实验报告

实验环境

论文实验环境:

Intel 2.8GHz 16-core CPU, 12GB DDR3 RAM with apeak bandwidth of 32GB/s, and a 500GB hard disk. TheL1 and L2 caches of the CPU are 32KB and 256KB

本次实验环境:

ubuntu kylin 14.04

Intel 2.50GHz 8-core CPU, 15.5GB DDR3 RAM with apeak bandwidth of 32GB/s, and a 1000GB machine disk. TheL1 and L2 caches of the CPU are 32KB and 256KB

实验原理：

**traces**: 三个数据集模拟hash调度

RandomInteger : 由生成器算法生成，算法源码作者没有说，只有一个相关的论文，在github找了一下，应该是这个<https://github.com/cslarsen/mersenne-twister>

DocWords： <http://archive.ics.uci.edu/ml/datasets/Bag+of+Words>

MacOS： <http://tracer.filesystems.org/>

**algorithm**: 四个算法对比说明本次算法性能

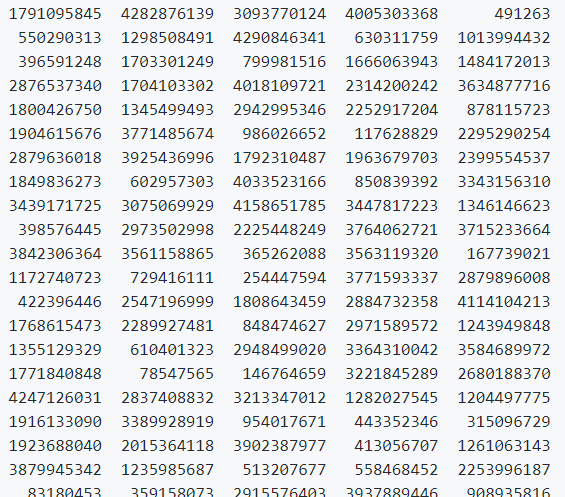
SmartCuckoo： <https://github.com/syy804123097/SmartCuckoo>

libcuckoo：<https://github.com/efficient/libcuckoo>

CHS (cuckoo hashing with a stash): 作为参照算法

BCHT 算法:

对于RandomInteger, 生成的随机数如下图



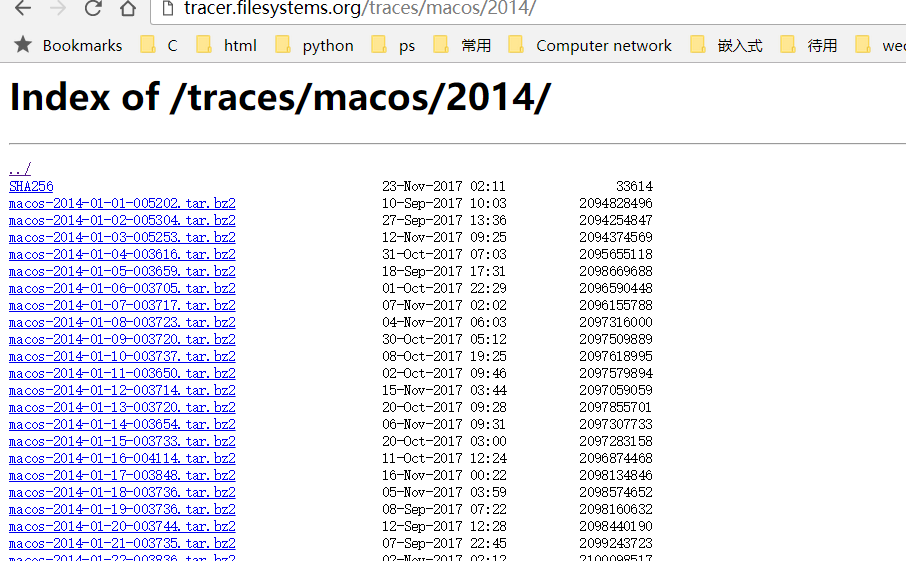
DocWords把DocID和WordID结合起来作为键插入hash table, MacOS将文件内容的md5加密值作为键插入到hash表。

