页面调度

- 当主存空间已满而又需要装入新页时，页式虚拟存储管理必须按照一定的算法把已在主存的一些页调出去
- 选择淘汰页的工作称为**页面调度**
- 选择淘汰页的算法称为**页面调度算法**
- 页面调度算法设计不当，会出现（刚被淘汰的页面立即又要调入，并如此反复）
- 这种现象称为**抖动**或**颠簸**

缺页中断率

- 假定进程P共n页，系统分配页架数m个
- P运行中成功访问次数为S，不成功访问次数为F，总访问次数A=S+F
- 缺页中断率定义为：f=F/A
- 缺页中断率是衡量存储管理性能和用户编程水平的重要依据

影响缺页中断率的因素

- 分配给进程的页架数：可用页架数越多，则缺页中断率就越低
- 页面的大小：页面尺寸越大，则缺页中断率就越低
- 用户的程序编制方法：在大数据量情况下，对缺页中断率也有很大影响

用户编程的例子

- 程序将数组置为“0”，假定仅分得一个主存页架，页面尺寸为128个字，数组元素按行存放，开始时第一页在主存

```
int A[128][128];
for(int j=0;j<128;j++)
  for(int i=0;i<128;i++)
    A[i][j]=0;
```

每执行一次赋值就要产生一次缺页中断，共产生(128×128－1)次缺页中断

```
int A[128][128];
for(int i=0;i<128;i++)
  for(int j=0;j<128;j++)
    A[i][j]=0;
```

共产生(128－1)次缺页中断

OPT页面调度算法

- 理想的调度算法是：当要调入新页面时，首先淘汰以后不再访问的页，然后选择距现在最长时间后再访问的页
- 该算法由Belady提出，称Belady算法，又称最佳算法（OPT）
- OPT只可模拟，不可实现

先进先出FIFO页面调度算法

- 总是淘汰最先调入主存的那一页，或者说主存驻留时间最长的那一页(常驻的除外)
- 模拟的是程序执行的顺序性，有一定合理性

最近最少用LRU页面调度算法

- 淘汰最近一段时间较久未被访问的那一页，即那些刚被使用过的页面，可能马上还要被使用到
- 模拟了程序执行的局部属性，既考虑了循环性又兼顾了顺序性
- 严格实现的代价大(需要维持特殊队列)

LRU算法的模拟实现

- 每页建一个引用标志，供硬件使用
- 设置一个时间间隔中断：中断时页引用标志置0
- 地址转换时，页引用标志置1
- 淘汰页面时，从页引用标志为0的页中间随机选择
- 时间间隔多长是个难点

最不常用LFU页面调度算法

- 淘汰最近一段时间内访问次数较少的页面，对OPT的模拟性比LRU更好
- 基于时间间隔中断，并给每一页设置一个计数器
- 时间间隔中断发生后，所有计数器清0
- 每访问页1次就给计数器加1
- 选择计数值最小的页面淘汰

时钟CLOCK页面调度算法

- 采用循环队列机制构造页面队列，形成了一个类似于钟表面的环形表
- 队列指针则相当于钟表面上的表针，指向可能要淘汰的页面
- 使用页引用标志位

CLOCK算法的工作流程

- 页面调入主存时，其引用标志位置1
- 访问主存页面时，其引用标志位置1
- 淘汰页面时，从指针当前指向的页面开始扫描循环队列
  - 把所遇到的引用标志位是1的页面的引用标志位清0，并跳过
  - 把所遇到的引用标志位是0的页面淘汰，指针推进一步