C计划

前言

网易高性能分布式存储系统Curve已在github开源，开源以来受到了业界的广泛关注，现招募在校学生贡献者加入我们的开发。

Who? 任何对分布式存储系统，Curve感兴趣的在校生，不限年级不限专业。

Why? Curve是新一代开源高性能分布式存储系统，通过参与Curve的开发，你可以收获：大型分布式系统开发经验，和业界大牛协作学习的机会，表现优异者可以获得在网易实习的机会（人数不限），特别优异者有机会直接获得校招offer。

How? 添加opencurve微信号联系我们，加微信号记得注明[报名C计划]。我们会提供分布式系统学习的Roadmap供大家参考，基于分布式存储系统的理论基础之上进行选题。另外我们会有定期线上会议，了解大家的想法并为为大家答疑解惑。近期12.25左右会有C计划启动会，具体时间我们会在微信群中公布。

Roadmap

因为缺乏专业背景知识，很多小伙伴看到Curve是一头雾水，看完Curve的简介后也许还是一头雾水，那我们该如何打怪升级呢？在开始C计划的选题之前，我们给参加C计划的小伙伴们一个分布式系统的学习的Roadmap，该Roadmap是Curve团队的新人培养实践沉淀，大家可以根据自身需要自行学习。

- **了解分布式存储系统的基本知识**

  目标：知道为什么需要分布式存储？分布式存储的难点在哪里？都有哪些解决方法

  参考资料：

  - [ ] Distributed systems（http://book.mixu.net/distsys/index.html）

- **熟悉分布式系统设计**

  目标：了解分布式系统一般架构，了解部分解决方案的技术细节

  参考资料：

  - [ ] The Google File System（http://static.googleusercontent.com/media/research.google.com/es//archive/gfs-sosp2003.pdf）
  - [ ] The Raft Consensus Algorithm （https://raft.github.io/）
  - [ ] MapReduce: Simplified Data Processing on Large Clusters（https://pdos.csail.mit.edu/6.824/papers/mapreduce.pdf）
  - [ ] CEPH: RELIABLE, SCALABLE, AND HIGH-PERFORMANCE DISTRIBUTED STORAGE（https://ceph.com/wp-content/uploads/2016/08/weil-thesis.pdf）
  - [ ] PacificA: Replication in Log-Based Distributed Storage Systems（https://www.microsoft.com/en-us/research/wp-content/uploads/2008/02/tr-2008-25.pdf）

- [ ] 6.824 Schedule: Spring 2020 （https://pdos.csail.mit.edu/6.824/schedule.html ，MIT的分布式系统课程，感兴趣的可以多看）

- **了解Curve**

  目标：掌握Curve总体设计，各模块的设计

  参考资料：

  - [ ] Curve主页 （https://opencurve.github.io/）
  - [ ] Curve系列讲座视频回放（https://space.bilibili.com/700847536/channel/detail?cid=153949）
  - [ ] Curve系列讲座ppt地址（https://github.com/opencurve/curve-meetup-slides/tree/main/2020）
  - [ ] Curve各模块介绍文档（https://github.com/opencurve/curve/tree/master/docs/cn）
  - [ ] Curve技术文档（https://zhuanlan.zhihu.com/p/311590077）
  - [ ] Curve代码阅读：建议先看下brpc的基本知识（https://github.com/apache/incubator-brpc/blob/master/docs/cn/brpc_intro.pptx）

- **掌握代码开发/测试工具**

  目标：掌握代码开发流程，熟练使用代码开发、调试、测试过程中所需要的工具

  参考资料：

  - [ ] Curve代码构建工具bazel（https://bazel.build/）
  - [ ] Curve代码测试框架gtest（https://github.com/google/googletest/blob/master/googletest/docs/primer.md）
  - [ ] Curve代码管理git （https://www.runoob.com/git/git-basic-operations.html）
  - [ ] 代码调试工具 gdb
  - [ ] 谷歌开源项目风格指南（https://zh-google-styleguide.readthedocs.io/en/latest/google-cpp-styleguide/
  - [ ] 编程规范：《Clean Code》

选题

难易级别：选题共有三个级别：easy，medium，hard

发放规则：这三个级别的选题我们会分三个阶段发布，第一阶段发布easy，第二阶段发布medium，第三阶段发布hard，每个阶段持续两个月，具体的时间会在微信群里通知。对选题的任何疑问或者需要帮助的都可以在群里咨询我们。

任务提交：

- 代码开发环境推荐使用docker镜像，参照：https://github.com/opencurve/curve/blob/master/docs/cn/build_and_run.md
- 对于所有的任务，开始之前请大家先提交issue，地址：https://github.com/opencurve/curve/issues

