Linux任督二脉之内存管理(四)

讲解时间：1月29日、1月30日、1月31日、2月1日、2月2日晚9点

宋宝华 <21cnbao@gmail.com>

扫描二维码报名

微信群直播：

https://mp.weixin.qq.com/s/wJye_syLYVZdzXo5U1fJug

麦当劳喜欢您来，喜欢您再来

扫描关注

Linuxer

内存与I/O的交换

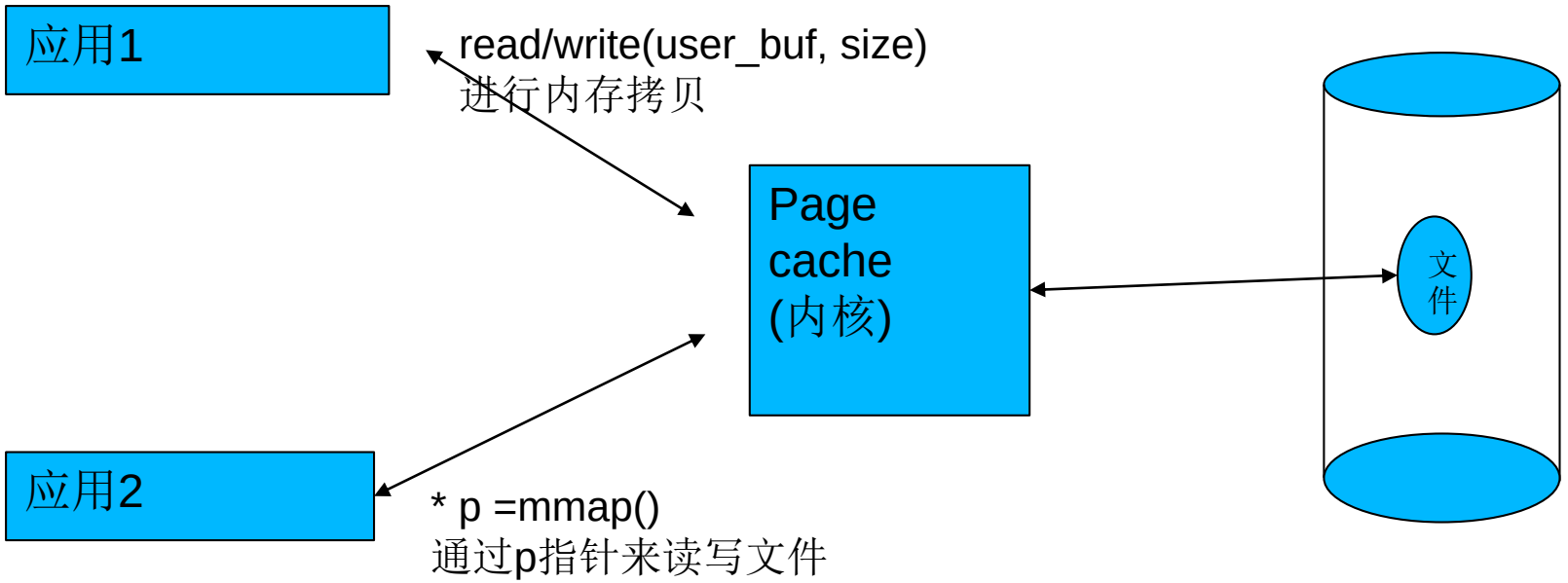
*page cache
*free命令的详细解释
*read、write和mmap
*file-backed的页面和匿名页
*swap以及zRAM
*页面回收和LRU

