2019 级

《物联网数据存储与管理》课程

实 验 报 告

姓　名　胡嘉慧

学　号　U201990067

班　号　计算机 1908 班

日　期　2022.04.14

目　　录

一、实验目的

1. 熟悉对象存储技术，代表性系统及其特性；
2. 实践对象存储系统，部署实验环境，进行初步测试；
3. 基于对象存储系统，分析性能问题，架设应用实践。

二、实验背景

对象存储也称为基于对象的存储，是一种将数据存储作为不同单元（称为对象）进行管理和操作的策略。这些对象保存在单个存储库中，不会与其他文件夹中的文件绑定到一起。相反，对象存储会将构成文件的数据片段合并到一起，将所有相关的元数据添加到该文件，并附加自定义标识符。

对象存储有不少的优势，例如:数据分析能力更为出色、可扩展性极高、数据检索速度更快、成本降低和资源优化。

对象存储系统（Object-Based Storage System）是综合了 NAS 和 SAN 的优点，同时具有 SAN 的高速直接访问和 NAS 的数据共享等优势，提供了高可靠性、跨平台性以及安全的数据共享的存储体系结构。

三、实验环境

实验环境如表 3.1。

表 3.1 实验环境

虚拟机操作系统	Ubuntu 18.04
虚拟机硬盘大小	40GB
对象存储服务端	Openstack Swift
对象存储客户端	python-swiftclient
评测工具	s3bench

四、实验内容

首先要安装所需的工具和语言，熟悉一下 git 的用法，并且能成功把文件名为学号的文件夹上传 Zhan / bigdata-storage-experiment-assignment-2022。

4.1 对象存储技术实践

安装 docker 容器后，安装配置服务端 OpenStack Swift 和客户端 python-swiftclient 进行测试。

4.2 对象存储性能分析

安装评测工具 COSBench，然后进行标准测试，探究对象尺寸如何影响性能、I/O 延迟背后的关键影响要素、如果客户端爆满将怎样等问题。

五、实验过程

5.1 系统搭建

1. 安装语言 python

安装成功如图 5.1 所示。

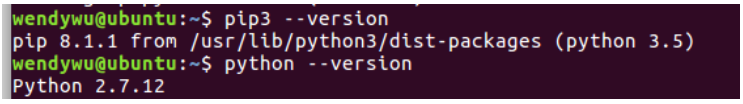


图 5.1 安装语言 python

2. 安装 git

安装成功如图 5.2 所示。

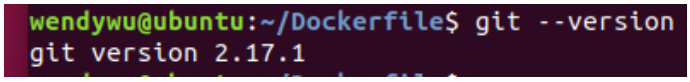


图 5.2 安装 git

3. 安装 docker 容器

首先先安装 curl，如图 5.3 所示。

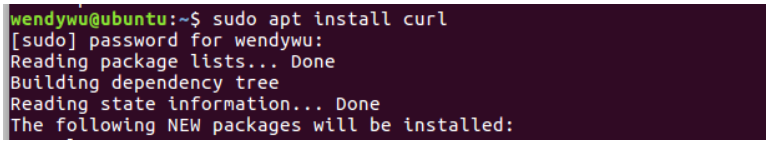


图 5.3 安装 curl

然後安装 docker 容器，但报出错误，原来是 ubuntu 版本过低，如图 5.4 所示。ubuntu 版本由 16.04 改为使用 18.04 后即安装成功，如图 5.5 所示。其版本号如图 5.6 所示。

