Ch-07 虚拟存储管理

虚拟页式存储管理

存储体系：寄存器、缓存、内存、磁盘 ⇒ 虚拟内存

由**操作系统**协调各存储器的使用

虚拟内存：

1. 把物理内存和磁盘结合起来使用，得到一个容量很大的“内存”，即虚存
2. 程序引用内存所使用的地址与内存物理地址是不同的，可被自动转换成物理地址
3. 虚存大小受计算机系统寻址机制和可用磁盘容量的限制

虚拟地址：虚拟内存中某一位置的地址，该位置可以被访问，仿佛它是内存的一部分

虚拟地址空间：分配给进程的**虚拟内存**

虚拟存储技术：当进程运行时，先将其一部分装入内存，另一部分暂时保存在磁盘；当要执行的指令或访问的数据不在内存时，由操作系统**自动完成**将它们从磁盘调入内存的工作

虚拟页式存储管理

基本思想：

1. 装载程序时，不是装入全部页面，而是装入几个甚至零个页面
2. 如果进程执行时需要的页面不在内存（**缺页**），则动态装入所需页面
3. 需要时，将内存中暂时不用的一些页面**交换**到磁盘，以便获得更多的内存空间

通常有两种方式：**请求调页**、**预先调页**

把内存与磁盘有机地结合起来使用，从而得到一个容量很大的“内存”，即**虚存**。

虚存是对**内存**的抽象，构建在存储体系之上，由**操作系统**协调各存储器的使用。

虚存提供了一个比物理内存空间大得多的地址空间

地址保护：防止地址越界

1. 确保每个进程有独立的地址空间
2. 确保进程访问合法的地址范围
3. 确保进程的操作是合法的

操作系统中的资源转换技术——以 CPU 时间和磁盘空间换取昂贵的物理内存空间

页表及页表项的设计

1. 页表表项设计

页表由**页表项**组成。页表项是由**硬件**设计的

页框号、有效位、访问位、修改位、保护位

1. 页框号（内存块号、物理页面号、页帧号）
2. 有效位（驻留位、中断位）：表示该页是在内存还是在磁盘
3. 访问位：引用位
4. 修改位：查看此页是否在内存中被修改过
5. 保护位：读/写/执行

