ТЕХНОСФЕРА

Методы распределенной обработки больших объемов данных в Hadoop

Лекция 2: Hadoop, основы

Москва 2014

BigData

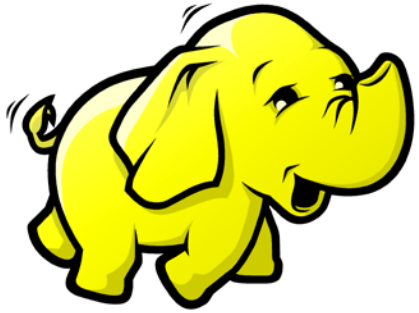
- Information Data Corporation (IDC) оценивает общий размер данных за 2010г как **1.2 Зеттабайт** (1.2 Триллиона Гигабайт)
- Компании продолжают генерить новые данные и рост их объема оценивается экспоненциальн

Hadoop

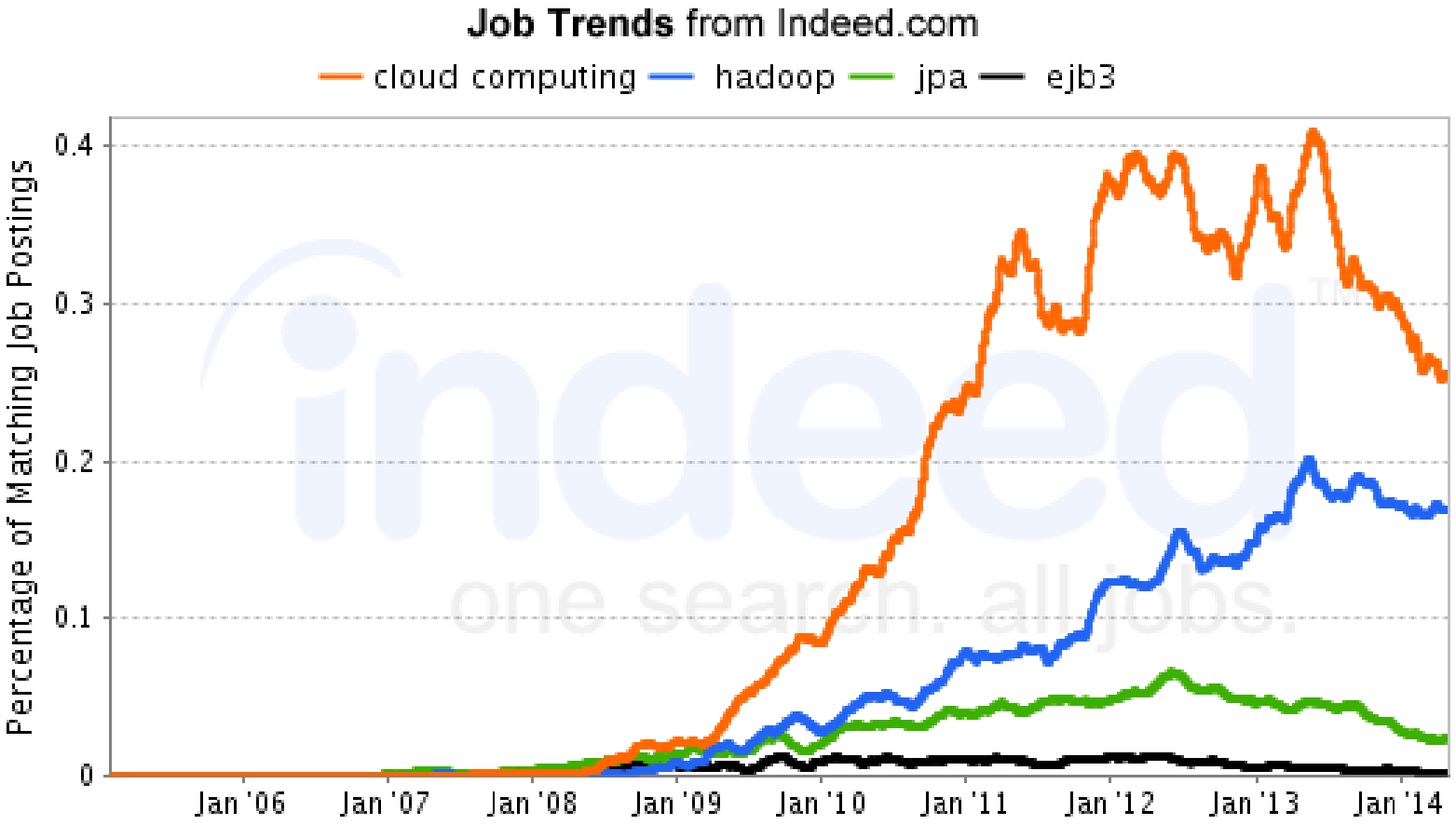
- Существующие средства на тот момент не были предназначены для работы с большими объемами данных

- Проект "The Apache™ Hadoop™“ разрабатывает open-source ПО для отказоустойчивых, масштабируемых и распределенных вычислений
  - Работает с BigData на обычных серверах
  - Сильное open-source комьюнити
  - Много различных продуктов и средств используют Hadoop

История Hadoop



- Начинался как подпроект в Apache Nutch
  - Nutch – это открытый Web Search Engine
  - OpenSource альтернатива Google
  - Начинал его Doug Cutting
- В 2004 году Google публикует статьи про GFS и MapReduce
- Doug Cutting и команда Nutch реализовала свой фреймворк на основе этих статей
- В 2006 Yahoo! Нанимает Doug Cutting для работы над Hadoop в своей команде
- В 2008 Hadoop становится Apache Top Level Project
  - http://hadoop.apache.org


Job Trends from Indeed.com
cloud computing
hadoop
jpa
ejb3
Percentage of Matching Job Postings
0.4
0.3
0.2
0.1
0
Jan '06
Jan '07
Jan '08
Jan '09
Jan '10
Jan '11
Jan '12
Jan '13
Jan '14
indeed
one search. all jobs.

