测试目的

测试不同文件系统(ext3, ext4, xfs, btrfs)在我们主要业务场景下的执行性能，对比分析各场景下不同文件系统的优劣，结合现状与业务需求选出最合适的单机文件系统类型。

业务场景

1, 目前使用的文件生成：
上传的策略，400kB-2MB 之间(web server)。
上传或从 kdb 获取的配置，4kB-2MB 之间(web server)。
生成的 tunnel_log，0-22MB 之间(calculate server)。
生成的计算中间结果文件， 目前采用/media/strategy_upload/output/test/d1a_ev_0410 等形式创建的中间结果，文件大小多为 4kB(web server)。

2, 主要有三个大表，行情表(shfedepth 等)，任务表(task_brief, task_detail)，结果表（task_result, turing_log）。
行情表通过文件 csv -> 入库，由计算程序通过网络链接查询。
任务表由服务端生成 task 时候，准备好关联数据，目前的实现将 task 写入数据库有 4 个进程。
结果表由服务端监听 Redis，将数据拿到后写入 mysql，目前的实现也是 4 个进程来回写结果。

用例设计

业务场景的特点是： 大规模行情数据连续读取，较多行情数据的短时间入库，小规模数据随机读写，大量小文件创建， 针对这几个场景设计出了以下两组测试用例。

A, 常用业务性能测试：

1. 行情并发查询。（1, 2, 4, 32, 64 个线程并发查询，使用与计算程序的方法一致）
2. 结果入库。（从线上环境的 mysql 中导入的一组结果，分别用 1, 2, 4, 8 个线程 insert 到数据库）
3. 行情入库。（一个线程，入库一个交易所一天的行情，测试时用多天求均值）

B, 文件系统基准测试：

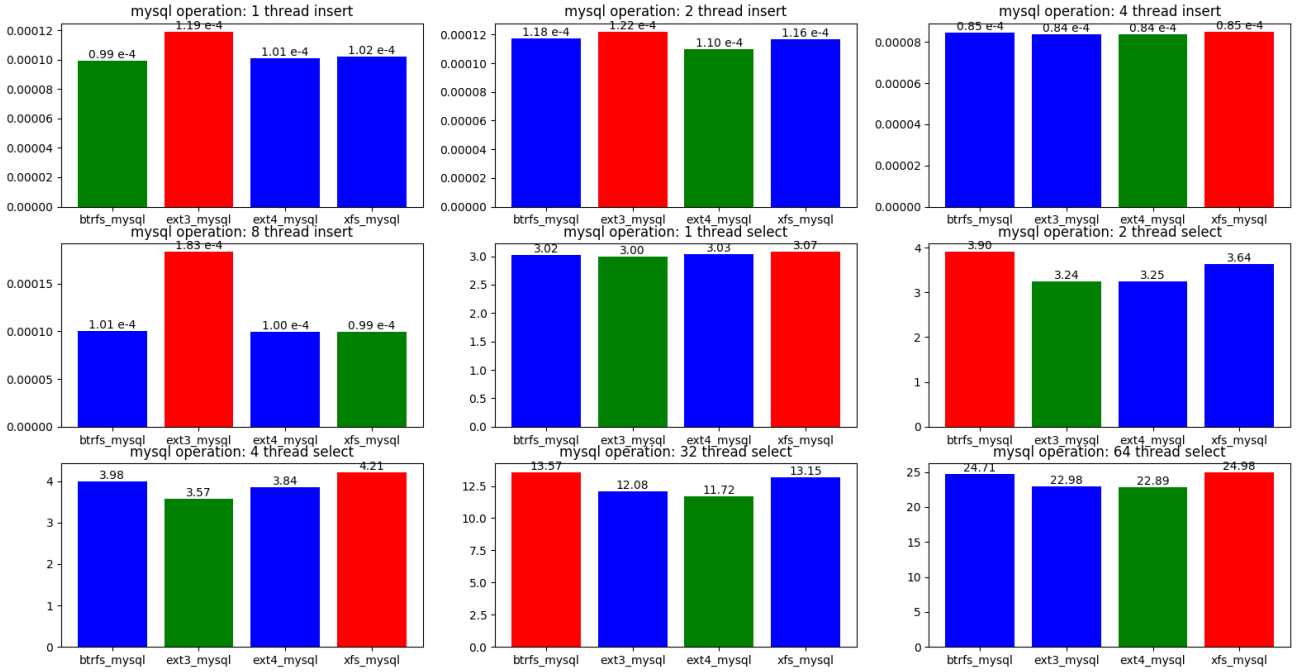
1. 不同大小，不同数量的文件 dd 执行连续读写速度测试。
2. 一定深度，大量中小文件的创建，删除速度测试。
3. 基于 sysbench 的数据库 oltp 测试。