练习题

*把hello，python运行两次，对比时间差；
*free，cat /dev/sda > /dev/null, free，观察变化，分析原因；

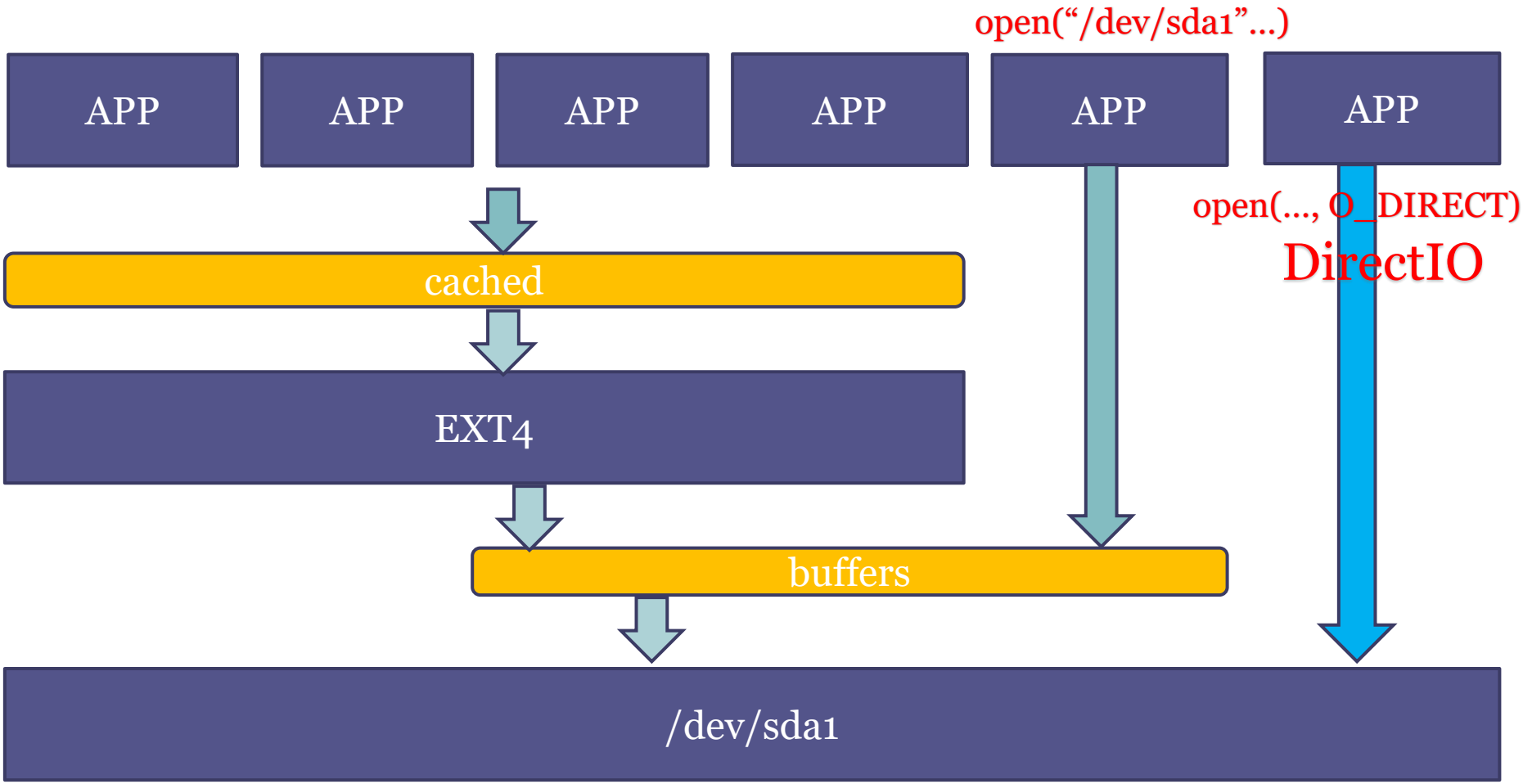
page cache

- 在Linux读写文件时，它用于缓存文件的逻辑内容，从而加快对磁盘上映像和数据的访问



Page cache的两种形式

- 以文件系统中的文件为背景：cached
- 以裸分区/dev/sdax等为背景：buffers



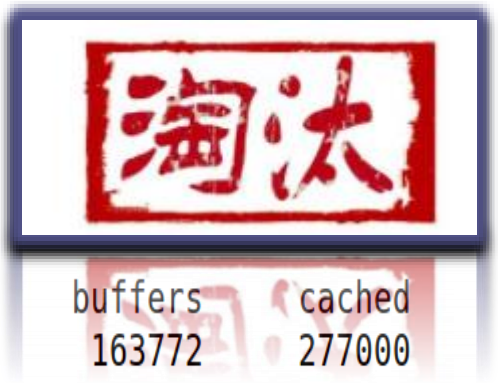
老版free命令

- Used5=1-3-4
- Free6=2+3+4

```
baohua@baohua-VirtualBox:~$ free
             total       used       free     shared    buffers     cached
Mem:       1024700    921372 1  103328 2    3276     163772 3  277000 4
-/+ buffers/cache:    480600 5  544100 6
Swap:       522236        360     521876
```

新版free命令

- 新版free命令加了available，删除了第2行




```
baohua@baohua-VirtualBox:~$ free
             total       used       free     shared    buffers     cached
Mem:       1024700     921372     103328       3276     163772     277000
