ТЕХНОСФЕРА

Методы распределенной обработки больших объемов данных в Hadoop

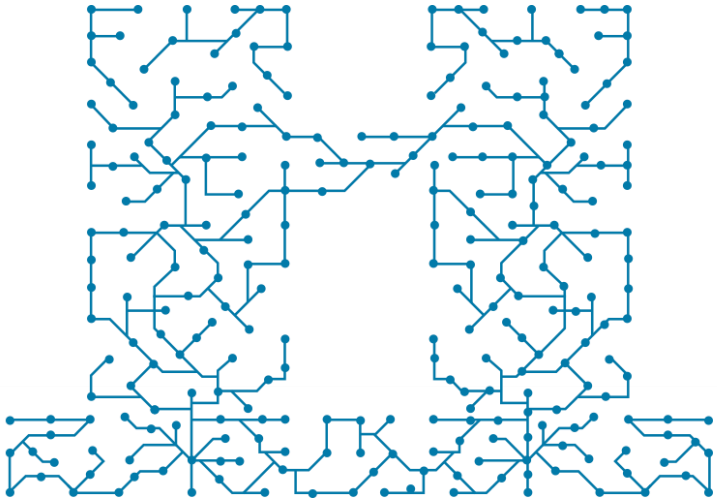
Лекция 1: Введение в BigData, Hadoop и MapReduce

Москва 2017

ЕВГЕНИЙ ЧЕРНОВ

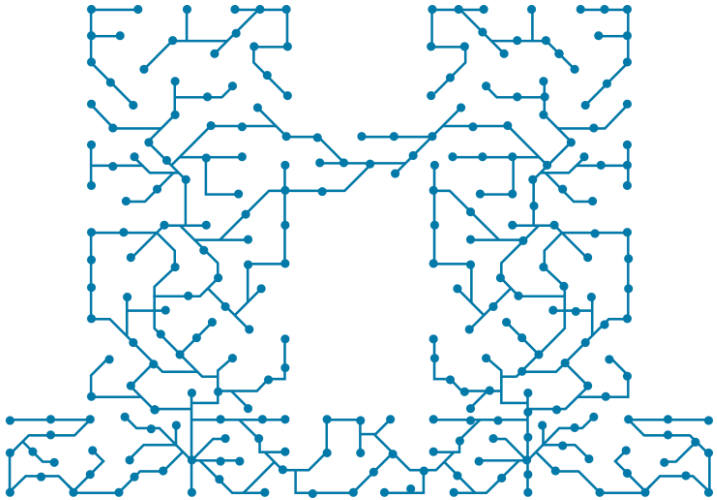
Руководитель отдела анализа запросов в проекте Поиск@Mail.Ru. Ранее отвечал за обработку поведенческих данных в отделе ранжирования в проекте Поиск@Mail.Ru. Обработка данных происходила в основном с помощью платформы Hadoop.

Выпускник факультета Управления и Прикладной Математики МФТИ в 2009 г, кафедра Системного программирования РАН.



Владимир Гулин

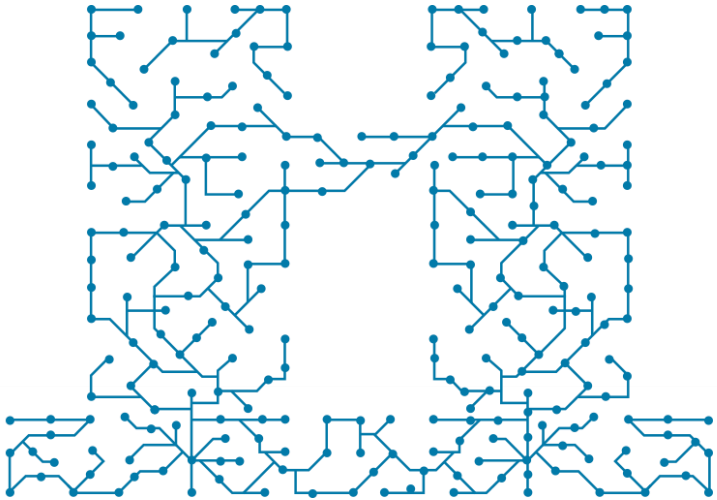
Руководитель Поиск@Mail.Ru.



ДЕНИС КЛЮКИН

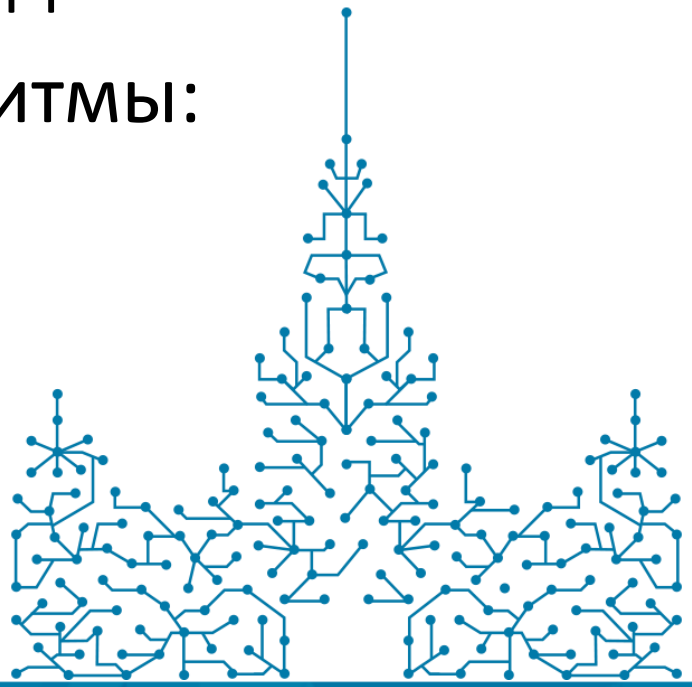
Разработчик отдела рекомендаций в проекте Поиск@Mail.Ru. Активно использует передовые технологии обработки больших объемов данных.

Выпускник кафедры Систем Автоматизированного Проектирования МГТУ им. Баумана в 2014 г.



Чему посвящен данный курс?

- Обработка больших объемов данных
  - “Big Data”
- Распределенная обработка данных
- Технологии Hadoop и алгоритмы:
  - MapReduce
  - HBase
  - Spark



Оценка результатов

- Набираете баллы за:
  - 4 домашних задания по 15 баллов
  - 4 рубежных контроля по 10 баллов
  - Экзамен (можно только улучшить оценку по РК)
- Получаете оценку:

  2: 0-54

  3: 55-70

  4: 71-85

  5: 86-100

Необходимые знания

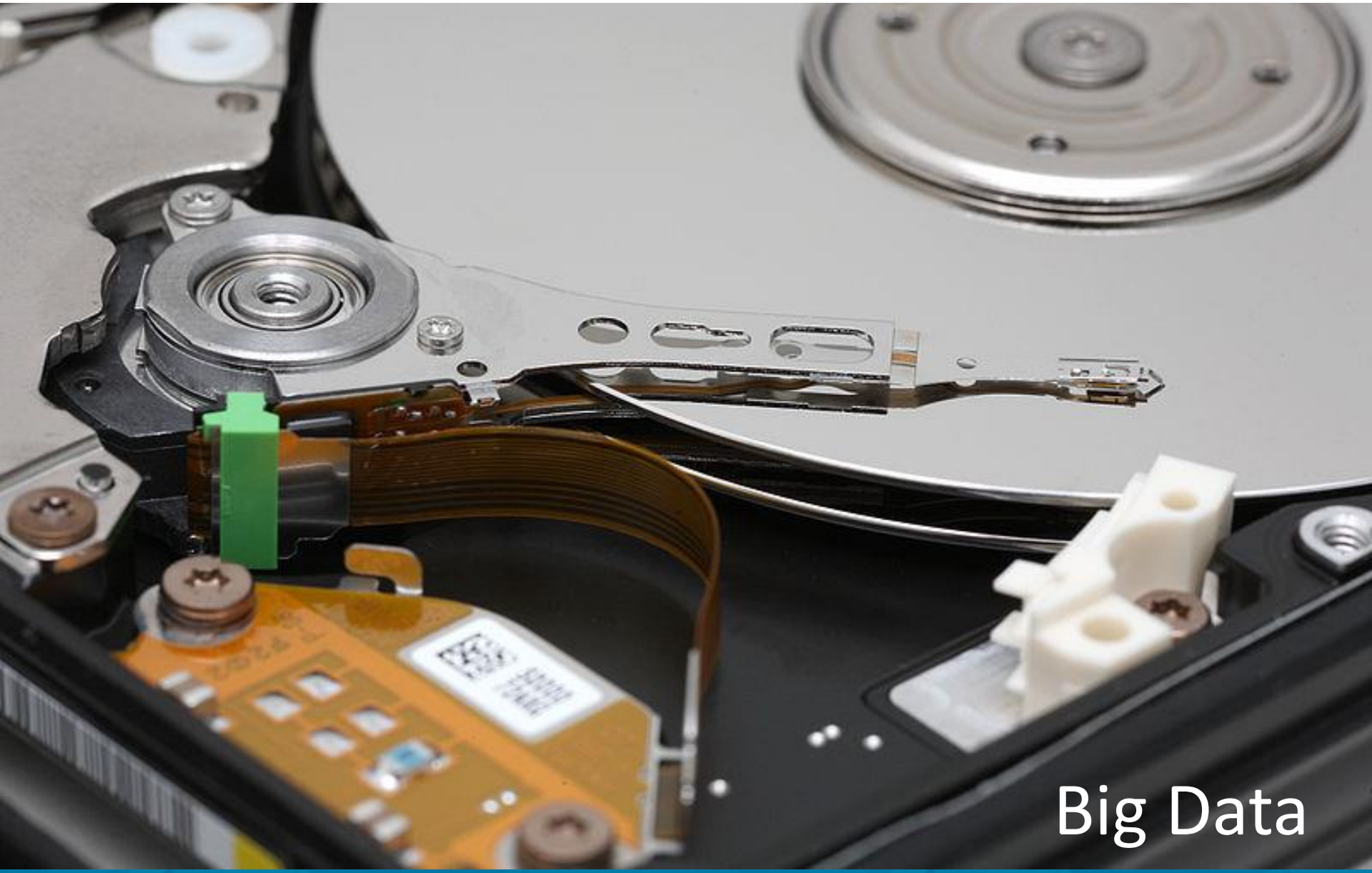
- Программирование на Java и Python
  - Этот курс не учит программировать
  - Фокус на “thinking at scale” и разработка/дизайн алгоритмов
  - Самостоятельная установка Hadoop (рекомендуется)
- Уметь отлаживать свой код
- Опыт работы в Linux (желательно)