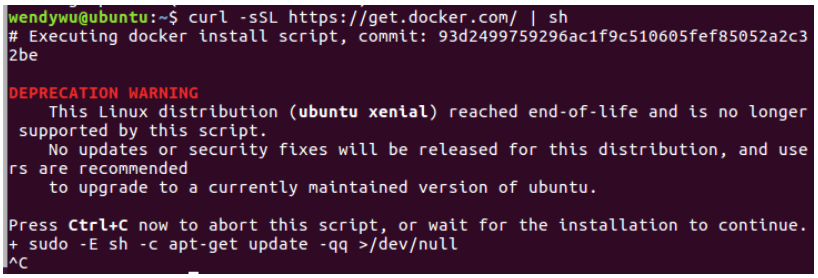


图 5.4 安装 docker 发生错误

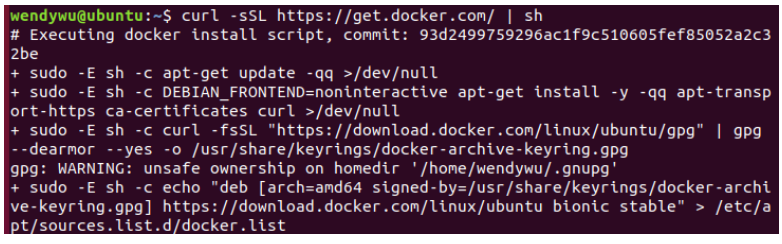


图 5.5 安装 docker 成功

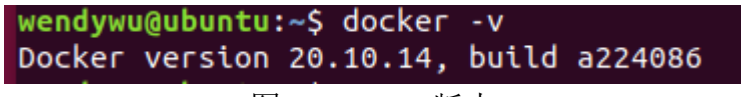


图 5.6 docker 版本

4. 安装服务端 OpenStack Swift

下载 openstack-swift-docker 的安装包并将其解压，如图 5.7 所示。

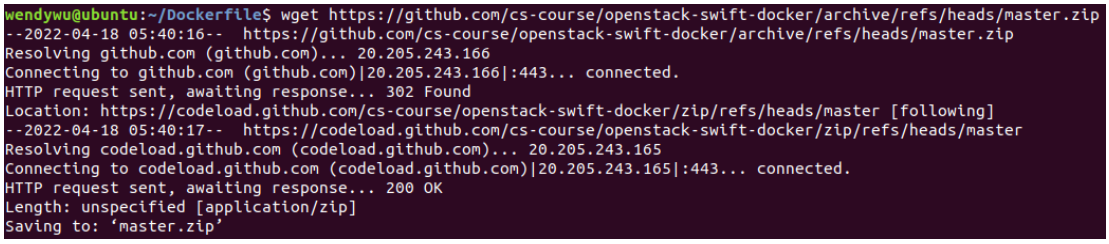


图 5.7 下载 openstack-swift-docker

按照 https://github.com/cs-course/openstack-swift-docker 上的步骤执行，如图 5.8~5.14 所示。

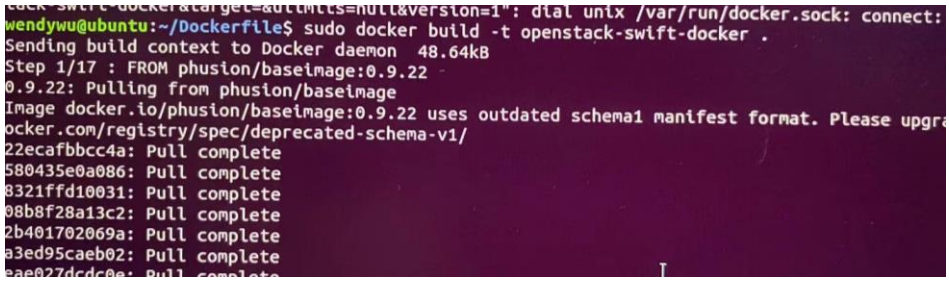


图 5.8 创建 docker 镜像

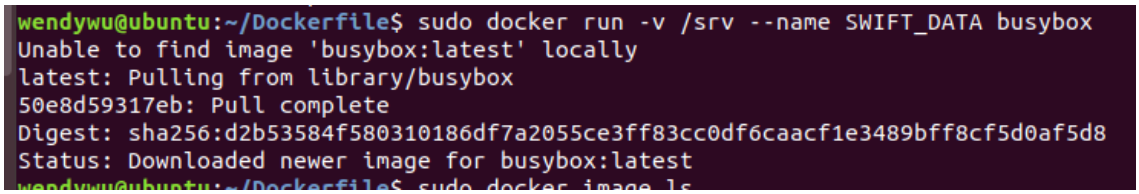


图 5.9 Prepare datavolume

5. 安装客户端 python3-swiftclient

在运行容器前要安装 python3-swiftclient，安装后运行容器。

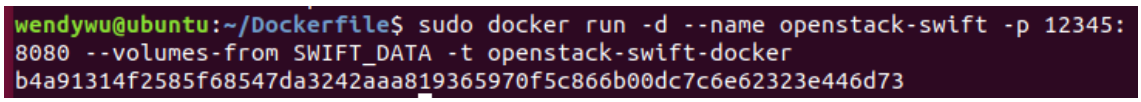


图 5.10 运行容器

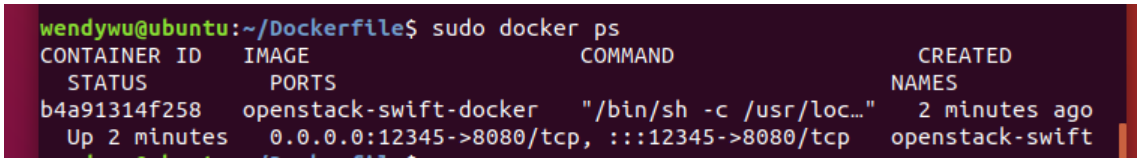


图 5.11 检查容器

```
wendywu@ubuntu:~/Dockerfile$ swift -A http://127.0.0.1:12345/auth/v1.0 -U test:
tester -K testing stat

                       Account: AUTH_test
                    Containers: 1
                       Objects: 0
                         Bytes: 0
Containers in policy "policy-0": 1
   Objects in policy "policy-0": 0
     Bytes in policy "policy-0": 0
                  X-Timestamp: 1650287764.35091
                 Content-Type: text/plain; charset=utf-8
                Accept-Ranges: bytes
                   X-Trans-Id: txb68ac7816302444589c46-00625e7ad5
```

