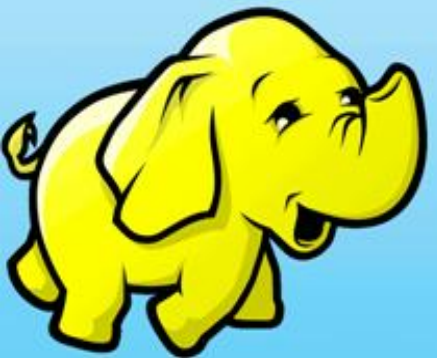
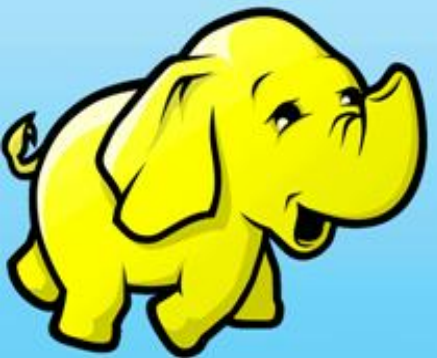
Технология Google MapReduce и ее реализация с открытым исходным кодом Hadoop



Владимир Орлов <vorl@codeminders.com>

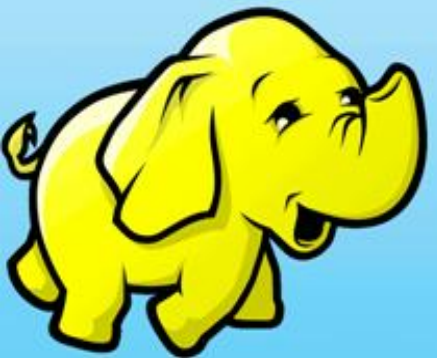
О чем я расскажу

- Технология Google Map Reduce: предназначение, суть
- Hadoop: особенности, применение
- Зоопарк Hadoop: обо всем что связано с Hadoop
- Hadoop workflow engines: Oozie, Azkaban, Hamake



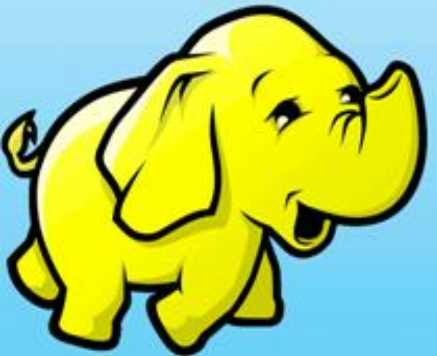
предназначение, суть

Google MapReduce



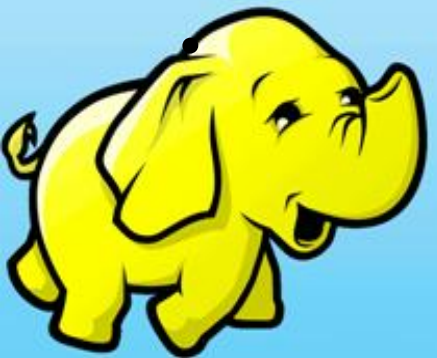
Данные...

- FaceBook: хранит на своих серверах примерно 10 млрд. картинок
- Фондовая биржа Нью-Йорка: ежедневно генерирует около 1 Тб новых данных
- Google: В 2008 году обрабатывал ежемесячно 400 ПБ
- Twitter: 55 миллионов "tweet" - сообщений в день и это только 0.5% от общего числа данных
- Yahoo: биллионы "транзакций" в день, 500M+ уникальных пользователей в месяц



Проблема обработки больших данных

- в 1990 году данные с винчестера объемом 1370 Мб и скоростью доступа 4,4 Мб/с можно было прочесть за 2,5 минуты
- у современных винчестеров скорость доступа около 100 Мб/с при среднем объеме в 1 Тб... данные можно прочесть за за 2,5 часа
- Решение?
- читать данные со ста машин одновременно, что займет 2 минуты