Как же стать гуру Hadoop?

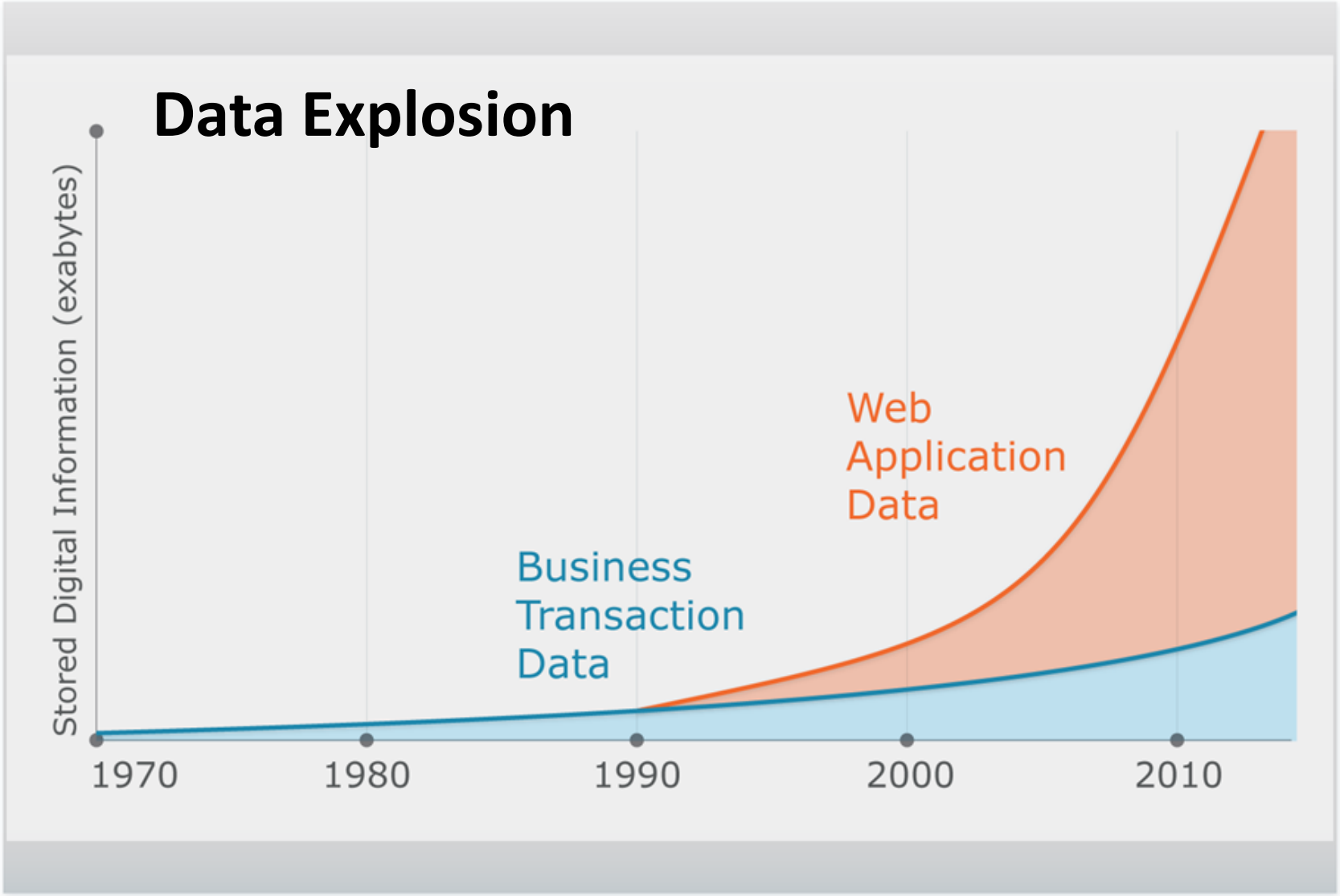
- Посещать занятия, выполнять ДЗ
- Читать книги
  - Tom White, “Hadoop: The Definitive Guide”
    - Есть издание на русском языке
  - Jimmy Lin and Chris Dyer, “Data-Intensive Text Processing with MapReduce”

Использование Hadoop

- Hadoop на вашем компьютере
- Hadoop в виртуальной машине:
  - https://www.cloudera.com/downloads/quickstart_vms/5-12.html
- Hadoop в нашей лаборатории



Big Data


Data Explosion
Stored Digital Information (exabytes)
Web
Application
Data
Business
Transaction
Data
1970
1980
1990
2000
2010

Общий размер данных (2010г)*:
1.2 Зеттабайт (1.2 Триллиона Гигабайт)

* По данным Information Data Corporation (IDC)

Обрабатывает 20 PB в день (2008)
Скачивает 20B веб-страниц в день (2012)

>10 PB данных,
75B DB (6/2012)

>100 PB польз. данных +
500 TB/день (8/2012)

S3: 449B объектов

JPMorganChase

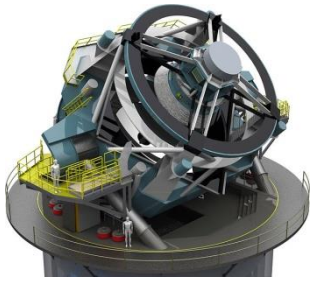
150 PB на 50k+ серверов
работает 15k apps (6/2011)

INTERNET ARCHIVE

Wayback Machine: 240B веб-страниц
в архиве, 5 PB (1/2013)