Кто использует Hadoop

Хранение данных

- Емкость дисков выросла экспоненциально, в отличии от скорости чтения
  - 1990
    - Емкость 1400 Мб
    - Скорость чтения 4.5 Мб/сек
    - Чтение всего диска за ~5 мин
  - 2010
    - Емкость 1Тб
    - Скорость чтения 100 Мб/сек
    - Чтение всего диска за ~3 часа

- Hadoop:
  - 100 HDD работающих одновременно могут прочитать 1Тб данных за 2 мин

Кластер Hadoop

- “Дешевое” обычное железо (Commodity Hardware)
- Соединенное по сети
- Расположено в одном месте
  - Сервера в стойках в датацентре

Обычное железо (Commodity Hardware)

- “Дешевое” обычное железо для серверов
  - Это не суперкомпьютеры
  - Это не десктопы

Системные принципы Hadoop

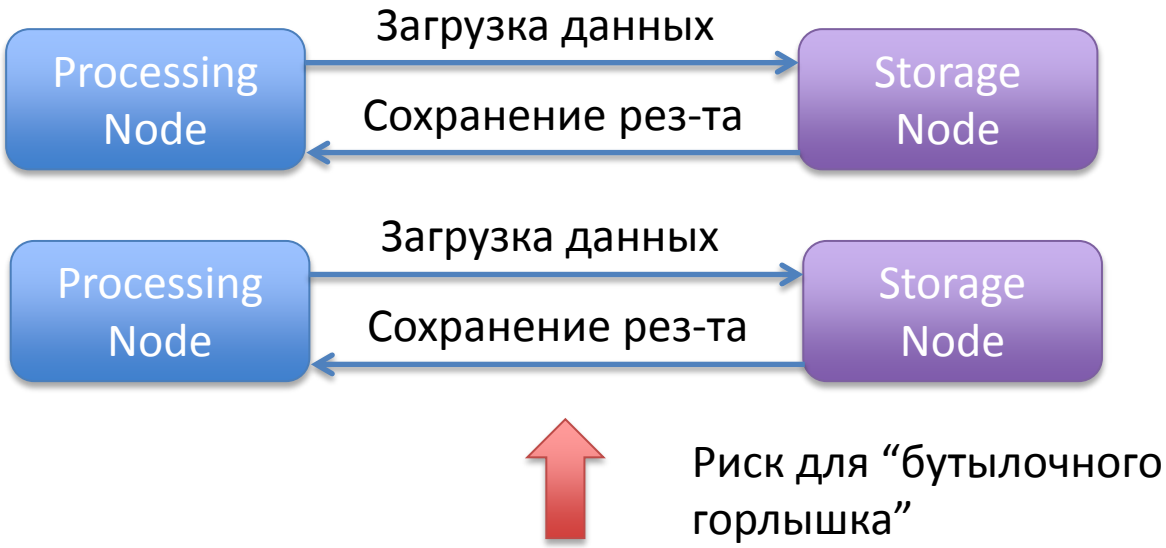
- Горизонтальное (Scale-Out) масштабирование вместо вертикального (Scale-Up)
- Отправляем код к данным
- Уметь обрабатывать падения и отказы оборудования
- Инкапсуляция сложности работы распределенных и многопоточных приложений

Горизонтальное масштабирование вместо вертикального

- Сложнее и дороже масштабироваться “вверх”
  - Добавить дополнительные ресурсы к существующему железу (CPU, RAM)
  - Закон Мура не успевает за ростом объема данных
  - Если нельзя улучшить железо, то надо покупать более мощное новое
  - Это вертикальное масштабирование
- Горизонтальное масштабирование
  - Добавить больше машин к существующему распределенному окружению
  - Уровень приложения поддерживает добавление/удаление нод
  - Hadoop исповедует такой подход – набор связанных нод
  - Так же очень просто масштабироваться “вниз”