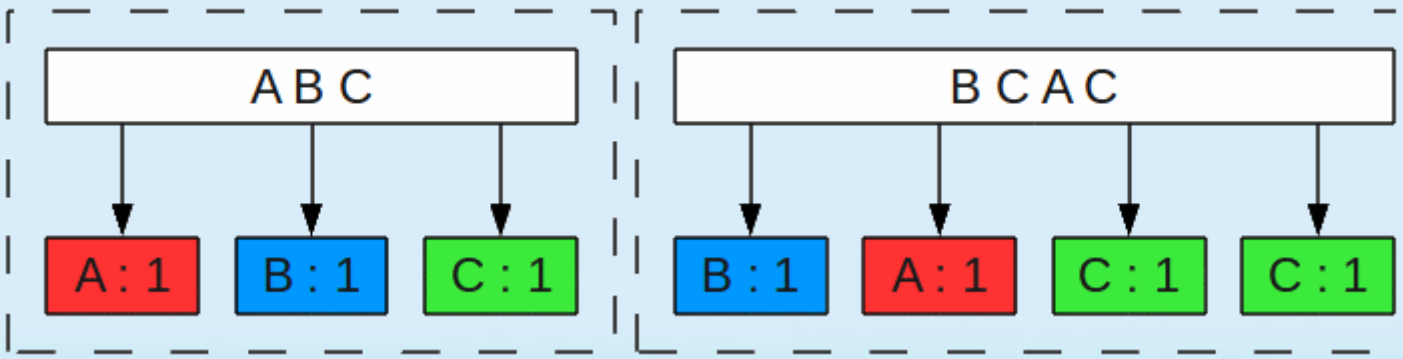


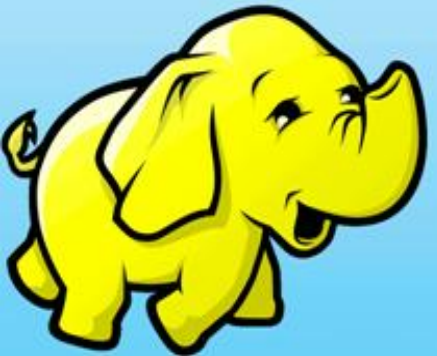
Google MapReduce

- Технология была создана в Google для сканирования и обработки веб-страниц из Internet
- абстрактная модель, которая позволяет выражать простые вычисления, в то же время пряча сложные детали параллелизации: обработку ошибок, распределение данных, балансировку нагрузки
- вы определяете функцию map, которая обрабатывает пары ключ/значение и генерирует промежуточные пары ключ/значение, которые далее агрегируются и обрабатываются в функции reduce
- Функции map и reduce выполняются на разных машинах
- очень хорошо подходит для решения распространенных задач – классификация данных, различные алгоритмы интеллектуального анализа данных (data-mining), борьба со спамом

Функция "Map"

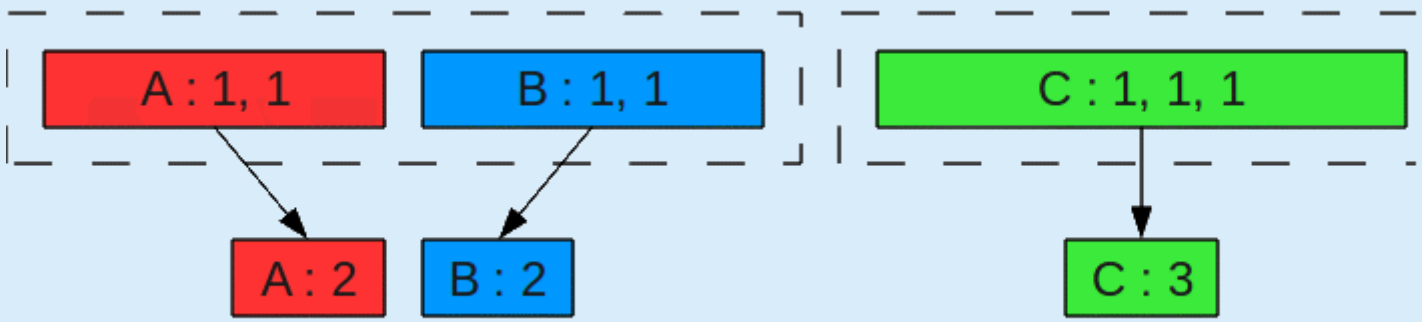
map (in_key, in_value) -> (inter_key, inter_value) list

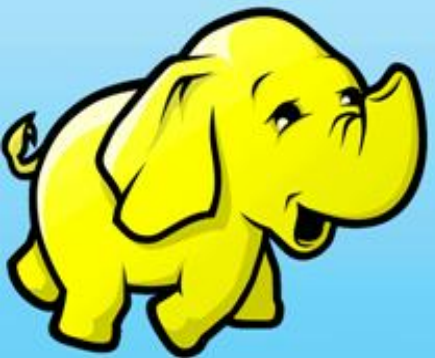




Функция "Reduce"

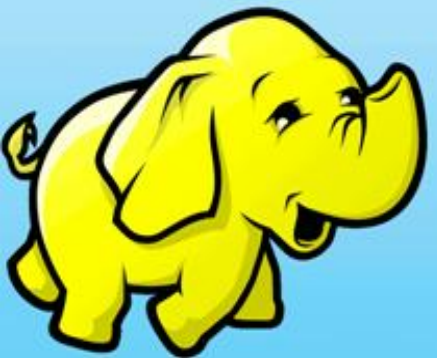
reduce (inter_key, inter_value list) -> (out_key, out_value) list