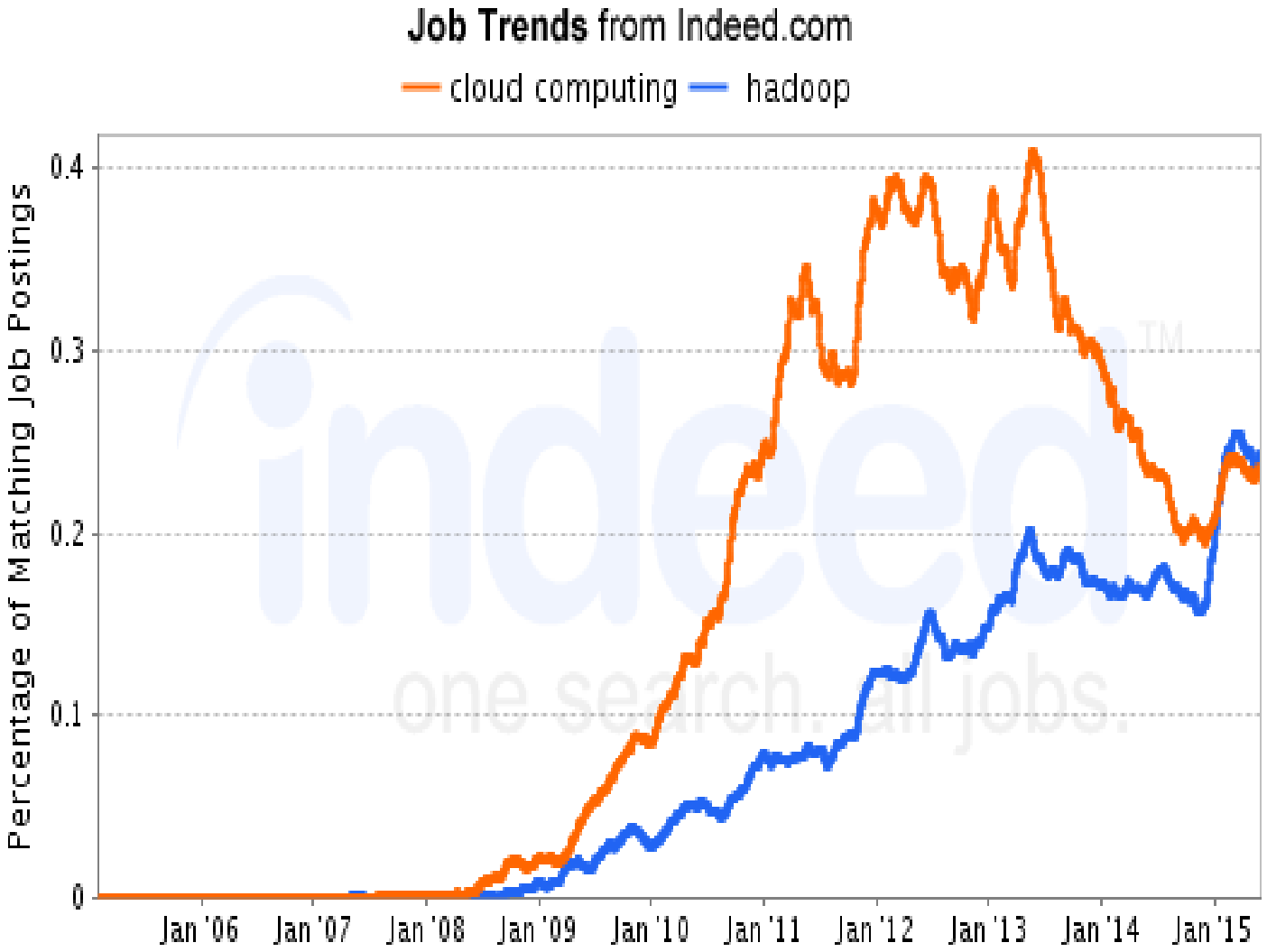
LHC: ~15 PB в год

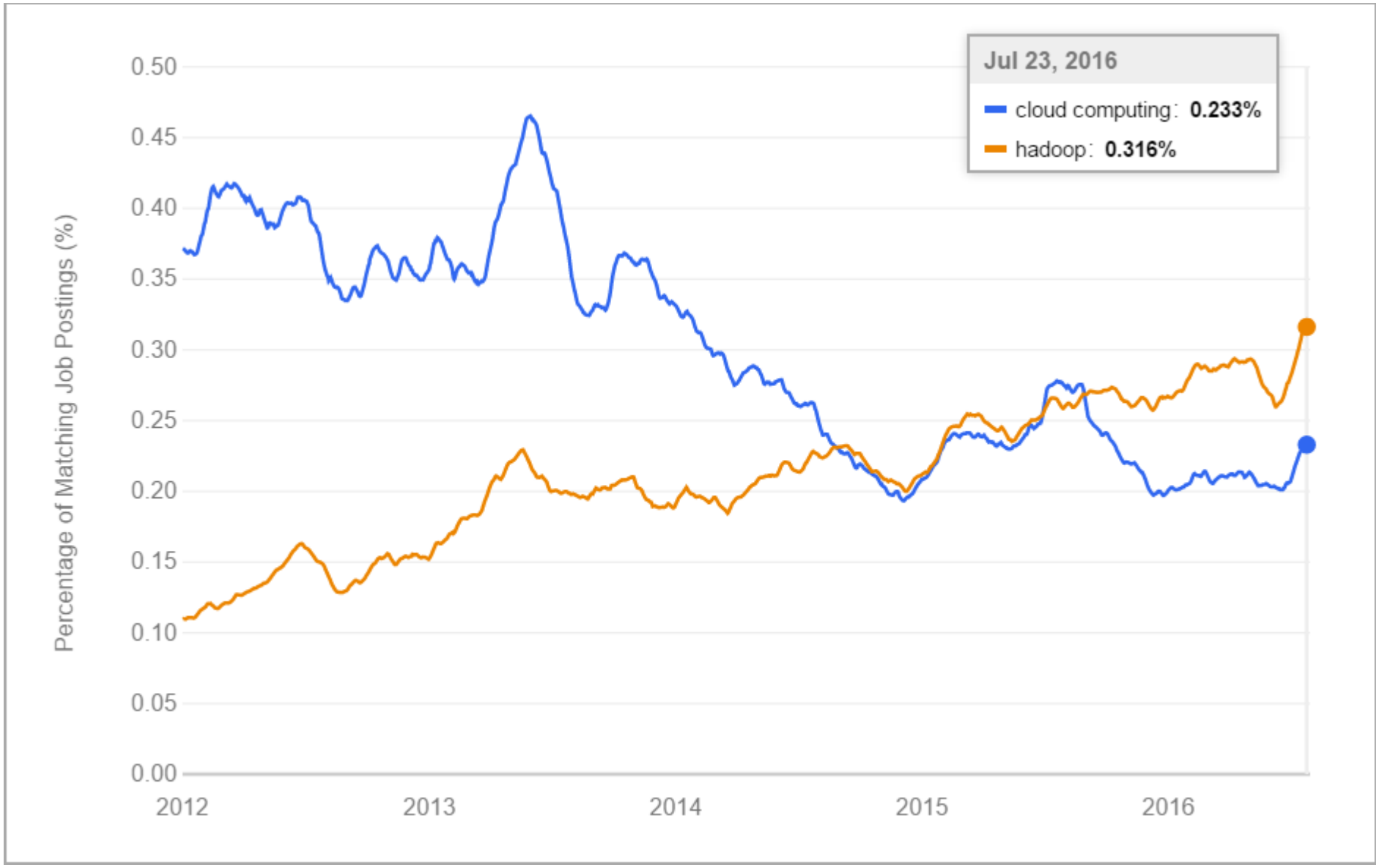
CERN

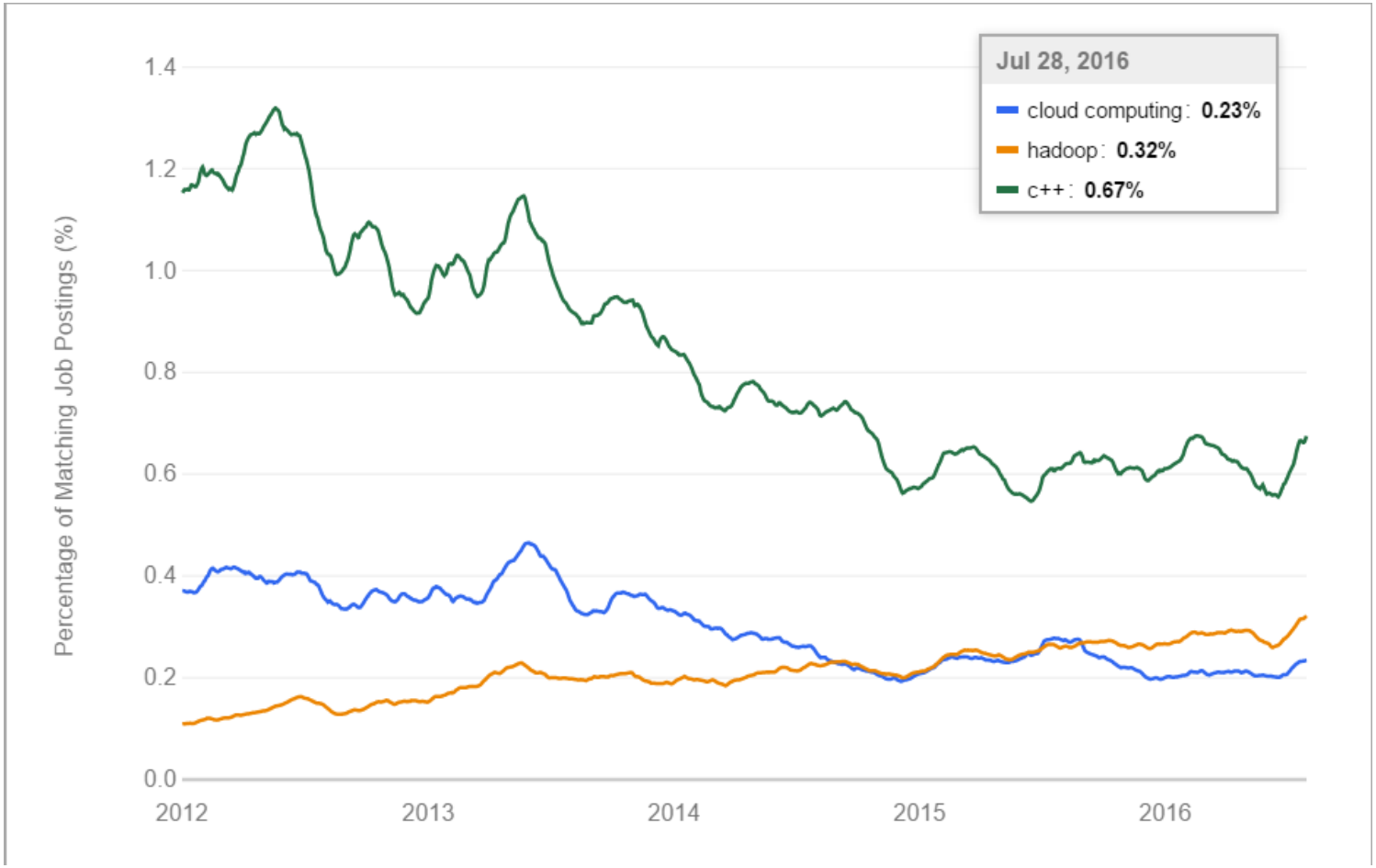


LSST: 6-10 PB в год (~2015)

Много данных – это сколько?