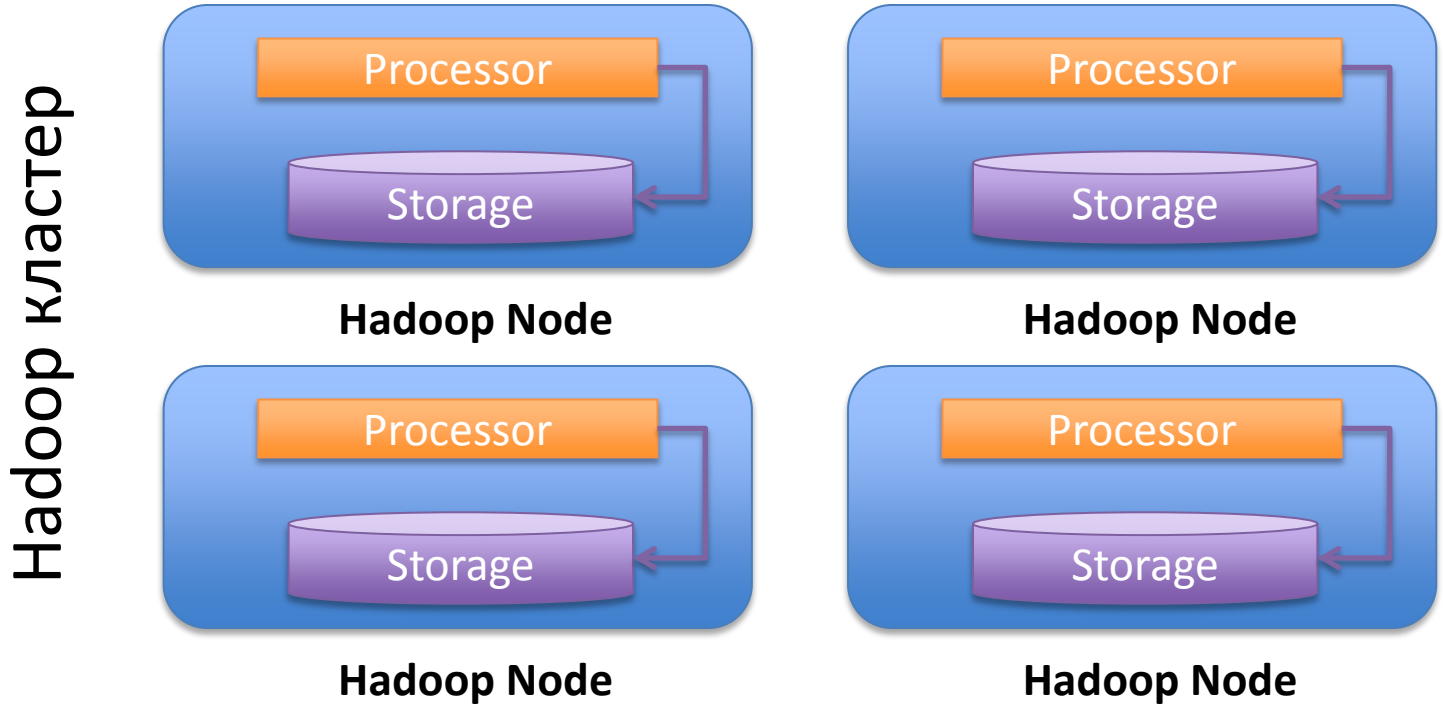
Код к данным

- Традиционная архитектура системы обработки данных
  - Ноды системы разделяются на вычислительные и стораджи, соединяются высокоскоростным линком
  - Многие приложения обработки данных являются CPU-bound, что приводит к проблемам с сетью



Код к данным

- Hadoop сближает вычислительный процессор и данные
  - Код копируется к данным (небольшой расход, Кб)
  - Процессор выполняет код и имеет доступ к локально расположенным данным



Отказы оборудования

- Чем больше количество машин, тем чаще будут отказы железа
  - На больших кластерах (сотни и тысячи машины) отказы будут еженедельно (и даже ежедневно!)
- Hadoop разрабатывался с учетом отказов железа
  - Репликация данных
  - Перезапуск тасков

Инкапсуляция сложности реализации

- Hadoop скрывает многие сложности распределенных и многопоточных систем
  - Небольшое число компонент
  - Предоставляет простой и хорошо определенный интерфейс для взаимодействия между компонентами
- Освобождает разработчика от заботы о проблемах системного уровня
  - Race conditions, ожидание данных
  - Организация передачи данных, распределение данных, доставка кода и т.д.
- Позволяет разработчику фокусироваться на разработке приложения и реализации бизнес-логики

Сравнение с СУБД (RDBMS)

- До недавнего времени многие приложения использовали Relational Database Management Systems (RDBMS) для batch processing
  - Oracle, Sybase, MySQL, Microsoft SQL Server и т.д.
  - Hadoop не заменяет полностью реляционные БД – многие архитектурные решения совмещают оба подхода
- Scale-Out vs. Scale-Up
  - RDBMS масштабируются “вверх”
    - Дорого для больших инсталляций
    - Появляются проблемы когда объем данных достигает сотен терабайт
  - Hadoop может масштабироваться до сотен машин и петебайт данных

Сравнение с СУБД (продолжение)

