CHINA LINUX KERNEL
中国Linux内核开发者大会

華中科技大學
网络空间安全学院
School of Cyber Science and Engineering, HUST

第19届中国Linux内核开发者大会

赞助单位

华中科技大学 | HUAWEI | 阿里云 | FUJITSU | 腾讯云
oppo | 火山引擎 | vivo | 龙芯中科 LOONGSON TECHNOLOGY | 蚂蚁集团 ANT GROUP

支持单位

清华大学 Tsinghua University | 迪捷软件 DIGIPROTO | OpenEuler | OpenAnolis 龙蜥社区 | OpenCloudOS

支持社区&媒体

CSDN | 云巅论剑 | sf 思否 | 阅码场 yomocode | InfoQ 极客传媒
51CTO | 开源江湖

2024年10月　湖北 • 武汉

華中科技大學

dma-buf 支持 DIO 的方案和收益

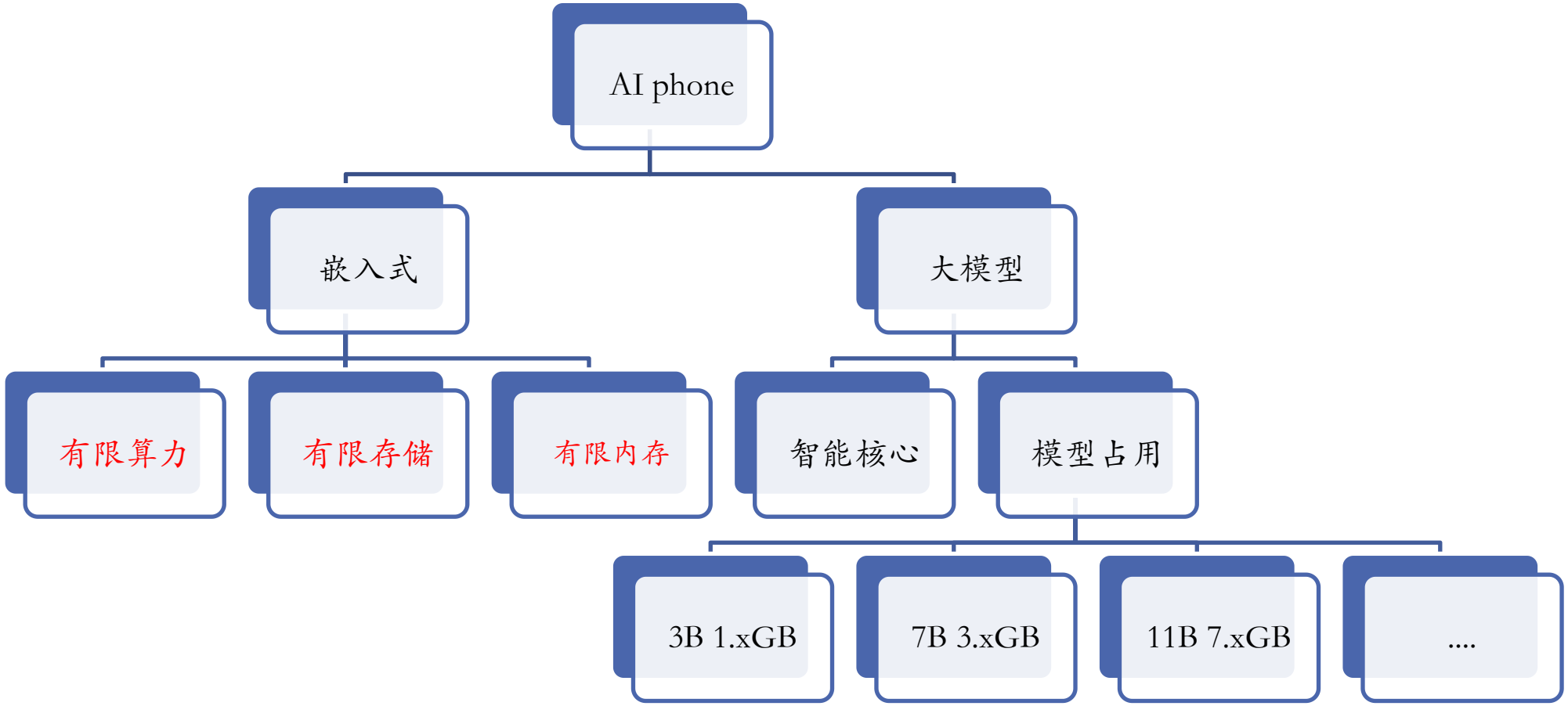
杨欢