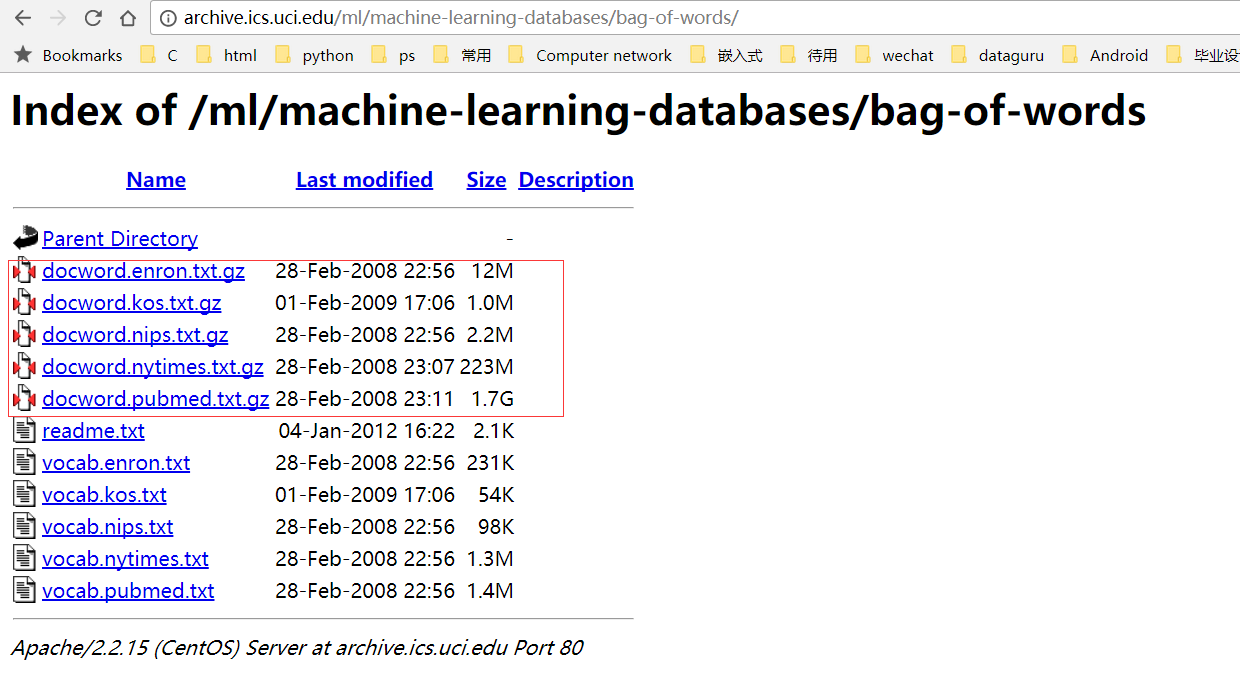
实验过程：

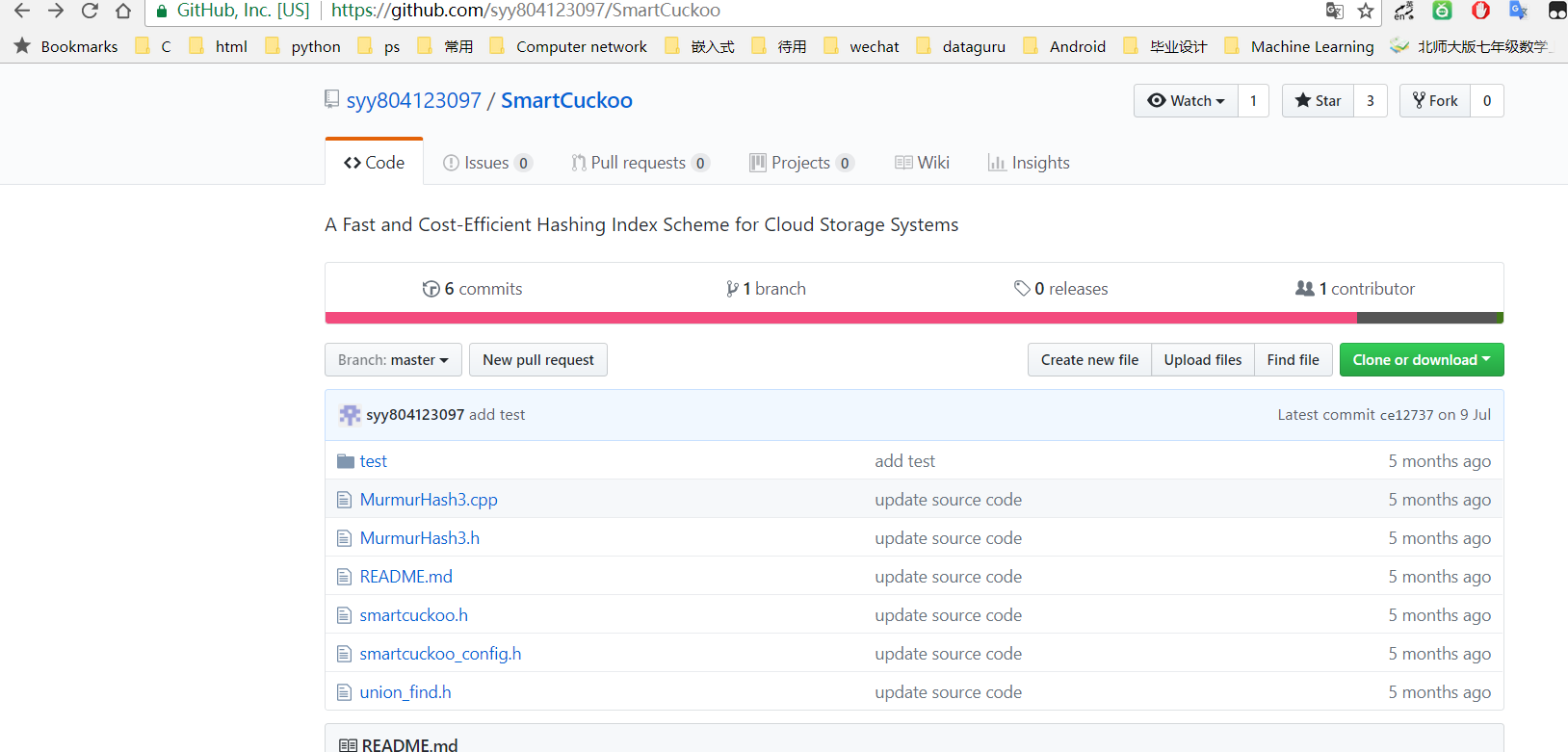
Linux 系统是以前装的，本次执行了

sudo apt-get update && apt-get upgrade

论文作者没有说数据集怎么用，加上数据集有点大，所以本次实验就Macos下载了一个2014年6月的一个，Docwords五个txt.gz都下载了，RandomInteger需要自己生成，不知道怎么利用到实验中…

