淘宝网
Taobao.com

TPPS:
支持资源隔离的高效 IO 调度器

董昊 <sanbai@taobao.com>

需求

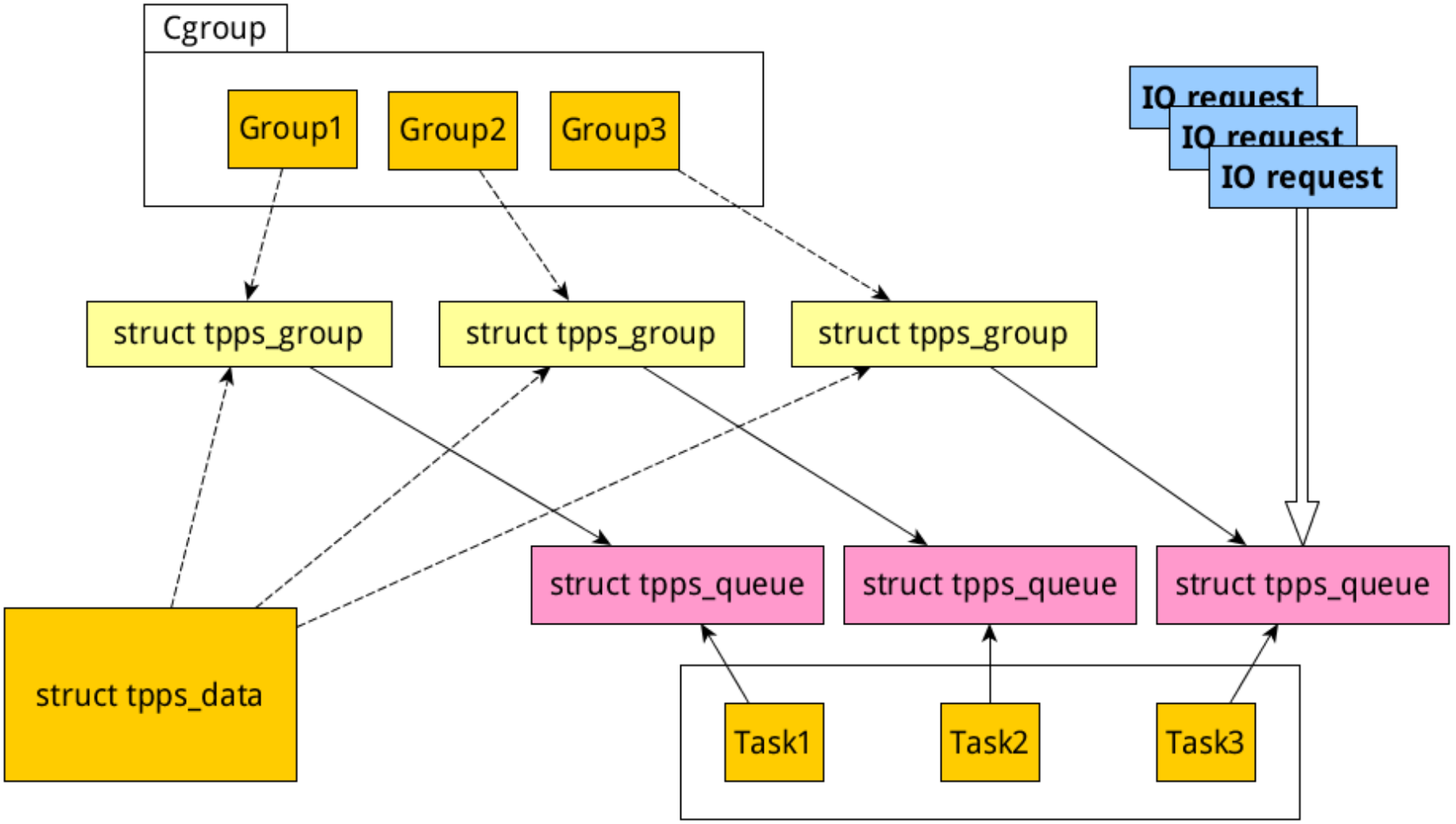
- 高效利用 SSD 的两个办法
  - 增大应用发出的 IO 深度（ aio ）
  - 在一个高速块设备上跑多个应用实例（数据库，虚拟机托管）
- 资源隔离
  - cfq 支持 cgroup
  - deadline 不支持 cgroup

相关工作

- 李少华 <shli@kernel.org> 的 FIOPS
  - FIOPS (Fair IO Based Scheduler)
  - https://lkml.org/lkml/2012/1/30/28
  - sync/async
  - RT/BE/IDLE

