Minimizing overhead of struct page

Muchun Song <songmuchun@bytedance.com>
STE - System Technology and Engineering

CLK 2021

ByteDance 字节跳动

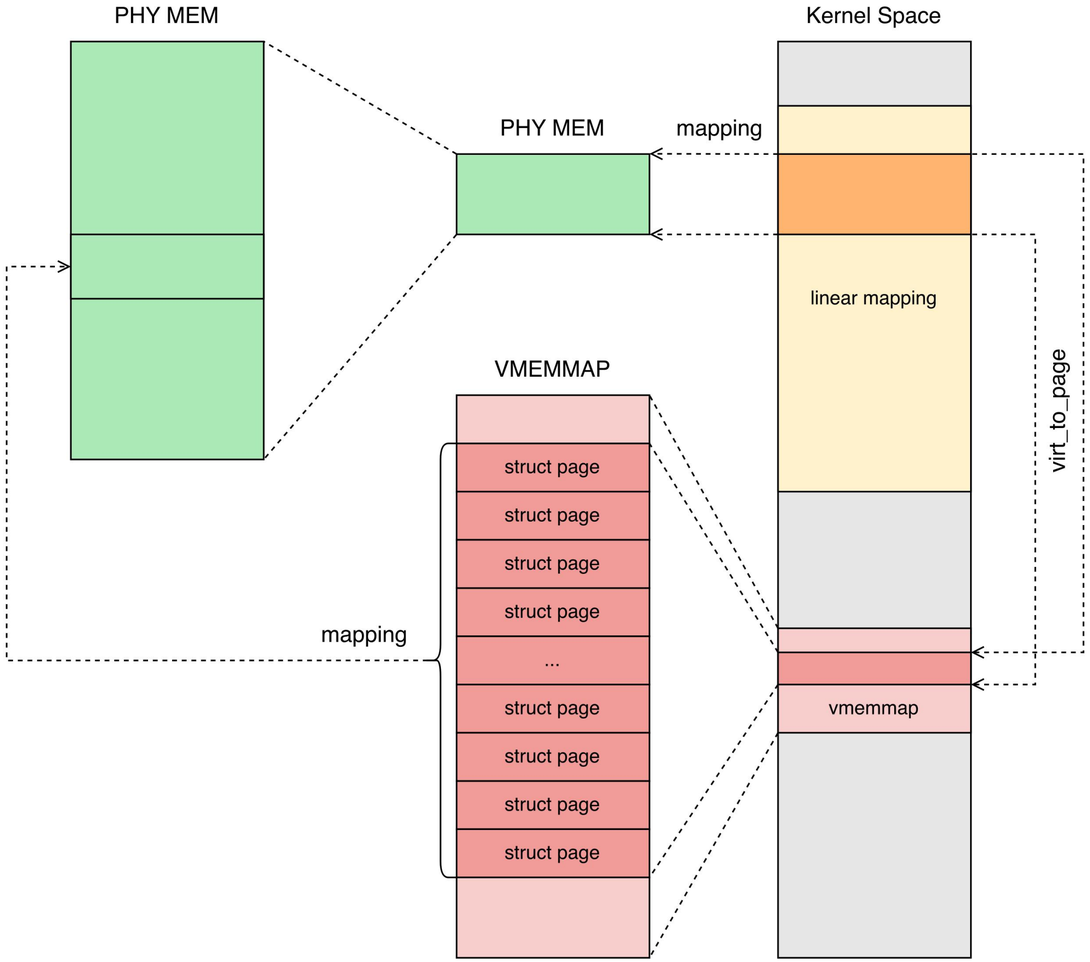
Agenda

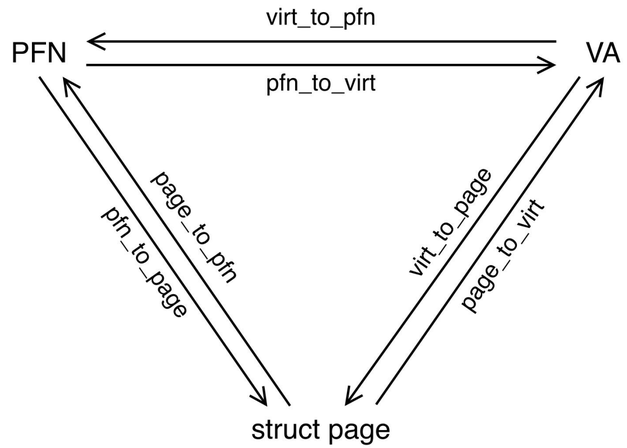
- Background
- Minimizing overhead of struct page
- Timeline of upstream work