vivo存储系统工程师

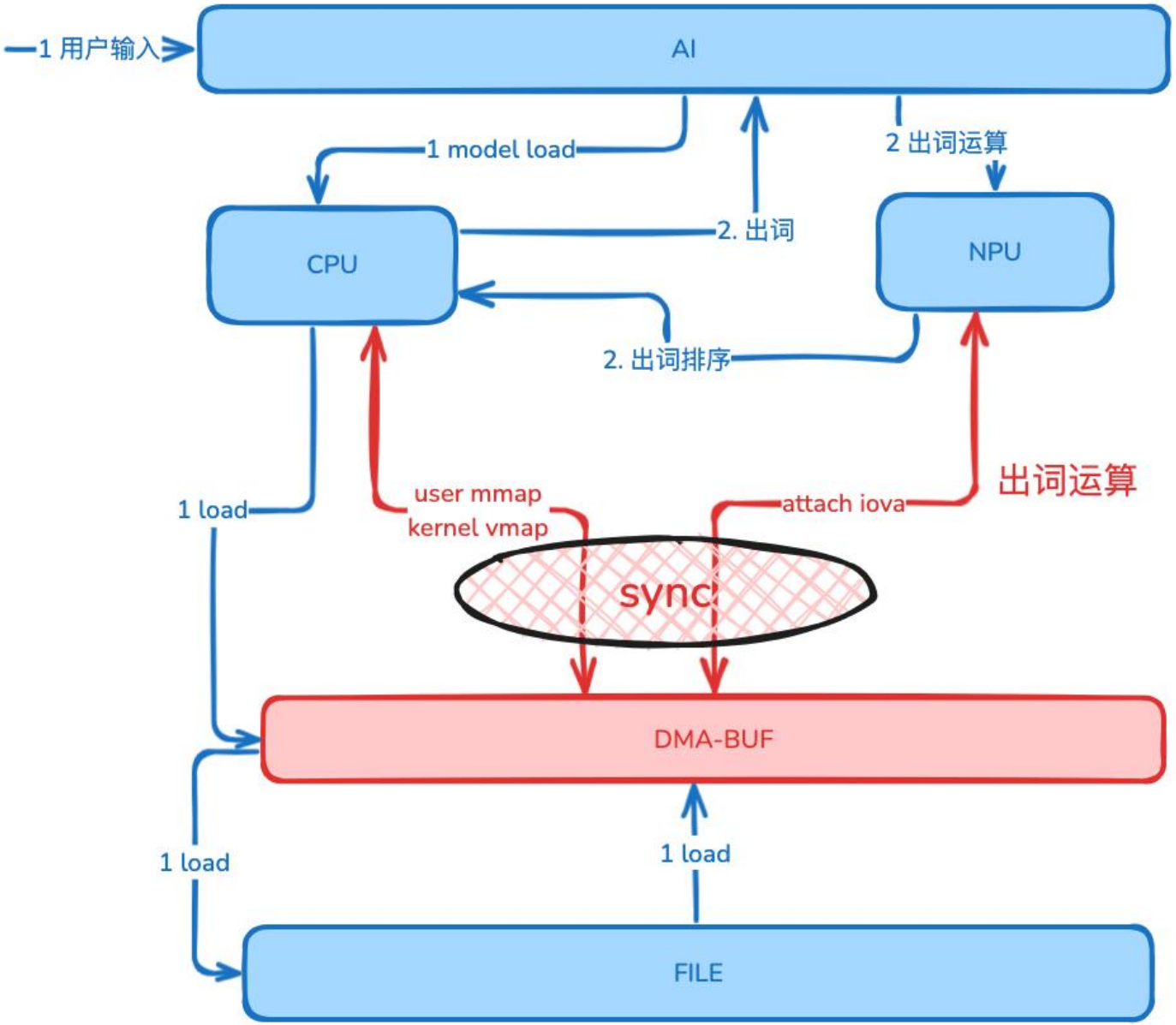
目录

- 背景介绍
- 优化方案
- 社区情况
- 未来展望

1. 背景 – 嵌入式有限资源的情况下流畅运行 AI 大模型遇到了巨大挑战



1. 背景 - dma-buf 作为模型文件在 CPU 和 DMA 设备间资源共享框架



CPU和NPU基于dma-buf高效协同进行模型运算

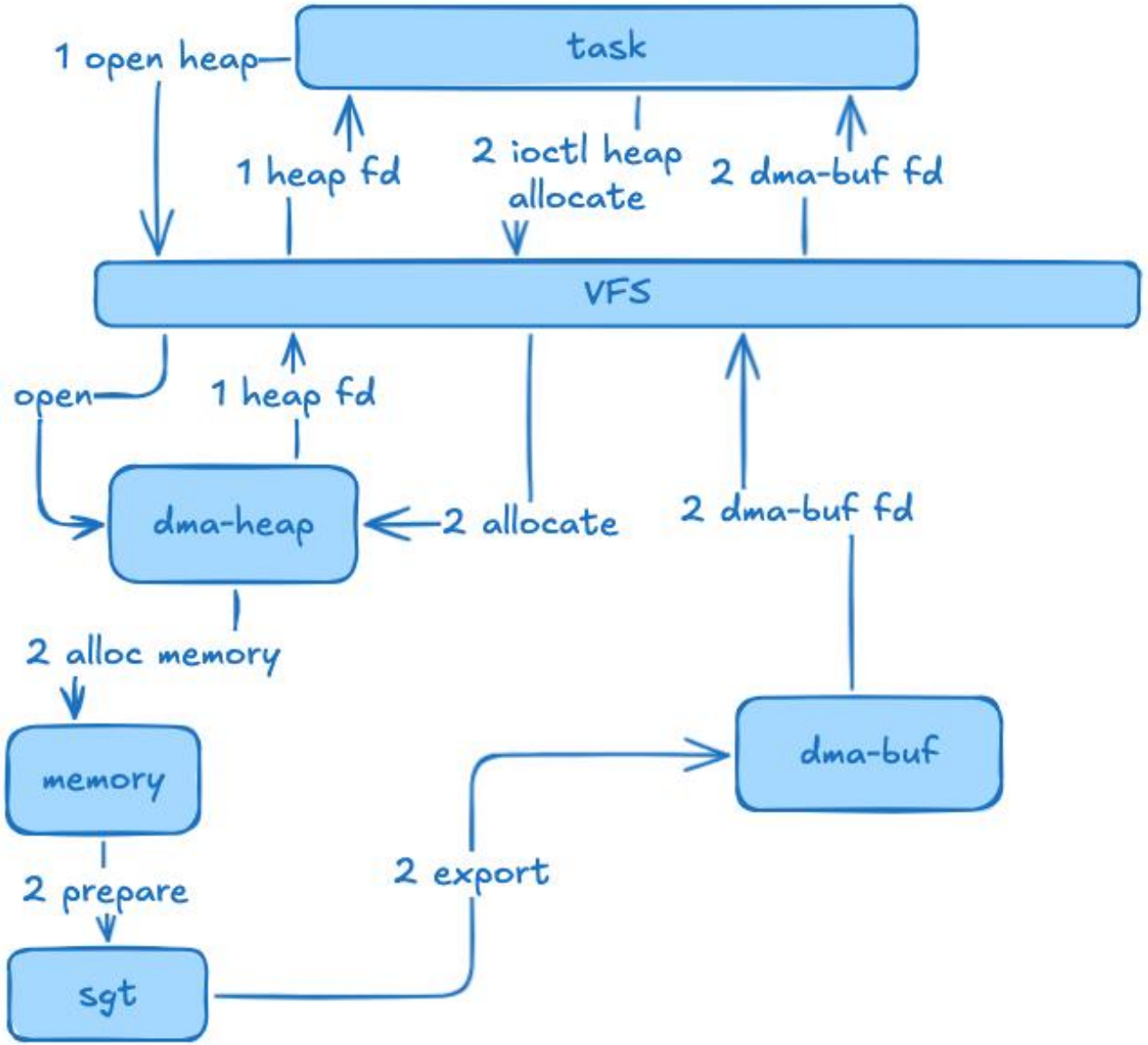
内存共享, 零拷贝

缓存管理

并发控制

1. 背景 — 需等 dma-buf 创建完毕后才能发起 IO

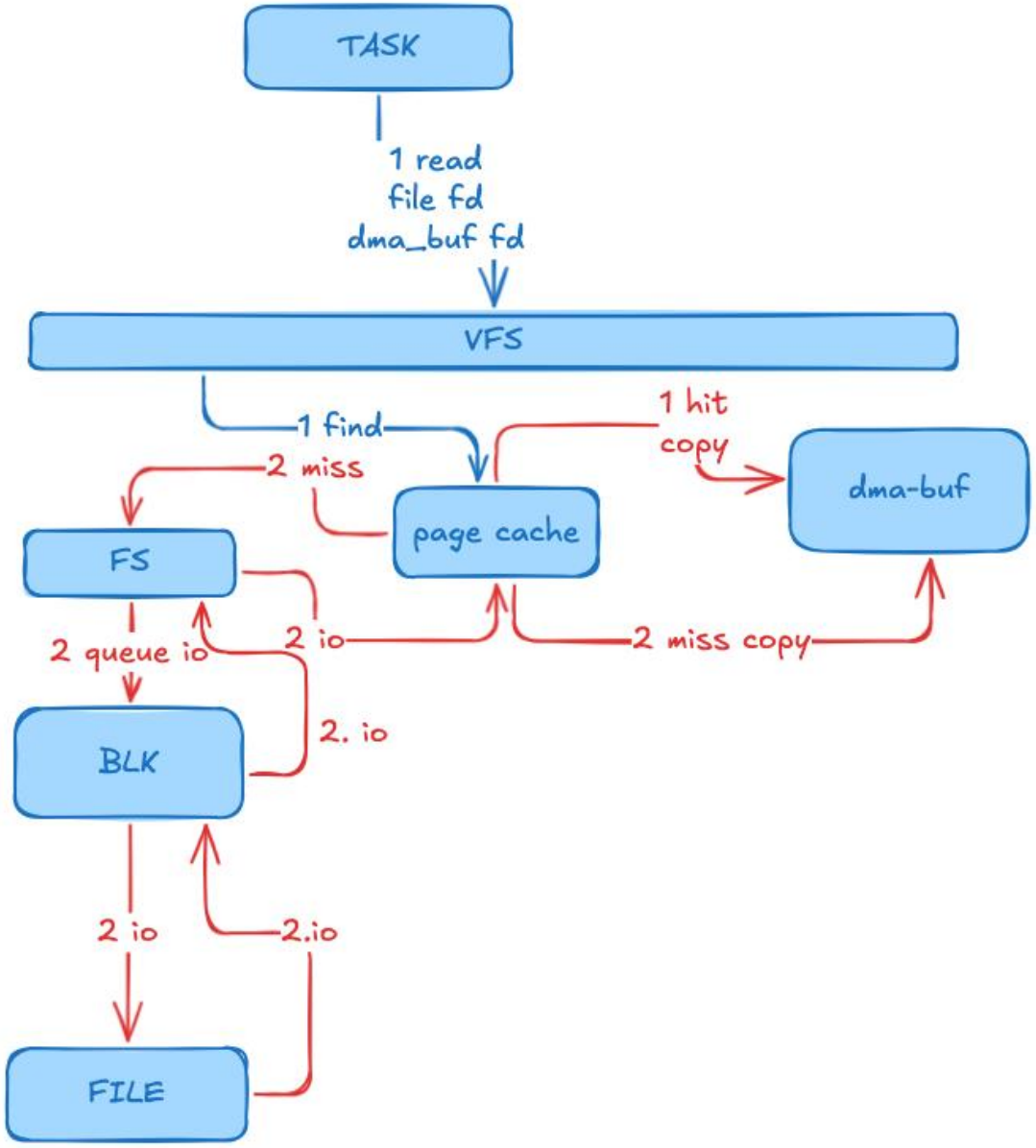
```c
// create heap_fd
int heap_fd = open("/dev/dma_heap/system", O_RDWR);

// get file and alloc size
int file_fd = open("file_path", O_RDONLY);
struct stat fstat;
fstat(file_fd, &fstat);
unsigned long file_size = fstat.st_size;

// create dma-buf fd and mmap it
struct dma_heap_allocation_data data = {
            .len = file_size,
            .fd_flags = O_RDWR | O_CLOEXEC,
            .heap_flags = 0,
};
ioctl(heap_fd, DMA_HEAP_IOCTL_ALLOC, &data);
int dma_buf_fd = (int)data.fd;
void *vaddr = mmap(NULL, fsize, PROT_WRITE, MAP_SHARED, dma_buf_fd, 0);

// trigger file read into dma-buf
read(file_fd, vaddr, fsize);
```



