LearnedFTL
Trade-Off of CMT and GTD ratio

2024 FTL Study

2024. 8. 28

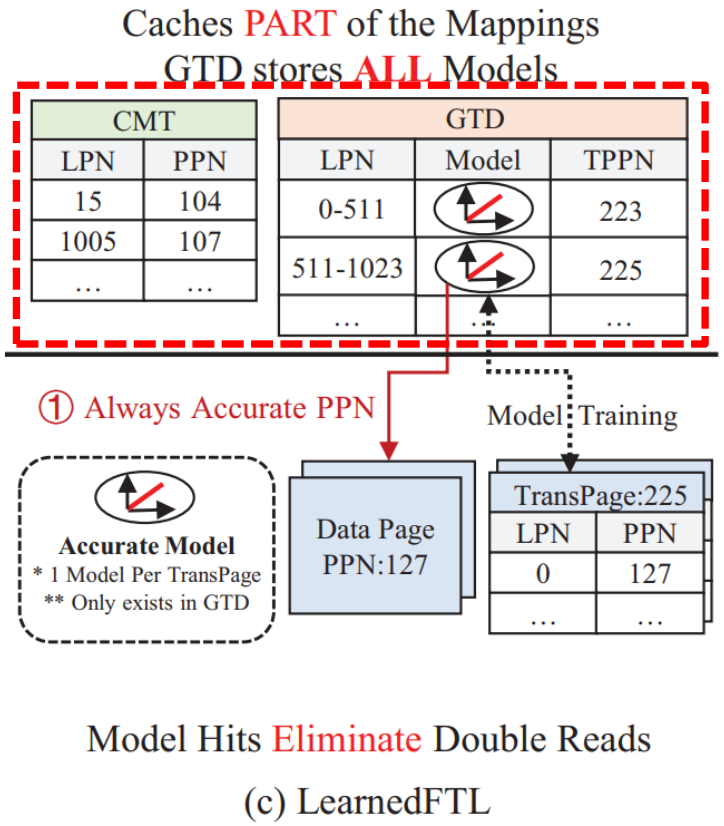
Presentation by MinSeong Kim

kms0509@dankook.ac.kr

Contents

1. Motivation
2. Evaluation

1. Motivation



"LearnedFTL은 모델의 예측 정확성을 보장하기 위해
rounding mode, bitmap filter를 사용하므로
그렇게 계산 정밀도가 높지 않다."

→ GTD 내 Model의 데이터 유형을 낮추고(Float → Float8)

→ 그만큼 CMT의 Size를 늘려보자!

→ 그렇다면 메모리의 크기는 그대로 유지하며
Random workload에서 hit ratio를 높일 수 있다 !

2. Evaluation

```
static void ssd_init_params(struct ssdparams *spp)
{
spp->tt_gtd_size = spp->tt_pgs / spp->ents_per_pg;
spp->tt_cmt_size = 8192;
spp->enable_request_prefetch = true;    /* cannot set false! */
spp->enable_select_prefetch = true;
```