新的调度器

- TPPS (Tiny Parallel Proportion Scheduler)
  - 通过 tpps_group 结构记录 group 的信息
  - 在 dispatch io 时，用该设备的 io request 容量（即 nr_request ）减去已经发出但还没有处理完成的 io 数（即 rq_in_driver ），得出的就是该设备还可处理的 io 数（即下面代码中的 quota ）
  - 然后根据这个“可处理 io 数”和各 group 的权重，算出各 group 的 list 上可以 dispatch 的 io 数，最后，按照这些数去 list 上取 io ，发出去


Cgroup
Group1
Group2
Group3
IO request
IO request
IO request
struct tpps_group
struct tpps_group
struct tpps_group
struct tpps_queue
struct tpps_queue
struct tpps_queue
struct tpps_data
Task1
Task2
Task3

```
static int tpps_dispatch_requests(struct request_queue *q, int force)
{
......
        if (unlikely(force))
                return tpps_forced_dispatch(tppd);

        if (!tppd->total_weight)
                return 0;

        quota = q->nr_requests - tppd->rq_in_driver;
        if (quota < MIN_DISPATCH_RQ)
                return 0;

        list_for_each_entry_safe(tppg, group_n, &tppd->group_list, tppd_node) {
                if (!tppg->nr_tppq)
                        continue;
                tpps_update_group_weight(tppg);
                grp_quota = (quota * tppg->weight / tppd->total_weight) - tppg-
>rq_in_driver;
......
```

开发

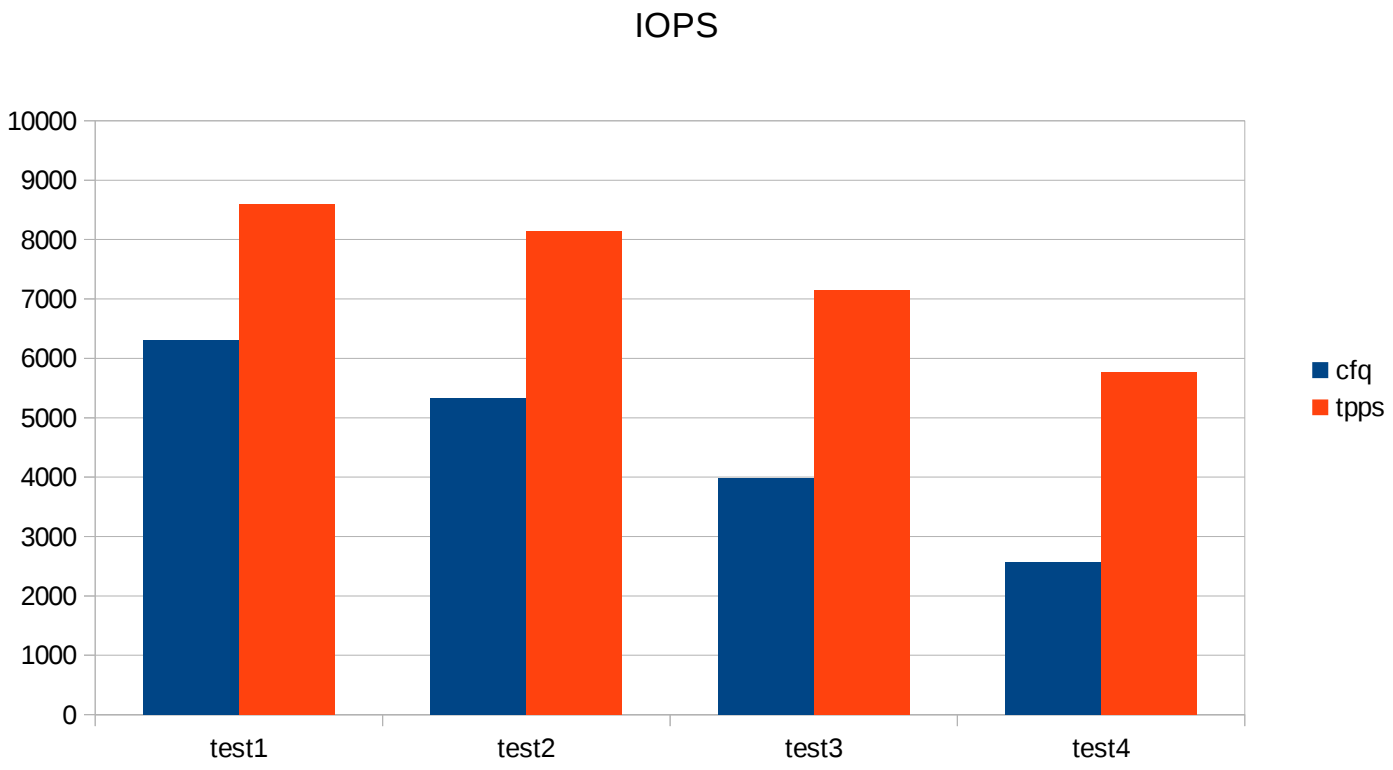
- 大部分框架借用 cfq——cfqq,cfqg 变成 tppq,tppg
- 用链表还是红黑树存放 io request ？