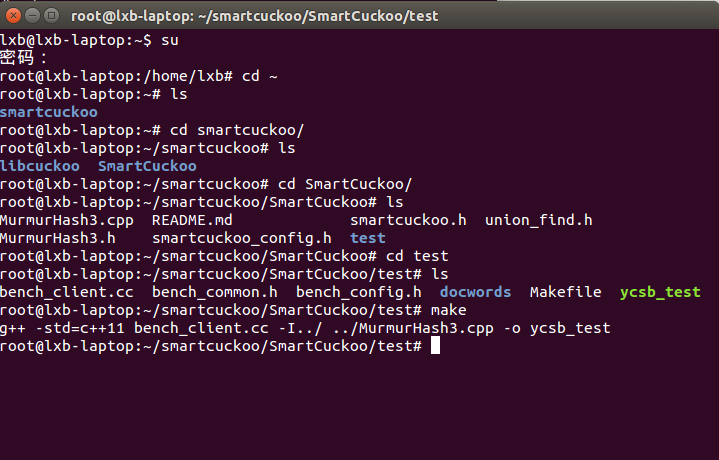


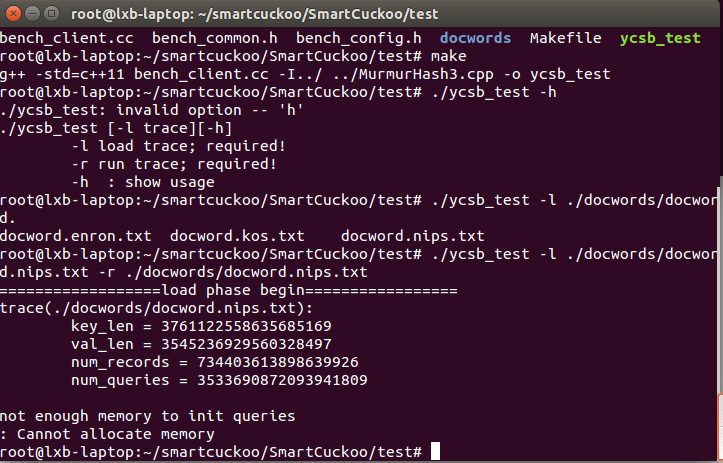


本次实验只实验了smartcuckoo和libcuckoo这两个开源的算法, 另外两个由于时间问题和个人能力问题尚未实验: 

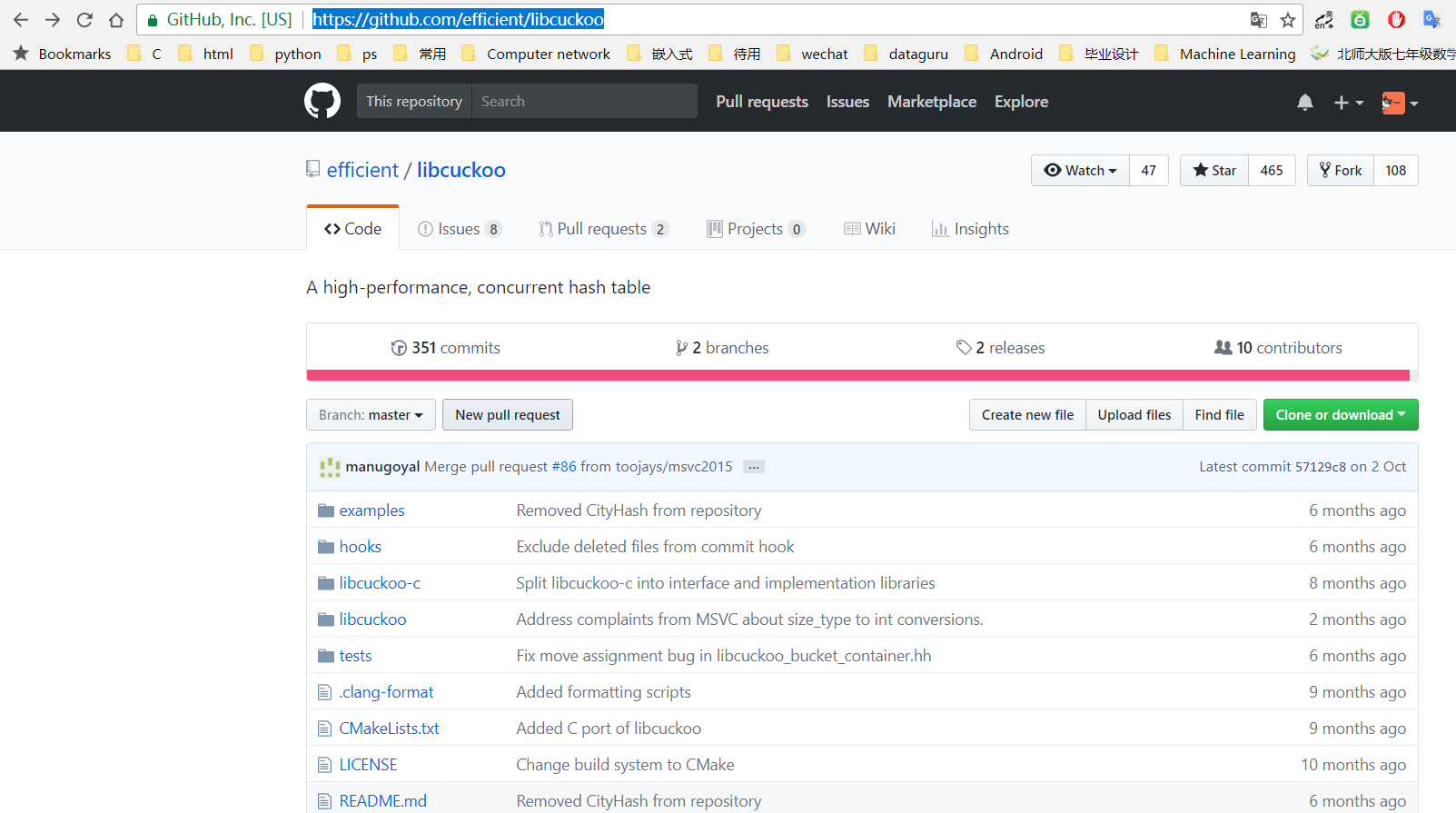
用git clone <https://github.com/syy804123097/SmartCuckoo.git> 后 进入该目录下,