图 5.12 验证功能(1)

先上传一个 txt 文件上去，如图 5.13 所示，然后便可看到 txt 文件已在 list 中，如图 5.14 所示。

```
wendywu@ubuntu:~/Dockerfile$ swift -A http://127.0.0.1:12345/auth/v1.0 -U test:
tester -K testing upload --object-name upload.txt SWIFT_DATA ./test.txt
upload.txt
```

图 5.13 上传文档

```
wendywu@ubuntu:~/Dockerfile$ swift -A http://127.0.0.1:12345/auth/v1.0 -U test:
tester -K testing list
SWIFT_DATA
```

图 5.14 验证功能(2)

6. 安装 COSBench(失败)

安装 java，如图 5.15 所示。

```
wendywu@ubuntu:~/0.4.2.c4$ java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-8u312-b07-0ubuntu1~18.04-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
```

图 5.15 安装 java

在 github 上下载 COSBench 的 v0.4.2.c4 版本的 zip 压缩包并将其解压，如图 5.16 所示。不要使用 v0.4.2.release 版本，会出现错误。然后运行 start-all.sh。但却一直卡在这个画面，如图 5.16 所示，一直解决不了这个问题，所以最终我决定 mock_s3、osm 和 s3bench 完成性能测试。

```
wendywu@ubuntu:~/Desktop/0.4.2.c4$ ./start-all.sh
Launching osgi framwork ...
Successfully launched osgi framework!
Booting cosbench driver ...
```

图 5.16 安装 COSBench 失败

5.2 性能测试

从老师提供的仓库里下载 s3bench 所需的文件，如图 5.17。

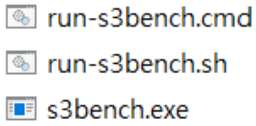


图 5.17 s3bench

运行脚本 run-s3bench.cmd 启动 s3bench 进行评测，如图 5.18 所示。

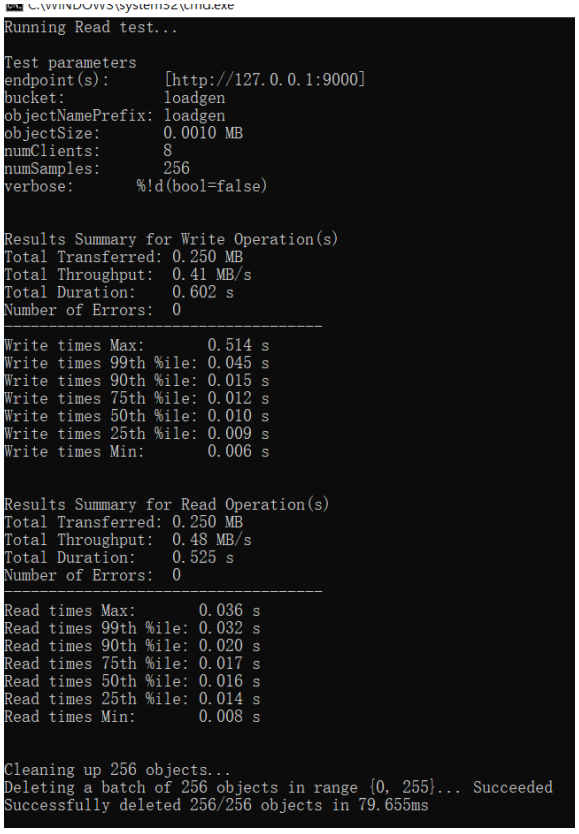


图 5.18 s3bench 评测结果显示

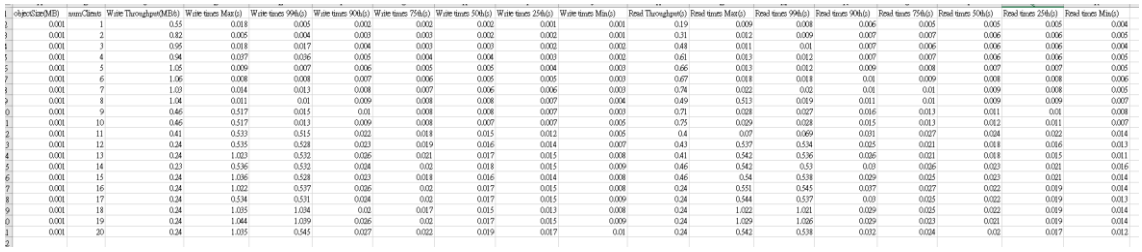
修改脚本 run-s3bench.cmd，保持其余值不变的情况下，改变 objectSize 数据记录在 excel 表格 s3bench data1.xlsx，如图 5.19 所示。

