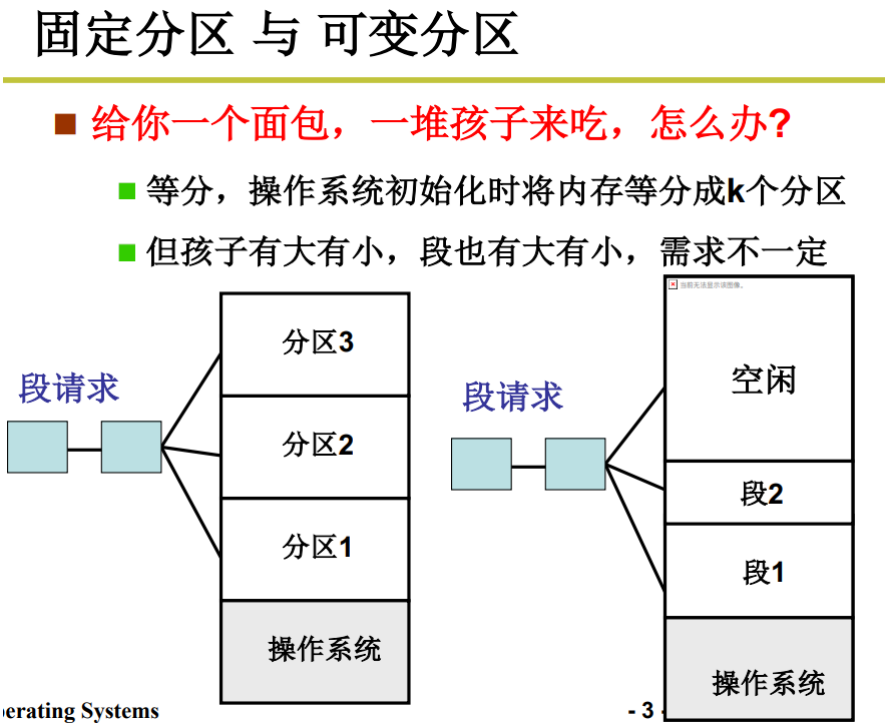
哈工大操作系统-L21内存分区与分页

- 我们要在内存中找一段空闲的区域，分配给程序。那么，我们如何找到这个空闲的分区? 因此，本次的核心就是，如何找到/分配空闲的分区
- **实际系统的物理内存一般是通过分页，非常少使用分区。 分区一般对虚拟内存进行管理。**
- **用户需要分段**，因为程序是按段组织的。**物理内存需要分页**，因为分页浪费少。因此，系统应该分页和分段都支持

1.固定分区与可变分区的抉择

固定分区容易造成浪费，因此一般使用可变分区

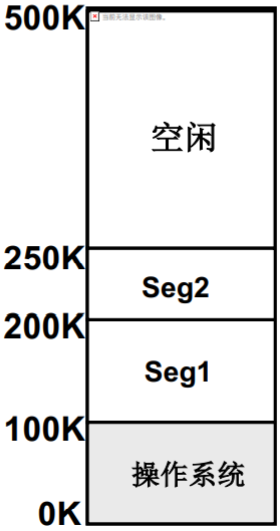


2.可变分区的管理

- 要维护内存数据结构
- 要设计分配的算法

2.1数据结构：空闲分区表+已分配分区表

可变分区的管理过程 — 核心数据结构



空闲分区表

始址	长度
250K	250K

已分配分区表

始址	长度	标志
0K	100K	OS
100K	100K	Seg1
200K	50K	Seg2

2.2管理过程--请求分配

可变分区的管理—请求分配

■ 段内存请求: reqSize = 100K

空闲分区表

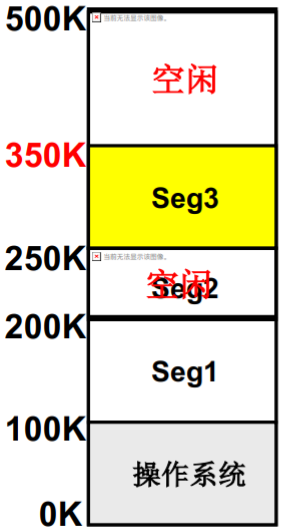
始址	长度
250K	250K

已分配分区表

始址	长度	标志
0K	100K	OS
100K	100K	Seg1
200K	50K	Seg2
250K	350K	Seg3

2.3管理过程--释放内存

可变分区的管理—释放内存



■ 段2不再需要，释放内存

空闲分区表

始址	长度
350K	150K
200K	50K

已分配分区表

始址	长度	标志
0K	100K	OS
100K	100K	Seg1
200K	50K	Seg2
250K	100K	Seg3

2.4管理过程--再次申请(多种解法，涉及到算法)

可变分区的管理—再次申请

■ 又一个段提出内存请求: reqSize=40K，怎么办?