Background

ByteDance 字节跳动

SPARSEMEM VMEMMAP





Features

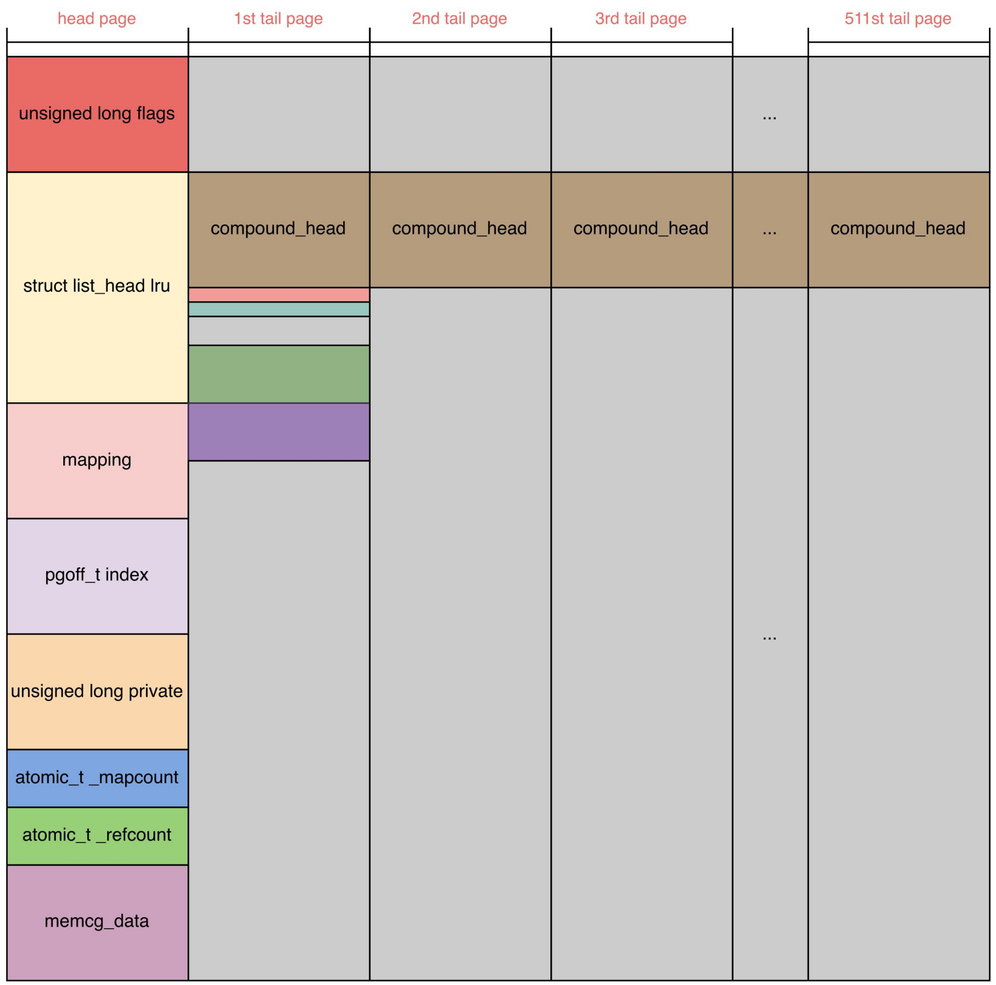
- One-to-one mapping between PFN and struct page.
- The status of physical page frames are tracked by using struct page.
- The size of struct page is 64 bytes on 64-bit system.
- PFN, struct page and VA (linear mapping) can be converted to each other.