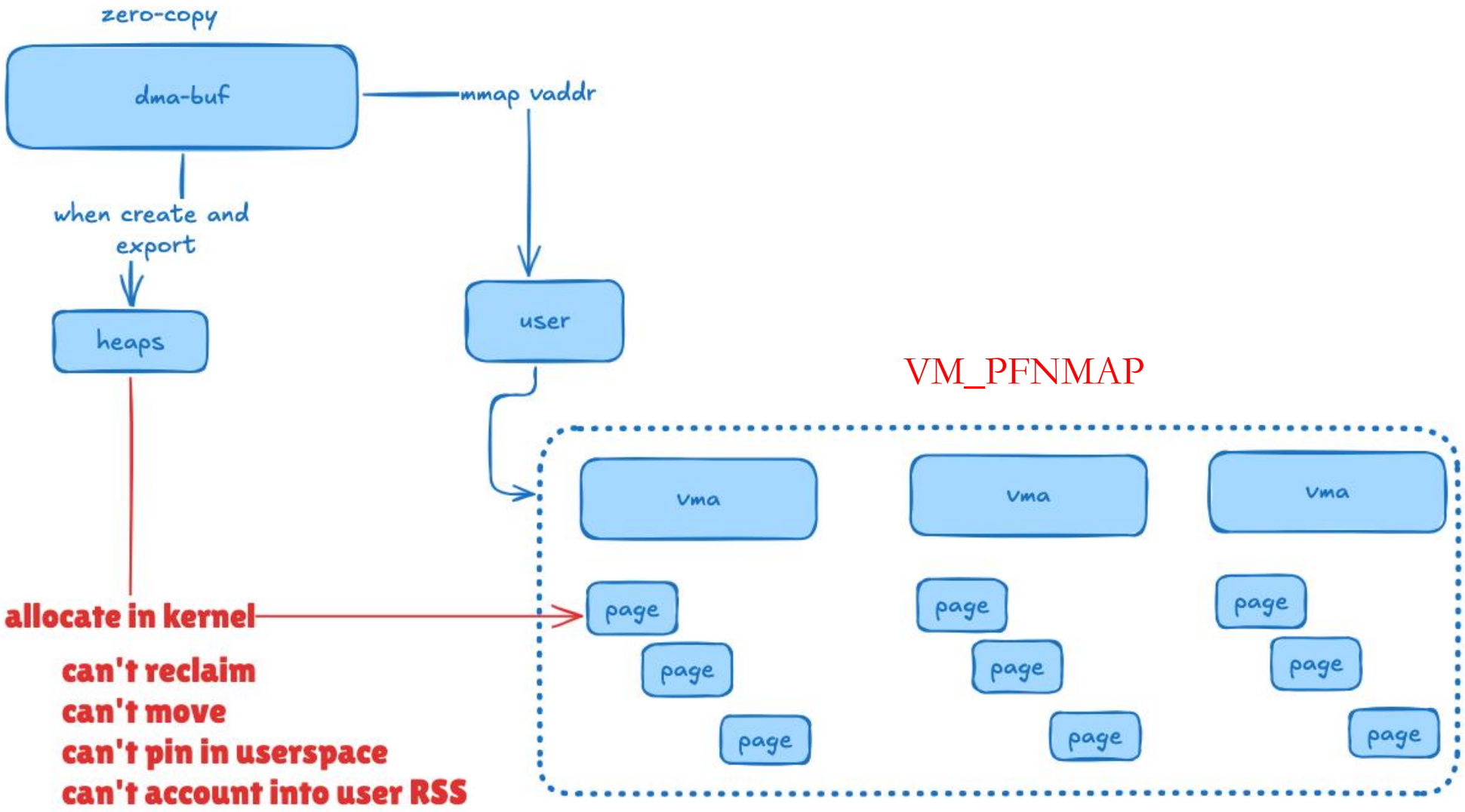
1. 背景 - dma-buf 采用 BIO 方式加载模型文件存在的两大问题

PAGE CACHE额外内存占用

COPY造成CPU额外消耗



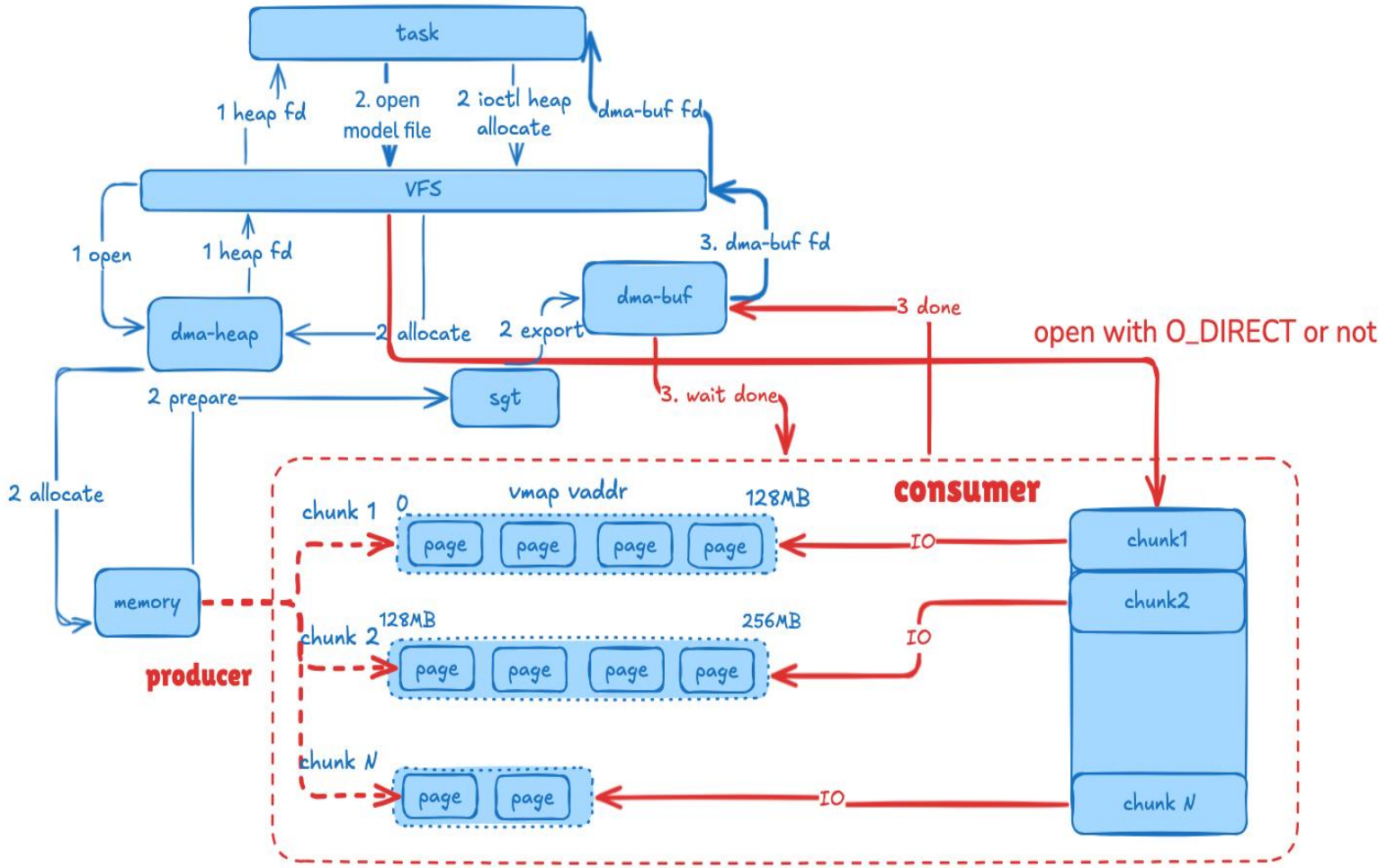
1. 背景 - dma-buf 用户态无法支持 DIO 的原因



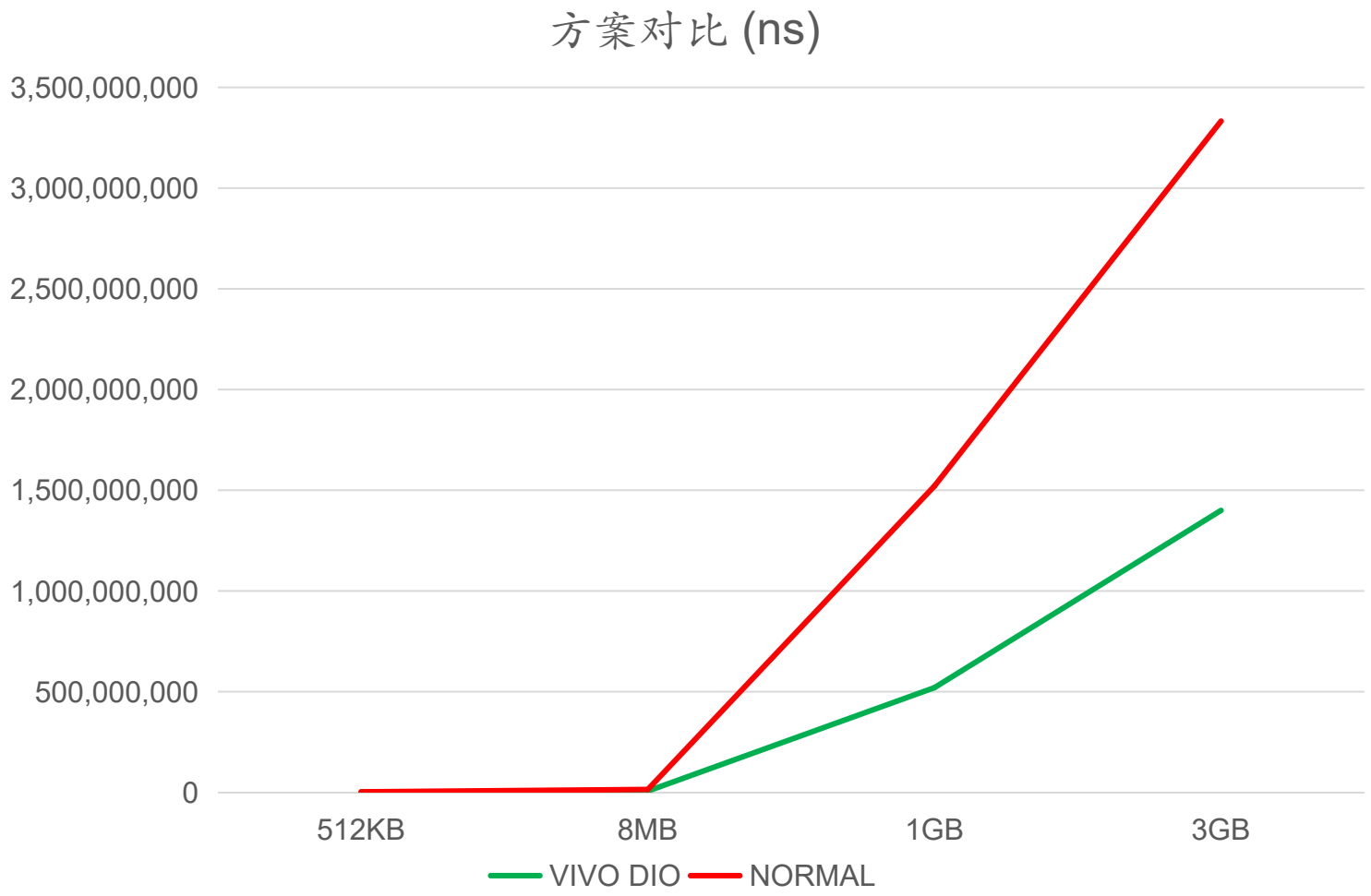
2. 优化方案 - 内核态并行读取

- 在内核态读取文件,struct page可管理
- 读取完再export dma-buf,避免并发竞争
- 内存申请和文件读取在生产-消费者模式下并行,提高效率

```
// create heap_fd
int heap_fd = open("/dev/dma_heap/system", O_RDWR);

// get file and alloc size with O_DIRECT or not is OK
int file_fd = open("file_path", O_RDONLY | O_DIRECT);

// create dma-buf fd and trigger file read
struct dma_heap_allocation_data data = {
            .file_fd = file_fd,
            .fd_flags = O_RDWR | O_CLOEXEC,
            .heap_flags = DMA_HEAP_ALLOC_AND_READ_FILE,
};
ioctl(heap_fd, DMA_HEAP_IOCTL_ALLOC, &data);
