeon per node
- 40*4/2=80Tb общий объем хранилища

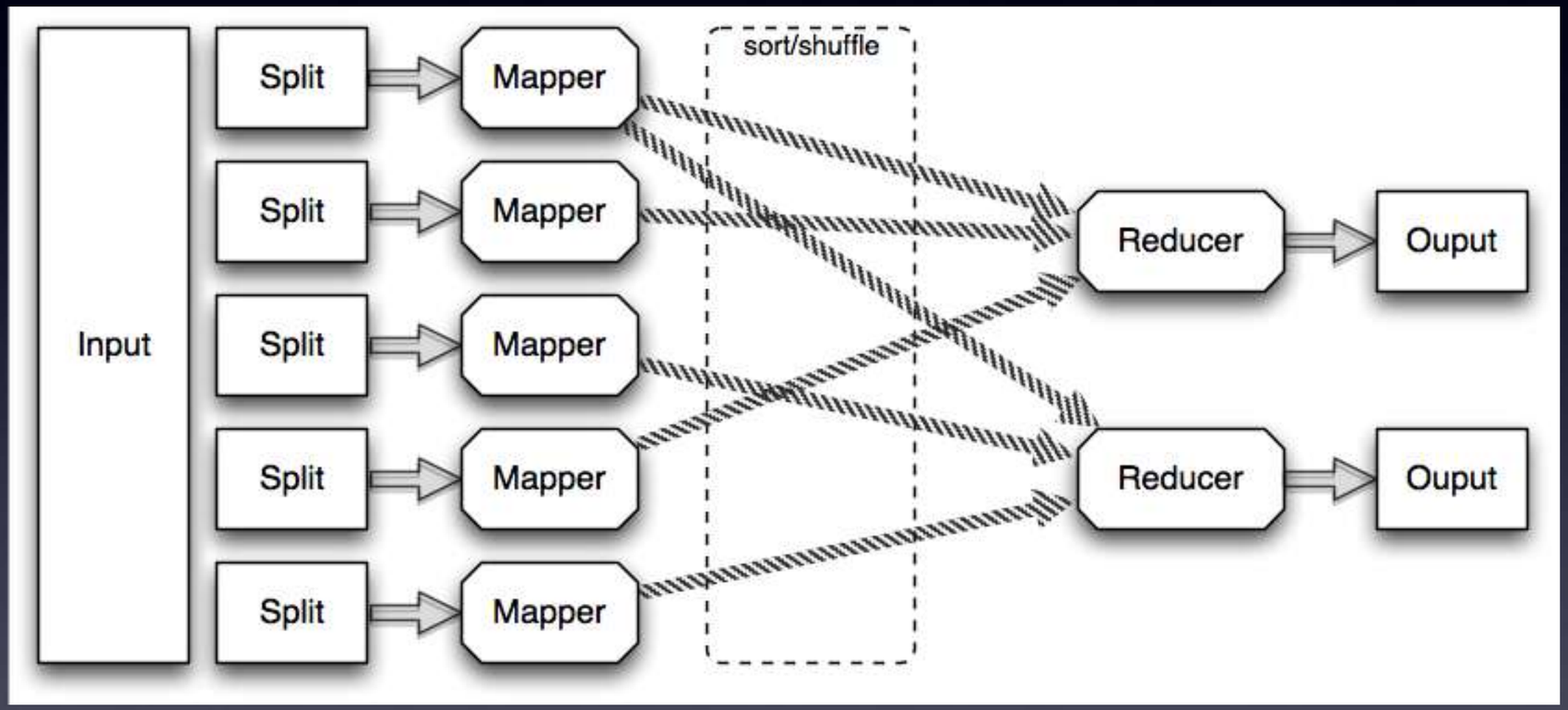
MapReduce

- Map: input record ⇒ (key, value)
- Reduce: (key, {v1, ..., vn}) ⇒ output record
- Данная парадигма применима к широкому спектру задач

Пример

- Как посчитать статистику по браузерам (Facebook)
- Map: log record ⇒ (Browser, 1)
- Reduce: (Browser, [1, .., 1]) ⇒ {Browser, sum}

Параллельность



Apache Hadoop

- 2004: Nutch - open source search engine
- 2006: Hadoop - отдельный проект
- 2006: Yahoo - research cluster
- 2008: Yahoo WebSearch использует Hadoop. Размер кластера - 4000 машин
- 2009 Hadoop выигрывает соревнование по сортировке 100Tb (на кластере Yahoo). 4000 машин, 173 секунды