Overhead of struct page

$$\frac{16\ GB}{1024\ GB} \qquad \frac{1\ TB}{64\ TB}$$

On x86_64, the overhead of struct page is about **1.56%** of total physical memory.

Organization of struct page of HugeTLB pages



Features

- The head struct page includes lots of main information.
- The 1st tail struct page includes some information as well.
- All tail struct pages except 1st only use filed of compound_head, and they all point to the head struct page.
- The struct page does not cross page boundaries.

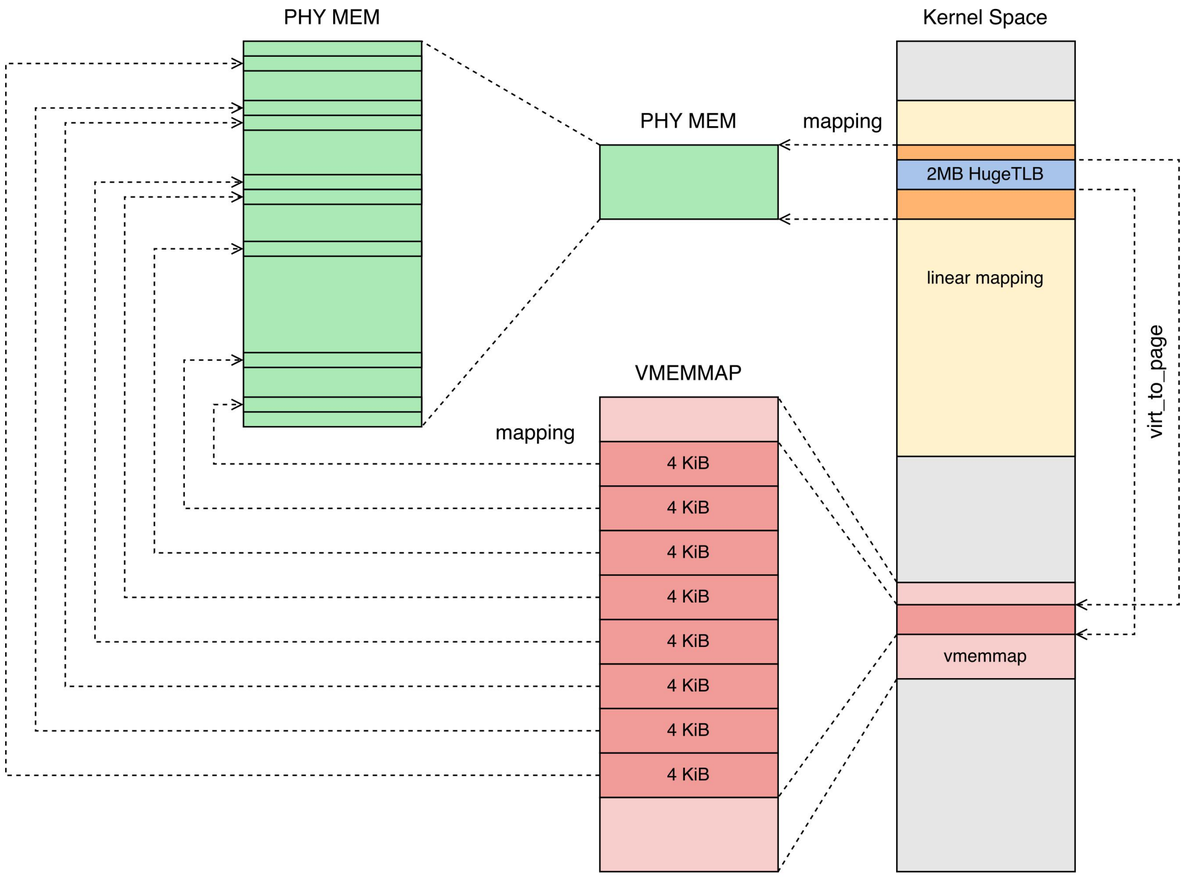
Challenges

Why do a huge page need **more than one** struct page?