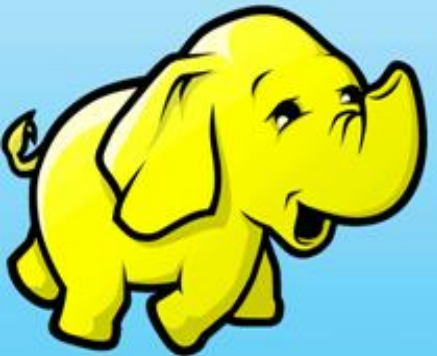
особенности, применение

Hadoop



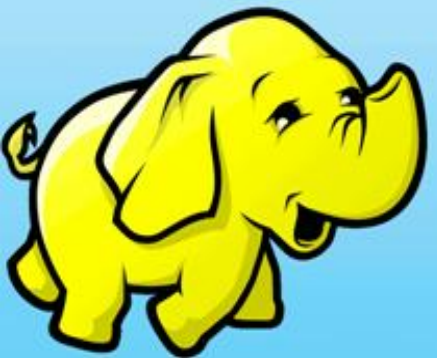
Hadoop

- Начат в Yahoo в 2004 Дугом Каттингом
- Потребность в Hadoop возникла во время разработки им системы Nutch
- Написан на Java, включает в себя:
  - 4 демона которые отвечают за работу системы
  - Скрипты для старта/остановки демонов
  - Распределенную файловую систему
  - API для написания MapReduce программ
  - Различные форматы входных/выходных данных



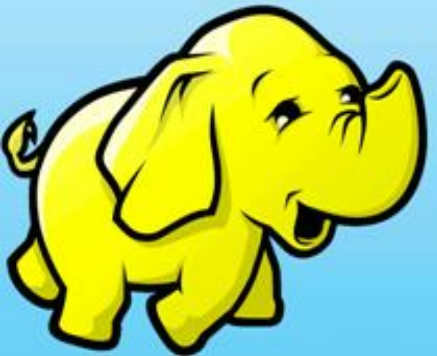
Зоопарк Hadoop

Проект в Google	Аналог с открытым исходным кодом
MapReduce	Hadoop
GFS	HDFS
Sawzall	Hive, Pig
BigTable	Hive
Chubby	ZooKeeper



Чтобы создать MapReduce программу надо:

1. создать 2 класса, наследуемых MapReduceBase и реализующих интерфейсы: Mapper (с вашей map-функцией) и Reducer (с вашей reduce-функцией)
2. сконфигурировать MapReduce-задание, создав экземпляр класса JobConf
3. передать в JobConf параметры: путь к входному файлу на HDFS, путь к директории с результатом, формат входных и выходных данных, класс с map-функцией, класс с reduce-функцией
4. вызвать метод JobConf.runJob()
5. дальше - за вас работает Hadoop



Пример "map"-функции

```
public static class Map extends MapReduceBase implements Mapper<LongWritable,
    Text, Text, IntWritable> {
     private final static IntWritable one = new IntWritable(1);
     private Text word = new Text();
     public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable>
         output, Reporter reporter) throws IOException {
           String line = value.toString();
           StringTokenizer tokenizer = new StringTokenizer(line);
           while (tokenizer.hasMoreTokens()) {
                 word.set(tokenizer.nextToken());
                 output.collect(word, one);
           }
     }
}
```

Пример "reduce"-функции

```
public static class Reduce extends MapReduceBase implements Reducer<Text,
    IntWritable, Text, IntWritable> {
     public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text,
         IntWritable> output, Reporter reporter) throws IOException {
           int sum = 0;
           while (values.hasNext()) {
                 sum += values.next().get();
           }
           output.collect(key, new IntWritable(sum));
     }
}
```

Пример конфигурации задания

```
public static void main(String[] args) throws Exception {
        JobConf conf = new JobConf(WordCount.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(Map.class);
        conf.setReducerClass(Reduce.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        JobClient.runJob(conf);
}
```

Hadoop workflow engines

	Hamake	Oozie	Azkaban
Разработчик	Codeminders	Yahoo!	Facebook
Статус проекта	Beta	Beta	Еще не было релиза
Язык описания workflow	XML	XML (основан на xPDL)	Текстовый файл с парами ключ-значение
Способ установки	не требует установки	Servlet/JSP engine + конфигурация	Servlet/JSP engine + конфигурация
Механизм определения зависимостей	основан на принципах dataflow	указывается явно	указывается явно
Версия Hadoop	0.18+	0.20	0.18+
Модель работы	коммандная строка	демон	демон

Дополнительная информация

- Hadoop: http://hadoop.apache.org/
- CloudEra: http://www.cloudera.com/
- Hamake: http://code.google.com/p/hamake/
- Книга Hadoop: The Definitive Guide
- Тренинг в Киеве в конце июля (дополнительная информация по адресу <vorl@codeminders.com>)