objectSize(MB)	numClients	Write Throughput(MB/s)	Write times Max(s)	Write times 99th(s)	Write times 90th(s)	Write times 75th(s)	Write times 50th(s)	Write times 25th(s)	Write times Min(s)	Read Throughput(s)	Read times Max(s)	Read times 99th(s)	Read times 90th(s)	Read times 75th(s)	Read times 50th(s)	Read times 25th(s)	Read times Min(s)
0.001	8	0.41	0.514	0.045	0.015	0.012	0.01	0.009	0.006	0.48	0.036	0.032	0.02	0.017	0.016	0.014	0.008
0.002	8	0.39	1.021	0.09	0.048	0.037	0.031	0.025	0.012	0.37	0.576	0.085	0.047	0.035	0.029	0.023	0.016
0.0048	8	1.98	0.515	0.04	0.027	0.019	0.016	0.013	0.008	2.29	0.026	0.023	0.02	0.018	0.017	0.015	0.01
0.0067	8	3.07	0.034	0.033	0.022	0.02	0.017	0.015	0.01	3.23	0.528	0.022	0.016	0.014	0.013	0.012	0.007
0.0086	8	3.29	0.055	0.05	0.028	0.022	0.018	0.016	0.011	3.42	0.027	0.027	0.024	0.022	0.019	0.018	0.012
0.0114	8	3.96	0.512	0.029	0.022	0.019	0.016	0.014	0.006	4.25	0.041	0.036	0.024	0.022	0.021	0.02	0.017
0.0134	8	6.08	0.035	0.029	0.022	0.02	0.017	0.015	0.007	5.03	0.027	0.027	0.024	0.022	0.021	0.02	0.013
0.0153	8	6.49	0.051	0.049	0.025	0.021	0.017	0.014	0.009	6.3	0.026	0.026	0.021	0.02	0.019	0.018	0.014
0.0172	8	7.26	0.049	0.038	0.024	0.021	0.018	0.016	0.009	5.65	0.079	0.054	0.03	0.024	0.022	0.021	0.015
0.021	8	9.66	0.035	0.028	0.022	0.019	0.017	0.015	0.007	7.45	0.029	0.028	0.025	0.023	0.022	0.021	0.016
0.0305	8	8.34	0.519	0.031	0.024	0.02	0.016	0.013	0.006	12.32	0.029	0.026	0.022	0.021	0.02	0.018	0.013
0.042	8	18.66	0.034	0.03	0.023	0.02	0.018	0.015	0.011	11.56	0.044	0.041	0.032	0.03	0.029	0.027	0.021
0.0515	8	22.6	0.529	0.043	0.022	0.018	0.015	0.013	0.007	20.54	0.531	0.022	0.02	0.019	0.018	0.017	0.013
0.0639	8	28.44	0.05	0.044	0.023	0.02	0.017	0.015	0.008	26.58	0.029	0.024	0.021	0.02	0.019	0.018	0.014
0.0744	8	31.32	0.056	0.052	0.024	0.021	0.018	0.015	0.007	24.83	0.533	0.028	0.024	0.023	0.022	0.021	0.017
0.083	8	37.09	0.038	0.029	0.022	0.02	0.018	0.015	0.008	32.29	0.531	0.023	0.02	0.018	0.016	0.015	0.012
0.0944	8	40.39	0.034	0.031	0.023	0.021	0.019	0.016	0.009	40.8	0.027	0.025	0.021	0.019	0.018	0.017	0.014
0.1144	8	48.53	0.035	0.028	0.023	0.021	0.019	0.016	0.011	44.58	0.538	0.024	0.021	0.02	0.018	0.017	0.013
0.2384	8	80.12	0.55	0.131	0.031	0.018	0.015	0.013	0.006	92.38	0.531	0.023	0.021	0.02	0.018	0.017	0.014
0.3624	8	102.39	0.335	0.334	0.029	0.02	0.017	0.015	0.007	97.71	0.042	0.039	0.033	0.031	0.029	0.027	0.024
0.4578	8	213.54	0.03	0.027	0.022	0.019	0.017	0.015	0.01	151.23	0.531	0.029	0.024	0.023	0.022	0.021	0.016
0.5627	8	244.31	0.522	0.034	0.019	0.017	0.015	0.014	0.009	196.51	0.029	0.027	0.025	0.024	0.023	0.022	0.017
0.6485	8	282.02	0.534	0.037	0.019	0.016	0.013	0.012	0.008	134.89	0.052	0.05	0.043	0.041	0.038	0.035	0.029
0.7439	8	332.09	0.034	0.026	0.022	0.02	0.018	0.016	0.01	264.99	0.527	0.027	0.022	0.02	0.018	0.017	0.013
0.8392	8	349.49	0.032	0.027	0.023	0.021	0.019	0.017	0.008	318.2	0.03	0.028	0.025	0.023	0.021	0.019	0.013
0.9346	8	377.89	0.032	0.03	0.024	0.021	0.019	0.018	0.012	316.91	0.031	0.029	0.026	0.025	0.023	0.022	0.016
1	8	269.28	0.526	0.04	0.025	0.022	0.019	0.017	0.013	358.98	0.03	0.028	0.024	0.023	0.022	0.021	0.017
2	8	15.44	1.391	1.391	1.033	1.029	1.024	1.02	1.016	628.32	0.045	0.038	0.029	0.027	0.025	0.023	0.019
4	8	29.99	1.53	1.224	1.053	1.042	1.036	1.032	1.021	902.19	0.064	0.057	0.043	0.038	0.034	0.031	0.024
6	8	45.52	1.201	1.175	1.061	1.052	1.046	1.042	1.028	1125.81	0.17	0.066	0.052	0.047	0.041	0.036	0.021
8	8	58.91	1.577	1.343	1.087	1.066	1.06	1.054	1.04	1241.53	0.087	0.085	0.064	0.057	0.049	0.044	0.035
10	8	72.94	1.347	1.343	1.106	1.08	1.072	1.065	1.047	1377.8	0.082	0.081	0.07	0.063	0.057	0.052	0.038
12	8	85.03	1.684	1.683	1.16	1.106	1.091	1.082	1.052	1383.12	0.111	0.098	0.085	0.075	0.068	0.061	0.047
14	8	97.99	1.67	1.66	1.152	1.111	1.097	1.089	1.064	667.09	0.225	0.207	0.187	0.177	0.165	0.155	0.116
16	8	111.94	1.76	1.517	1.168	1.116	1.108	1.099	1.08	1423.06	0.415	0.384	0.107	0.095	0.083	0.075	0.037