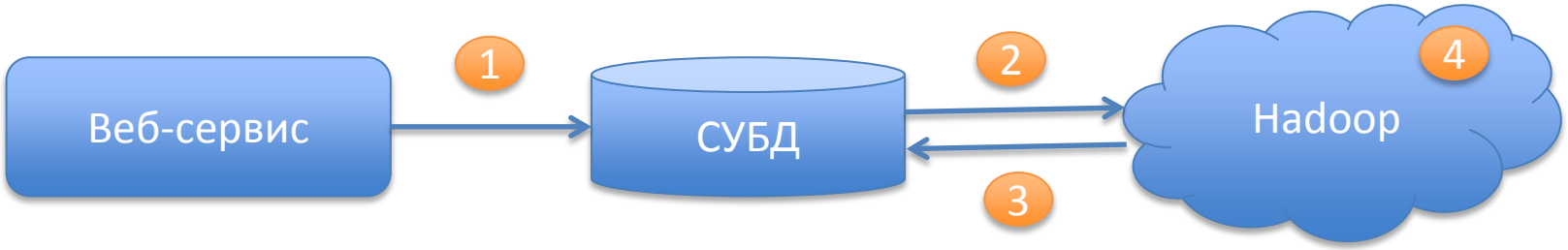
- Структурированные, слабоструктурированные и “сырые” данные
  - СУБД оптимально работают со структурированными данными
    - Таблицы в БД определяются схемой данных
  - Для Hadoop лучше всего подходят слабоструктурированные и “сырые” данные
    - Слабоструктурированные данные могут некоторую схему, которая не всегда соблюдается
    - “Сырые” данные вообще не имеют общей схемы и обычно это просто текстовые блоки или изображения
    - В процессе обработки таких данных пару key/value определяет сам пользователь
  - Определенные типы данных нет так то просто преобразовать в реляционную схему.
    - Напр., картинки, XML, JSON и т.д.

Сравнение с СУБД (продолжение)

- Offline batch vs. Online транзакции
  - Hadoop не был изначально предназначен для *real-time* или *low latency* запросов
  - Продукты, которые предоставляют возможность выполнения *low latency* запросов (напр., HBase), имеют ограниченный функционал
  - Hadoop лучше всего выполняет *offline batch processing* на больших объемах данных
  - СУБД лучше всего подходит для выполнения online транзакций и *low latency* запросов
  - Hadoop предназначен для обработки больших файлов и больших объемов данных
  - СУБД лучше всего работает с небольшими размерами записей

Сравнение с СУБД (продолжение)

- Hadoop и СУБД часто дополняют друг друга в рамках одной архитектуры
- Напр., веб-сервис
  - у которого небольшое число пользователей
  - но он генерит много логов с информацией



1. СУБД используется для предоставления данных пользователям и гарантии целостности данных
2. СУБД генерит большое кол-во логов, которые периодически отправляются в Hadoop-кластер
3. Все логи хранятся в Hadoop, поверх низ запускаются аналит. Задачи
4. Результат копируется обратно в СУБД, который затем используется веб-сервисом

Экосистема Hadoop

Экосистема Hadoop

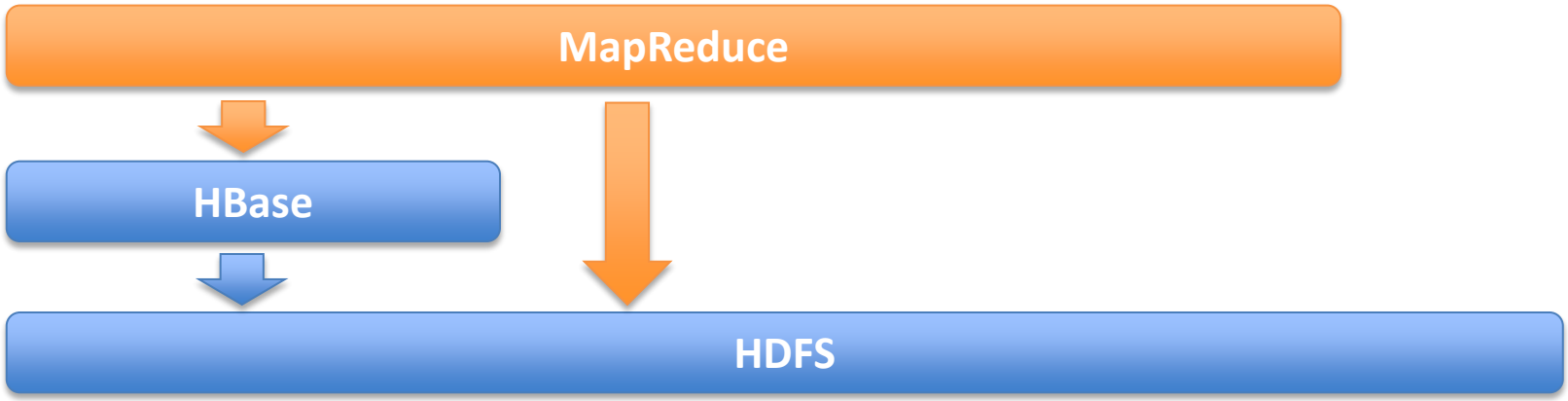
- Изначально Hadoop был известен в основном из-за двух ключевых компонент:
  - HDFS: Hadoop Distributed FileSystem
  - MapReduce: Фреймворк распределенной обработки данных
- Сейчас, в дополнении к этому, также известны следующие продукты:
  - Hbase: Column-oriented DB, поддержка последовательного и произвольного чтения, поддержка простых запросов
  - Zookeeper: Highly-Available Coordination Service
  - Oozie: Диспетчер задача для Hadoop
  - Pig: Язык обработки данных и среда выполнения
  - Hive: Data warehouse с SQL интерфейсом