Minimizing overhead of struct page

ByteDance字节跳动

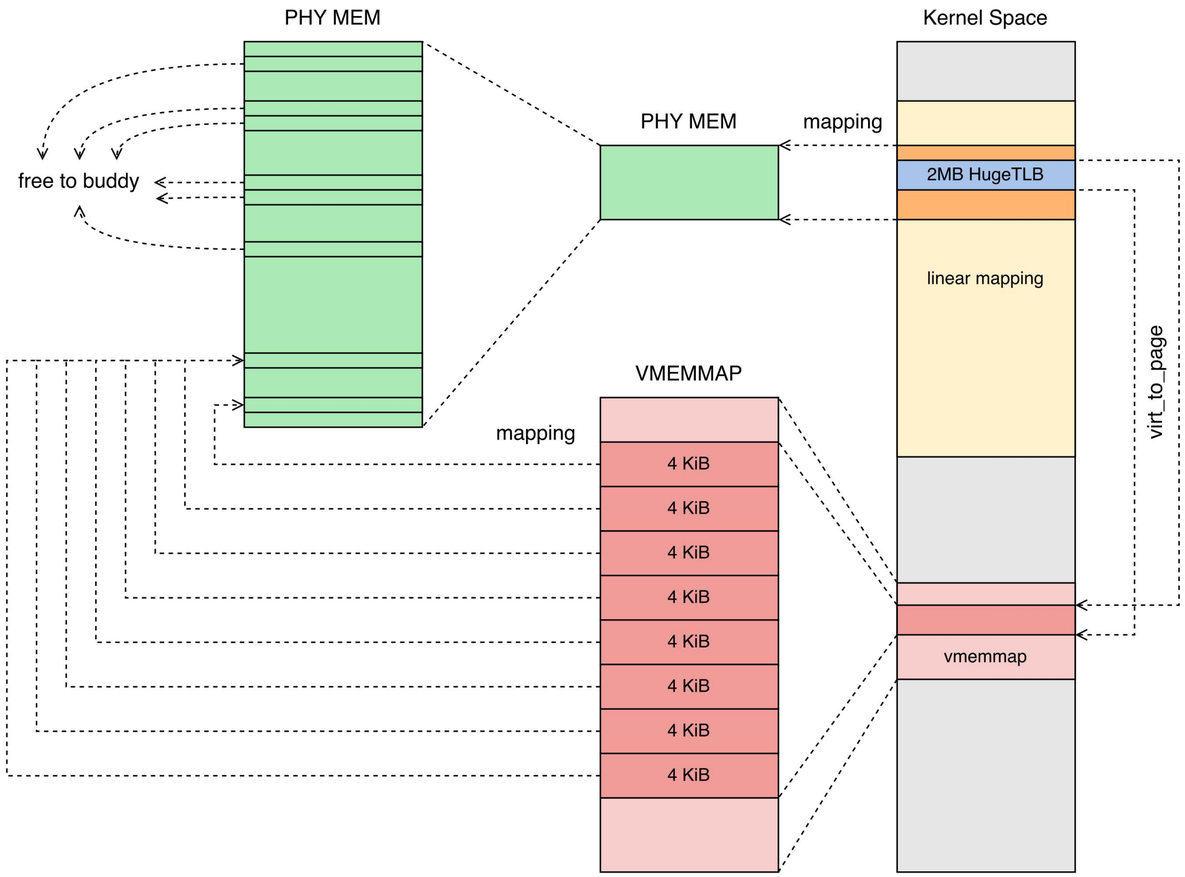
Vmemmap mapping



Features

- There are 32 KiB vmemmap pages associated with each 2 MB HugeTLB page.
- Every vmemmap page contains 64 struct page structures.

Optimization



Remap vmemmap pages 2 to 7 to page 1 and reuse 1st tail vmemmap page.