图 5.19 改变 objectSize 数据记录

修改脚本 run-s3bench.cmd，保持其余值不变的情况下，改变 numClients 数据

记录在 excel 表格 s3bench data2.xlsx，如图 5.20 所示。



objectSize(MB)	numClients	Write Throughput(MB/s)	Write times Max(s)	Write times 99th(s)	Write times 90th(s)	Write times 75th(s)	Write times 50th(s)	Write times 25th(s)	Write times Min(s)	Read Throughput(s)	Read times Max(s)	Read times 99th(s)	Read times 90th(s)	Read times 75th(s)	Read times 50th(s)	Read times 25th(s)	Read times Min(s)
0.001	1	0.55	0.018	0.005	0.002	0.002	0.002	0.001	0.001	0.19	0.009	0.008	0.006	0.005	0.005	0.005	0.004
0.001	2	0.82	0.005	0.004	0.003	0.003	0.002	0.002	0.001	0.31	0.012	0.009	0.007	0.007	0.006	0.006	0.005
0.001	3	0.95	0.018	0.017	0.004	0.003	0.003	0.002	0.002	0.48	0.011	0.01	0.007	0.006	0.006	0.006	0.004
0.001	4	0.94	0.037	0.036	0.005	0.004	0.004	0.003	0.002	0.61	0.013	0.012	0.007	0.007	0.006	0.006	0.005
0.001	5	1.05	0.009	0.007	0.006	0.005	0.005	0.004	0.003	0.66	0.013	0.012	0.009	0.008	0.007	0.007	0.005
0.001	6	1.06	0.008	0.008	0.007	0.006	0.005	0.005	0.003	0.67	0.018	0.018	0.01	0.009	0.008	0.008	0.006
0.001	7	1.03	0.014	0.013	0.008	0.007	0.006	0.006	0.003	0.74	0.022	0.02	0.01	0.01	0.009	0.008	0.005
0.001	8	1.04	0.011	0.01	0.009	0.008	0.008	0.007	0.004	0.49	0.513	0.019	0.011	0.01	0.009	0.009	0.007
0.001	9	0.46	0.517	0.015	0.01	0.008	0.008	0.007	0.003	0.71	0.028	0.027	0.016	0.013	0.011	0.01	0.008
0.001	10	0.46	0.517	0.013	0.009	0.008	0.007	0.007	0.005	0.75	0.029	0.028	0.015	0.013	0.012	0.011	0.007
0.001	11	0.41	0.533	0.515	0.022	0.018	0.015	0.012	0.005	0.4	0.07	0.069	0.031	0.027	0.024	0.022	0.014
0.001	12	0.24	0.535	0.528	0.023	0.019	0.016	0.014	0.007	0.43	0.537	0.534	0.025	0.021	0.018	0.016	0.013
0.001	13	0.24	1.023	0.532	0.026	0.021	0.017	0.015	0.008	0.41	0.542	0.536	0.026	0.021	0.018	0.015	0.011
0.001	14	0.23	0.536	0.532	0.024	0.02	0.018	0.015	0.009	0.46	0.542	0.53	0.03	0.026	0.023	0.021	0.016
0.001	15	0.24	1.036	0.528	0.023	0.018	0.016	0.014	0.008	0.46	0.54	0.538	0.029	0.025	0.023	0.021	0.014
0.001	16	0.24	1.022	0.537	0.026	0.02	0.017	0.015	0.008	0.24	0.551	0.545	0.037	0.027	0.022	0.019	0.014
0.001	17	0.24	0.534	0.531	0.024	0.02	0.017	0.015	0.009	0.24	0.544	0.537	0.03	0.025	0.022	0.019	0.013
0.001	18	0.24	1.035	1.034	0.02	0.017	0.015	0.013	0.008	0.24	1.022	1.021	0.029	0.025	0.022	0.019	0.014
0.001	19	0.24	1.044	1.039	0.026	0.02	0.017	0.015	0.009	0.24	1.029	1.026	0.029	0.023	0.021	0.019	0.014
0.001	20	0.24	1.035	0.545	0.027	0.022	0.019	0.017	0.01	0.24	0.542	0.538	0.032	0.024	0.02	0.017	0.012