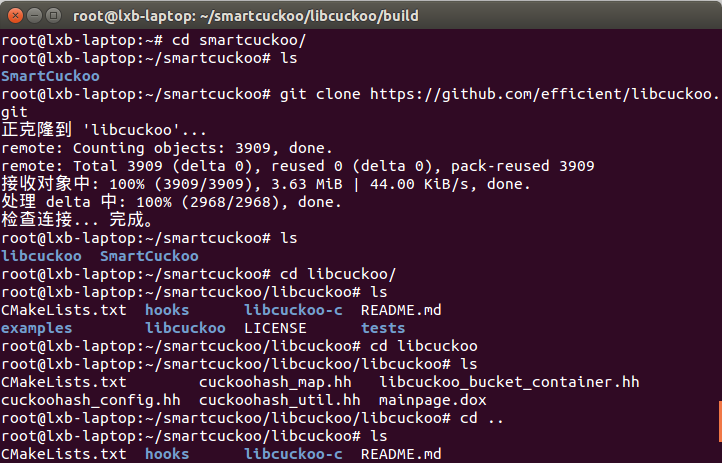
作者有了一个test目录, 里面有一个编好的程序, make 生成了ycsb\_test, 看了下源码bench\_client.cc 感觉应该是这个, 但是执行后发现了内存不足的错误.. 好像是需要定义一下key的格式, 看了一下配置是字符串, 读的话直接读的文件根据num\_queries来申请内存,但是输出表示num\_queries是10^16级别的, 内存肯定是不够的了, 自己还是哪里没有注意到, 本次实验是失败的...Orz