测试环境

发行版：CentOS Linux release 7.2.1511 (Core)
型号: ProLiant DL380 Gen9
kernel：3.10.0-327.el7.x86_64
CPU: Intel(R) Xeon(R) CPU E5-2643 v3 @ 3.40GHz
Memory: 8G * 8
Network: HP Ethernet 1Gb 4-port 331i
GCC: 4.8.5 20150623 (Red Hat 4.8.5-4) (GCC)
SSD：750 series pcie 1.2TB [顺序 128 kb 读2500MB/s，写1200MB/s]
[随机 4K 读460,000 IOPS，写 290,000 IOPS]
mkfs 均采用默认选项格式化（如mkfs -t ext4 /dev/nvme0n1）。
mount 也均采用默认选项挂载(如mount -t ext4 /dev/nvme0n1 /mnt/ssd)。

测试结果

直方图对比中，绿色表示最好，红色表示最差，蓝色表示中间状态，测试过程中文件系统挂载均采用默认挂载参数。
A1,A2 多线程行情查询与结果写回性能对比结果如下所示：

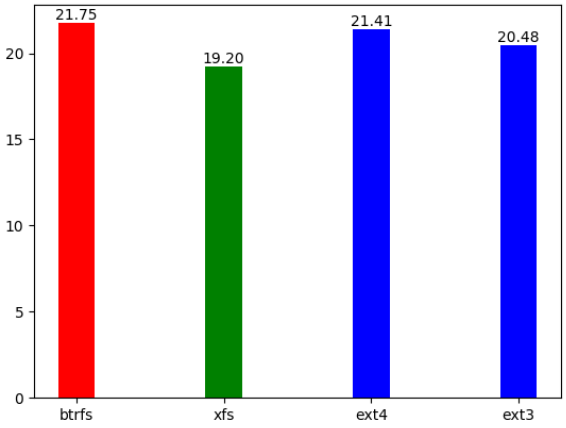


说明：Y 轴是响应时间，单位为 s

以不同线程数写结果到数据库，线程数为 1 时候 btrfs > ext4 > xfs > ext3，当线程增加到 2,4,8 时，ext4 与 xfs 差异均不大。

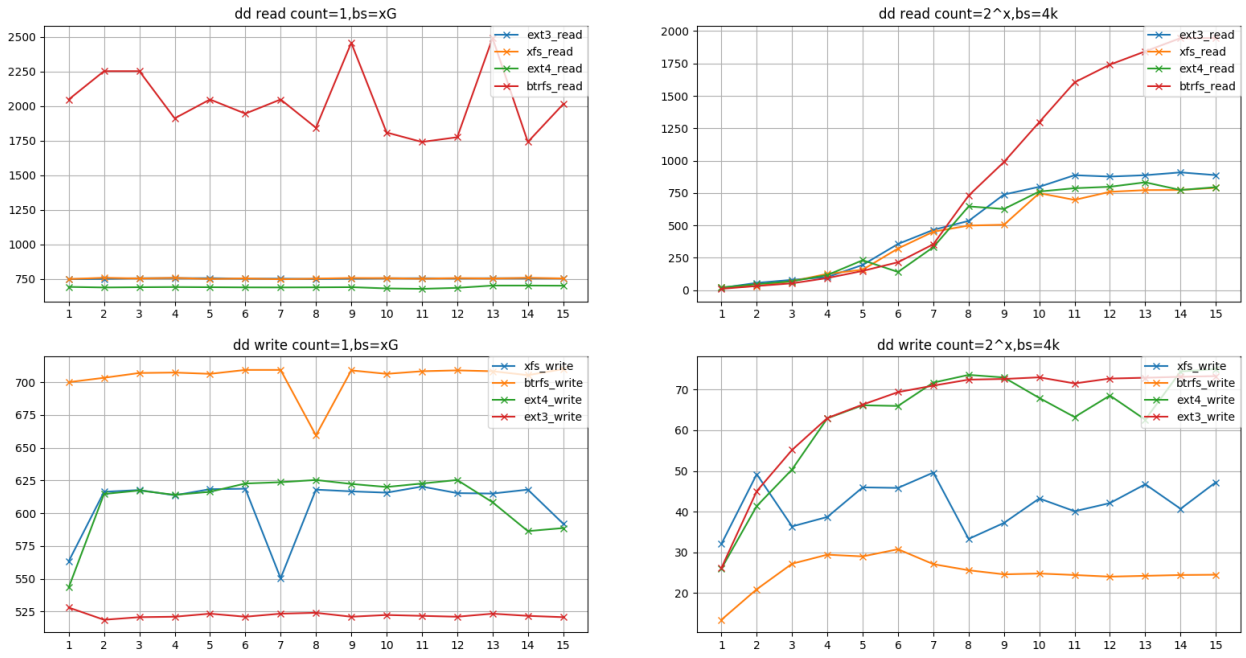
以不同线程查询行情，单线程下 ext3 > btrfs > ext4 > xfs，多线程下 ext3 和 ext4 表现相对更好，另外，在 32 线程查询时 CPU 已经打满。

