2021-07-15

mimalloc에 bitmap 포팅

- mimalloc 자체를 수정하여 bitmap 포팅에 성공하였습니다.
- mimalloc 자체 stress test 통과

포팅 자세한 사항

- allocation
  - 기존 mimalloc에서 사용하는 thread-local free list를 bitmap으로 교체하였습니다.
  - bitmap search를 빠르게 하기 위하여 bitmap index (bitmap of bitmap)을 추가하였습니다.
  - 또한, instruction lock이 수반되는 atomic `fetch_or` / `fetch_add` 를 사용하지 않고, atomic `load` / `store` 를 사용했습니다. (1 writer / 1 reader 이기 때문에 가능)
    - `BTS` / `BTR` 을 사용할 수도 있겠지만, atomic을 사용하면 instruction lock이 필요하고, relexed로 사용하면 `load` / `store` 와 혼용할 수 있을 지 모르겠습니다.
- free
  - mimalloc에서는 세개의 리스트를 이용하여 free를 구현합니다.
    - free list: 아직 allocation 되지 않은 free chunk
    - local free list: thread-local하게 free된 chunk, 주기적으로 free list로 merge 됨. 몇 프로그래밍 언어에서 `defered free` 를 구현하기 위해 사용됨
    - cross thread free list: thread를 건너가는 free
  - 다음과 같이 변경했습니다.
    - free list: bitmap으로 대체했습니다.
    - local free list: 새로운 bitmap을 할당하기에는 메모리 오버헤드가 생기므로, `defered free` 기능을 포기하고 free list와 합쳤습니다. (바로 free list로 free함)
    - cross thread free list: 리스트를 계속 이용합니다. bitmap을 사용하면 메모리 오버헤드가 생기고, 기존 리스트에서 사용하는 cross thread 동기화 매커니즘을 사용하지 못합니다.
  - 따라서, 다음 차이가 발생합니다.
    - 25-1024 크기의 allocation은 bitmap을 사용.
    - allocation시 bitmap을 참조하는 한번의 indirection이 발생
    - `defered free` 의 호출 빈도가 적어짐. (사용하지 않는 기능)
    - bitmap이 페이지의 최신 상태를 반영하지 않을 수 있음 (다른 thread에서 free된 항목이 alloc되어 있음) -> false positive이므로 bitmap을 이용하여 evicition하더라도 대역폭에 낭비가 조금 있을 뿐 correctness에는 문제가 없음

!! 4K Fetching vs SG bitmap fetching

Scatter-gather page fetch / eviction

- sg 방식으로 page를 fetch / eviction 하는 방식을 디자인 / 구현 했습니다.

RDMA scatter-gather API

하나의 WR에 보낼수 있는 SG 벡터는 다음과 같습니다.

```
struct ibv_send_wr {
 ...
 struct ibv_sge         *sg_list;                /* Pointer to the s/g
array */
 int                     num_sge;                /* Size of the s/g
array */
 ...
 uint64_t        remote_addr;
 uint32_t        rkey;
 ...
};


struct ibv_sge {
  uint64_t                addr;                   /* Start address of
the local memory buffer or number of bytes from the
                                                                    start
of the MR for MRs which are IBV_ZERO_BASED */
  uint32_t                length;                 /* Length of the
buffer */
  uint32_t                lkey;                   /* Key of the local
Memory Region */
};
```

- 하나의 WR로는 local memory의 SG 리스트를 지정할 수 있을 뿐, remote memory의 SG 리스트를 지정할 수는 없습니다.
- 따라서, 로컬의 SG 리스트로 지칭하는 메모리는 리모트에는 연속적으로 저장되게 됩니다.
- 둘 이상의 WR를 사용하기에는 QP의 길이가 길어지고, 프로그램 로직이 복잡해 지게 됩니다.

Page Table

- SG를 이용하여 Fetch를 구현하기 위해서는 어떤 subpage 영역을 가져올 것인가를 저장해 두어야 합니다.
- 이 정보를 page table entry에 저장을 했습니다.
- 기존에는 remote address를 가리키던 page table entry를 bitmap으로 변경하였습니다.

- remote address는 로컬 virtual address와 일치시켜 virtual address의 형태로 저장하게 하였습니다.
- PTE에 임베딩한 비트맵 정보를 보고 1로 세팅된 영역만 SG 리퀘스트로 가져오게 됩니다.
- 따라서 64 바이트 (4096 / 64) 단위로 가져올 수 있으며, 첫 128 바이트는 무조건 가져옵니다.

변경된 PTE

```
|              |    |     |    |         | w | v |
|              | 63 | ... | 48 | ....... | 1 | 0 |
| ------------ | -- | --- | -- | ------- | - | - |
| Local        |                          | X | 1 |
| Remote (all) |  all ones                | 0 | 0 |
| Remote (bit) |  bitmap                  | 0 | 0 |
| Empty        |  all zero                | 0 | 0 |
| Protect      |  CPU id                  | 1 | 0 |
```

Remote Driver

- 변경된 비트마스크를 사용하여 fetch / push 하는 리모트 IO driver를 제작하였습니다.

Eviction (구현중)

- bitmap을 page table에 임베딩하기 때문에 eviction이 일어나는 시점에 hint가 필요합니다.
- 이 힌트를 upcall 인터페이스를 이용하여 구현할 예정입니다.

Mimalloc Eviction (구현중)

- Mimalloc에서 사용하는 비트맵을 이용하여 eviction bitmap을 구성해야 합니다.
- mimalloc bitmap을 빠르게 eviction bitmap으로 바꿀 방법을 생각하고 있습니다.