Экосистема Hadoop

- Для разработки приложения необходима файловая система
  - В мире Hadoop это обычно Hadoop Distributed File System (HDFS)
  - В Linux это может быть ext3 и ext4
- Также, дополнительно к хранилищу данных, нужен удобный интерфейс для работы с данными
  - Hbase: Это key/value хранилище, реализованное поверх HDFS
  - В обычной системе используется СУБД, которая работает поверх локальной файловой системы

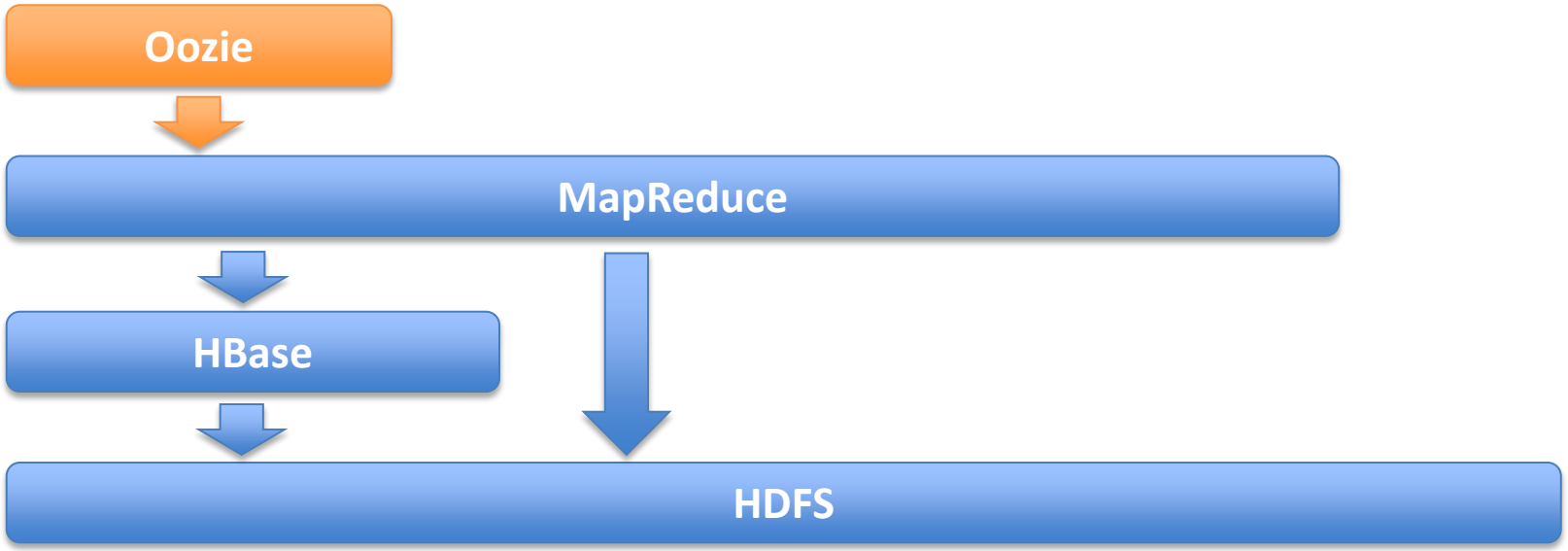
Экосистема Hadoop

- Для пакетной обработки данных надо использовать фреймворк
  - В мире Hadoop это MapReduce
  - MapReduce значительно упрощает реализацию распределенных приложений, которые должны работать на кластере