图 5.20 改变 numClients 数据记录

1. objectSize 对存储系统性能的影响

修改 objectSize 的大小，numClients 数量不变，一直为 8。从图 5.21 中可看出 objectSize 越大时读吞吐率就越大，但写吞吐率却比较平缓。

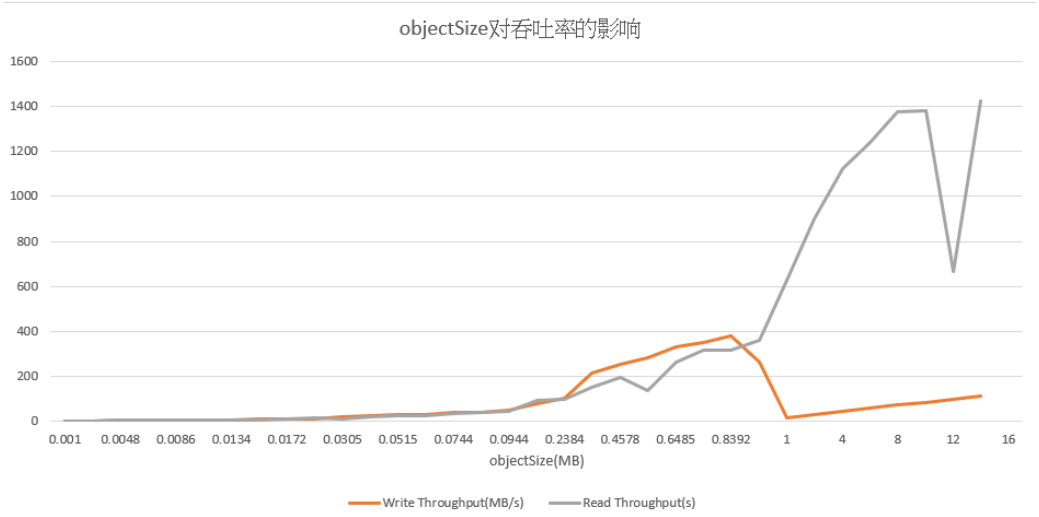


图 5.21 objectSize 对吞吐率的影响

Write times Max 和 Write times 99th 的起伏比较大，其他都是从 objectSize 大小是 1 后写延迟才突然急增，如图 5.22 所示。