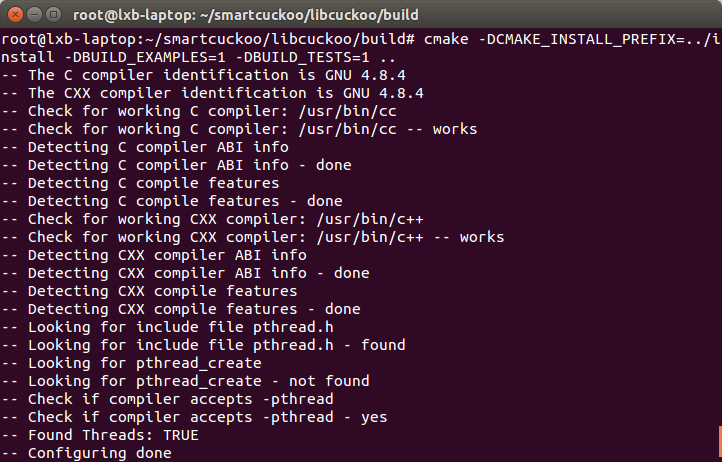




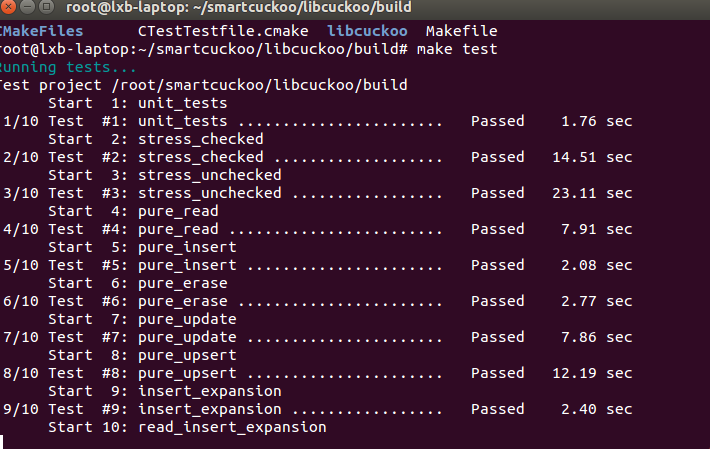
然后试着编译了一下libcuckoo, 首先的问题是cmake版本太低, Ubuntu 14.04 最高只能更新到cmake 2.8, 手动编译了一个cmake3.2, 然后编译后运行了一下这个算法

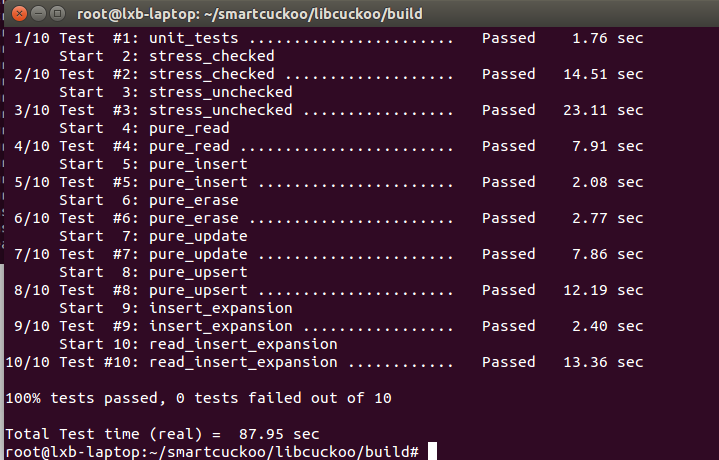






使用make test来进行算法的一次模拟, 终于能够正确的输出一个结果了:





实验总结:

本次实验总体来说很是失败,四个算法只成功运行了一个算法, 原来看了一下论文,感觉都有源码, 会很简单, 结果开始的太晚, 后来发现并不简单, 需要额外看好几篇论文,一周时间也没能搞定, 需要好好的反思一下. 自己在学术这边能力太弱,需要好好加强一下, 写了半年的web, 感觉linux编程能力感觉也生疏了, 算法没能自己实现出来, 接下来会继续看相关的论文, 希望能把其他几个算法做出来吧...Orz