处理大数据问题：离不开分流，要么是哈希函数把大文件的内容分配给不同的机器，要么是哈希函数把大文件拆成小文件，然后处理每个小数量的集合

1.有一个包含20亿个全是32位整数的大文件，在其中找到出现次数最多的数（内存限制2GB)

把包含20亿个数的大文件用哈希函数分成16个小文件，根据哈希函数的性质，同一种数不可能哈希到不同小文件上，同时每个小文件中不同的数一定不会大于2亿种，假设哈希函数足够好，然后对每一个小文件用哈希表来统计其中每种数出现的次数，这样就得到了16个小文件中各自出现次数最多的数，还有各自的次数统计，接下来只要选出16个小文件中各自的第一名中谁出现的次数最多即可

分为16个小文件的原因：由于哈希表的key为4B，value也是4B，那么一条记录需要占用8B，当记录数为2亿个时至少需要1.6GB的内存，限制内存为2GB，所以分为了16个小文件

2.包含40亿个非负整数的文件中中找到没出现的数（内存限制1GB)

32位无符号整数的范围是0-4294967295，所以必然有没有出现的数
使用bit map的方式，申请一个长度为4294967295（2的32次方）的bitArr，bitArr上每个位置只可以表示0或者1状态，8个bit为1B，所以数组占用500MB的空间
遍历这40亿个数，如果遇到7000，就将bitArr[7000]设为1，遇到所有数时就把相应位置上的值设为1，最后遍历这个数组，哪个位置上没有设为1，哪个数没有出现

进阶：当内存限制为10M时

1.根据10MB的内存限制，确定统计区间的大小，就是第二次遍历时的bitArr

把数的范围平均分成64个区间，申请长度为64的整形数组遍历40亿个数统计每个区间上数的个数，当遍历完后，遍历整形数组，如果某个位置上的值小于67108864，表示该区间至少有一个数没出现过
2.找到计数不足的区间，申请区间大小的bit数组，再遍历这40亿个数，对不足区间做映射将相应值设为1，
3.最后再遍历bit数组，找到没出现数即可

3.布隆过滤器

不安全网页的黑名单中包含100亿个黑名单网页，每个网页的URL最多占用64B，设计一个网页过滤系统，根据网页的URL判断该网页是否在黑名单上（限制：允许有万分之一的判断失误率，额外空间不超过30GB)

类似问题：垃圾邮件过滤系统、爬虫网址判重系统（空间要求比较严格，系统容忍一定程度失误率）
布隆过滤器的优势：使用很少的空间将准确率做到很高
思想：长度为m的bit类型数组，k个哈希函数（函数输出域都大于等于m）
对每一个输入对象都经过k个哈希函数计算，算出的每一个结果都对m取余，然后在bit 数组中将相应位置设为1（至此完成了100亿个url到bit数组上的映射）
检查阶段时，假设一个对象为a，把a通过k个哈希函数计算k个值，然后取余，看得到结果的相应位置上是否为1，如果有位置不为1则不在该集合中
缺陷：可能误判，不在该集合中的对象误判为在该集合中（原因可能是集合中输入对象过多，bitMap过小 导致绝大多数位置都已设为1），但不会漏掉
确定k和m的方式：
m=-(n*lnp)/(ln2)2(2代表平方)
k=ln2*m/n
所以用25GB的bitMap再单独实现14个哈希函数即可
公式的由来：是根据检查k个位置都为黑的概率得出失误率与k的函数

4.一个包含100亿个URL的大文件，每个URL占用64B,找出所有重复URL

可以采用哈希分流的方式，将100亿字节的文件通过哈希函数分配到100台机器上，然后每台机器分别统计是否有重复的URL，或者在单机上将大文件哈希函数拆成1000个小文件，对每个小文件再利用哈希表遍历找出重复的url,或者再分给机器活着拆分完文件后进行排序，排序后看是否有重复的URL

5.某搜索公司一天搜索词汇是海量的（百亿数据量），设计求出一种求出每天最热top100词汇的可行办法

把包含百亿数据量的词汇文件分流到不同机器上（具体几台根据限制而来），如果分到的数据量依旧很大，再用哈希函数把每台机器上的分流文件拆成更小的文件处理，利用哈希表处理每个小文件统计每种词及词频，再遍历哈希表，遍历哈希表的过程中使用大小为100的小根堆来选出来选出每一个小文件的top100,将小根堆里的词按照词频排序，就可以得到每个小文件排序后的top100，然后再利用外排或者继续利用小根堆选出每台机器上的top100，然后再继续外排或者利用小根堆选出整体中的top100

6.一致性哈希算法原理

数据缓存设计方案：将数据id通过哈希函数转换成哈希值的范围是2的32次方，将这些数字首尾相连构成一个闭合的环形，则一个数据id对应环中一个位置，机器在环中的位置通过机器id计算得出的哈希值决定，每条数据就归属到顺时针找离它最近的机器

这样当删除或者添加机器时调整代价是较小的

但当机器负载不均时，可以引入虚拟节点机制，即对每一台机器通过不同的哈希函数计算出多个哈希值，每一个位置上都放一个服务节点