Job Trends from Indeed.com
cloud computing
hadoop
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
Jan '15


Jul 23, 2016
cloud computing: 0.233%
hadoop: 0.316%
Percentage of Matching Job Postings (%)
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2012
2013
2014
2015
2016

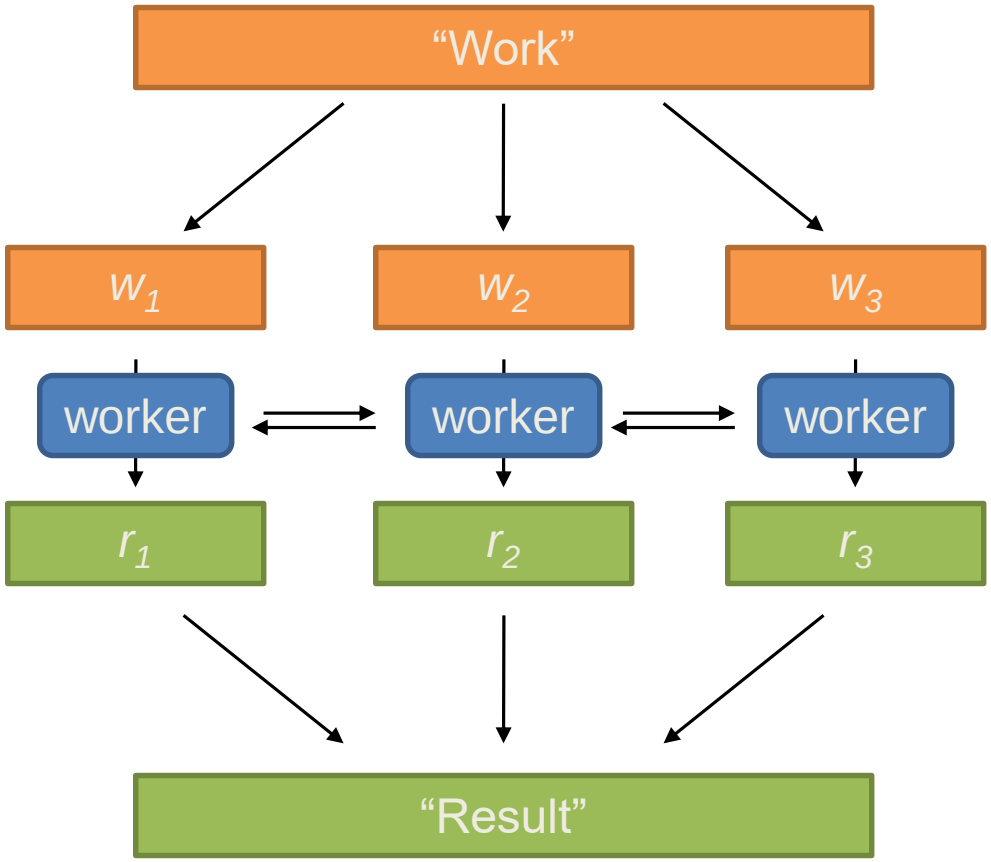


Top Job Trends:

1. HTML5
2. MongoDB
3. iOS
4. Android
5. Mobile app
6. Puppet
7. Hadoop
8. jQuery
9. PaaS
10. Social Media

Работаем с Big Data

Divide and Conquer



Partition

Combine

Вопросы параллелизма

- Как назначать части work по воркерам?
- А что если у нас больше частей work чем воркеров?
- А что если воркерам надо обмениваться промежуточными результатами?
- Как мы объединяем промежуточные результаты?
- Как мы узнаем, что все воркеры отработали?
- А что если воркер умрет?

Какая общая проблема объединяет эти вопросы?

Общая проблема

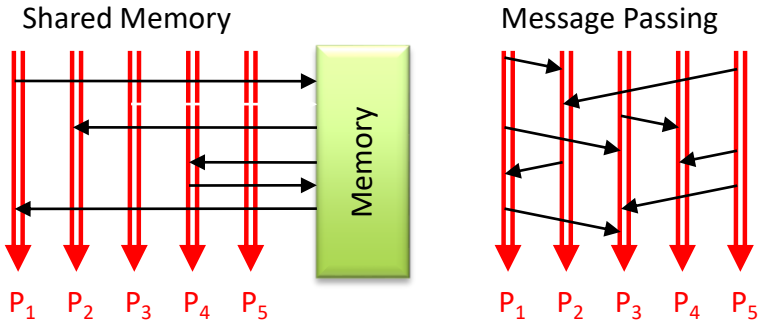
- Проблемы параллелизма возникают из-за:
  - Взаимодействия между воркерами
    - например, обмен состояниями
  - Доступ к общим ресурсам
    - например, данные
- Т.о. нам нужен механизм синхронизации

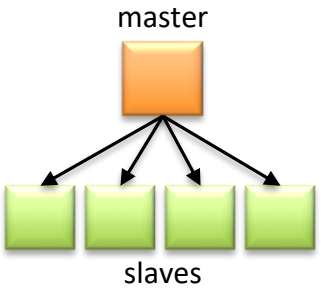
Управление множеством воркеров

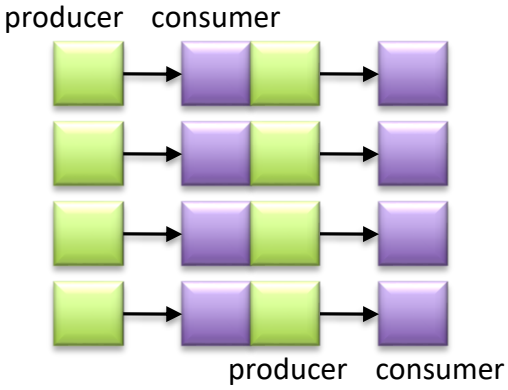
- Трудно потому что:
  - Мы не знаем порядок, в котором воркеры запускаются
  - Мы не знаем, когда воркеры прерывают друг друга
  - Мы не знаем, когда воркерам надо обмениваться промежуточным результатом
  - Мы не знаем порядок, в котором воркеры имеют доступ к общим данным
- Поэтому, нам нужны:
  - Семафоры (lock, unlock)
  - Conditional variables (wait, notify, broadcast)
  - Barriers
- Все равно много проблем:
  - Deadlock, livelock, race conditions...
  - Dining philosophers, sleeping barbers, cigarette smokers...
- Нужно быть очень осторожным!

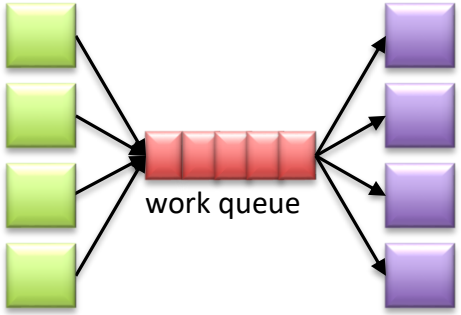
Текущие средства

- Программные модели
  - Shared memory (pthreads)
  - Message passing (MPI)
- Design Patterns
  - Master-slaves
  - Producer-consumer flows
  - Shared work queues