    - issue的标题格式【C计划-选题*】描述清楚选做哪部分
    - 在提issue之前可以浏览下其他已有的issue，是否有一样的任务，尽量选择不同任务，如果非常感兴趣，也可以重复选择
- 对于完成的任务，请大家将代码/文档以pr的形式提交至Curve的仓库，我们会定期查收并进行review给大家提出相应建议。对于优秀的提交，我们会合入代码仓库。

必选题：curve项目部署与编译

任务说明

- 每个参与者必须要实际部署和编译
- 所需技能：docker、读懂curve部署/编译文档

任务描述

部署和编译是参与开发前必须要掌握的。编译是代码开发、调试与测试过程中必要步骤，部署是学会使用curve、理解curve使用场景的途径。所以希望每个人都按照下列参考文档实际动手操作一下。任务提交：编译/部署成功的截图，通过issue的方式进行提交。如果在编译/部署过程中有对文档的改进建议，也可以一并添加进来。

参考资料

单机部署：https://github.com/opencurve/curve/blob/master/docs/cn/deploy.md#单机部署

编译：https://github.com/opencurve/curve/blob/master/docs/cn/build_and_run.md

选题一：代码解读

任务说明

- 单人参与，这是一个系列任务
- 所需技能：熟悉Curve代码

任务描述

在阅读Curve代码的过程中写一些源码解读或者自己的心得体会，一方面作为自己学习的沉淀，另一方面可以供他人参考

参考资料

建议在了解Curve的整体架构基础上去看代码

（B站直播视频：https://space.bilibili.com/700847536/channel/detail?cid=153949）

（ppt下载：https://github.com/opencurve/curve-meetup-slides/tree/main/2020）

选题二：代码翻译

任务说明

- 单人参与，这是一个系列任务
- 所需技能：了解Curve代码框架和书写规范，github使用

任务描述

针对curve github仓库中的Curve各模块的代码注释进行中文到英文的翻译，注意翻译的完整性和准确性。Curve代码中的mds模块已经都是英文注释，剩下的为 curve/src/chunkserver 、curvesrc/client、curve/nebd/part1、curve/nebd/part2。大家可以按照一个头文件和cpp对应实现为单位进行翻译，比如common.h、common.cpp。

选题三：清理代码中的TODO

任务说明

- 单人参与，这是一个系列，每个TODO可以作为一个任务
- 所需技能：github的使用，google c++编程规范，curve文档，编译工具bazel的使用，测试工具gtest的使用

任务描述

curve的代码在开发的过程中遗留了一些TODO，可以对这些TODO进行一些清理。清理范围，include,src,test目录下，排除thirdparties目录下的第三方组件的TODO。用“// TODO”作为关键字，搜索代码中的TODO。这些TODO有些比较简单，有些难度比较大。建议先从简单的开始修复，熟悉代码的修复合入流程，再慢慢挑战比较复杂的TODO。

参考资料

github的使用，google c++编程规范，curve文档，编译工具bazel的使用，测试工具gtest的使用

这里举例几个简单的TODO任务，也可以自己搜索代码中的TODO。

1. curve/include/chunkserver/chunkserver_common.h，把kOpRequestAlignSize放到配置文件中。

```
// TODO(wudmeiao): 是否需要考虑可配置
const uint32_t kOpRequestAlignSize = 4096;
```

2. curve/src/chunkserver/copyset_node.cpp，Init copyset对应的raft node options放到nodeOptions的init中。

```
/**

* Init copyset对应的raft node options

*/

nodeOptions_.initial_conf = conf_;

nodeOptions_.election_timeout_ms = options.electionTimeoutMs;

nodeOptions_.fsm = this;

nodeOptions_.node_owns_fsm = false;

nodeOptions_.snapshot_interval_s = options.snapshotIntervalS;

nodeOptions_.log_uri = options.logUri;

nodeOptions_.log_uri.append("/").append(groupId)

.append("/").append(RAFT_LOG_DIR);
```

3. curve/src/client/libcbd_libcurve.cpp，cbd_libcurve_filesize调用StatFile4Qemu接口时，判断StatFile4Qemu的返回值。

```
int64_t cbd_libcurve_filesize(const char* filename) {
struct FileStatInfo info;
memset(&info, 0, sizeof(info));

    // TODO(wuhanqing): 判断返回值
StatFile4Qemu(filename, &info);
return info.length;
}
```

4. curve/src/mds/nameserver2/curvefs.cpp，RenameFile接口，把oldFileName改成sourceFileName，newFileName改成destFileName。

```
// TODO(hzchenwei3): change oldFileName to sourceFileName
//                   and newFileName to destFileName)
StatusCode CurveFS::RenameFile(const std::string & oldFileName,
const std::string & newFileName,
uint64_t oldFileId, uint64_t newFileId)
```