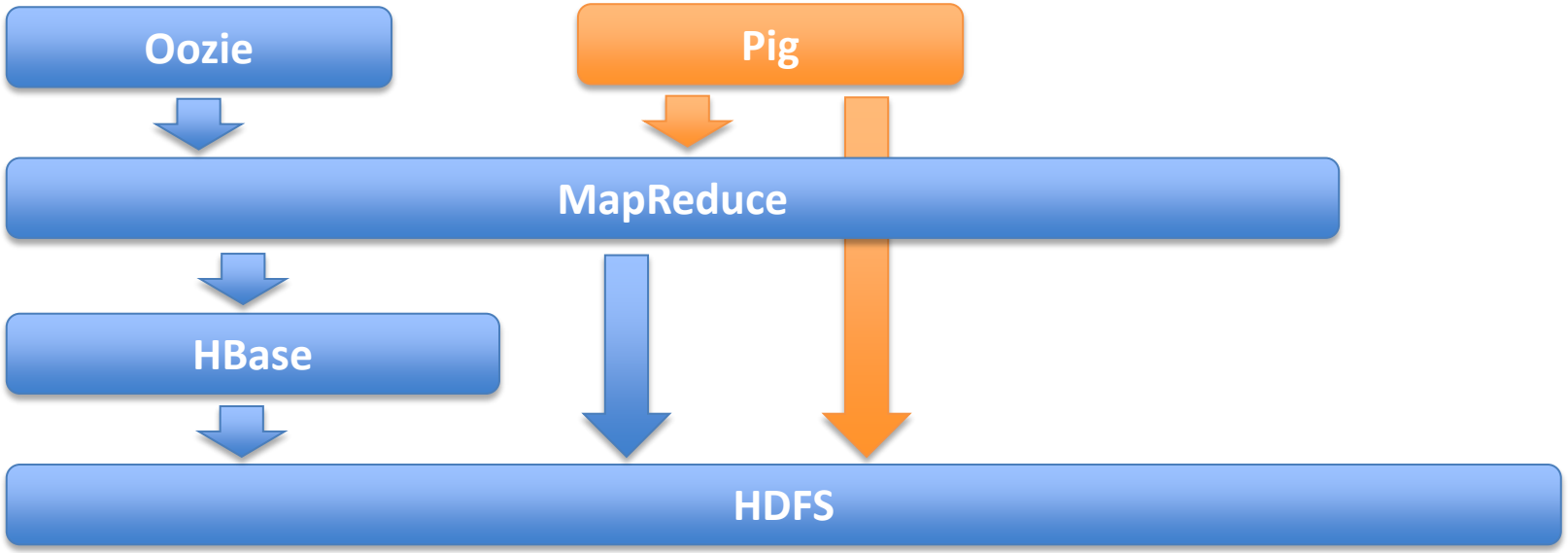
Экосистема Hadoop

- Решение многих задач подразумевает запуск несколько MapReduce задач
  - Apache Oozie: популярный продукт для координации рабочего процесса MR задач



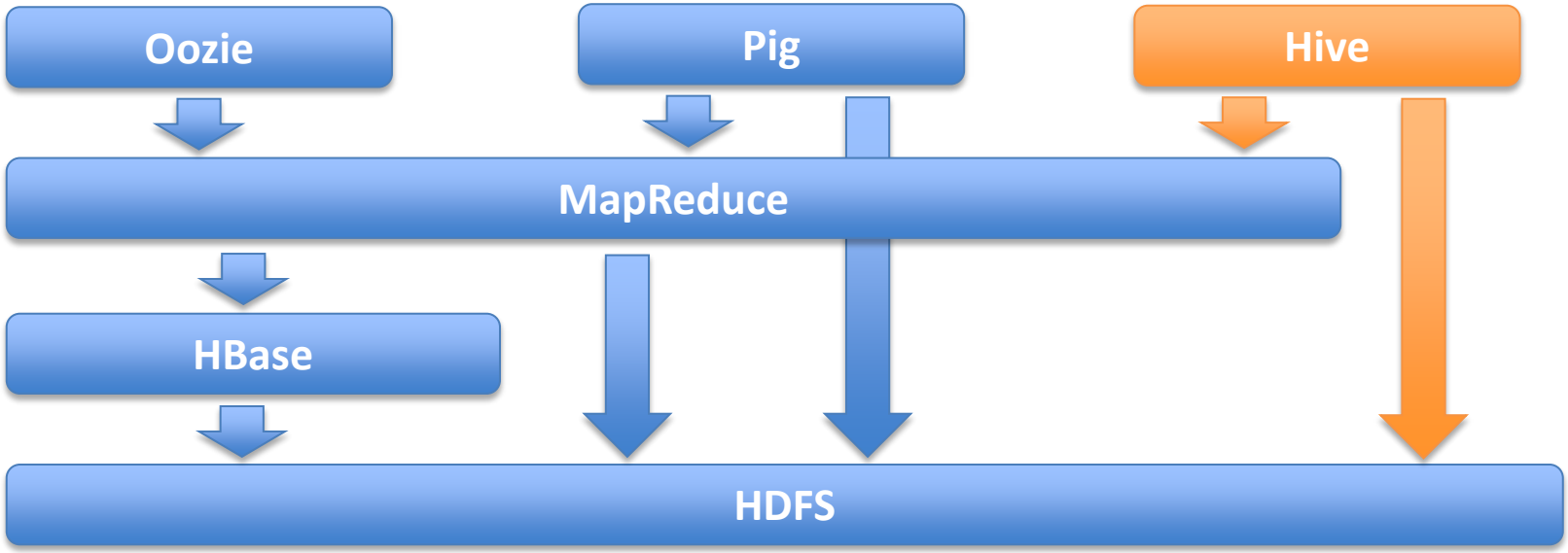
Экосистема Hadoop

- Фреймворк MapReduce не всегда удобно использовать для аналитиков и data scientists
  - Apache Pig является дополнением для высокоуровневой обработки данных используя специальный язык написания скриптов



Экосистема Hadoop

- Для тех, кто привык работать с SQL
  - Apache Hive позволяет организовать *data warehouse* и работать с ним через SQL-подобный интерфейс



Дистрибутивы Hadoop

- Сначала скачиваем Hadoop's HDFS и MapReduce с http://hadoop.apache.org/
- Все работает хорошо, но затем мы решаем начать использовать Hbase
  - Скачиваем HBase http://hadoop.apache.org/
  - Но оказывается, что эта версия HBase может работать только с предыдущей версией HDFS, поэтому мы устаналиваем нужную версию HDFS и все работает
- Еще чуть позже мы решаем установить Pig
  - Но, к сожалению, версия Pig не работает с нашей версий HDFS и ее надо проапгрейдить
  - Но если мы это сделаем, то перестанет работать HBase…

Дистрибутивы Hadoop

- Дистрибутивы Hadoop призваны решить проблему несовместимостей версий
- Вендоры дистрибутивов обеспечивают
  - Интеграционные тесты компонентов Hadoop
  - Инсталляционные пакеты в различных форматах
    - rpm, tarballs и т.д.
  - Могут включать дополнительные скрипты для запуска
  - Некоторые вендоры могут делать backport фич и исправлений багов из Apache
  - Обычно, найденный баги исправляются коммитерами, которые работают на вендоров и отправляются в основной Apache репозиторий