// by dma_buf_fd created, file also loaded into dma-buf
int dma_buf_fd = (int)data.fd;
```



2. 优化方案 – DIO 并行大幅提升大模型加载性能



优化后对比正常方式在3GB级别下有近65%的性能提升

3. 社区意见 – NAK, 有限场景下不同意修改内核

Thread overview: 26+ messages / expand[flat|nested] mbox.gz Atom feed top

```
2024-07-30  7:57 Huan Yang [this message]
2024-07-30  7:57 ` [PATCH v2 1/5] dma-buf: heaps: Introduce DMA_HEAP_ALLOC_AND_READ_FILE heap flag Huan Yang
2024-07-31 11:08   ` kernel test robot
2024-07-30  7:57 ` [PATCH v2 2/5] dma-buf: heaps: Introduce async alloc read ops Huan Yang
2024-07-30  7:57 ` [PATCH v2 3/5] dma-buf: heaps: support alloc async read file Huan Yang
2024-07-31 14:44   ` kernel test robot
2024-07-30  7:57 ` [PATCH v2 4/5] dma-buf: heaps: system_heap alloc support async read Huan Yang
2024-07-30  7:57 ` [PATCH v2 5/5] dma-buf: heaps: configurable async read gather limit Huan Yang
2024-07-30  8:03 ` [PATCH v2 0/5] Introduce DMA_HEAP_ALLOC_AND_READ_FILE heap flag Christian König
2024-07-30  8:14   ` Huan Yang
2024-07-30  8:37     ` Christian König
2024-07-30  8:46       ` Huan Yang
2024-07-30 10:43         ` Christian König
2024-07-30 11:36           ` Huan Yang
2024-07-30 13:11             ` Christian König
2024-07-31  1:48               ` Huan Yang
2024-07-30 17:19     ` T.J. Mercier
2024-07-31  1:47       ` Huan Yang
2024-07-30  8:56 ` Daniel Vetter
2024-07-30  9:05   ` Huan Yang
2024-07-30 10:42     ` Christian König
2024-07-30 11:33       ` Huan Yang
2024-07-30 12:04     ` Huan Yang
2024-07-31 20:46       ` Daniel Vetter
2024-08-01  2:53         ` Huan Yang
2024-08-05 17:53           ` Daniel Vetter
```