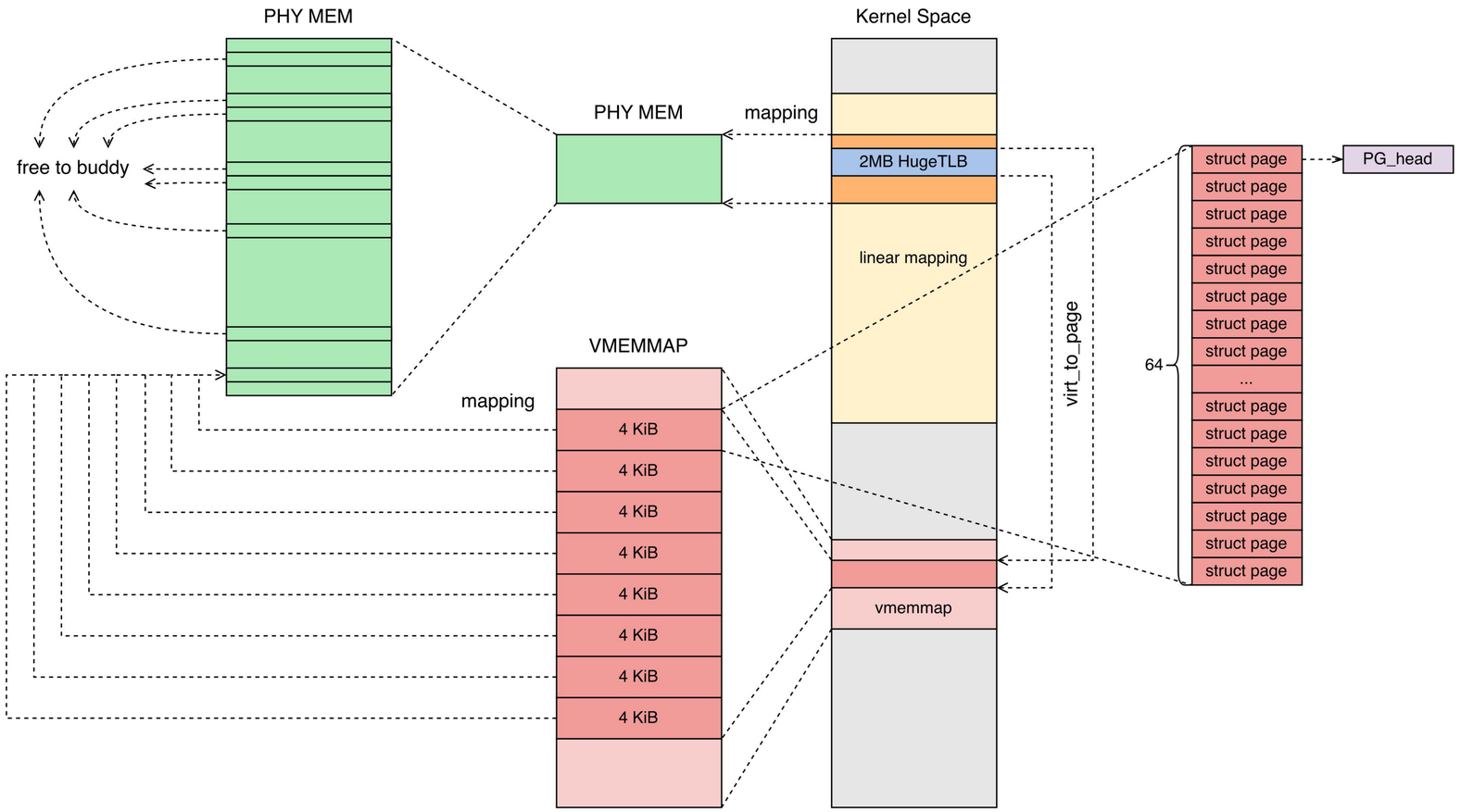
Memory gain

75% 2 MB

100% 1 GB

The overhead is reduced by about

Further optimization



How to identify a fake head page?

```
if (test_bit(PG_head, &page->flags)) {
        unsigned long head = READ_ONCE(page[1].compound_head);

        if (head & 1) {
                if (head == (unsigned long)page + 1)
                        ==> head page
                else
                        ==> tail page
        } else
                ==> head page
}
```

Remap vmemmap pages 1 to 7 to page 0 and reuse head vmemmap page.