Вендоры дистрибутивов

- Cloudera Distribution for Hadoop (CDH)
- MapR Distribution
- Hortonworks Data Platform (HDP)
- Apache BigTop Distribution
- Greenplum HD Data Computing Appliance

Cloudera Distribution for Hadoop (CDH)

- Cloudera является лидером в распространии дистрибутивов Hadoop
  - Cloudera проповедует ту же политику, что и RedHat в популяризации Linux в свое время
- Самый популярный дистрибутив
  - http://www.cloudera.com/hadoop
  - 100% open-source
- В Cloudera работает большой процент коммитеров Hadoop
- CDH распространяется в различных форматах
  - RPM, Virtual Machine Images и tarballs

Cloudera Distribution for Hadoop (CDH)

- Включает большинство популярных продуктов Hadoop
  - HDFS, MapReduce, Hbase, Hive, Pig, Oozie, Mahout, Sqoop, Zookeeper, Flume

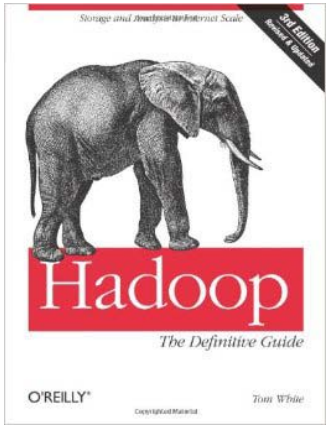
Поддерживаемые операционные системы

- Каждый дистрибутив поддерживает свой собственный набор операционных систем
- Обычно поддерживаются
  - Red Hat Enterprise
  - CentOS
  - Oracle Linux
  - Ubuntu
  - SUSE Linux Enterprise Server

Ресурсы

- Apache Hadoop Documentation
  - http://hadoop.apache.org
- Каждый отдельный продукт имеют свою собственную документацию
- Каждый вендор Hadoop предоставляет свою документацию
  - https://ccp.cloudera.com/display/DOC/Documentation

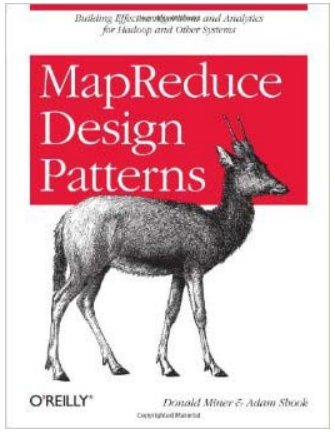
Книги



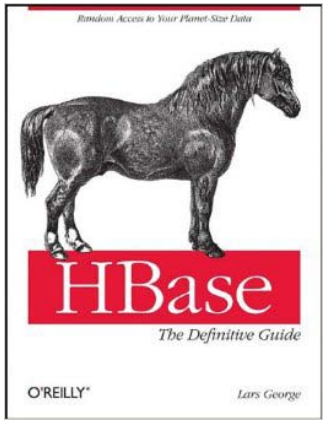
Hadoop: The Definitive Guide
Tom White (Author)
O'Reilly Media; 3rd Edition

Hadoop. Подробное руководство

MapReduce Design Patterns
Donald Miner (Author), Adam Shook (Author)
O'Reilly Media

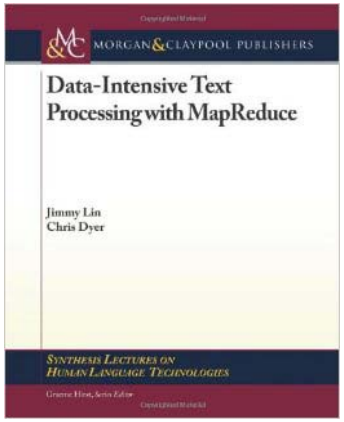


Книги

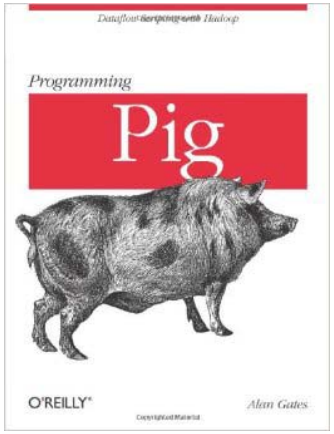


HBase: The Definitive Guide
Lars George (Author)
O'Reilly Media; 1 edition

Data-Intensive Text Processing with MapReduce
Jimmy Lin and Chris Dyer (Authors)
(April, 2010)
http://lintool.github.com/MapReduceAlgorithms/index.html

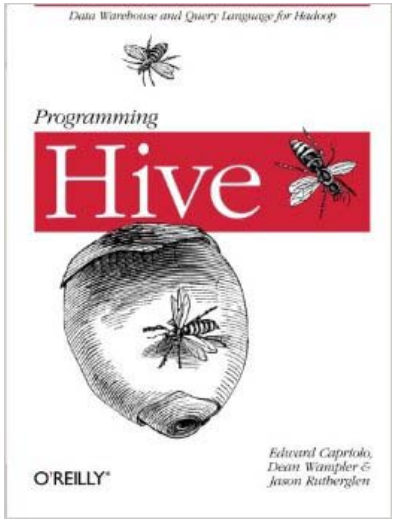


Книги



Programming Pig
Alan Gates (Author)
O'Reilly Media; 1st Edition

Programming Hive
Edward Capriolo, Dean Wampler,
Jason Rutherglen (Authors)
O'Reilly Media; 1 edition