-/+ buffers/cache:     480600     544100
Swap:       522236        360     521876
```

available

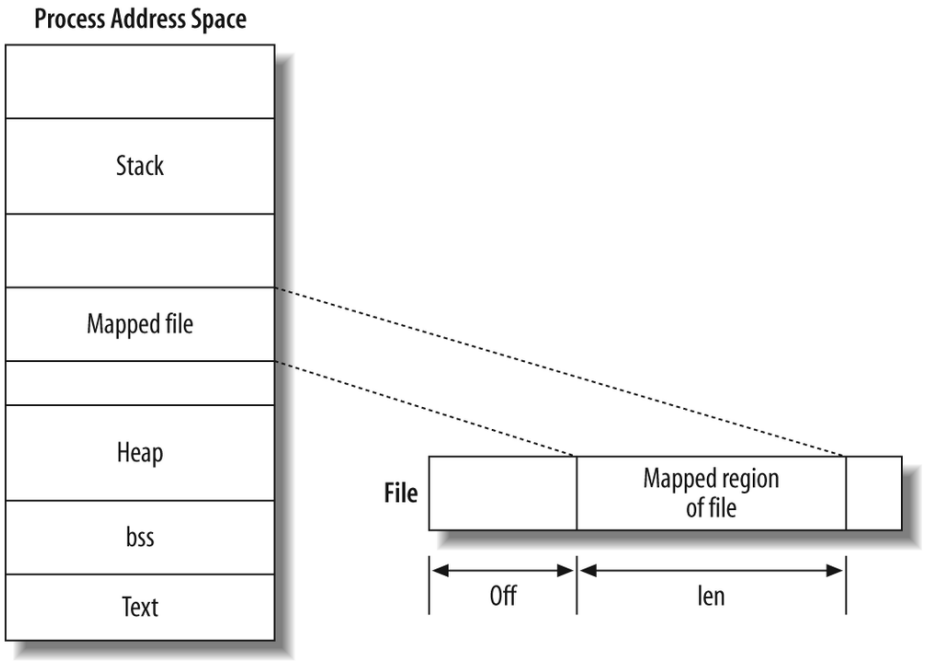
Estimation of how much memory is available for starting new applications, without swapping. Unlike the data provided by the cache or free fields, this field takes into account page cache and also that not all reclaimable memory slabs will be reclaimed due to items being in use (MemAvailable in /proc/meminfo, available on kernels 3.14, emulated on kernels 2.6.27+, otherwise the same as free)

新版

```
baohua@ubuntu:~$ free
              total        used        free      shared  buff/cache   available
Mem:        1016060      722200       78704       28200      215156       96308
Swap:       1046524      109904      936620
```