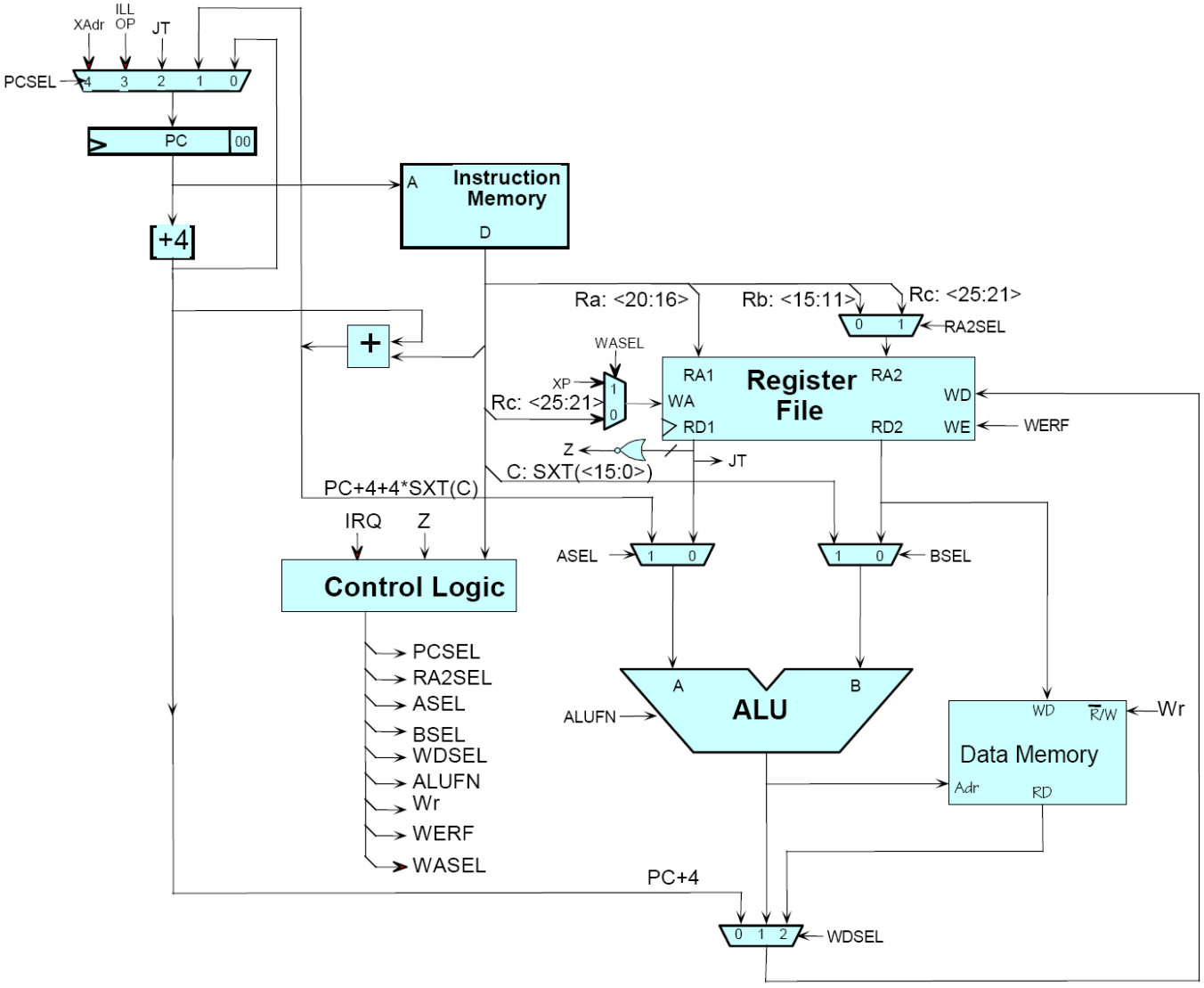


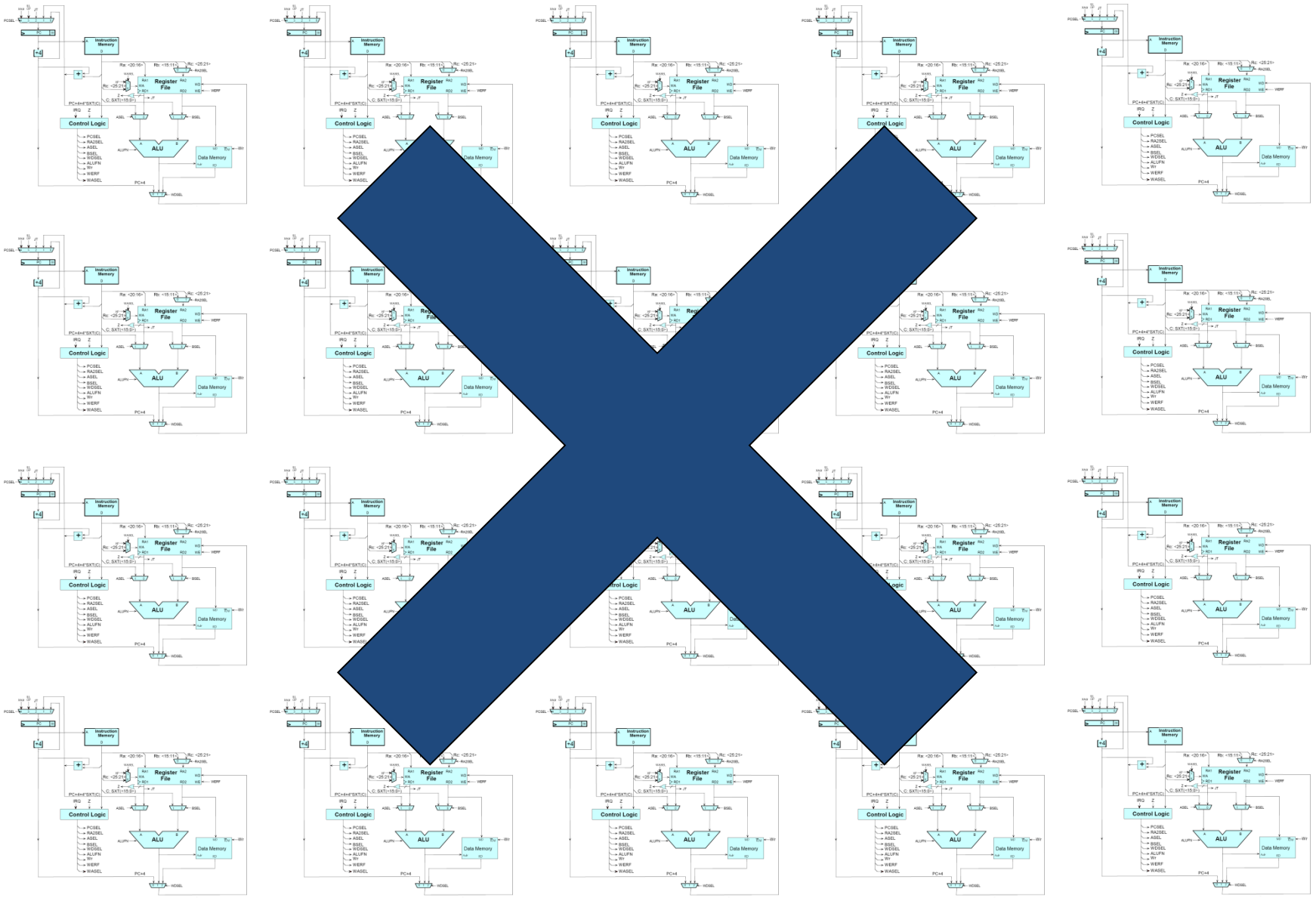




Момент истины

- Concurrency сложная вещь
- Особенно в случае:
  - работы в нескольких датацентрах
  - при наличии отказов
- Сложная отладка и тестирование приложений
- Реальность:
  - Множество одноразовых и узкозаточенных решений
  - Написание своих специальных библиотек, затем их использование
  - Бремя программиста все это поддерживать

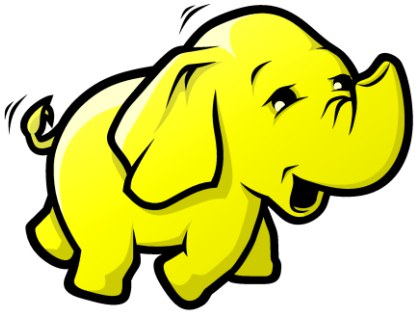

XAdr
ILL OP
JT
PCSEL
4 3 2 1 0
PC
00
+4
A
Instruction Memory
D
Ra: <20:16>
Rb: <15:11>
Rc: <25:21>
0 1
RA2SEL
WASEL
XP
1
0
Rc: <25:21>
RA1
WA
RD1
Register File
RA2
WD
RD2
WE
WERF
Z
JT
C: SXT(<15:0>)
PC+4+4*SXT(C)
IRQ
Z
Control Logic
ASEL
1 0
1 0
BSEL
PCSEL
RA2SEL
ASEL
BSEL
WDSEL
ALUFN
Wr
WERF
WASEL
A
B
ALUFN
ALU
WD
R/W
Wr
Data Memory
Adr
RD
PC+4
0 1 2
WDSEL


Instruction Memory
Register File
Control Logic
ALU
Data Memory


hadoop

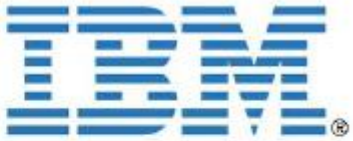
- Проект "The Apache™ Hadoop™“ разрабатывает open-source ПО для отказоустойчивых, масштабируемых и распределенных вычислений
- Особенности:
  - Работает с BigData на обычных серверах
  - Сильное open-source комьюнити
  - Много различных продуктов и средств используют Hadoop