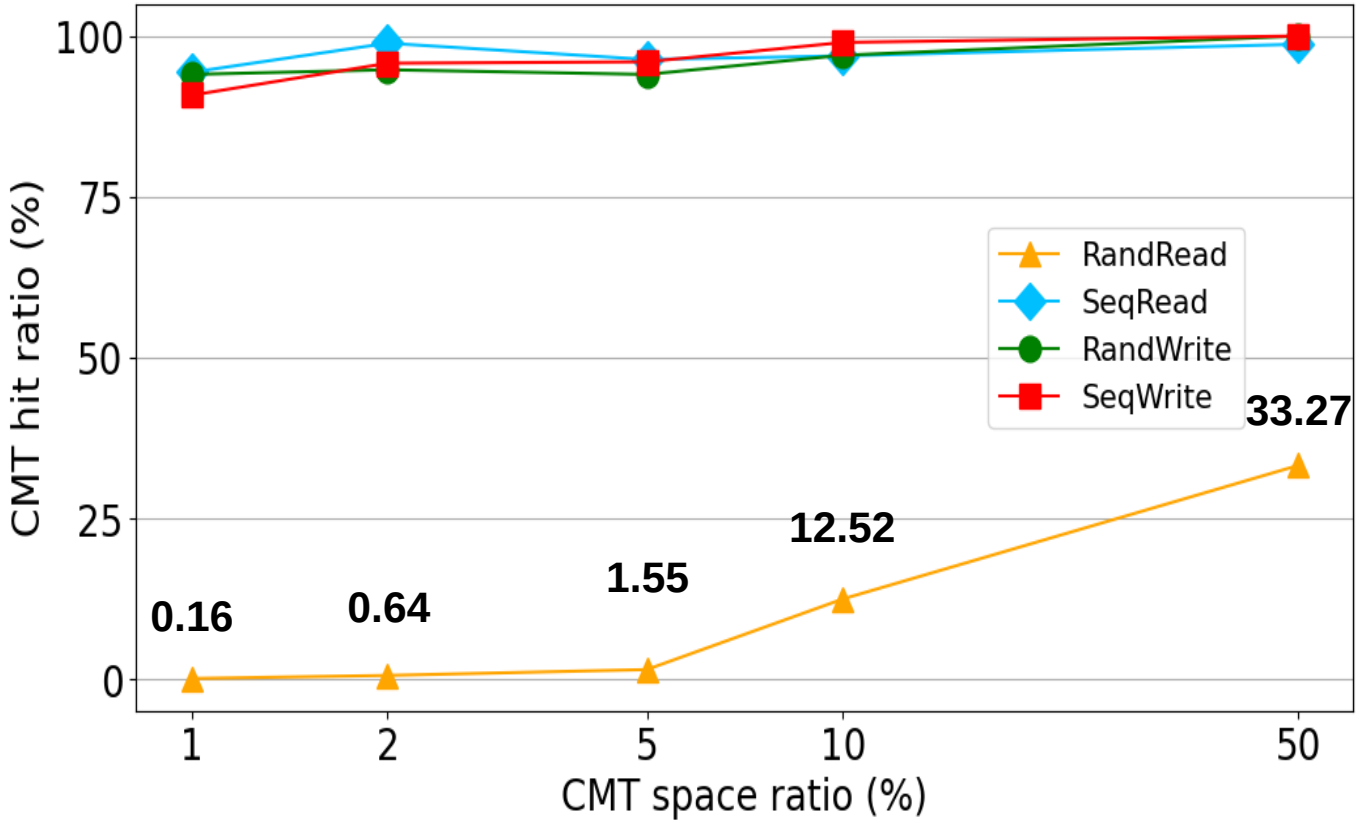
```
typedef struct cmt_entry {
    uint64_t lpn;// Logical Page Number
    uint64_t ppn;// Physical Page Number
    int dirty;// 엔트리의 수정 여부
    // int hotness;
    QTAILQ_ENTRY(cmt_entry) entry;// 큐에서의 위치
    bool prefetch;// 프리페치 여부
    uint64_t next_avail_time;// 다음 사용 가능 시간
    struct cmt_entry *next;// 해시 테이블에서의 다음 엔트리 (충돌 처리용)
} cmt_entry;
```

CMT의 개수: 8192(entry)

