

1 赛题名称

大容量SSD数据管理算法设计与实现

2 赛题描述

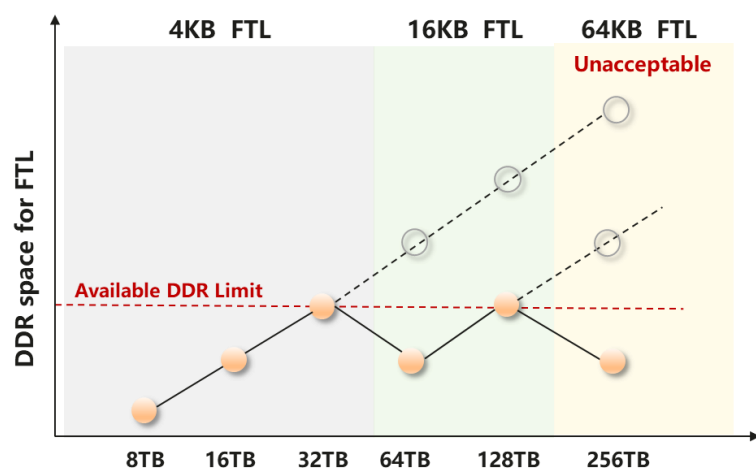
随着AI本地化部署密集推进，企业级存储正在向大容量、高密度持续进阶。全球主要存储制造商积极布局大容量存储市场，SSD迎来120TB+ 超高容量时代，且QLC技术已成为构建大容量 SSD 的主流方向。

针对大容量盘受限盘上内存限制，当前大容量SSD盘通常采用16KB、64KB等的大IO粒度：

SSD容量	FTL粒度	FTL表项个数	内存空间占用
256TB	4KB	64G	512GB @ 64bit
	16KB	16G	128GB @ 64bit
	32KB	8G	64GB @ 64bit
	64KB	4G	16GB @ 32bit

然而，系统侧应用通常采用小粒度 4KB FTL 数据管理方式，造成系统的适配和写入放大成本增加。通过FTL的优化设计可以保证小粒度FTL访问，同时保证SSD低时延特征。

本课题的主要挑战是：如何在内存容量受限情况下，仍然提供小粒度的FTL，从而提升大容量盘在系统侧的易用性。



3 相关要求

- 1) 设计一个低内存使用的SSD FTL模型，用于管理更大容量的SSD：以256TB容量SSD为例，可以额外使用盘上SSD容量空间，使用内存容量不高于16GB，用于访问IO大小为4KB，要求能够支撑正常的数据读写访问。
- 2) 参赛人员最终提交算法代码库、方案设计文档（含实验效果说明），方案设计文档需完整描述方案设计思路，达成效果及原理。
- 3) 提交的代码以专家组验证效果为最终效果，方案设计文档需经过专家组评审，以确保结果符合参赛的实验效果。

4 评分标准

判断不同work load下FTL算法的准确性、索引结构的内存容量占用以及访问时延，

复赛评分方法：基于准确率、时延、内存占用进行综合评分

复赛总分 = IO准确率得分+内存占用得分+时延性能得分

每个测试用例设置一定数量的预热写IO和 1M 测试IO，用例设置包括连续型、离散型、随机型以及混合型；

5 示例代码

5.1 任务说明

参赛选手将会负责设计和实现 FTL 索引结构，并提供 FTLRead 和 FTLModify 两个算法接口函数，其中，FTLRead 的输入为 lba，该函数负责从 FTL 中获取该 lba 对应的 ppn；FTLModify 的输入为 lba 和 ppn，该函数负责在 FTL 中建立 lba 和 ppn 的索引关系，FTL 粒度为 4KB。

示例代码提供了简单的 FTL 实现，输入一批盘内 IO 请求，要求对于写请求，执行 FTLModify，对于读请求，执行 FTLRead，并记录所获取的结果；最后统计运行时间、内存使用情况以及结果准确率。



输入文件：

- 1) input.txt：第一行为该文件包含的 IO 数量，后面的每一行为 [IOType, lba, ppn]，其中， $0 \leq lba \leq 2^{36} - 1$ ， $0 \leq ppn \leq 2^{36} - 1$ ；
- 2) val.txt：对应 input.txt 中所有读 IO 的结果；

要求输出：

output.txt：遍历 input.txt 中的所有 IO，记录由 FTL 算法获取的所有读 IO 的结果。

读写 IO 请求示例如下：

```
IOUint {
```

```
    IOType: IO_READ,  
  
    lba: 12,  
  
    ppn: 0, // 无意义  
}
```

```
IOUint {  
  
    IOType: IO_WRITE,  
  
    lba: 20120692,  
  
    ppn: 44451196877,  
}
```

5.2 接口函数

(1) 参赛选手需提供两个算法接口函数 FTLRead 和 FTLModify:

```
/**  
 * @brief FTLRead接口, 获取lba对应的ppn  
 * @param lba      输入参数: 逻辑地址  
 * @return uint64_t 返回 lba 对应的 ppn  
 */  
uint64_t FTLRead (uint64_t lba)
```

```
/**  
 * @brief FTLModify接口, 记录lba对应的ppn  
 * @param lba      输入参数: 逻辑地址  
 * @param ppn      输入参数: 物理地址  
 * @return bool     返回是否成功  
 */  
bool FTLModify (uint64_t lba, uint64_t ppn)
```

(2) 其他的 FTL 功能相关函数, 由参赛选手按需自行修改实现。

5.3 代码编译

```
mkdir build
```

```
cd build
```

```
cmake ..
```

```
make
```



5.4 运行测试

```
./project_hw -i ../dataset/input.txt -o ../dataset/output.txt -v ../dataset/val.txt
```