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

Кто использует Hadoop



Системные принципы Hadoop

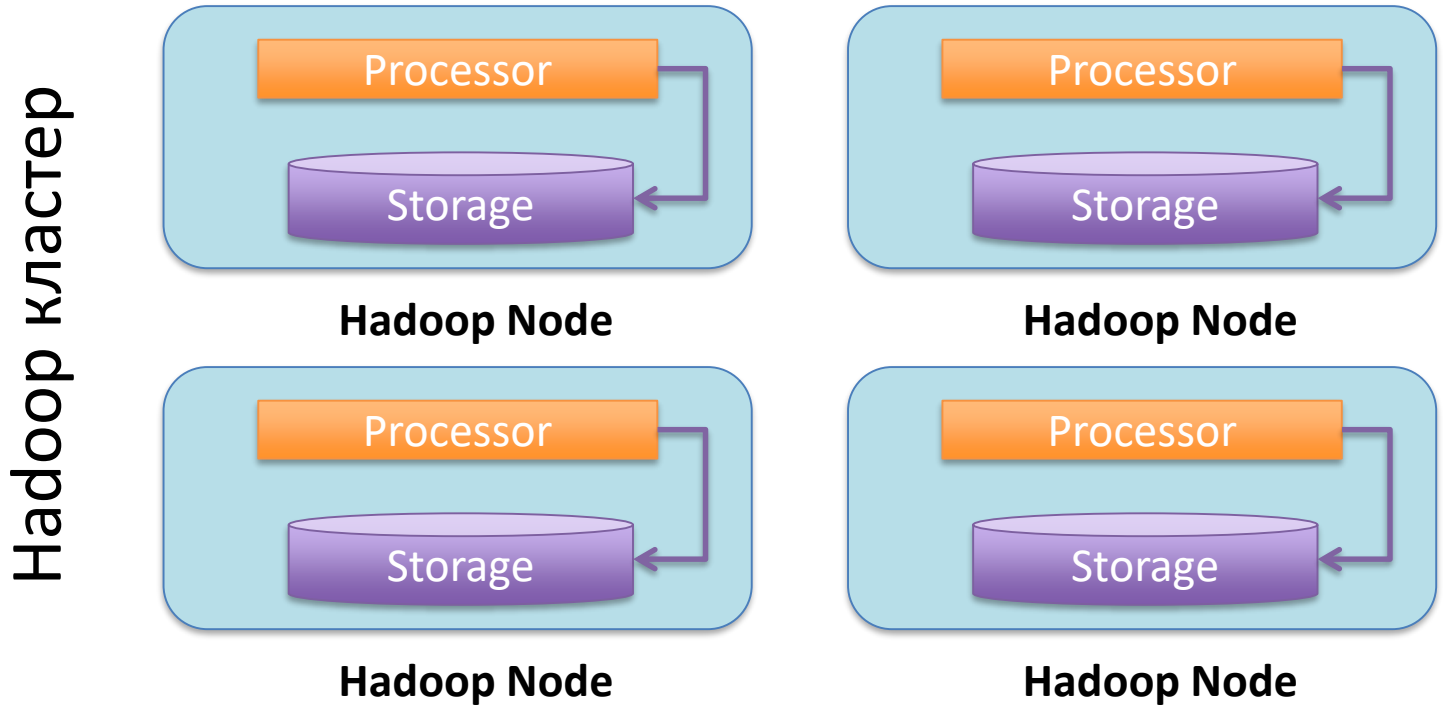
- Горизонтальное (Scale-Out) масштабирование вместо вертикального (Scale-Up)
- Отправляем код к данным
- Умение обрабатывать падения нод и отказы оборудования
- Инкапсуляция сложности работы распределенных и многопоточных приложений

Масштабирование

- Вертикальное:
  - Добавить дополнительные ресурсы к существующему железу (CPU, RAM)
  - Если нельзя улучшить железо, то надо покупать более мощное новое
  - Закон Мура не успевает за ростом объема данных
- Горизонтальное:
  - Добавить больше машин к существующему кластеру
  - Приложение поддерживает добавление/удаление серверов
  - Просто масштабироваться “вниз”

Данные к коду

Код к данным


Hadoop кластер
Processor
Storage
Hadoop Node
Processor
Storage
Hadoop Node
Processor
Storage
Hadoop Node
Processor
Storage
Hadoop Node

Отказы оборудования

- Чем больше количество машин, тем чаще будут отказы железа
- Hadoop разрабатывался с учетом отказов железа
  - Репликация данных
  - Перезапуск тасков

Инкапсуляция сложности реализации

- Hadoop скрывает сложности распределенных систем
- 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 логики приложения

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
  - Не суперкомпьютеры
  - Не десктопы
- Соединенное по сети
- Расположено в одном месте
  - Сервера в стойках в датацентре

Экосистема Hadoop

Экосистема Hadoop

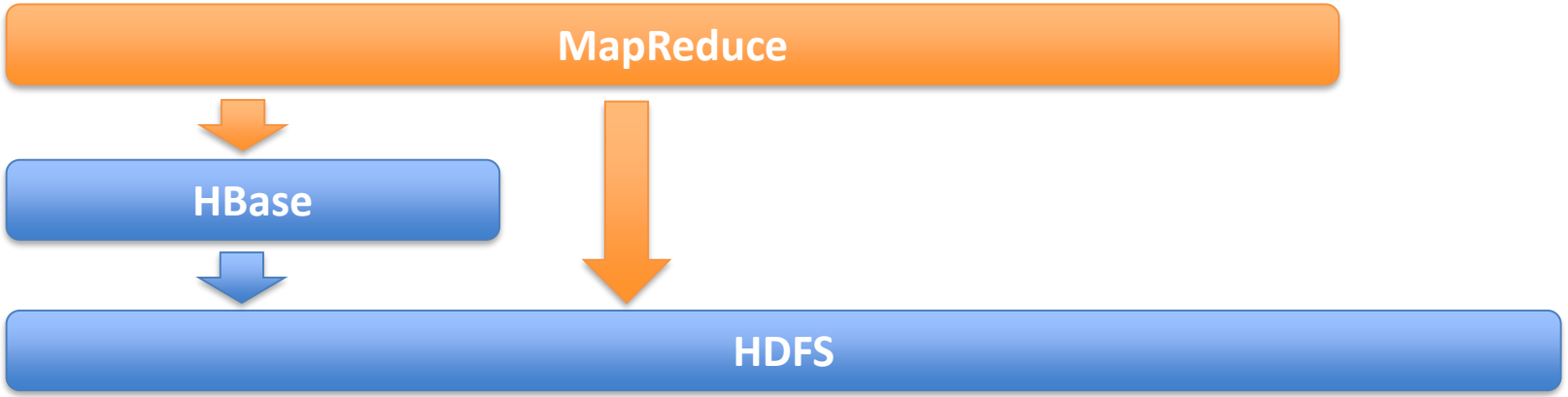
- Главные компоненты Hadoop:
  - HDFS: Hadoop Distributed FileSystem
  - MapReduce: Фреймворк распределенной обработки данных
- Другие компоненты :
  - Hbase: Column-oriented NoSQL DB
  - Zookeeper: Highly-Available Coordination Service
  - Oozie: Диспетчер задача для Hadoop
  - Pig: Язык обработки данных и среда выполнения
  - Hive: Data warehouse с SQL интерфейсом

Экосистема Hadoop

- Для разработки приложения необходима файловая система
  - В Linux: ext3 и ext4
  - В мире Hadoop обычно Hadoop Distributed File System (HDFS)
- Также нужен удобный интерфейс для работы с данными
  - Реляционная СУБД поверх локальной файловой системы
  - Hbase: Это key/value хранилище, реализованное поверх HDFS