X = **524KB**

각 entry의 크기: 64B

2. Evaluation



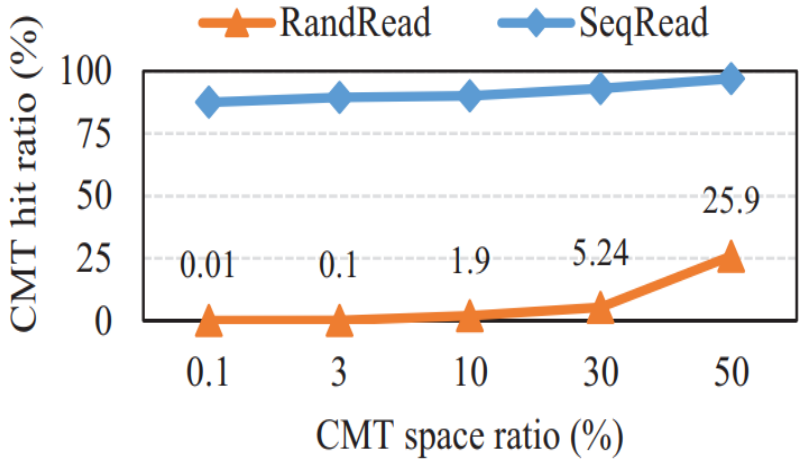
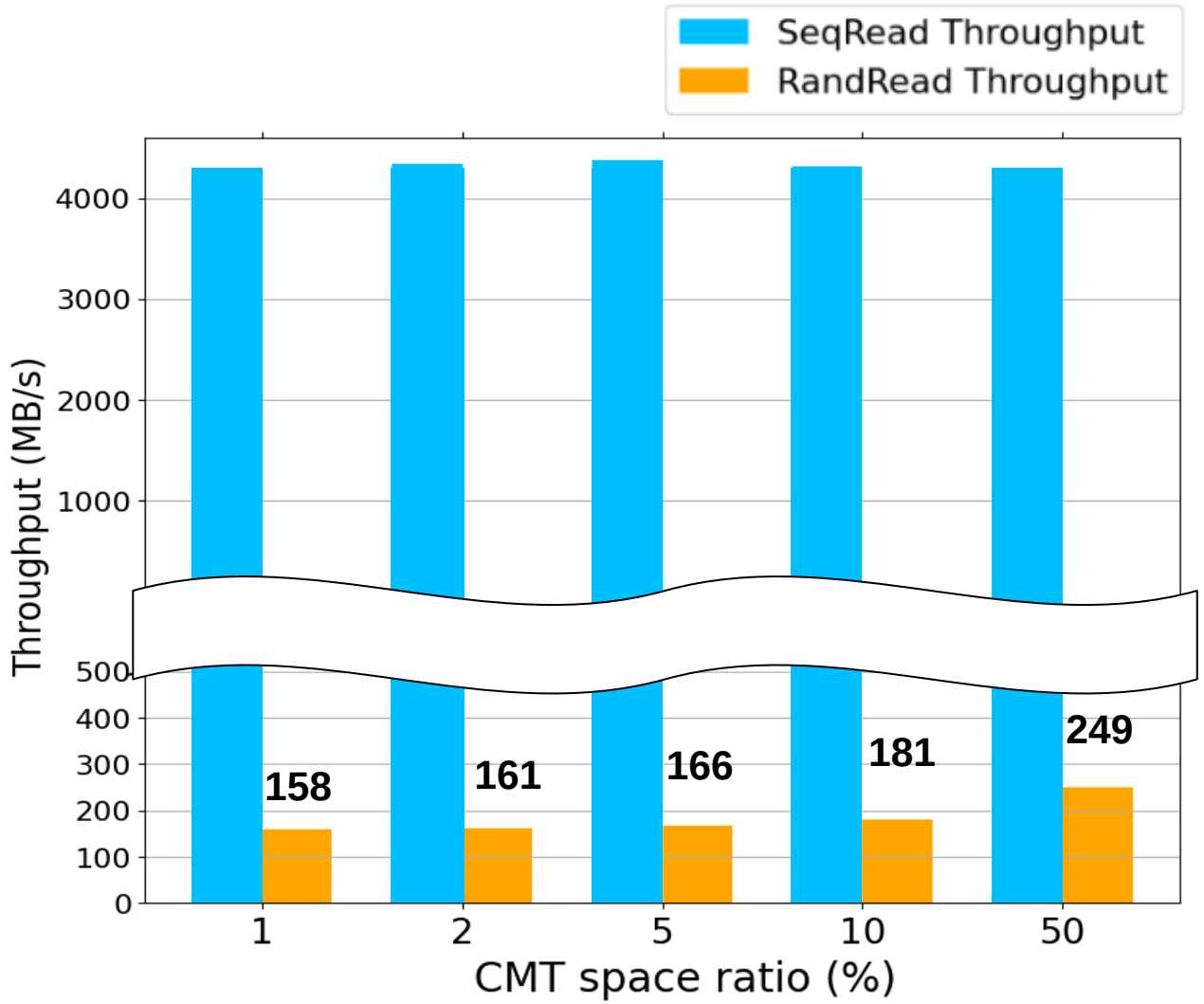


Fig. 3: The hit ratio of TPFTL under different CMT space.