Final memory gain

87.5% 2 MB

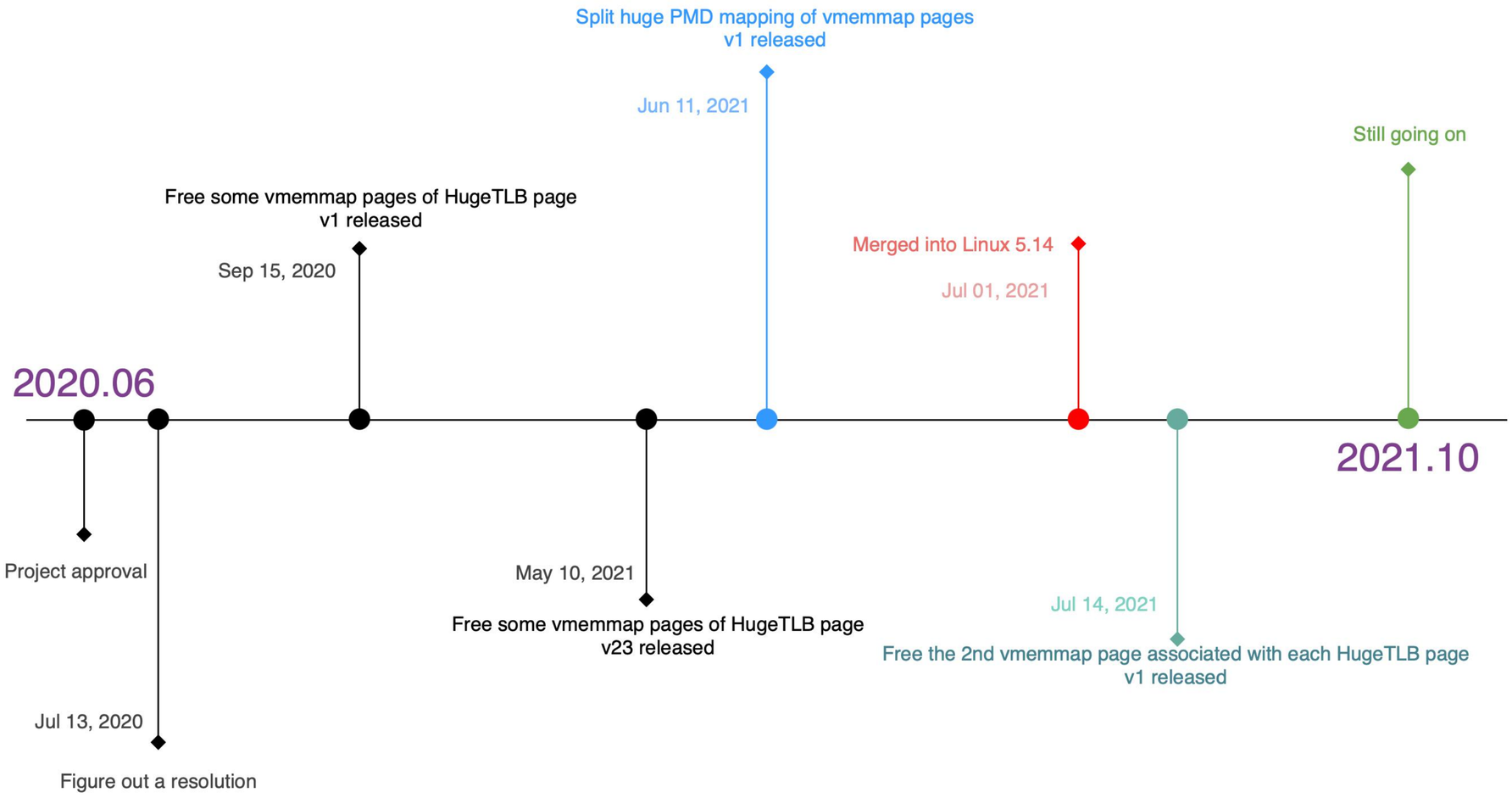
100.0% 1 GB

The overhead is reduced by about

Timeline of upstream work

ByteDance字节跳动

Timeline



THANKS

ByteDance字节跳动