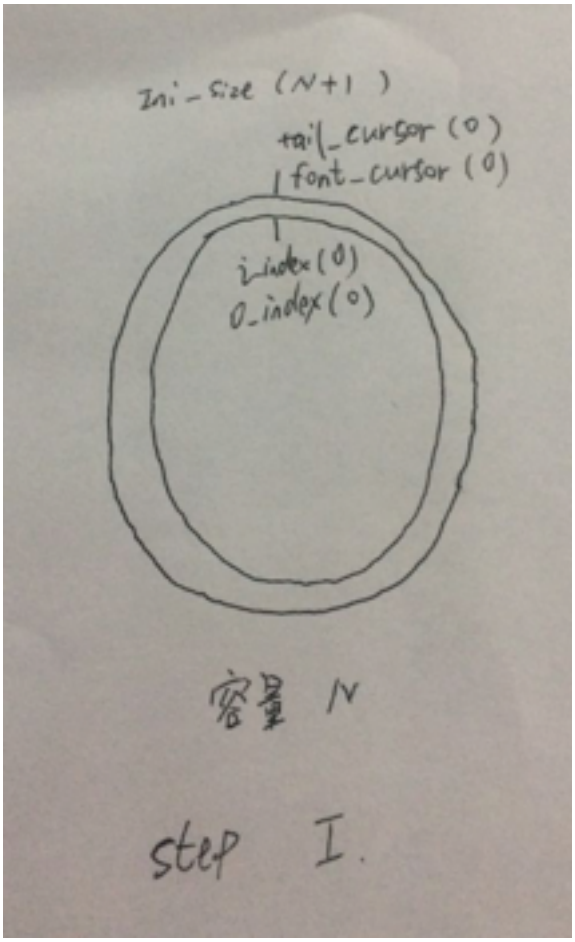
一种并放队列的设计
1，设计背景
2，原理
3，该设计能适应什么场景
4，还有一些改进的地方

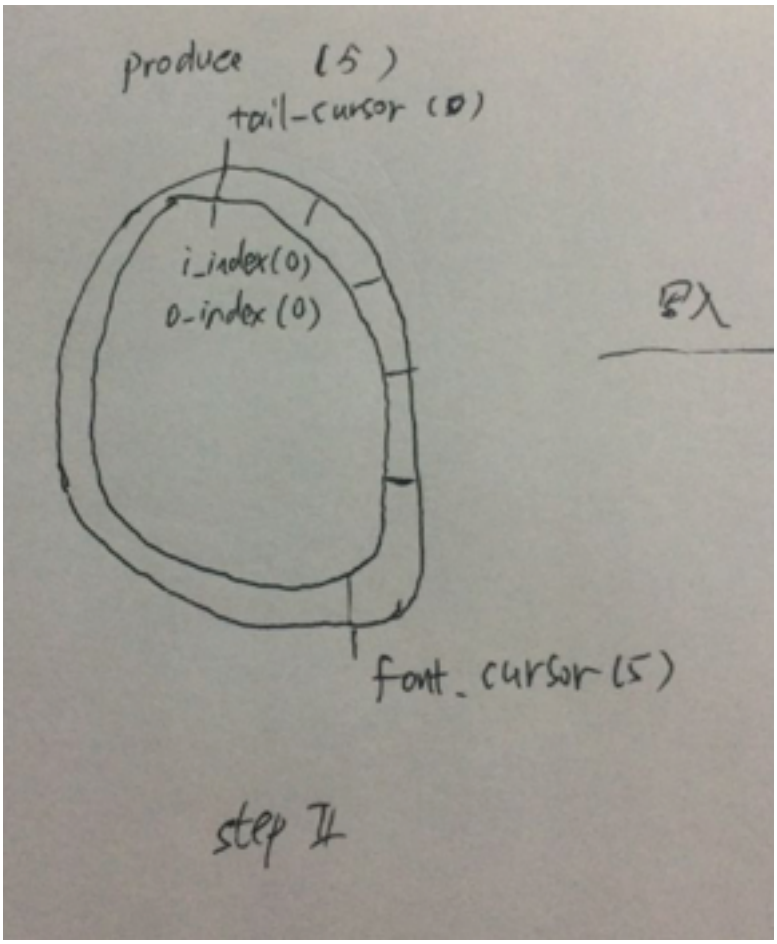
源码地址 ： https://github.com/randomprin/tutorial/tree/master/demo

1，为支持服务器多线程的开发，设计了该并发队列。设计灵感来源于互联网。1，阿里巴巴的技术博客的环形设计（reachtb，貌似关了，核心的思想也许和他一致，但是我现在考证不了。）。2，无锁队列的概念（非内存屏障，只是cas）。3，mapreduce的一些类比思想