选题四：单元测试

任务说明

- 单人参与，这是一个系列任务
- 所需技能：c++基础，gtest使用

任务描述

目前Curve很多代码的单元测试覆盖率不够，（具体情况见59.111.93.165:8080/job/curve_untest_job/HTML_20Report/），希望大家在现有单元测试代码（位于Curve代码的test目录）基础上，添加测试用例，使其覆盖率达到CI标准，代码行覆盖85%及以上，代码分支覆盖75%及以上。

Directory	Line Coverage		Functions		Branches	
chunkserver	83.5 %	3984 / 4769	93.5 %	492 / 526	62.8 %	1062 / 1692
chunkserver/concurrent_apply	96.0 %	97 / 101	100.0 %	19 / 19	85.7 %	12 / 14
chunkserver/datastore	96.3 %	1405 / 1459	93.4 %	127 / 136	88.2 %	465 / 527
chunkserver/raftlog	70.6 %	688 / 975	79.3 %	65 / 82	61.7 %	250 / 405
chunkserver/raftsnapshot	78.0 %	716 / 918	93.6 %	132 / 141	59.6 %	286 / 480
client	92.6 %	4380 / 4731	96.4 %	729 / 756	80.4 %	797 / 991
common	81.8 %	731 / 894	92.0 %	150 / 163	69.2 %	164 / 237
common/concurrent	93.5 %	287 / 307	86.4 %	95 / 110	79.5 %	35 / 44
fs	94.2 %	262 / 278	92.7 %	51 / 55	88.5 %	85 / 96
kvstorageclient	90.1 %	173 / 192	91.3 %	21 / 23	71.4 %	20 / 28
leader_election	95.5 %	42 / 44	92.9 %	13 / 14	100.0 %	4 / 4
mds/chunkserverclient	98.6 %	285 / 289	95.0 %	19 / 20	82.1 %	64 / 78
mds/common	100.0 %	4 / 4	100.0 %	2 / 2	-	0 / 0
mds/copyset	84.5 %	239 / 283	83.0 %	44 / 53	80.0 %	64 / 80
mds/heartbeat	94.2 %	438 / 465	96.2 %	51 / 53	77.1 %	168 / 218
mds/main	87.9 %	51 / 58	100.0 %	5 / 5	56.2 %	45 / 80
mds/nameserver2	82.0 %	2697 / 3288	94.6 %	247 / 261	68.6 %	952 / 1388
mds/nameserver2/allocstatistic	97.6 %	201 / 206	100.0 %	22 / 22	82.8 %	53 / 64
mds/nameserver2/helper	93.0 %	53 / 57	100.0 %	14 / 14	66.7 %	4 / 6
mds/nameserver2/idgenerator	90.7 %	49 / 54	82.1 %	23 / 28	80.0 %	16 / 20
mds/schedule	91.0 %	1660 / 1825	97.7 %	215 / 220	75.3 %	663 / 880
mds/schedule/scheduleService	100.0 %	48 / 48	100.0 %	8 / 8	100.0 %	4 / 4
mds/server	97.0 %	351 / 362	100.0 %	37 / 37	43.8 %	7 / 16
mds/topology	89.5 %	3217 / 3593	92.6 %	437 / 472	67.2 %	952 / 1417
snapshotcloneserver	85.6 %	714 / 834	75.6 %	34 / 45	66.3 %	281 / 424
snapshotcloneserver/clone	80.5 %	1377 / 1711	86.9 %	139 / 160	60.4 %	400 / 662
snapshotcloneserver/common	89.3 %	792 / 887	82.0 %	223 / 272	84.4 %	81 / 96
snapshotcloneserver/snapshot	86.5 %	1280 / 1479	86.6 %	142 / 164	71.1 %	355 / 499
tools	86.6 %	3216 / 3715	77.6 %	298 / 384	70.3 %	1320 / 1878

参考资料

RoadMap中「掌握代码开发/测试工具」所列出的资料

选题五：捉虫计划

任务说明

- 单人参与，这是一个列任务，每找到1个bug，相当于完成了一个任务
- 所需技能：github使用，熟悉curve部署、使用、代码

任务描述

金无足赤，人无完人，代码也没有不存在bug的代码。在代码开发过程中，虽然工程师们采用了各种方式来减少bug，但是总有一些漏网之鱼。各位小伙伴，一起撸起袖子来捉虫吧。在curve部署、使用、阅读代码过程中，如果发现了bug，请通过issue的方式记录下来，如果有解决方案，欢迎向我们提交代码。

参考资料

RoadMap中「了解Curve」所列出的资料

奖励

表现优异者可以获得在网易实习的机会（人数不限），特别优异者有机会获得校招offer。我们会根据整个计划过程中提交任务的质量、参与度、提交任务的数量等为依据进行评估。