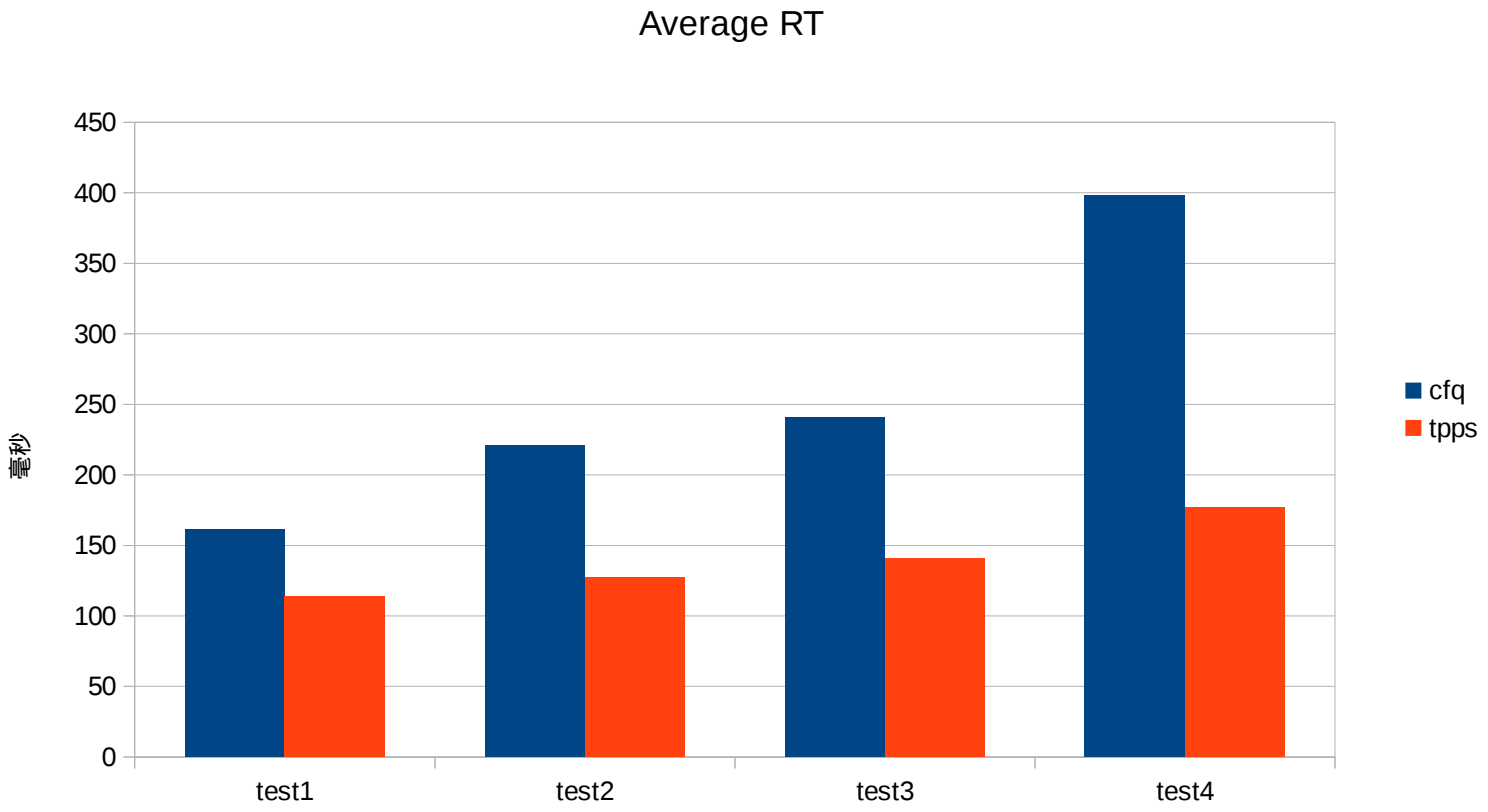
实施

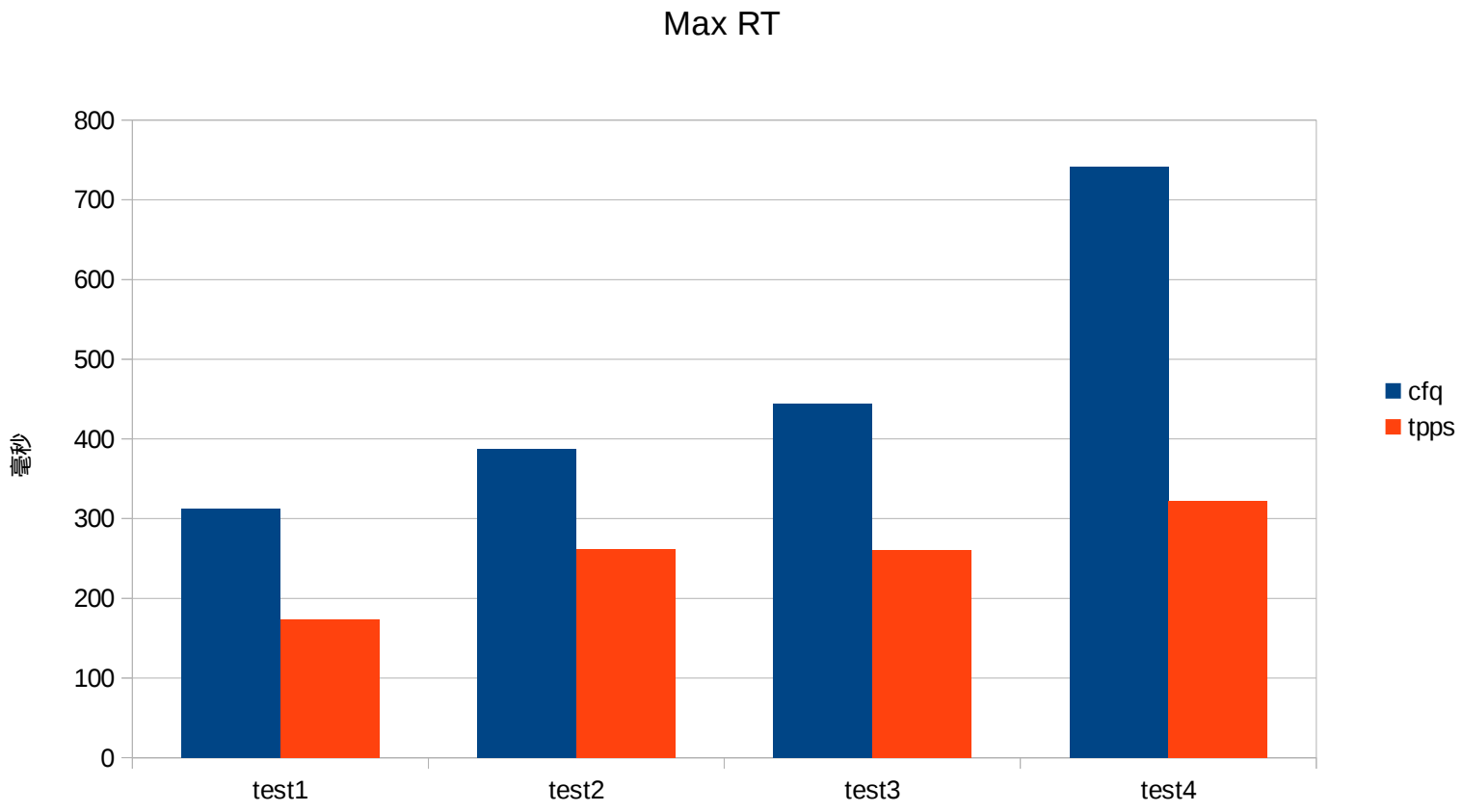
- https://github.com/alibaba/ali_kernel/commit/f3c5b7bbe26831eed44a18c8452a3803e67e7025
- 两个月开发，一个月测试，上线测试半年多

- FIO 测试脚本

```
[global]
direct=1
ioengine=libaio
runtime=20
bs=4k
rw=randwrite
iodepth=1024
filename=/dev/sdX
numjobs=4

[test1]
cgroup=test1
cgroup_weight=1000

[test2]
cgroup=test2
cgroup_weight=800

[test3]
cgroup=test3
cgroup_weight=600

[test4]
cgroup=test4
cgroup_weight=400
```


IOPS
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
test1
test2
test3
test4
cfq
tpps


Average RT
450
400
350
300
250
200
150
100
50
0
毫秒
test1
test2
test3
test4
cfq
tpps


Max RT
800
700
600
500
400
300
200
100
0
毫秒
test1
test2
test3
test4
cfq
tpps

思考

- 来自 flash 设备的挑战
  - 更快的速度
  - 随机性
- 来自应用程序方的挑战

淘宝网
Taobao.com

谢谢