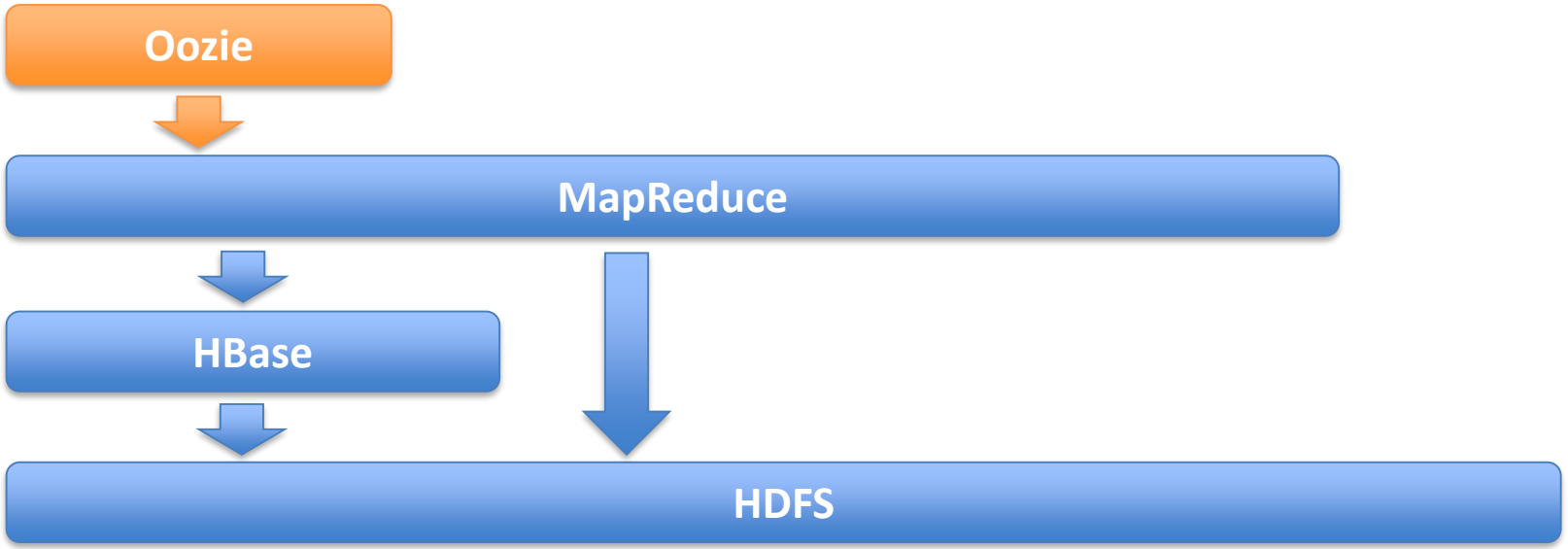
Экосистема Hadoop

- Фреймворк для запуска MapReduce задач



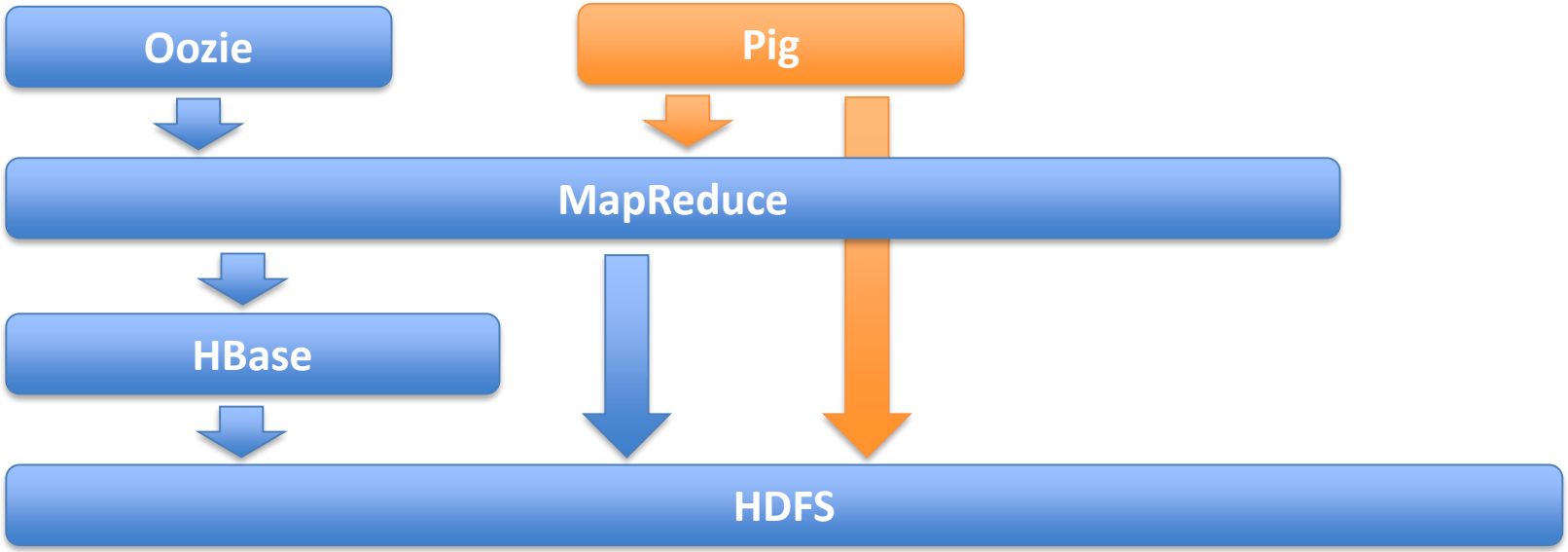
Экосистема Hadoop

Apache Oozie: популярный продукт для координации рабочего процесса MR задач



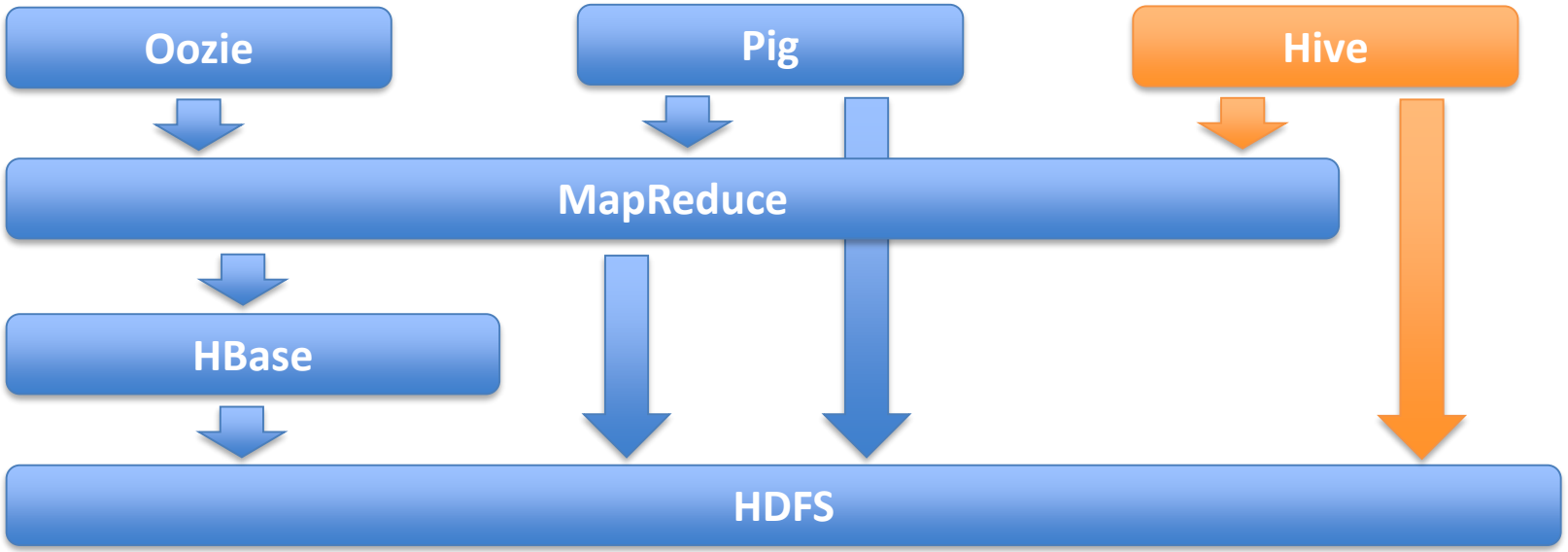
Экосистема Hadoop

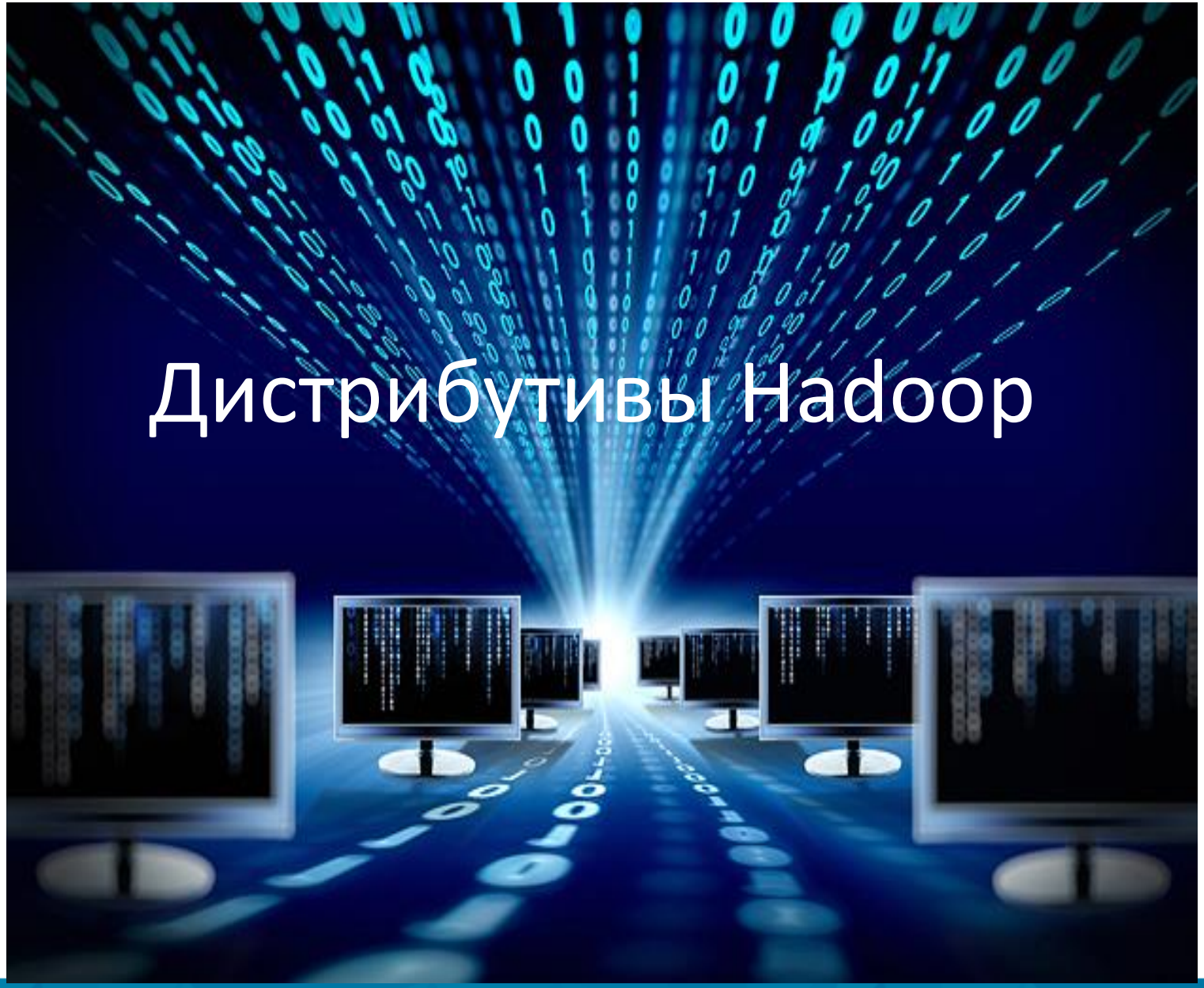
Apache Pig инструмент для обработки данных с помощью высокоуровневых команд