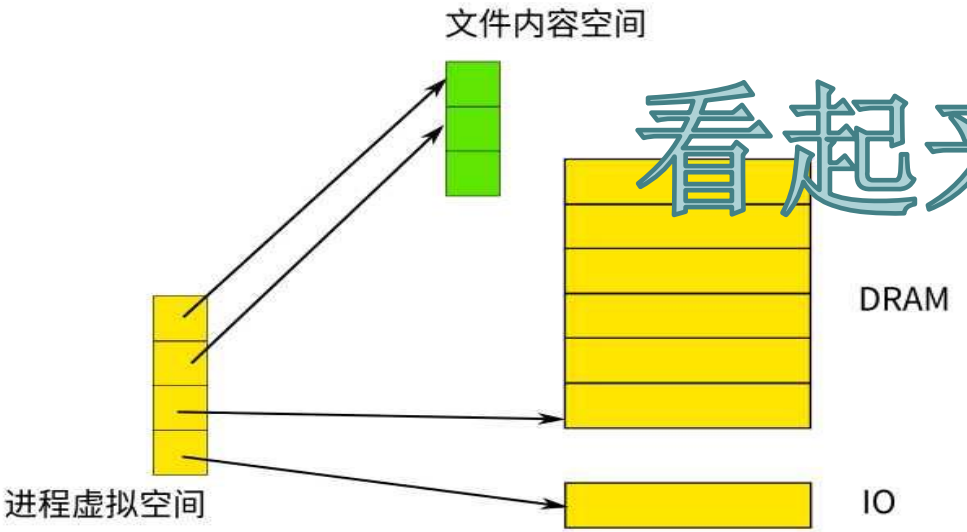
mmap

- 将文件的一部分内容map到进程虚拟地址空间



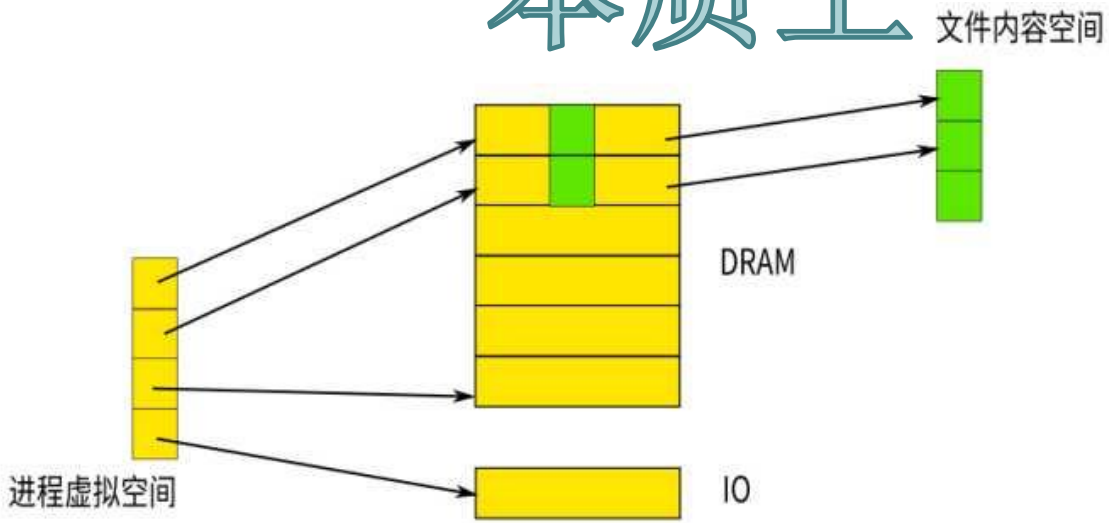

```
int main (int argc, char *argv[])
{
        ...
        fd = open (argv[1], O_RDONLY);
        ...

        p = mmap (0, sb.st_size, PROT_READ,
MAP_SHARED, fd, 0);
        ...

        for (len = 0; len < sb.st_size; len++)
                putchar (p[len]);

        if (munmap (p, sb.st_size) == -1) {
                perror ("munmap");
                return 1;
        }
        ...
}
```