- 有2个空闲分区，选哪一个?
- 首先适配: (350,150)
- 最佳适配: (200,50)
- 最差适配: (350,150)

空闲分区表

始址	长度
350K	150K
200K	50K

- **最佳适配**--长度与申请长度适配好；但由于适配好，剩余内存不多，内存会越割越小o(n)
- **最差适配**--内存会割的比较均匀o(n)
- **首先适配**--查表快o(1)

问题：如果某操作系统中的段内存请求很不规则，有时候需要很大的一个内存块，有时候又很小，此时用哪种分区分配算法最好? (B)

A. 最先适配

B. 最佳适配

C. 最差适配

D. 没有区别

切割后会不均匀，但也会留下大的内存块

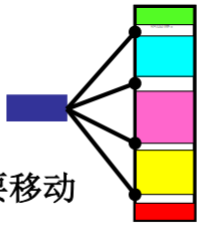
这个，操作系统到最后没有非常大的内存块

3.分页

3.1可变分区容易造成碎片

可变分区造成的问题

- 发起请求reqSize=160K怎么办?
  - 总空闲空间>160，但没有一个空闲分区>160，怎么办?
  - 这就是内存碎片
  - 将空闲分区合并，需要移动1个段(复制内容)：内存紧缩
  - 内存紧缩需要花费大量时间，如果复制速度1M/1秒，则1G内存的紧缩时间为1000秒≈17分钟



空闲分区表

始址	长度
350K	150K
200K	50K

ating Systems - 10 -

内存紧缩的时候，单核CPU是执行不了上层用户的进程的。

3.2将内存分成页

从连续到离散...

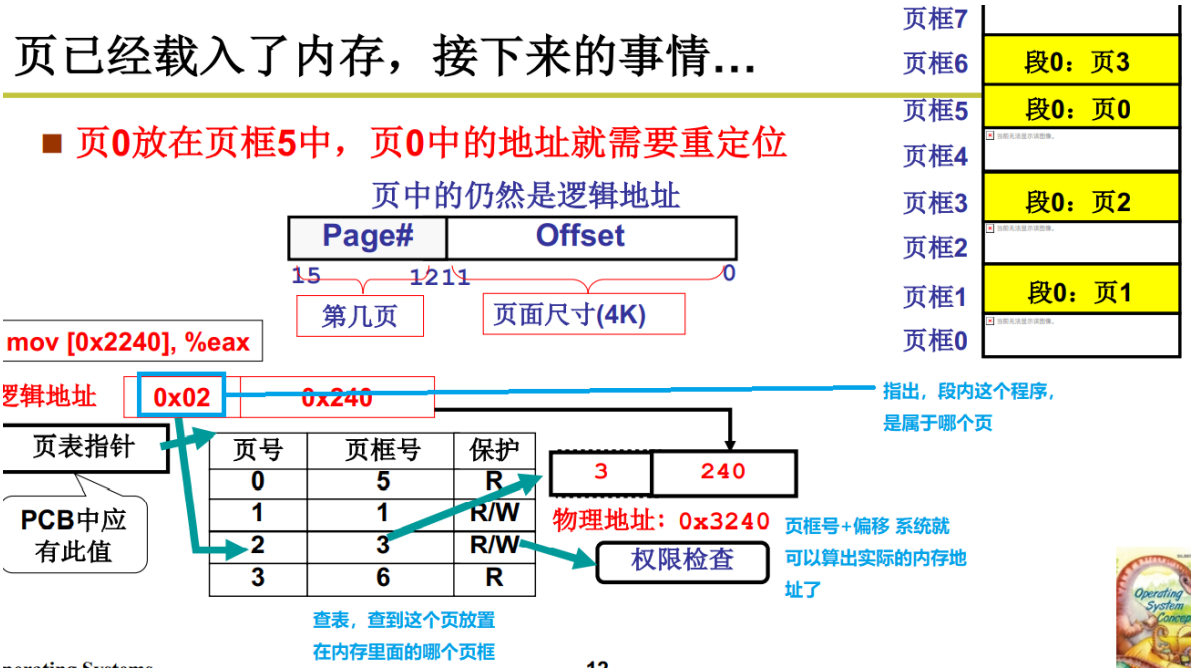
让给面包没有谁都不想要的碎末

- 将面包切成片，将内存分成页
- 针对每个段内存请求，系统一页一页的分配给给这个段

问题：此时需要内存紧缩吗? 最大的内存浪费是多少?

- 把内存分成页，**页是内存的分配单位**
- 搬移时，把**程序段拆分成页**，一部分一部分的往页里面塞
- 每个程序段最多浪费的也就一。假设页的大小是4K，那么塞不满的页也就最多浪费4k。因此，浪费少，不需要内存紧缩。

3.3程序塞入内存的例子



- **页框**：内存中容纳一页的地方。指的就是内存空间，而这个内存空间可以容纳一页
- **页号**：一个索引号，段被分成多页之后，每个页叫索引。搬移时，每页都放到内存中空闲的页框
- 页框页号映射：逻辑地址由两部分组成了，页号+偏移。使用页号可以找到内存中存放该页对应的页框号，再使用偏移我们就可以找到物理地址了。
- 这个例子中，页尺寸是4k，因此偏移的地址可以占3个16进制位。