Модули Hadoop

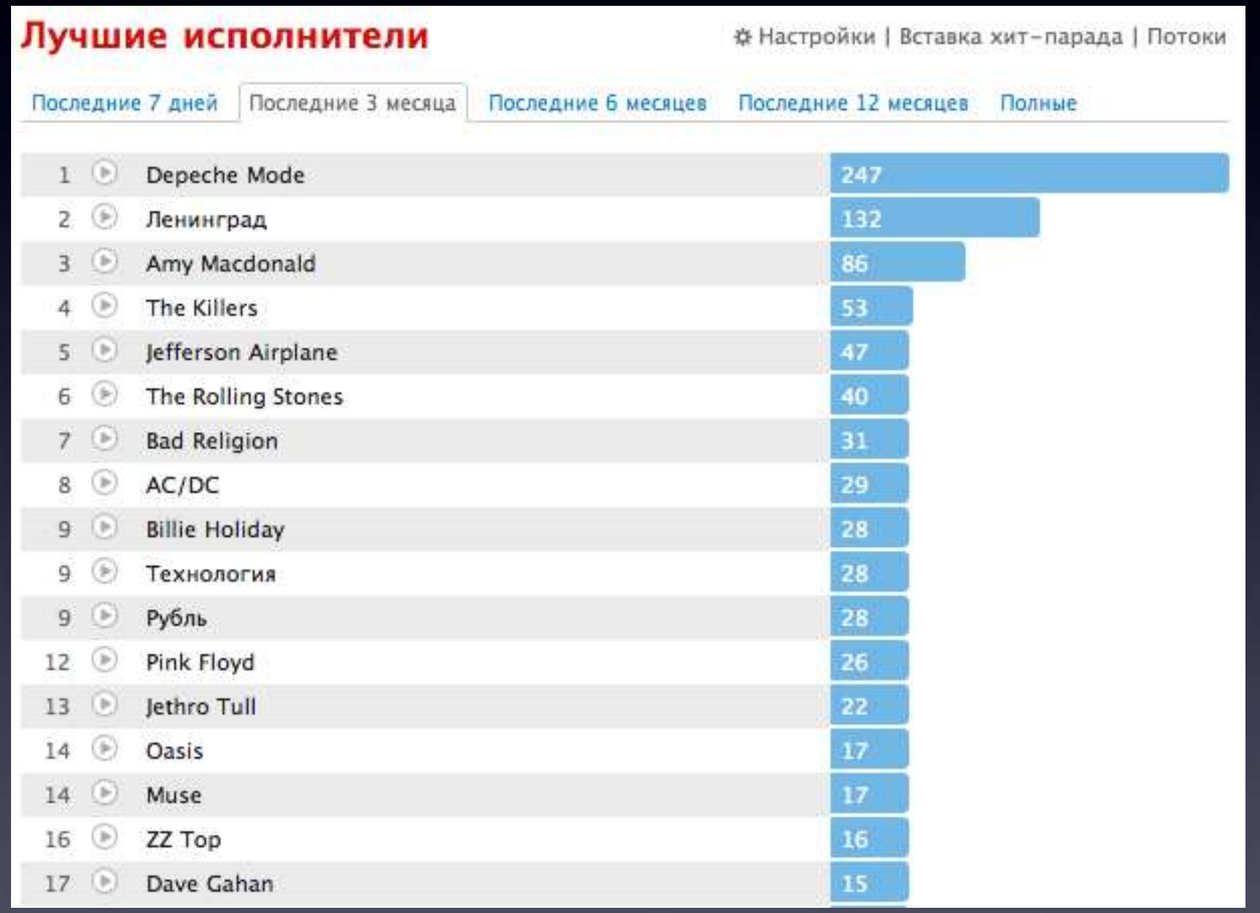
- HDFS: Hadoop distributed file system
- MapReduce

Yahoo: web graph

- Map: page ⇒ (Target URL, {sourceURL, link text})
- Reduce: {Target URL, SourceURL, Link Text}

Target URL	SourceURL	Text
hadoop.apache.org	reddit.com	MapReduce OpenSource
hadoop.apache.org	wikipedia.org/MapReduce	MapReduce OpenSource
hadoop.apache.org	sun.java.com	MapReduce

Last.fm


Лучшие исполнители
Настройки | Вставка хит-парада | Потоки
Последние 7 дней
Последние 3 месяца
Последние 6 месяцев
Последние 12 месяцев
Полные
1 Depeche Mode 247
2 Ленинград 132
3 Amy Macdonald 86
4 The Killers 53
5 Jefferson Airplane 47
6 The Rolling Stones 40
7 Bad Religion 31
8 AC/DC 29
9 Billie Holiday 28
9 Технология 28
9 Рубль 28
12 Pink Floyd 26
13 Jethro Tull 22
14 Oasis 17
14 Muse 17
16 ZZ Top 16
17 Dave Gahan 15

Last.fm

- Пользователь слушает песню
- Информация о прослушивании записывается в HDFS: {user, band, track} (строчка в log-файле)
- Map: {userId, band, track} ⇒ (user_band, 1)
- Reduce (user_band, [1, ... , 1]) ⇒ (user_band, sum)

SQL

- SELECT f1, f2, sum(a) WHERE expr GROUP BY f1, f2
- Map: line ⇒ ({f1, f2}, a) if expr
- Reduce: ({f1, f2}, [a1, ..., an]) ⇒ ({f1, f2}, sum)