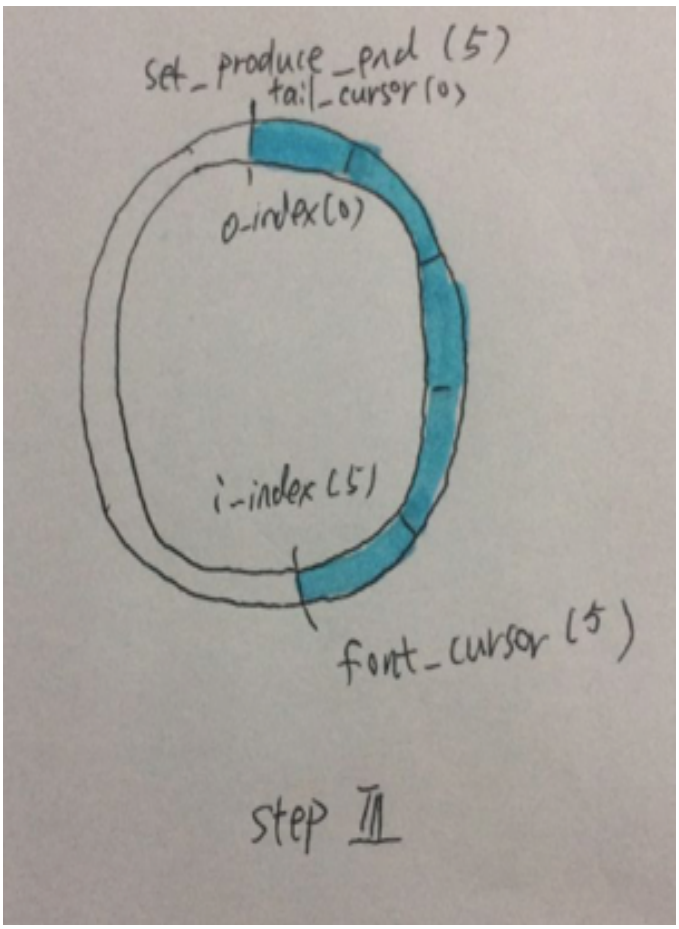
2，在设计中有两类操作，读和写！而每一种操作有两种状态，预备 / 执行。而锁设计在预备阶段。具体如下图：



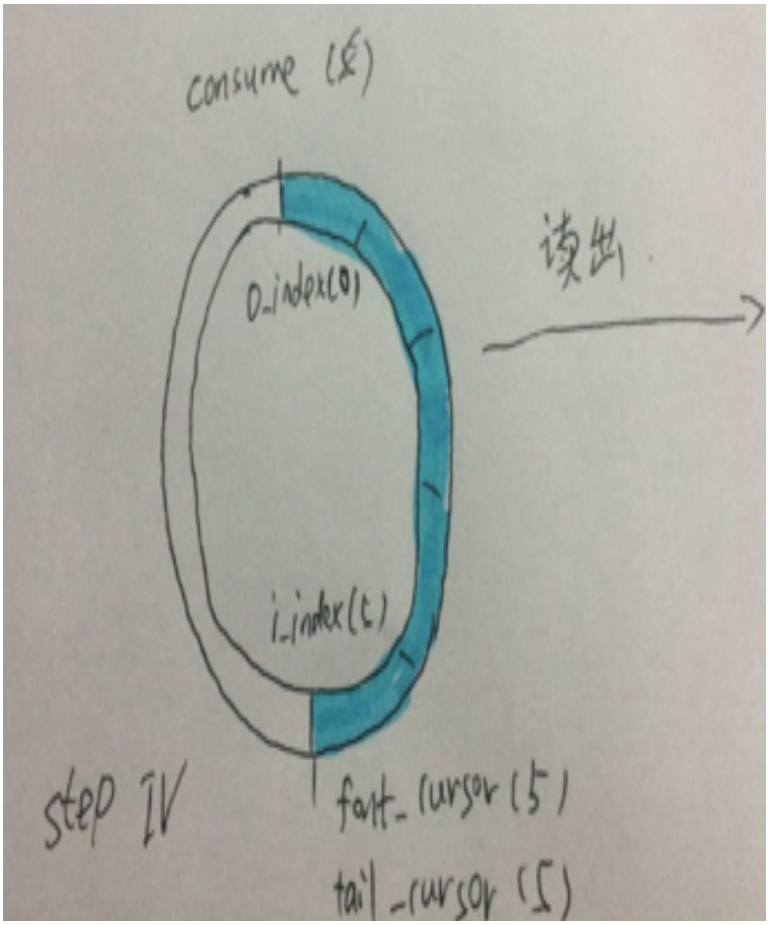
第一步初始化一个容量为（N＋1）的环形内存，实际大小为N，因为空和满标记位重合了，很有无中生有的味道。