A3 行情入库性能对比(Y 轴为响应时间，单位为 s)：



行情入库操作中，可以看 xfs > ext3 > ext4 > btrfs，其中 xfs 约比 ext4 快 10%+。

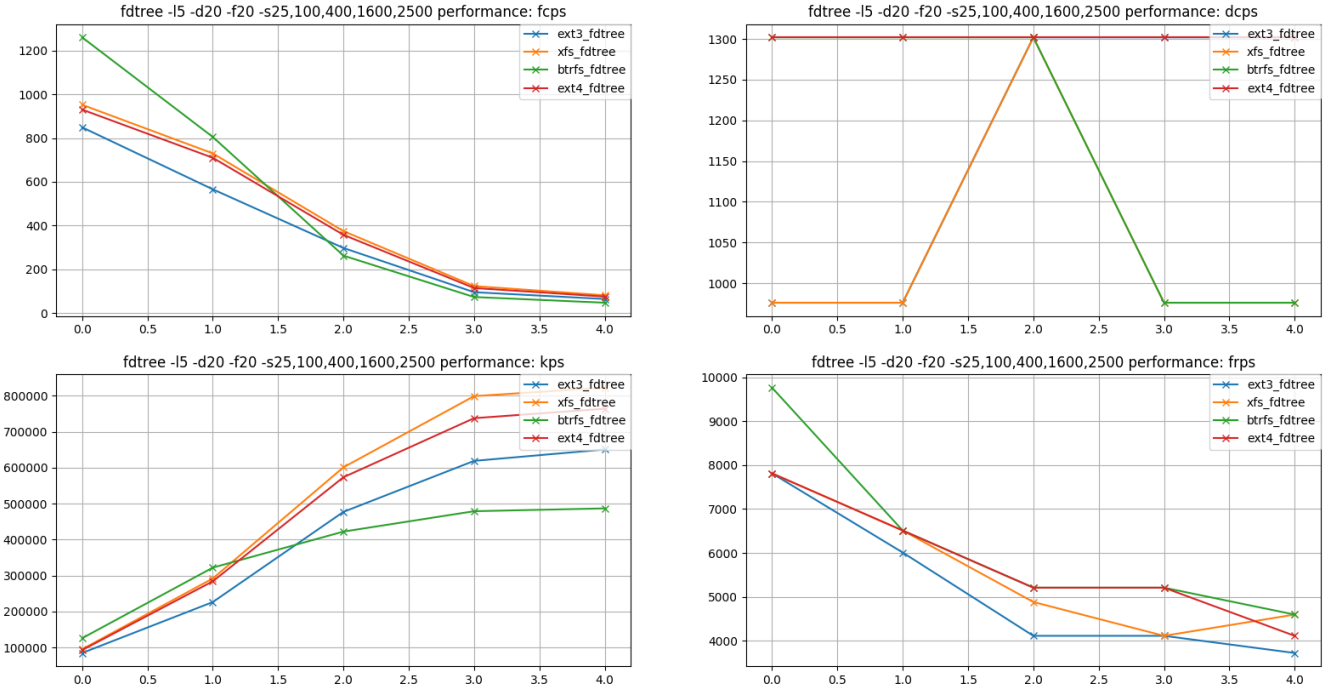
B1 测试文件系统不同大小文件的连续读写能力(dd write 使用了 oflag=dsync)。