Экосистема Hadoop

Apache Hive обработка данных с помощью SQL-подобных запросов





Дистрибутивы Hadoop

Установка Hadoop

- Скачиваем и устанавливаем HDFS и MapReduce с http://hadoop.apache.org/
- Потом скачиваем HBase, не работает с текущим HDFS -> переустанавливаем HDFS
- Устанавливаем Pig -> не работает с нашим HDFS
- Меняем HDFS – перестает работать HBase

Дистрибутивы Hadoop

- Решают проблему несовместимости версий
- Вендоры дистрибутивов обеспечивают:
  - Интеграционные тесты компонентов Hadoop
  - Инсталляционные пакеты в различных форматах
  - Некоторые вендоры делают дополнительные фичи и исправляют баги в стандартной версии Hadoop

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 распространении Hadoop
- 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

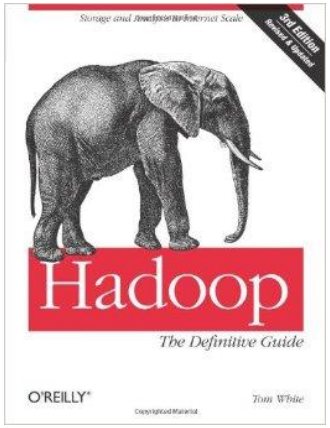
Поддерживаемые операционные системы

- 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

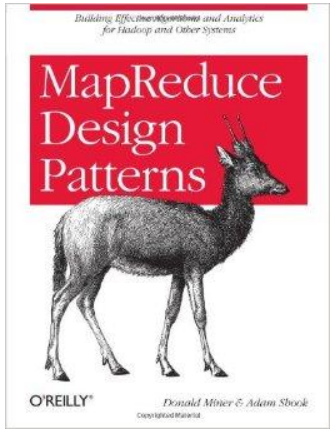
Книги



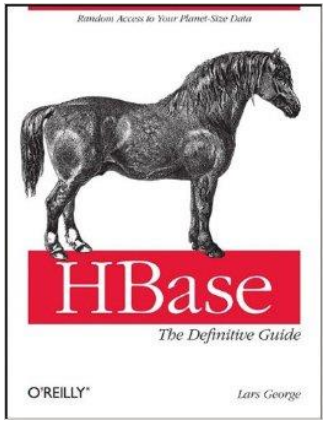
Hadoop: The Definitive Guide
Tom White (Author)
O'Reilly Media; 3rd Edition

Hadoop. Подробное руководство

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

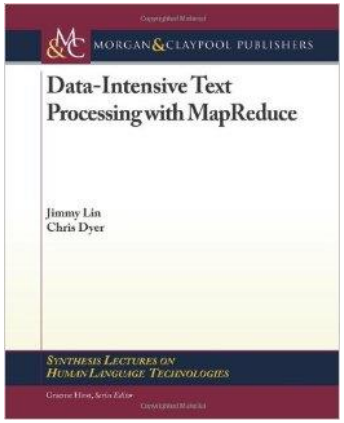


Книги

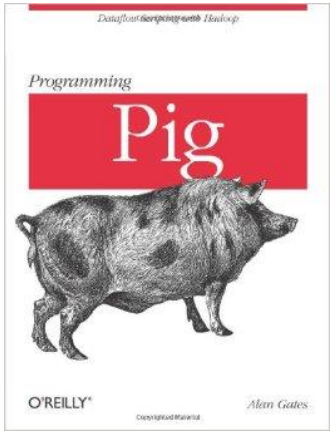


HBase: The Definitive Guide
Lars George (Author)
O'Reilly Media; 1 edition

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

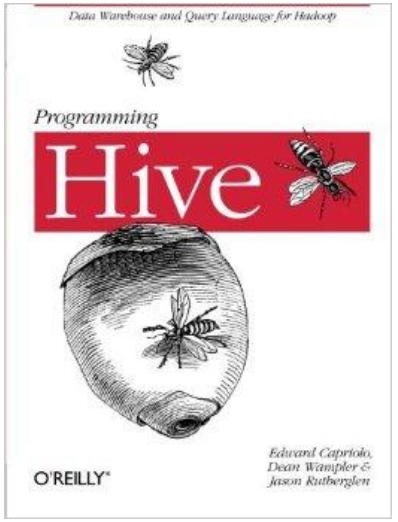


Книги



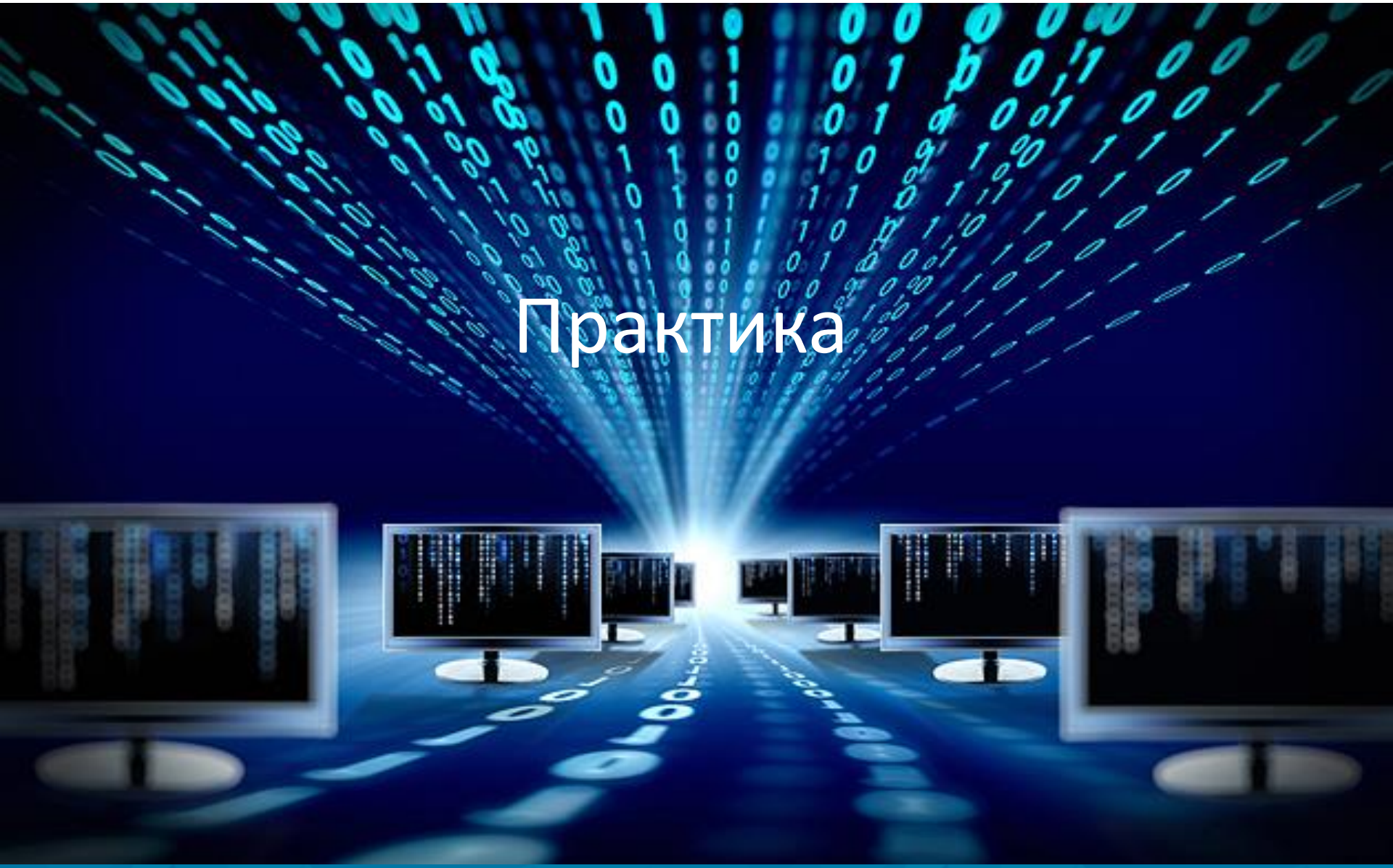
Programming Pig
Alan Gates (Author)
O'Reilly Media; 1st Edition

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



Спасибо за внимание!

Отмечайтесь и оставляйте отзыв


Практика

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
    - https://www.cloudera.com/downloads/quickstart_vms/5-12.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:
  - http://www.cloudera.com/documentation/other/tutorial/CDH5/topics/ht_wordcount1.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 к 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
```