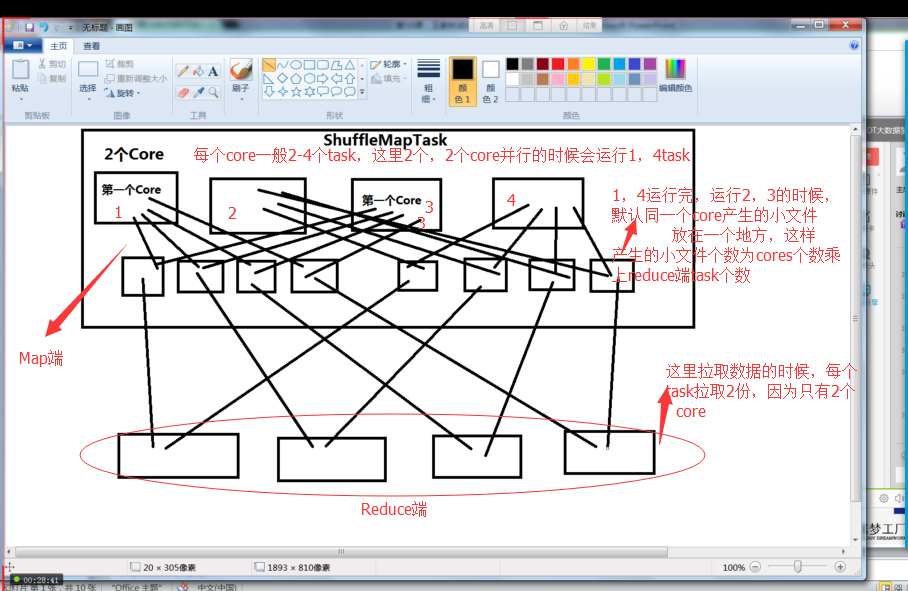
一：Shuffle性能调优：因为不同的shuffle方式，有不同的具体的性能优化的差异点。所以本节课追求通用性，适用于任何方式的shuffle。据说spark钨丝计划可以把spark性能提升6倍左右，从根本上解决GC的问题。spark sql上写sql语句和写操作RDD的话，它底层都与钨丝计划有关系，

1. 问题：Shuffle output file lost？真正的原因是GC导致的！！！就是下一个stage找上一个stage要数据，读数据肯定耗cpu，lost就是读不到数据。为什么读不到呢?有可能是磁盘有问题，但概率不是很大。read的时候，下一个stage都会向上一个stage去请求线程，如果GC尤其是Full GC产生通常会导致线程停止工作，就请求不到数据，这个时候下一个Stage的Task在默认情况下就会找上一个stage尝试重试来获取数据，一般重试3次每次重试的时间为5s，也就是说默认情况下15s内如果还是无法抓到数据的话，就会出现Shuffle output file lost等情况，进而会导致Task重试，甚至会导致Stage重试，最严重的是会导致App失败；在这个时候首先就要采用高效的内存数据结构和序列化机制、JVM的调优来减少Full GC的产生，新生代，老年代等内存的调整；



1. 在Shuffle的时候，Reducer端获取数据会有一个指定大小的缓存空间，如果内存足够大的情况下，可以适当的增大该缓存空间，否则会spill到磁盘上，影响效率。

此时可以调整（增大）spark.reducer.maxSizeInFlight参数；这是一个reduce中的reduceTask占用的缓存空间大小。默认48M，一般调到128M以上。具体根据内存和任何进行计算，一般越大越好。同理，Mapper端的shuffleMapTask写数据到磁盘上也有缓存空间。不会有一点数据就立即写的。

1. 在ShuffleMapTask端通常也会增大Map任务的写磁盘的缓存，默认情况下是32K；

spark.shuffle.file.buffer 32k，为什么默认情况下这么小呢？考虑在最小的硬件情况下都把它部署成功。

1. 调整获取Shuffle数据的重试次数，默认是3次，通常建议增大重试次数8-10次；

调整获取Shuffle数据重试的时间间隔，默认5s，都在spark-conf或者配置文件中配置。强烈建议提高该时间，spark.shuffle.io.retryWait 5s

1. 在reducer端做Aggregation的时候，默认是20%的内存用来做Aggregation，如果超出了这个大小就会溢出到磁盘上，建议调大百分比来提高性能；