MapReduce big data computing 的 框架 -- 很多公司 都会用 Map Reduce 框架 -- 要不就用 MapReduce，要不就用 Spark

统计单词的出现次数：

* 应用： 微博 热搜词（根据词出现的统计次数， 统计 top K （前10个））。这里先做 第一步应用，统计单词的出现次数
* given a line 1“HELLO HOW ARE YOU” line 2 “”I am fine you?“”
* 土方法一： HashMap<String, Integer> wordCount; Foreach word in webpage: for 循环 wordCount[word]++; 这个方法 很简单， working solution，如果要统计整个微博的热搜词， 恐怕就不行，要实时更新 热搜词， 就要用多台机器解决
  + 限制： 只能在一台机器上处理，如果统计一个网站的热搜词，会比较慢
* 土方法二：多台机器For 循环
  + 第一台机器处理第一行的句子，for循环
  + 第二台机器处理第二行的句子，for循环
  + 机器三来汇总单词词频，汇总输出
  + 问题： 机器三会成为**瓶颈**，所有的机器For循环处理得结果汇总到机器三，汇总的时候并没有并行**汇总**， 仅仅是并行**处理.**如果 是 1000台机器处理1000行，则最后交给1台机器去汇总，其实并不是 in parallel ， 而是 sequential execution
* 方法三：  **处理 和 汇总 都是 并行的 Map Reduce （处理 和 汇总 都是 并行的）**
  + 处理时 多台机器 处理
  + 汇总：多台机器进行汇总时，进行 data sharding，对于所有跟字母 a, b相关的，对应到 Machine3汇总； 对于所有跟c,d 相关的，对应到Machine4进行汇总
    - Map（分）: 拆的过程。利用Hash将数据分开，把data拆分成一个一个单词
    - Reduce（和）： 和的过程。分别进行汇总，合并
  + MapReduce 框架 有 如下 6个阶段：
    - Input： 输入文件 (0：ABACDD, 1：ABCCDB)
    - Split： 把文件拆分： （e.g. 拆成一行一行， 一个机器 处理第一行，一个机器处理第二行）
    - Map： Map成每个单词出现的次数
      * Map时是否会进行小汇总？
        + e.g. 在当前行， a出现了2次，b出现了3次。如果这样小汇总，需要使用HashMap,可能会耗费大量的**内存**。
        + Instead，map机器并不关心当前机器中当前行每个单词共出现几次，看到a出现了一次直接transfer到reduce机器上 a = 1， b = 1, a = 1, a = 1, c=1...d=1…....
    - Transfer: 传输到各个用于汇总的机器 a b 传输到 机器 1， cd 传输到 机器2
    - Reduce： 分别进行汇总 a 的 word count
    - Output
  + 面试时会出现什么code challenge? 实现什么内容：
    - 其实 就是 实现Map & Reduce 函数， MapReduce framework 会 提供 6个步骤，我们只需要提供 map function （如何将数据分开） 和 reduce function（如何 将数据合并）
      * 实现 Map 函数
      * 实现 Reduce 函数
      * 定义Input Output： 输入/输出必须是 Key, value 形式
        + 输入： key 文章地址 value 文章内容
        + reduce 输入： key： map 输出的key value：map输出的value
        + 输出：输出流控制，输出到 硬盘里面
    - Map方法1（不太好的方法）：HashMap 的方法：几乎不care总输入的key，而 输入的value 是有用的。保存HashMap的方法：for 循环加入 hashTable， aggregate之后 output.collect(key, count);这样不太好，如果文章特别大，就很容易超过限制 （XMLReader Coresight Server就是个悲催的例子）。输出是 流控制: OutputCollector<String, Integer> , load一部分在内存中，最终是存储在硬盘中
    - Map方法2：直接输出： output.collect(key, 1); 每个单词 出现的 次数
    - Reduce： 每次load 一小部分 从 硬盘 到 内 中。不会 全部load 到 内存中。用Iterator load 一部分到内存中，统计一下，然后输出。
  + MapReduce 去做统计类操作 会 比较容易，做 sort 这样的操作会麻烦一点。
  + 在现实工程中， Map/Reduce 机器数量 可以自己控制， 统计10T数据词频， 起了1000个 map，1000个 reduce，跑了2/3小时
  + 可以在 Hadoop 官方网站 开 很多线程 模拟这个过程。
  + 机器越多越好吗？
    - 机器越多，总处理数据就越快
    - 但是启动时间会变长
    - 不考虑启动时间， Reduce的机器越多，一定越快吗？
      * Key的数目就是reduce的上限
      * 比如，统计 数据里面 男生 几个 女生几个，最多就3台机器。所以key的数目决定了reduce的上限。Map Reduce 和 GFS 的 结合？ 下半节课讲
  + 传输 / 整理 详细操作：
    - 问题描述，map 端有 很多机器： 每篇文章 有 单词/1次， 传输 到 reduce 机器 上进行 aggregate
    - Map端： 传输前 先做 Partition & Sort
      * map 端 Sort的原因是什么？
        + 如果不在map端小排序，就没法再 reduce 端 做 merge K sorted lsit，就要在reduce端做大外排序，
        + 为什么map 端 数据量小于 reduce 端 数据量：因为Map端机器数量可以>> reduce端机器，map的上限跟key无关 可以按 行均分，reduce端的上限就是key。Reduce端不能做均分（比如the这个词，数据量很慢），所以数据量会很大，map端可以均分（按照行）
      * 排序方法： 外排序
        + a, b 先整理到一起，传输到机器3去Reduce
        + c，d整理到一起，出书到机器4去Reduce
        + 每次产生 a, 1, 都不是写入到内存，是写入到硬盘中，使用外排序进行排序