copy_file_range

- 需要源文件和目标文件为同一文件系统

splice/sendfile

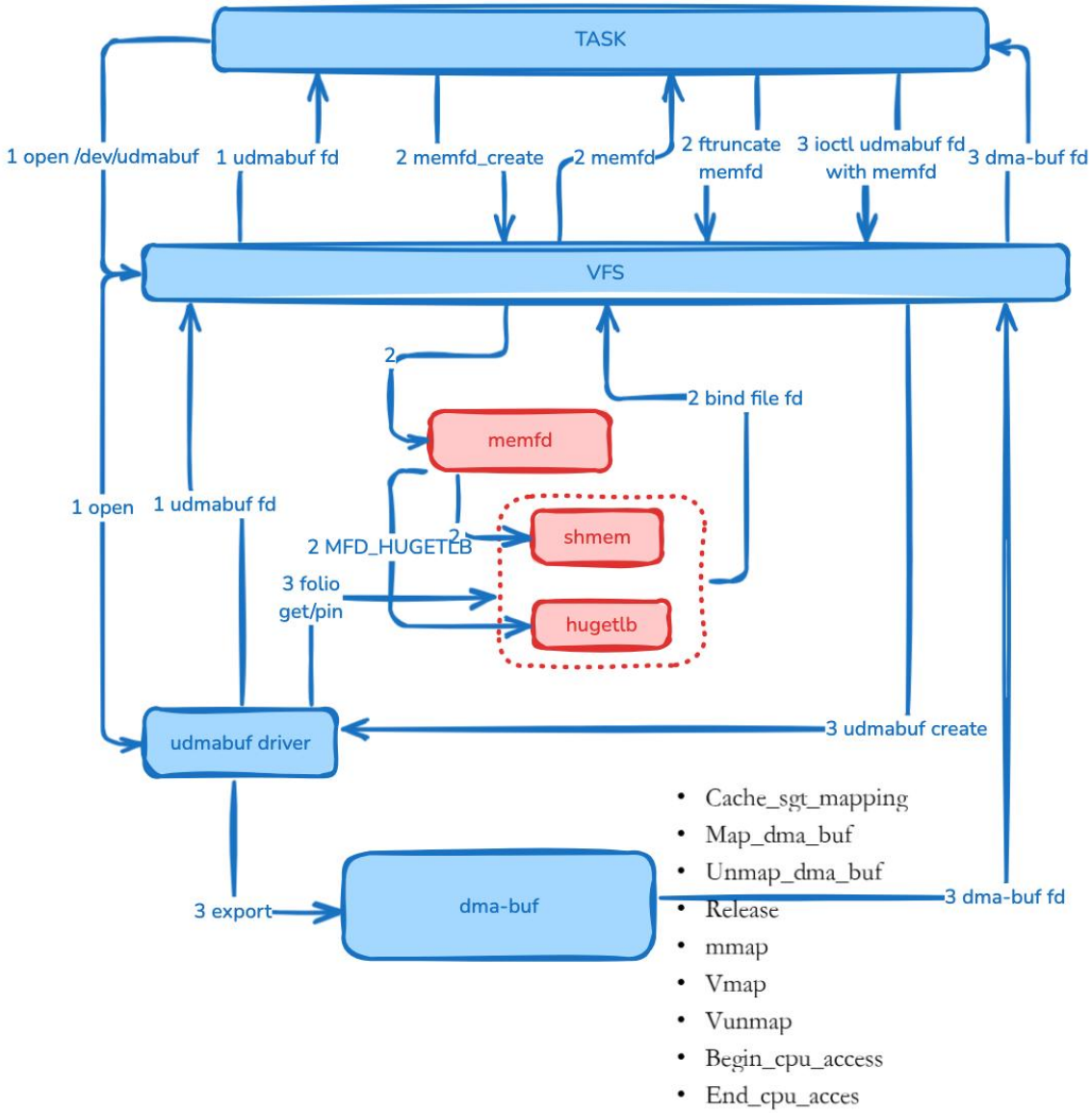
- 扩展splice_write等可以实现
- 基于pipe_buffer,即便使用DIO,依然需要中间形态的pipe_buffer
- pipe_buffer大小64KB,需要等待一个pipe_buffer read file完毕才能发起下一笔,效率很低

udmabuf

- memfd作为内存来源
- memfd支持DIO
- memfd读取文件过程可以和udmabuf创建过程并行

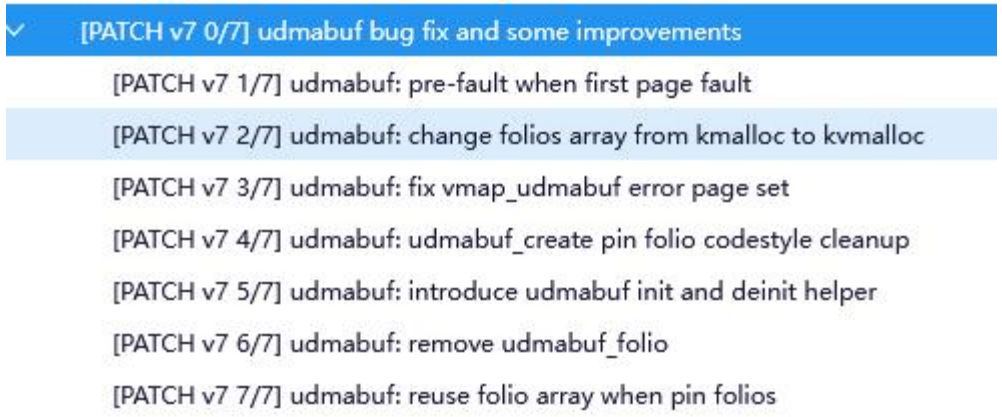
3. 社区情况 – udmabuf 可完成并行 DIO 读取

```c
int devfd = open("/dev/udmabuf", O_RDWR);

int memfd = memfd_create("udmabuf-test", MFD_ALLOW_SEALING);

int file_fd = open("./model.txt", O_RDONLY | O_DIRECT);
struct stat ftat;
fstat(file_fd, &ftat);
size = (ftat.st_size + getpagesize()) & ~(getpagesize());
ftruncate(memfd, size);

            //async thread
            void *vaddr = mmap(NULL, size, PROT_WRITE | PROT_READ,
                               MAP_SHARED, memfd, 0);
            read(file_fd, vaddr, size);

struct udmabuf_create create;
create.memfd  = memfd;
create.offset = 0;
create.size   = size;
int dma_buf_fd = ioctl(devfd, UDMABUF_CREATE, &create);
// wait async read done
```



3. 社区情况 – vivo 对 udmabuf 的提交

- pre-fault加速mmap page的获取
- 修复udmabuf size超过2G创建失败问题（buddy alloc导致）
- vmap等适配HVO，避免用page struct，而是使用pfn
- 对于create过程的代码简化和性能提升
- google后续将在安卓上开启udmabuf



> udmabuf is good, but I think our oem driver can't suit it. (And, AOSP do not open this feature)