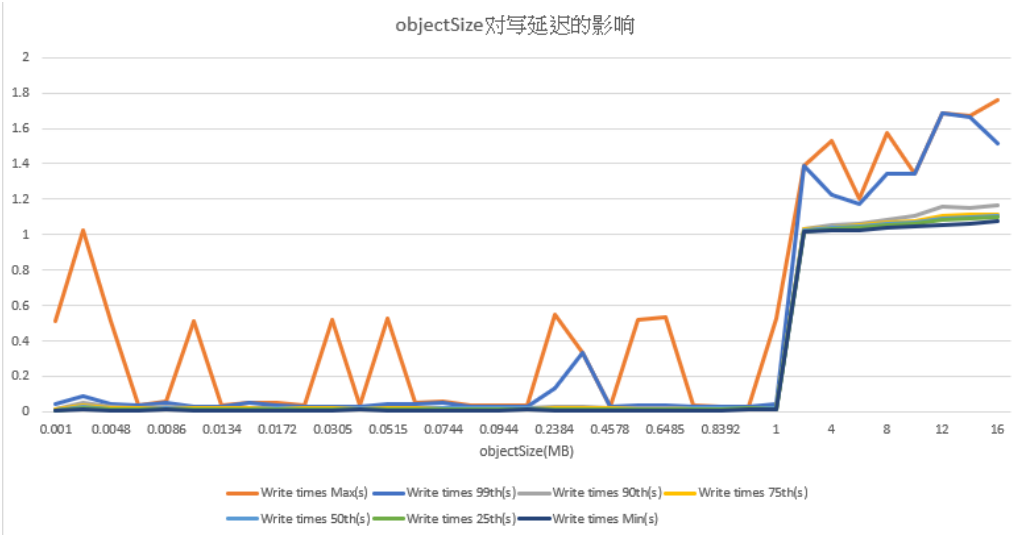


图 5.22 objectSize 对写延迟的影响

只有 Read times Max 的起伏比较大，其他的读延迟都趋近平缓，如图 5.23 所示。

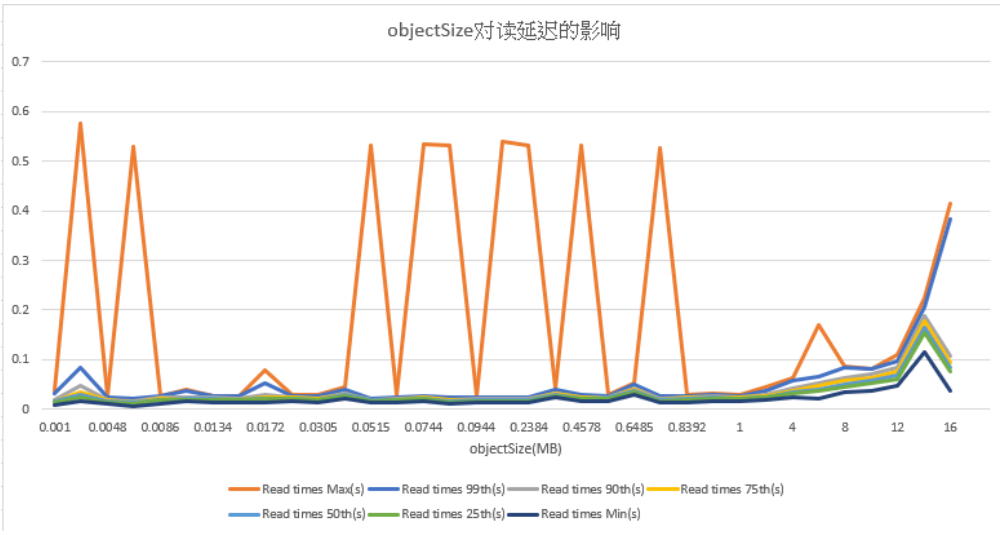


图 5.23 objectSize 对读延迟的影响

2. numClients 对存储系统性能的影响

修改 numClients 的数量，objectSize 大小不变，一直为 0.001MB。从图 5.24 中可看出在 numClients 为 16 前，读写吞吐率的差别很大，但 numClients 为 16 后，它们的值都一样为 0.24。

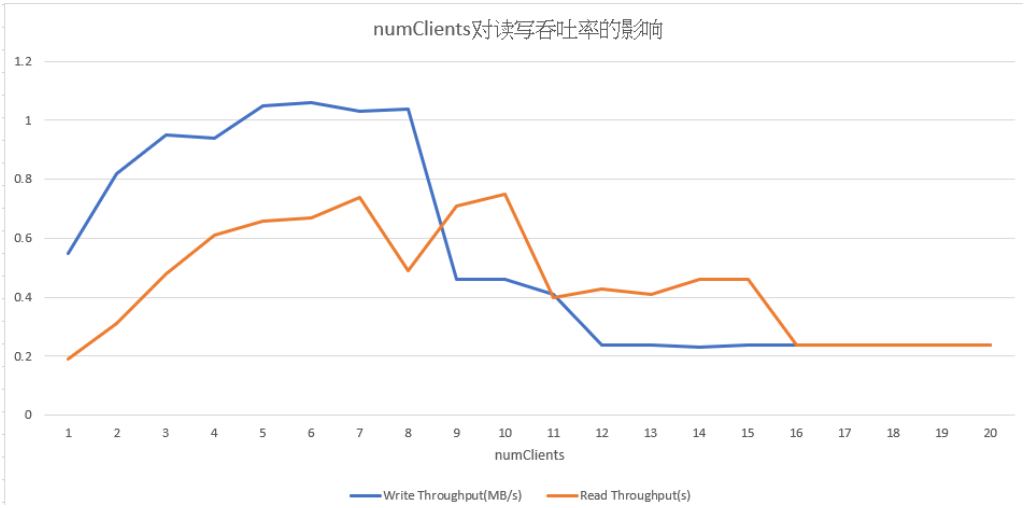


图 5.24 numClients 对吞吐率的影响

除 Write times Max 和 Write times 99th 的在 numClients 为 8 后起伏较大外，其他数值无论 numClients 的值都差别不大，如图 5.25 所示。