说明： Y 轴是速度单位为 MB/s，X 轴内容参考 title，比如 bs=xG,即 X 轴数字对应多少 G

单个大文件 dd 连续读速度表现，btrfs > xfs > ext3 > ext4，多个小文件连续读速度表现 btrfs > ext3 > ext4 > xfs；单个大文件写方面，btrfs > ext4 > xfs > ext3，多小文件写方面 ext3 > ext4 > xfs > btrfs。

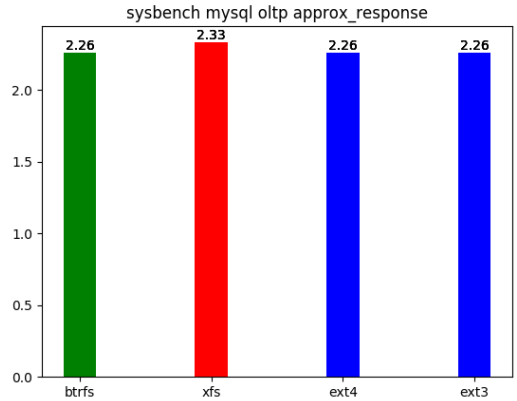
B2 带一定深度的目录和小文件的创建 & 删除测试，得到目标文件系统下的元数据操作性能。



说明：X 轴分别是在 5 层目录，每层 20 个文件夹，每个文件夹 20 个文件，[25,100,400,1600,2500] * 4K 个文件大小对应的性能参数，（221）的 Y 轴是个创建文件数/s，(222)的 Y 轴是创建文件夹个数/s，（223）的 Y 轴是速度，单位 kb/s，（224）的 Y 轴是删除文件数/s

结果来看，创建大量不同大小小文件，速度 xfs > ext4 > ext3（与写大量文件速度一致），创建路径速度 ext3 == ext4 > xfs 降低，删除文件速度大致为 ext4 > xfs> ext3。

B3 常规的数据库基准测试（针对多线程，大量数据查询用例）。
结果显示，针对多表（4），多数据(500000 条数据一个表)，多线程（100），随机查询，btrfs 表现最好，xfs 表现相对差一点， btrfs，ext3，ext4 的响应时间几乎一致（Y 轴是单个请求响应时间，单位为 ms）。



相关参考

1. 文件系统综述：
   https://en.wikipedia.org/wiki/File_system

2. Google 将单机文件系统由 ext2 切换为 ext4:
   http://www.phoronix.com/scan.php?page=news_item&px=Nzg4MA

3. ssd 下 ext4 vs xfs 性能测试：
   https://www.percona.com/blog/2012/03/15/ext4-vs-xfs-on-ssd/

4. xfs vs ext4 差异：

   https://www.unixmen.com/review-ext4-vs-btrfs-vs-xfs/

5. btrfs 不用于生产的原因主要是它还不稳定：

   https://www.rath.org/btrfs-reliability-a-datapoint.html

6. openbenchmarking 近 1 年的文件系统测试关注度，ext4 仍然是热度最高文件系统：

   http://openbenchmarking.org/s/File-System

7. Hadoop 默认用 ext3，Yahoo 线上 Hadoop 用的 ext3，Hadoop 与 ext3 配合已通过大量测试：
   https://community.hortonworks.com/articles/14508/best-practices-linux-file-systems-for-hdfs.html

8. 大多发行版采用默认文件系统仍然是 ext4:

   https://en.wikipedia.org/wiki/Comparison_of_Linux_distributions

9. 最新 kernel 4.10 中不同文件系统在 SSD 下性能测试得到：单位时间小文件创建 xfs 最快（与我们测试结果一致），BlogBench（模拟 file server,随机读写和重写）测试 xfs 读较 ext4 好，ext4 写较 xfs 好。

   http://www.phoronix.com/scan.php?page=article&item=linux-410-earlyfs&num=2

总结与结论

建议使用 ext4 来作为线上文件系统。

主要依据：

1，多个发行版的默认文件系统（Debian，Fedora，Ubuntu 等）。

2，在大量并发查询行情时，ext4 性能受影响最小，在高并发行情请求时相比 xfs 高 10%+性能，在其他查询场景下与 xfs 性能很接近，行情导入 xfs 性能相对比 ext4 好，但目标业务场景行情查询的数据量远大于写结果入库，行情入库的数据量。

3，Google 2010 年由做了文件系统切换 ext2 → ext4。

存在的问题：

1．未从机制上分析 xfs 在多线程环境下比 ext4 快的原因。

2．生产环境使用的 Centos，而 Centos 的默认文件系统是 xfs。

3．测试基于 linux kernel 3.10，对于最新版的 kernel 没有做相应的测试。