SQL: Принцип

- GROUP BY: как ключ в Map
- WHERE: вычисляется в фазе Map
- SUM/AVG как значение в Map
- SUM/AVG: окончательное значение вычисляется в Reduce
- JOIN: Reduce или Map
- HAVING: как фильтрация в окончательной фазе Reduce

SQL: partitioning

- WHERE: для ключевых полей имеет смысл делать partitioning
- В случае анализа исторических partitioning обычно делается по дате
- Часть условия WHERE, имеющее отношение к дате вычисляется до запуска и ограничивает объем входных данных

Apache Hive

- Фреймворк на базе Apache Hadoop
- Транслирует SQL запросы в MapReduce jobs
- Используется как основной R&D инструмент в Facebook

Apache Pig

- Researchers не привыкли писать на Java
- Аналог данного скрипта да Java: 3 класса, 200 строк

top_5.pig

```
users = load 'users.csv' as (username: chararray, age: int);
users_1825 = filter users by age >= 18 and age <= 25;

pages = load 'pages.csv' as (username: chararray, url: chararray);

joined = join users_1825 by username, pages by username;
grouped = group joined by url;
summed = foreach grouped generate group as url, COUNT(joined) AS views;
sorted = order summed by views desc;
top_5 = limit sorted 5;

store top_5 into 'top_5_sites.csv';
```

Области применения

- Research: как фронтенд для людей, занимающихся исследованием данных
- Data mining: построение моделей для дальнейшего использования Real Time
- Reporting: построение отчетов

Достоинства

- Гладкая масштабируемость: для 2x произвоительности досточно 2x оборудования (почти)
- Нулевая стоимость software
- Доступность on-demand как Amazon Cloud Service — удобно для research задач

Недостатки

- Высокая стоимость поддержки и администрирования
- В отличие от SQL, необходим штат квалифицированных Java-developer’ов
- Нестабильность
- Низкая скорость,
- Не real-time

Real-Time?

- Окончательный результат можно загружать в SQL/MemCache
- Однако, SQL/MemCache не будет работать если объем данных, к которому необходим Real-Time доступ остается большим
- Другое решение: column oriented database (HBase)

Column oriented databases

- В SQL-подходе хранения данных есть определенные проблемы
- Данные должны быть хорошо структурированы, ALTER TABLE - “дорогая” операция
- Структурированность данных в многих случаях является плюсом. Но, когда она не нужна, можно хранить данные более эффективно

BigTable

- Дизайн представлен компанией Google в 2004-ом году
- Принцип 1: на всю таблицу есть одно индексное поле называемое row key (аналог primary key)
- Принцип 2: данные во всех остальных полях не индексируются. Таблица может иметь сколько угодно полей, добавление нового поля — затрагивает только отдельные row.

BigTable

- Удобнее представлять хранилище не как таблицу
- А как соответствие: (row key, column name) ⇒ value
- Так же во многих реализациях данные имеют версионность по времени
- (row key, column name, timestamp) ⇒ value

BigTable: пример

- Задача: хранить информацию о посетителях сайта
- Простое решение: Cookie
- Недостаток: размер Cookie ограничен
- BigTable: (UserUID, поле) ⇒ значение
- В Cookie хранится только UserUID
- Возможные поля: дата последнего визита, история посещений, история показа рекламных объявлений. Новое поле добавить очень легко

BigTable: дизайн

- Row keys сортируются, данные храняться на кластере
- Каждый сервер (region server) хранит определенный диапазон ключей
- Клиент обращается к master node и определяет на каком сервере лежат интересующие его данные
- Чтение идет напрямую с region server

HBase

- Построен на платформе Apache Hadoop
- Для хранения данных используется HDFS
- Map Reduce процессы могут быть использованы для загрузки большого объма данных
- На этапе Reduce выполняется загрузка данных в таблицу
- Reduce процесс выполняется на соответствующем region server — происходит исключительно локальная запись данный

HBase: производительность

- 7 server cluster (16Gb RAM, 8x core CPU, 10K RPM HD)
- Таблица из 3 миллиардов rows, от 1 до 5 колонок
- Размер каждого row — около 300 байт
- 300 параллельных запросов
- Средние: 18ms — чтение, 8ms — запись

HBase: недостатки

- Около 1% процента запросов работают сильно больше среднего (порядка 300ms)
- Возможность индексировать только по одному полю (row key)
- Нестабильность: в последней самой производительной версии возможна потеря данных

Hadoop: области использования

- MapReduce — там, где некритична скорость получения результата: обработка лог-файлов, построение стаитстических моделей, построение индексов, research
- HBase — там, где некритична небольшая потеря данных и не обязательно гарантированое время ответа (например, хранение информации о пользователе в online advertising)

Где не стоит использовать Hadoop

- Точные вычисления
- Биллинг
- Трейдинг
- Банковские операции

Спасибо за внимание!