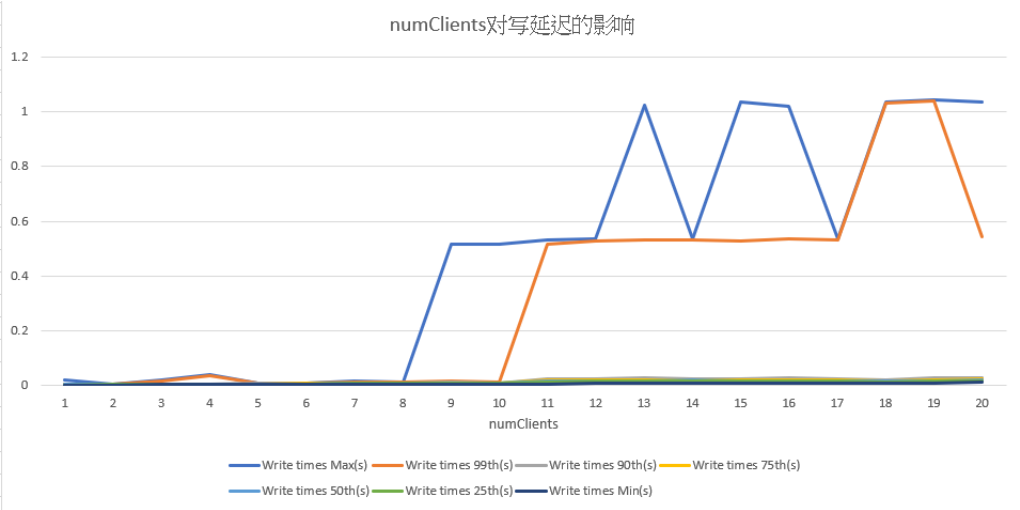


图 5.25 numClients 对写延迟的影响

除 Write times Max 和 Write times 99th 的在 numClients 为 7 后起伏较大外，其他数值无论 numClients 的值都差别不大，如图 5.26 所示。

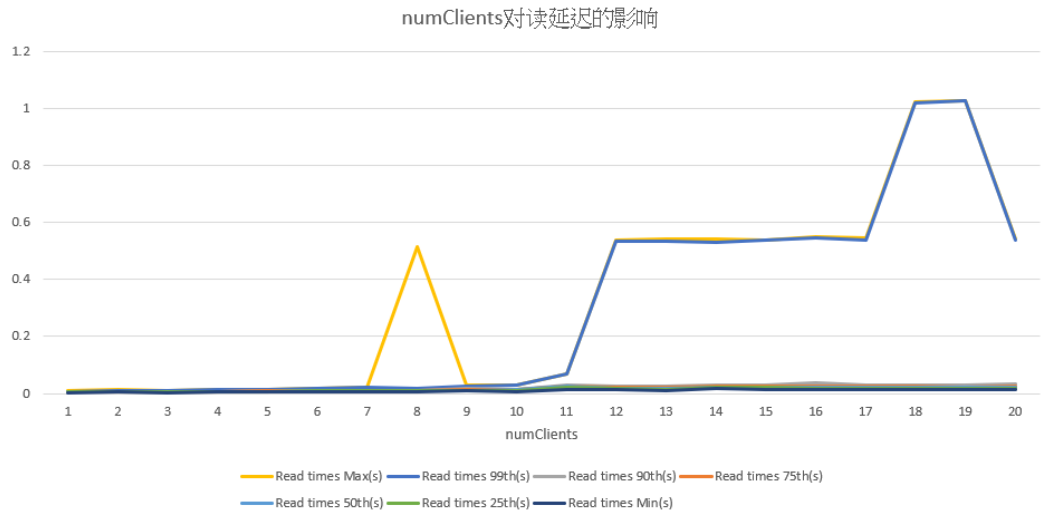


图 5.26 numClients 对读延迟的影响

六、实验总结

通过这次的实验我知道了对象存诸的意义，还学会了 docker 的使用方法，在实验的过程中我以为 dockerFile 是会自动生成的，当系统提醒我找不到 dockerFile 时我还以为是选择的路径不对，原来是要下载在 openstack-swift-docker 的 dockerFile 来能正确运行。

参考文献

[1] ZHENG Q, CHEN H, WANG Y 等. COSBench: A Benchmark Tool for Cloud Object Storage Services[C]//2012 IEEE Fifth International Conference on Cloud Computing. 2012: 998－999.
[2] ARNOLD J. OpenStack Swift[M]. O'Reilly Media, 2014.
[3] WEIL S A, BRANDT S A, MILLER E L 等. Ceph: A Scalable, High-performance Distributed File System[C]//Proceedings of the 7th Symposium on Operating Systems Design and Implementation. Berkeley, CA, USA: USENIX Association, 2006: 307－320.
[4] Dean J, Barroso L A. Association for Computing Machinery, 2013. The Tail at Scale[J]. Commun. ACM, 2013, 56(2): 74－80.
[5] Delimitrou C, Kozyrakis C. Association for Computing Machinery, 2018. Amdahl's Law for Tail Latency[J]. Commun. ACM, 2018, 61(8): 65－72.