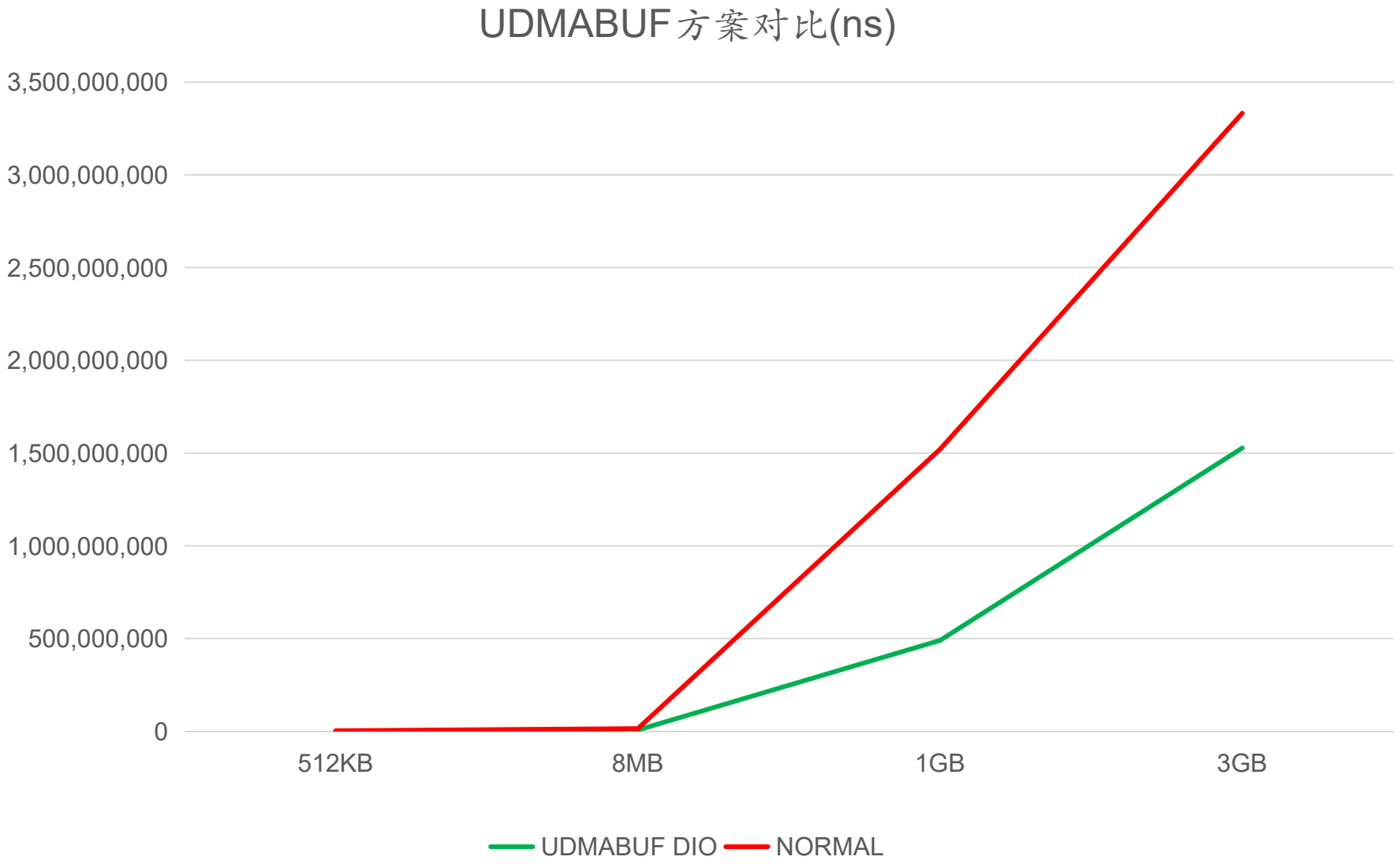
Hi Huan,

We should be able to turn on udmabuf for the Android kernels. We don't have CONFIG_UDMABUF because nobody has wanted it so far. It's encouraging to see your latest results!

-T.J.

https://lore.kernel.org/all/20240918025238.2957823-1-link@vivo.com/

3. 社区情况 – udmabuf 并行DIO收益



4. 展望

- udmabuf驱动是固化的，是否可以集成到dma heaps中，提高扩展性?
- 让一些架构的dma_mmap强制设置VM_SPECIAL, 避免有些驱动未使用VM_PFNMAP映射dma-buf导致问题
- DIO能否判断page是pfn base? 如果是 就不尝试pin page 而是直接发起?

```
On Wed, Jul 10, 2024 at 04:14:18PM +0200, Christian König wrote:
> Am 10.07.24 um 15:57 schrieb Lei Liu:
> > Use vm_insert_page to establish a mapping for the memory allocated
> > by dmabuf, thus supporting direct I/O read and write; and fix the
> > issue of incorrect memory statistics after mapping dmabuf memory.
>
> Well big NAK to that! Direct I/O is intentionally disabled on DMA-bufs.
>
> We already discussed enforcing that in the DMA-buf framework and this patch
> probably means that we should really do that.

Last time I looked dma_mmap doesn't guarantee that the vma end sup with
VM_SPECIAL, and that's pretty much the only reason why we can't enforce
this. But we might be able to enforce this at least on some architectures,
I didn't check for that ... if at least x86-64 and arm64 could have the
check, that would be great. So might be worth it to re-audit this all.

I think all other dma-buf exporters/allocators do only create VM_SPECIAL
vmas.
-Sima
--
Daniel Vetter
Software Engineer, Intel Corporation
http://blog.ffwll.ch
```

https://lore.kernel.org/all/ZpTjR-7dabdyREXS@phenom.ffwll.local/#t

Thank You~