因为内存很小，数据很大，都在disk中，不能用Quicksort或mergesort，需使用外排序进行排序 （课后看wiki， external sort）

* + - Reduce端Fetch and then merge
      * 限制： 内存很小，disk data量很大
      * merge的方法：
        + 使用堆 PriorityQueue， 方法类似于 Merge K Sorted List 这道题
    - Suffle 其实就是 map端的 external sort 和 reduce 端的 merge k sorted list

做项目： Hadoop： 情感倾向project: Sentiment Analysis Project on IntelliJ

MapReduce 应用：

* inverted index: 倒排索引 - 在搜索引擎中用的非常多

（正排索引： car单词出现在书1和书2中: 1-> car, river。 2-> deer, car）

倒排： {car： {1， 2}}

搜索引擎就是这么找（九章 这个关键字 出现在 哪几个文章中），倒排索引 解决了 找的 问题， 具体 怎么排序由 另外 的 算法 解决

* 如何定义Map的output key:
  + key: word
  + value: 文章id
* Reduce的output：
  + key: Bear
  + value: id0, id2
* 如何 shuffle 哪个map 机器 到 reduce 机器， 这个由 map reduce framework 用consistent hashing 都做好了。
* 如何去重：
  + 比如Map 机器上 River ： 2 出现了两次
  + 使用HashTable去重不好，可能会出现内存不够用
  + 在Reduce端使用for循环去重，因为Map Machine已经Partition and sort了，在reduce端 进行merge k sorted list 的时候 直接可以用 for 循环解决
* 所有reduce机器等 map 机器之后运行

Anagram - Map Reduce (lint, intl, inlt, abc, 返回 [lint, intl, inlt])  
 - Map输出的key是什么： sorted keyword， value是original keyword, 我们希望 key 相同的，挤在一堆

* Map端: 拿到单词，sort 生成key， output
* Reduce端， 拿到key,aggregate

Interviewer： Design a MapReduce System 帮助理解 map reduce 原理，不会考 整个系统怎么工作的

* Input Data: 计算word cnt 时，有很多web pages
* Output Data: 输出到 /home/word
* 主函数 会 配置 map reduce 参数 ，然后 run，
* Given很多台机器：使用什么框架来管理 Master / Slave
  + Master 去assign slave的工作： 做为Map机器还是Reduce机器
* Master用来拆分文件，分配给Map （只是需要 配置 map Input format （key value pair））
* GFS和MapReduce的关系：MapReduce framework 的总的 Input / output 都是放在GFS里面。
  + 多台机器的分布式读取（High I/O）速度够快
  + 存储够大
* Map端的输出是写在local disk还是GFS里？
  + 数据大吗，数据怕被损坏吗，数据写入快吗？有没有必要对 数据做 replica。被破坏后无所谓，重新做一下就好。一台机器的输出，再大大不到哪儿去。输出到本机比输出到 GFS快的多
  + 写在local disk （因为写入local disk比写入DFS快，不需要考虑大数据问题 （因为一个机器就能搞），无需考虑破坏的问题（因为非关键数据））
* 每个Map端的输出会进行Partition和外排序，Reduce端进行Remote Read and merge sort
* 最终的输出是否有瓶颈 （都输出到/home/word里）
  + 分别输出到不同directory（home/word/output0\_of\_2, home/word/output1\_of\_2）
* GFS机器和Map Reduce机器是否是一台：
  + 不是。 GFS和Map Reduce是完全不一样的机器。GFS是由很多台机器组成的 服务器群。
* 为什么GFS不怕被破坏：
  + GFS有replica / 和 duplicate
* Mapper和Reducer是同时工作还是Mapper先工作？
  + Mapper要结束了之后Reducer才工作
* 运行过程中，Mapper或Reducer挂掉了怎么办？
  + 备胎池：master 在启动整个服务的时候，会多启动其他机器。使用备胎机器替换挂掉了Reducer/Mapper
* 一个Reducer的key特别大怎么办？
  + 如果map的输出key是一个网址： [www.facebook.com,](about:blank) 对应的 key value pair 特别多 （假设mapReduce计算的是 访问量，这个 网址访问量特别高）
  + run一遍，发现问题，改进
  + 拿word count作为一个例子： 在Map端，如果key.equals(facebooks.com), 给key加一个random后缀。 平分到Reducer机器上。
* Mapper和Reducer可以是同一台机器码？
  + 有可能。
* 另外两个例子： Top K 问题：微博热搜词， 照片选出前几个， 看short URL
* PageRank： 看short URL

MapReduce六个步骤很重要，需要破解Map和Reduce

* 多台机器如何管理：
  + AdHoc： 读cassandra 的文章
  + Master / Slave, 有大哥，有小弟 GFS的实现方式
  + Map Reduce 的输入很多可以 是 GFS， BigTable 或者 Cassandra