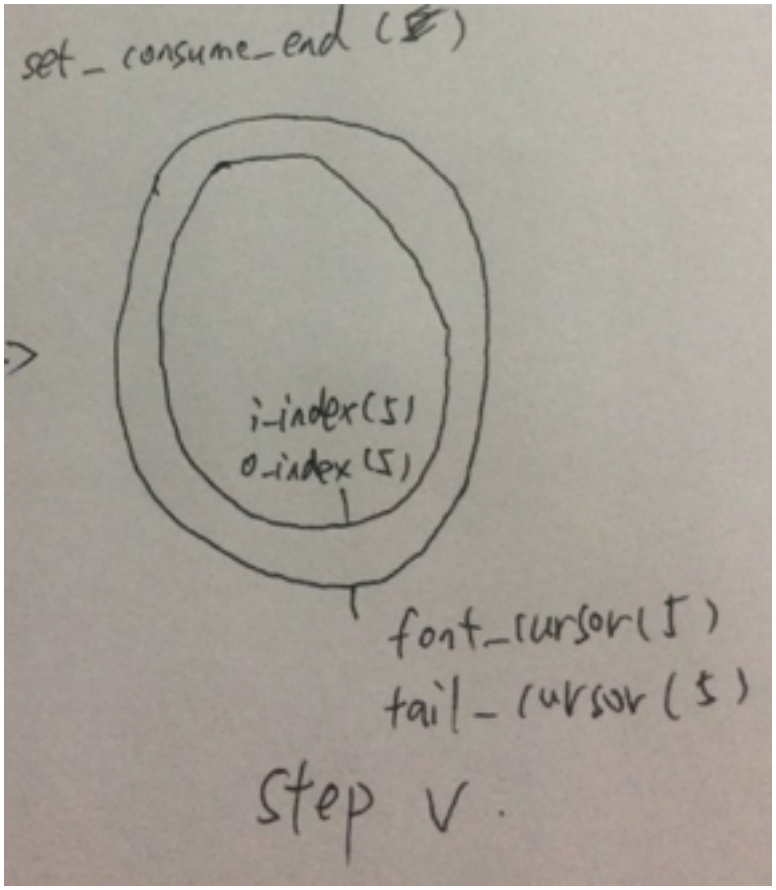
第二步，生产线程请求到达，预分配5个内存给该线程，四个原子数的变化如上图，并且生产线程拿到内存的首地址和唯一的一个token（font_cursor）.



第三步：线程写完毕后，把唯一的token归还给环，原子数变化如上图。



第四步消费线程去请求内存的内容，消费内存拿到需要处理内存的首地址和消费的唯一token（tail_cursor）



第五步消费线程完成处理之后归还token，原子数变化如上图。

上面2，3步是一组，4，5步是一组，可以由任意组，任意序列同时处理，唯一不太满意的是第4步有个环的次序（tail_cursor依次增大，不可以跳，如果当前位置的写卡住了，即使之后的位置已经完成写操作，那也要等到之前的那个位置写完成。），
上面几幅图的索引都应该为4，在这边说明下。

3，场景具有几个特点，1，大量无序写 / 请求。2，写 / 请求操作存在逻辑单元，那些单元相对独立，彼此竞争。3，对逻辑单元的信息处理相对耗时。4，一个定长内存池。
比如web服务器 / 游戏服务器 / 物联网的服务器。在我的想象中也应该能模拟网站的秒杀买卖的并发。

4，　　1，多余的参数要去掉。
　　2，命名规则（如果能忍受的话可以忽略）
　　3，加入线程信号量通知，最好再封装一组线程池，消费线程等待通知，而不是轮询。
　　4，读出的话有多少读出多少可以改进成定量读出
　　5，写入可以多条线同时写入，但是标记写成功缺必须按照写入动作的次序完成，这部分有两种优化方式，一个是写入的颗粒小一点，这样不会因某个大量的写入而堵塞后续的操作（类似拆分大sql语句），二个是把标记位分段存储类似邻接表（但是这个也有问题，涉及标记的合并处理，一旦写入的颗粒很小，那么合并处理和读的处理势必被其所累，可以根据具体的业务具体设计。）