1. 海量日志数据，提取某日访问百度次数最多的那个IP？

分而治之+hash：

①IP地址最多有2^32个，总大小为4G

②按照IP地址的HASH（ip）映射到1024个小文件中，每个文件4M

③对于小文件可以找到频率最高的IP

④得到1024个出现较多的IP，再排序一下得到访问最多的IP

1. 有一个1G大小的文件，里面每一行是一个词，词的大小不超过16字节，内存限制是1M。请返回频率最高的100个词？

顺序读文件，对每个词去hash(x)%5000，得到5000个小文件。每个文件大概200k，如果文件中有超过1M的，可以继续hash下去分直到所有小文件都<1M。

对于每个小文件统计词&词频（trie树或者hash表），并取出频率最大的100个词（可以用最小堆），并把100个词存入文件，这样就又得到5000个文件，下一步就是对5000个文件进行归并（类似归并排序）。

1. 有10个文件，每个文件1G，每个文件的每一行存放是都是用户的query，每个文件的query都有可能重复。要求按照query的频率排序。

方案①顺序读取10个文件，按照hash(query)将结果写入10个文件中，大约1G，找一台2G内存的服务器对其进行query频率统计并排序，重新输入到新的10个文件中，最后对这10个文件进行归并排序（内排序+外排序）。

方案②一般来说query的总量是有限的，只是重复的很多而已，对于所有的query，可以一次性加载到内存，这样就可以用hash表或者trie树来直接统计query频率，最后按照频率来排序就好了。

方案③mapreduce

1. 在2.5亿个整数中找出不重复的整数，注意内存不足以容纳2.5亿个整数？

方案①位图2-Bitmap（每个数分配2bit，00表示不存在，01表示出现1次，10表示多次，11表示无意义），共需要2^32\*2bit=1G内存。

方案②也可以采用划分小文件，在小文件中找不重复的整数，然后进行归并。

5.搜索引擎会通过日志文件把用户每次检索使用的所有检索串都记录下来，每个查询串的长度是1-255字节。假设目前有一千万个记录（重复度比较高），请统计最热门的10个查询串，要求使用的内存不能超过1G。

TopN的算法，可以用最小堆来维护，而最小堆需要的是key和频率，频率怎么来？可以用hash，可以用trie树，本题有大量重复的记录，所以可以在内存中直接计算频率，如果出现内存放不下的情况，可以采用划分小文件的方式获取词频。

1. 给定a、b两个文件，各存放50亿个url，每个url占64字节，内存限制是4G，请找出a、b文件共同的url？

50亿=5000000000大约=5G个字节，所以一共5\*64=320G，肯定不能加载到内存。采用分而治之的策略。

方案①对文件a、b分别进行hash映射为1000个小文件，大约300M，由于hash映射所以相同的文件只能存在于相对应的文件对中，针对每一对文件对，把一个文件放到set中，遍历另一个文件即可确定重复的url。

方案②如果允许一定的误差，可以采用bloom filter，4G的内存可以容纳的bit数是340亿，将文件a中的url映射到这个位数组中之后，然后挨个读取文件b中的url，判断是否重复（有一定的错误率）

1. 给40亿个不重复的unsigned int整数，没排过序，再给一个数，如何判断这个数是否在那40亿个数当中？

一般的思维是：快排+二分

方案①int整形的范围是-2\*\*31-2\*\*32-1，所以unsigned int整形的范围是0-2\*\*32，而2\*\*32大小是42亿，而512M内存=512\*1024\*1024\*8=42亿，所以512M正好存储40亿个unsigned int整数，读取40亿个数，在对应的bit上设为1，当新的数进来时就可以判断是不是重复了。

方案②将这40亿个数中的每一个用32位表示（这是肯定的），一开始全部都在一个文件中，接下来将这40亿个数分为两类：根据最高位是0还是1，这样将分成两个20亿数据量的 文件了；接着根据这个新的数的最高位跟那个文件吻合再决定下一步跟那个文件对比，同样的套路根据次高位是0还是1分为10亿数据量.......最后logn的次数判断是否存在。

十个海量数据处理的方法

1. Bloom Filter

适用于实现数据字典，进行数据判重或者集合求交集

1. Hashing——分而治之

快速查找，条件是总数据量可以放进内存

1. bit-map

数据快读查找、判重

可以认为bloomfilter是bit-map的一种扩展

1. 堆

用于海量数据topN，可以放入内存

1. 多层通划分——分而治之

使用于第K大，中位数、不重复或重复的数字

和上面的hashing类似，只不过多层通可以用于数据分布不均，其实可以认为是多层文件分割，比如上面的问题6的方案②就是多层桶的应用，对于求中位数也是一样，按照二进制的高位比较分割文件直到文件可以加载到内存+排序

1. 数据库索引

大数据量的增删改查

1. 倒排索引

搜索引擎、关键词查询

1. 外排序

大数据排序去重，比如归并排序，排序会专门整理

1. trie树

使用数据量大、重复多的字符串时即数据种类小可以放入内存，可以轻松存储查找计数

1. 分布式mapreduce