mmap的内存视图



看起来

本质上

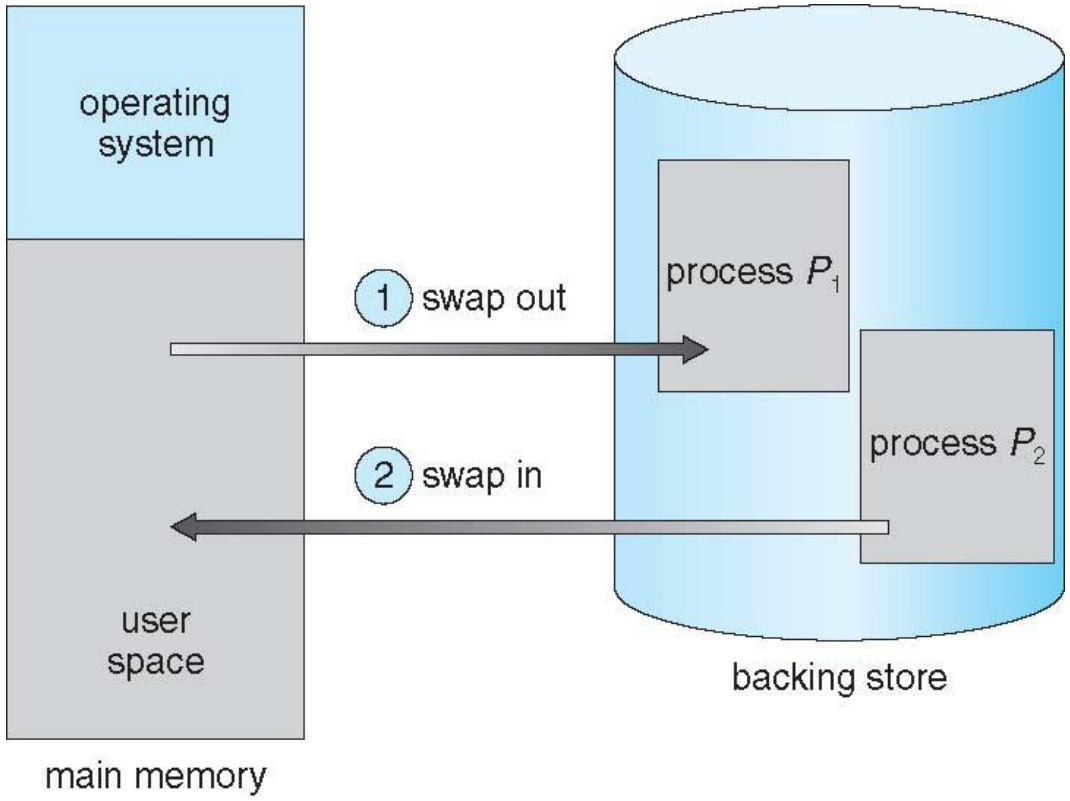


File-backed和anonymous page

- File-backed映射把进程的虚拟地址空间映射到files
  - ✓比如代码段
  - ✓比如mmap一个字体文件
- Anonymous 映射是进程的虚拟地址空间没有映射到任何file
  - ✓Stack
  - ✓Heap
  - ✓CoW pages