2. Evaluation



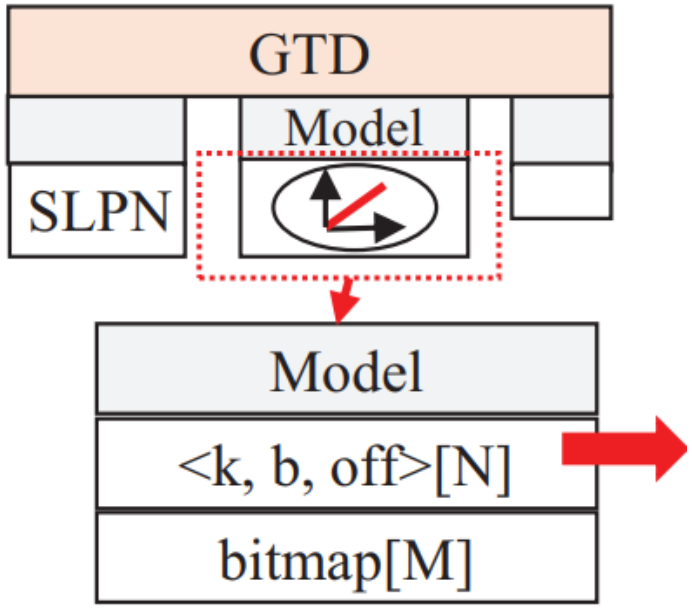
Sequential Read: 큰 변화폭 X

Random Read: CMT space ratio ↑
성능이 증가

2. Evaluation

CMT를 늘렸으니 GTD의 모델의 크기를 줄여야 한다.

GTD의 구조와 내부 Model의 크기를 살펴보면 한 entry의 모델의 크기는 **152B**이다.




```
typedef struct lr_breakpoint {
    float w;
    float b;
    int key;
    int valid_cnt;
}lr_breakpoint;
```

```
typedef struct lr_node {

    lr_breakpoint brks[MAX_INTERVALS];
    uint64_t start_lpn;
    uint64_t start_ppa;
    uint8_t u;
    uint8_t less;
    float success_ratio;
}lr_node;
```

GTD 내 entry의 개수를 확인해보면 16384인 것을 확인 가능
전체 Model의 크기: 152B X 16384 = 약 **2.5MB**

```
spp->addr_size = 8;
spp->pg_size = spp->secsz * spp->secs_per_pg;//512 * 8
spp->ents_per_pg = spp->pg_size / spp->addr_size; //512 * 8 / 8 = 512
spp->tt_trans_pgs = spp->tt_pgs / spp->ents_per_pg;//8388608 / 512 = 16384
```

2. Evaluation

그럼 어떻게 Model의 크기를 줄일까?

→ 모델의 매개변수를 Float8과 같은 낮은 데이터 유형을 사용

```c
typedef struct {
    uint8_t mantissa : 3;  // 유효 숫자 (3비트)
    uint8_t exponent : 4;  // 지수 (4비트)
    uint8_t sign : 1;      // 부호 (1비트)
} Float8;

Float8 float_to_float8(float f);
float float8_to_float(Float8 f8);
Float8 float8_add(Float8 a, Float8 b);
Float8 float8_subtract(Float8 a, Float8 b);
Float8 float8_multiply(Float8 a, Float8 b);
Float8 float8_divide(Float8 a, Float8 b);
```

ld-tpftl.h

```c
typedef struct lr_breakpoint {
    float8 w;
    float8 b;
    int key;
    int valid_cnt;
}lr_breakpoint;
```

```c
static void ssd_init_all_models(struct ssd *ssd) {
    struct ssdparams* sp = &ssd->sp;
    int avg_valid_cnt = sp->ents_per_pg / MAX_INTERVALS;

    for (int i = 0; i < sp->tt_gtd_size; i++) {

        ssd->lr_nodes[i].u = 1;
        for (int j = 0; j < MAX_INTERVALS; j++) {
            lr_breakpoint* brk = &ssd->lr_nodes[i].brks[j];
            brk->w = float_to_float8(1.0f);
            brk->b = float_to_float8(0.0f);

            // * all models' valid cnt is zero, to facilitate
            brk->valid_cnt = 0;
            brk->key = j * avg_valid_cnt;
        }
    }
}
```

ld-tpftl.c

2. Evaluation

Float8을 사용한 모델에서는 CMT space ratio를 증가시면
Sequential Read에서는 비슷한 성능을 보였다.