Hadoop на Cloudera VM

- Cloudera предоставляет пакеты для инсталляции Hadoop на вирутальной машине.
  - Т.о. можно использовать Hadoop для учебных целей на различны ОС
- Скачать
  - **VirtualBox**: программа для виртуализации на локальной машине
    - https://www.virtualbox.org/wiki/Downloads
  - **Cloudera QUickstart Virtual Machine (VM)**: Single-node Hadoop кластер
    - http://www.cloudera.com/content/support/en/downloads/quickstart_vms/cdh-5-1-x1.html

Импорт и запуск VM

- В *VirtualBox Manager* выбрать *File->Import Appliance*
- Затем в окне ввода выбрать файл с *Cloudera VM*
- Импортировать *Cloudera VM*
- После успешного импорта в *VirtualBox Manager* надо запустить VM выбрав из контекстного меню пункт “Start”

Компиляция wordcount.jar

- После запуска VM откройте терминал и введите:

```
[cloudera@localhost ~]$ pwd
/home/cloudera

#если другая директория, то перейдите в нужную
[cloudera@localhost ~]$ cd /home/cloudera/
```

- Затем откройте текстовый редактор (*gedit*) и скопируйте туда код отсюда:
  - https://www.cloudera.com/content/cloudera-content/cloudera-docs/HadoopTutorial/CDH4/Hadoop-Tutorial/ht_wordcount1_source.html
- Затем сохраните файл с именем “/home/cloudera/WordCount.java”

Компиляция wordcount.jar

```
#export CLASSPATH
[cloudera@localhost ~]$ export CLASSPATH=/usr/lib/hadoop/client-
0.20/\*:/usr/lib/hadoop/\*

#display the value of CLASSPATH
[cloudera@localhost ~]$ echo $CLASSPATH
/usr/lib/hadoop/client-0.20/*:/usr/lib/hadoop/*

#make a directory to store the to-be-compiled class
[cloudera@localhost ~]$ mkdir wordcount_classes

#compile the class, save it to the wordcount_classes directory
[cloudera@localhost ~]$ javac -d wordcount_classes/ WordCount.java
```

Компиляция wordcount.jar

```
#make the .jar file, which is to be used for directing word count job in Hadoop
[cloudera@localhost ~]$ jar -cvf wordcount.jar -C wordcount_classes/ .
added manifest
adding: org/(in = 0) (out= 0)(stored 0%)
adding: org/myorg/(in = 0) (out= 0)(stored 0%)
adding: org/myorg/WordCount.class(in = 1546) (out= 749)(deflated 51%)
adding: org/myorg/WordCount$Map.class(in = 1938) (out= 798)(deflated
58%)
adding: org/myorg/WordCount$Reduce.class(in = 1611) (out= 649)(deflated
59%)

#list files in the current directory. Now you should see the wordcount.jar file
listed there.
[cloudera@localhost ~]$ ls
```

Копирование файлов в HDFS

- В качестве примера мы запустим задачу подсчета слов в файле
- Для этого создадим несколько текстовых файлов и скопируем их в HDFS

```
[cloudera@localhost ~]$ echo "Hello World Bye World" >file0
[cloudera@localhost ~]$ echo "Hello Hadoop Bye Hadoop" >file1

[cloudera@localhost ~]$ hadoop fs -mkdir /user/cloudera/wordcount
[cloudera@localhost ~]$ hadoop fs -mkdir /user/cloudera/wordcount/input

[cloudera@localhost ~]$ hadoop fs -put file0 /user/cloudera/wordcount/input
[cloudera@localhost ~]$ hadoop fs -put file1 /user/cloudera/wordcount/input
```

Запуск MapReduce задачи в Hadoop

- Запустим задачу в терминале
  - Для этого надо указать путь в jar-файлу, основной, input и output директории в HDFS
- Важно: output директория не должна существовать в HDFS т.к. это приведет к ошибке запуска задачи

```
[cloudera@localhost ~]$ hadoop jar wordcount.jar org.myorg.WordCount
/user/cloudera/wordcount/input /user/cloudera/wordcount/output
```

Проверка результатов

- Результат будет находиться в отдельном файле в output директории в HDFS

```
[cloudera@localhost ~]$ hadoop fs -ls /user/cloudera/wordcount/output
Found 3 items
-rw-r--r--    3 cloudera cloudera    0 2014-03-15 11:56 /user/cloudera/wordcount/output/_SUCCESS
drwxr-xr-x   - cloudera cloudera    0 2014-03-15 11:56 /user/cloudera/wordcount/output/_logs
-rw-r--r--    3 cloudera cloudera  31 2014-03-15 11:56 /user/cloudera/wordcount/output/part-00000

[cloudera@localhost ~]$ hadoop fs -cat /user/cloudera/wordcount/output/part-00000
Bye 2
Hadoop  2
Hello   2
World   2
```

Вопросы?