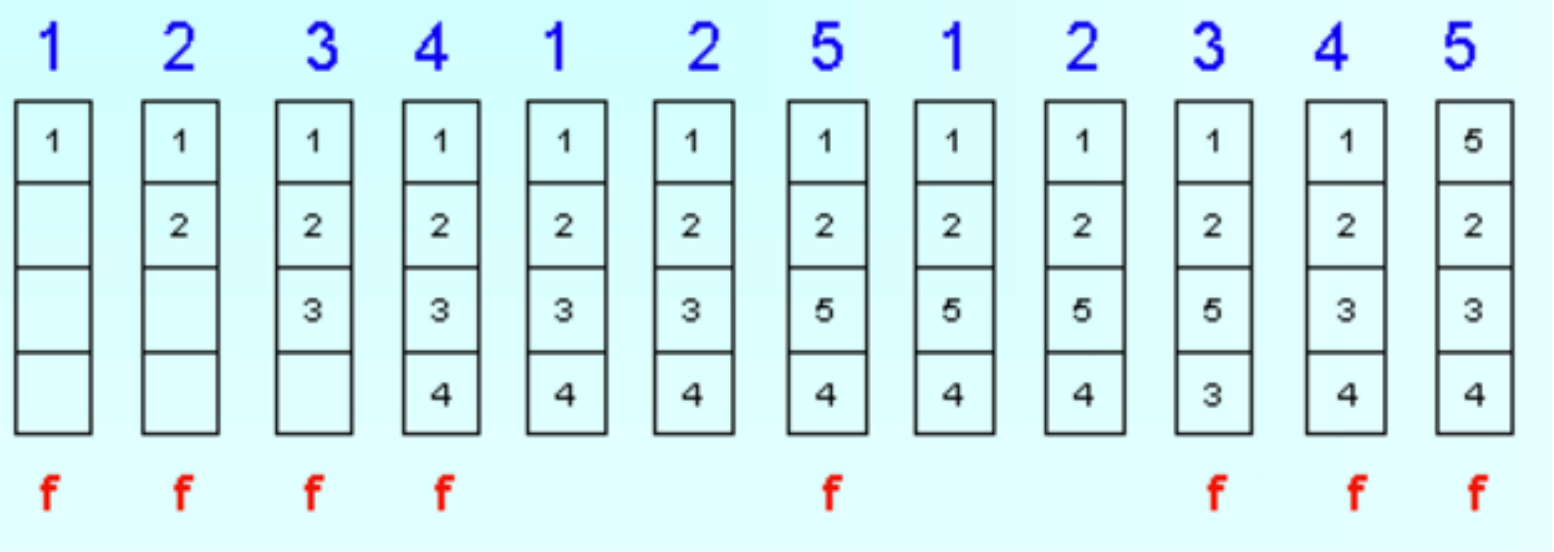
匿名页和Swap

- anonymous pages (没有任何文件背景) 分配一个 swapfile文件或者一个swap分区，来进行交换到磁盘的动作



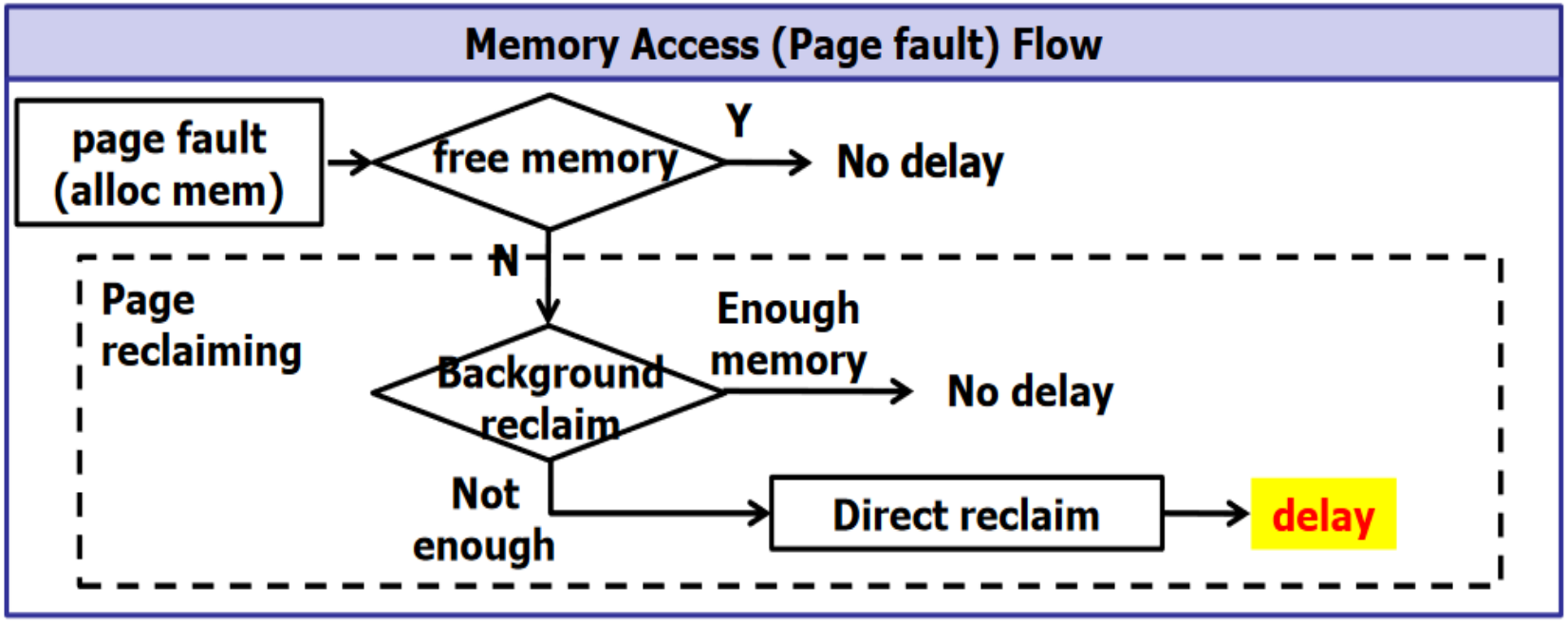
Linux Page Replacement

- 用LRU算法来进行swap和page cache的页面替换



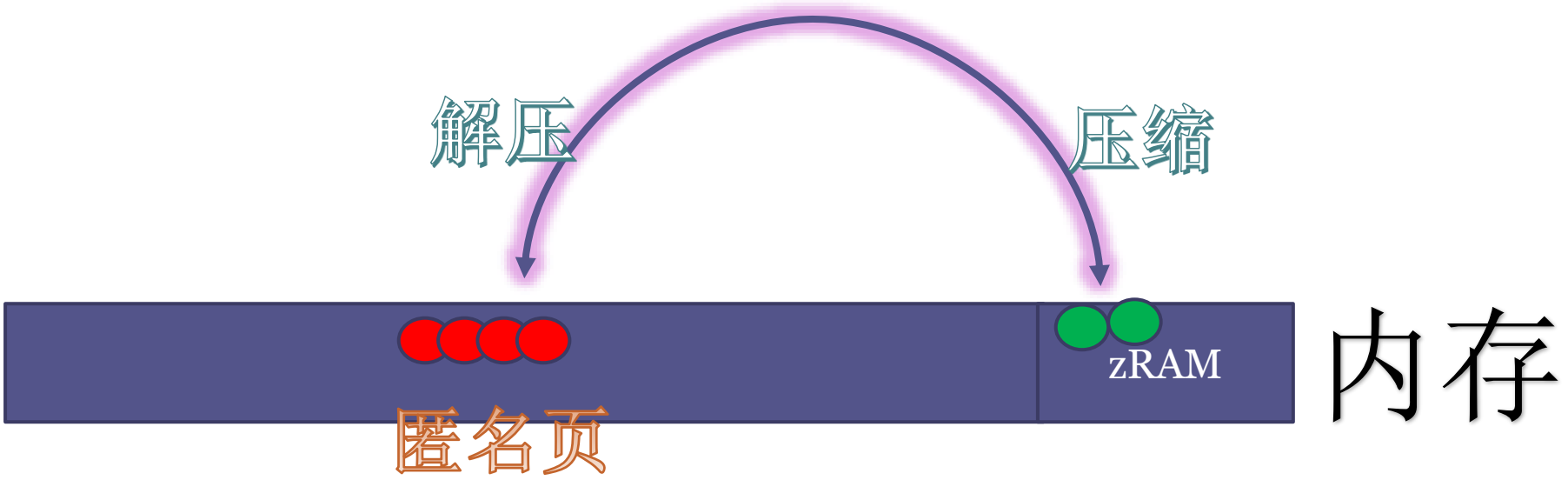
内存回收(reclaim)

- Background后台回收 kswapd
- Direct reclaim，在内存紧缺的时候，直接堵住进程空间回收



zRAM Swap

- zRAM 因为需要开辟一小块内存作为 compressed block 使用，这样的swap访问速度可以提高很多
- 压缩需要占用 CPU 时间



课程练习源码

https://github.com/21cnbao/memory-courses

更早课程

- 《Linux总线、设备、驱动模型》录播：
http://edu.csdn.net/course/detail/5329

- 深入探究Linux的设备树
http://edu.csdn.net/course/detail/5627

- Linux进程、线程和调度
http://edu.csdn.net/course/detail/5995

- C语言大型软件设计的面向对象
https://edu.csdn.net/course/detail/6496

谢谢！