2. 多级页表

32 位虚拟地址空间：页面大小为 4K，页表项大小为 4 字节，用户地址空间 2G

一个进程的页表的各页在内存中若不连续存放，则需要引入地址索引 $\rightarrow$ **页目录**

页表本身也放在虚存中（进程运行时，部分页表映射到内存）

二级页表可以表示 4G 的地址空间

3. 引入反转（反置、反向）页表

地址转换

- 从虚拟地址空间出发：虚拟地址 → 查页表 → 得到页框号 → 形成物理地址
- 每个进程一张页表

反转页表思路

- 从物理地址空间出发，系统建立一张页表
- 页表项记录进程 i 的某虚拟地址（虚页号）与页框号的映射关系

4. 地址转换

MMU 将虚拟地址转换为物理地址

5. 快表（TLB）的引入

相连存储器

- 特点：按内容并行查找

问题：

1. 页表 → 两次或两次以上的内存访问
2. CPU 的指令处理速度与内存指令的访问速度差异大，CPU 的速度得不到充分利用

快表的作用：**加快地址映射速度，改善系统性能**

程序访问的局部性原理 → 引入快表（TLB）

工作原理：采用联想映射技术按内容同时查找

快表是 CPU 中引入的高速缓存，是一种相连存储器，特点是按内容并行查找。

6. 缺页异常处理

是页错误的一种

在地址映射过程中，硬件检查页表时发现所要访问的页不在内存，则产生**缺页异常**

7. 驻留集管理

驻留集：操作系统给每个进程分配多少个页框

- **固定分配**策略

  进程创建时确定、根据进程类型需要确定
- **可变分配**策略

  根据缺页率评估进程局部性表现

  缺页率高 → 增加页框数；缺页率低 → 减少页框数

页面置换算法

1. 置换问题

置换范围：计划置换页面的集合是局限在产生缺页中断的进程，还是所有进程的页框。

- 局部置换策略：仅在产生本次缺页的进程的驻留集中选择
- 全局置换策略：将内存中所有未锁定的页框都作为置换的候选

置换策略：在计划置换的页框集合中，选择换出哪一个页框

所有策略的目标：**置换最近最不可能访问的页**

根据**局部性原理**，最近访问历史和最近将要访问的模式间存在相关性，大多数置换策略基于**过去的行为**预测**将来的行为**。

置换策略设计得越精致、越复杂，实现的软硬件开销就越大

页框锁定：

- 给每一页框增加一个锁定位
- 通过设置相应的锁定位，避免因交换产生的不确定的延迟
- 如操作系统核心代码、关键数据结构、I/O 缓冲区

2. 页面置换算法

最优算法 → 最近未使用 → 先进先出 → 第二次机会 → 时钟算法 → 最近最少使用 → 最不经常使用 → 老化算法 → 工作集 → 工作集时钟

最优置换算法

置换以后不再需要的或最远的将来才会用到的页面

先进先出页面置换算法

选择在内存中驻留时间最长的页并置换它

实现：页面链表法

第二次机会置换算法

按照先进先出算法选择某一页面，检查其访问位 R，如果为 0，置换该页；如果为 1，则将访问位置为 0，并给第二次机会

时钟算法

将页框组成循环缓冲区场景

最近未使用算法

选择在最近一段时间内未使用过的一页并置换

实现：设置页表项的访问位（R）、修改位（W）

启动一个进程时，R、M 位置为 0，R 位定期清零（复位）

最近最少使用算法（LRU）

选择最后一次访问时间距离当前时间最长的一页并置换，即置换**未使用时间最长的一页**

实现：时间戳或维护一个访问页的栈，开销较大

最不经常使用算法

选择访问次数最少的页面置换：LRU 的一种软件解决方案

老化算法

计数器在加 R 前先右移一位，R 位加到计数器的最左端

4. $Belady$ 现象

$FIFO$ 页面置换算法会产生 $Belady$ 现象，即当分配给进程的物理页面数增加时，缺页次数反而增加

5. 影响缺页次数的因素

- 页面置换算法
- 页面大小
- 程序的编制方法
- 分配给进程的物理页面数

颠簸（抖动）

虚存中，页面在内存与磁盘之间频繁调度，使得调度页面所需的时间比进程实际运行的时间还多，这样导致系统效率急剧下降，这种现象称为颠簸或抖动。

6. 工作集模型

基本思想：根据程序的局部性原理，一般情况下，进程在一段时间内总是集中访问一些页面，这些页面被称为**活跃页面**，如果分配给一个进程的页框太少了，使该进程所需的活跃页面不能全部装入内存，则进程在运行过程中将频繁发生中断。

如果能为进程提供与活跃页面相等的页框数，则可减少缺页中断次数

工作集 (t, Δ) = 该进程在过去的 Δ 个虚拟时间单位中使用的虚拟页面集合。

工作集取决于三个因素：

1. 访页序列特性
2. 当前时刻 t
3. 工作集窗口长度 (Δ)

驻留集：当前时刻，进程实际驻留在内存中的页框集合

工作集与驻留集的关系：

- 工作集是进程在运行过程中固有的性质
- 驻留集取决于系统分配给进程的页框数和页面置换算法

工作集算法：

找出一个不在工作集中的页面并置换它

7. 清除策略

清除策略指的是从进程的驻留集中收回页框。

虚拟页式系统工作的**最佳状态**是，发生缺页异常时，系统中有大量的空闲页框

结论：保存一定数目的空闲页框比使用所有内存并在需要时搜索一个页框有更好的性能。

设计一个**分页守护进程**，多数时间睡眠，定期唤醒检查内存状态。

如果空闲页框过少，分页守护进程通过预设的页面置换算法选择页面换出内存。

如果页面装入内存后被修改过，则将它们写回磁盘。分页守护进程可保证所有的空闲页框都是“干净”的。

当需要一个已置换出的页框时，如果该页框还没有被新的内容覆盖，将它从空闲页框集合中移出即可恢复该页面。

页缓冲技术

目的：提高性能

页缓冲技术：

- 不丢弃置换出的页，而是放入两个表之一：如果未被修改，放到**空闲页链表**中，如果修改了，则放到**修改页链表**中。
- 被修改的页以簇方式写回磁盘，减少了 I/O 操作的数量，从而减少了磁盘访问时间
- 被置换的页仍然保留在内存中，一旦进程又要访问该页，可以迅速将它加入该进程的驻留集合

8. 加载控制

系统并发度：驻留在内存中的进程数目

通过调节并发进程数进行系统负载控制

内存映射文件

基本思想：进程通过一个系统调用将一个文件映射到其虚拟地址空间中的一个区域，访问这个文件就像访问内存中的一个大数组，而不是对文件进行读写。

- 映射共享的页面并不会实际读入页面的内容，而是在访问页面时，页面才会被每次一页的读入，磁盘文件则被当做后备存储。
- 当进程退出或显式地解除文件映射时，所有被修改页面会写回文件

写时复制：新复制的页面对执行写操作的进程是私有的，对其他共享写时复制页面的进程是不可见的。

策略与机制分离

控制系统复杂度的重要方法——把策略从机制中分离出来

存储管理系统分为三个部分：

1. 底层 MMU 处理程序
2. 作为内核一部分的缺页中断处理程序
3. 运行在用户空间中的外部页面调度程序