Operating System Principle, OS

《操作系统原理》

华中科技大学网安学院

2023年11月-2024年01月

实验三：第7章 内存管理

- 一、实验目的
  - （1）理解页面淘汰算法原理，编写程序演示页面淘汰算法。
  - （2）验证Linux虚拟地址转化为物理地址的机制
  - （3）理解和验证程序运行局部性的原理。
  - （4）理解和验证缺页处理的流程。
- 二、实验内容
  - （1）Win/Linux编写二维数组遍历程序，理解局部性的原理。
  - （2）Windows/Linux模拟实现OPT或FIFO或LRU淘汰算法。
  - （3）研读并修改Linux内核的缺页处理函数do_no_page 或页框分配函数get_free_page，并用printk打印调试信息。注意：需要编译内核。建议优麒麟或麒麟系统。
  - （4）Linux下利用/proc/pid/pagemap技术计算某个变量或函数虚拟地址对应的物理地址等信息。建议优麒麟或麒麟系统。
- 三、实验要求
  - 第2，4必做，第1，3选做。寝室提前做完，老师机房检查和答疑。

实验三：第7章 内存管理

- 四、实验指南
  - （1）在Linux/Windows下编写二维数组的遍历程序，理解程序运行局部性的原理。【区别：64位或32位或编译选项，是否有差别自己探索，答案不唯一】
    - ◆提示1：数组尽可能开大一些，并尝试改变数组大小，改变内外重循环次序。例如从[2048] X[2048]变化到[10240] x [20480]，观察它们的遍历效率。

```
1  int MyArray[10000][20000];
2  for(int i=0;i<10000;i++)
3      for(int j=0;j<20000;++)
4          MyArray[i][j] = 0;
```

```
1  int MyArray[10000][20000];
2  for(int i=0;i<20000;i++)
3      for(int j=0;j<10000;++)
4          MyArray[j][i] = 0;
```

    - ◆提示2：在任务管理器或利用/proc中文件系统观察它们的缺页次数。

实验三：第7章 内存管理

- 四、实验指南
  - （2）Windows/Linux模拟实现OPT或FIFO或LRU淘汰算法。
  - [以下模拟过程仅供参考，不是唯一方案！百度参考其他方案！]
    - ◆提示1：程序指令执行过程采用遍历数组的操作来模拟；
    - ◆提示2：用1个较大数组A（例如2400个元素）模拟进程，数组里面放的是随机数，每个元素被访问时就使用printf将其打印出来，模仿指令的执行。数组A的大小必须是设定的页大小（例如10个元素或16个元素等等）的整数倍。
    - ◆提示3：用3-8个小数组(例如数组B，数组C，数组D等)模拟分得的页框。小数组的大小等于页大小（例如10条指令的大小，即10的元素）。小数组里面复制的是大数组里面的相应页面的内容（自己另外构建页表，描述大数组的页与小数组序号的关系）。
    - ◆提示4：利用不同的次序访问数组A，次序可以是：顺序，跳转，分支，循环，或随机，自己构建访问次序。不同的次序也一定程度反映程序局部性。
    - ◆提示5：大数组的访问次序可以用 rand( )函数定义，模拟产生指令访问序列，对应到大数组A的访问次序。然后将指令序列变换成相应的页地址流，并针对不同的页面淘汰算法统计“缺页”情况。缺页即对应的“页面”没有装到小数组中（例如数组B，数组C，数组D等）。
    - ◆提示6：实验中页面大小，页框数，访问次序，淘汰算法都应可调。

实验三：第7章 内存管理

- 四、实验指南
  - （3）研读并修改Linux内核的缺页处理函数do_no_page 并用printk打印调试信息。注意：需要编译内核。建议优麒麟或麒麟系统。
    - ◆提示1：编写2个类似hello world或简单的for循环应用程序作为测试目标。
    - ◆提示2：在Linux内核的缺页处理函数do_no_page( )或类似函数中（不同版本的此函数名有差异）采用printk方式添加调试信息，打印特定进程（以可程序的程序名为过滤条件）的缺页信息，统计缺页次数。
    - ◆提示3：也可以在Linux内核的分配物理页框函数get_free_page( )或类似函数中（不同版本的此函数名有差异）采用printk方式添加调试信息，打印特定进程（以可程序的程序名为过滤条件）运行过程中分配新页框的信息，统计相关信息。

实验三：第7章 内存管理

- 四、实验指南
  - （4）Linux下利用/proc/pid/pagemap技术计算某个变量或函数虚拟地址对应的物理地址等信息。
    - ◆提示1：Linux的/proc/pid/pagemap文件允许用户查看当前进程虚拟页的物理地址等相关信息。
      - □每个虚拟页包含一个64位的值
      - □注意分析64位的信息
    - ◆提示2：获取当前进程的pagemap文件的全名

```
49    pid  = getpid();
50    sprintf(buf, "%d", pid);
51    strcat(pagemap_file, "/proc/");
52    strcat(pagemap_file, buf);
53    strcat(pagemap_file, "/pagemap");
```

    - ◆提示3：可以输出进程中某个或多个全局变量或自定义函数的虚拟地址，所在页号，所在物理页框号，物理地址等信息。
    - ◆思考：（1）如何扩充实验（写个通用函数）展示指定进程的指定虚拟地址对应的物理地址。（2）如何扩充实验验证不同进程的共享库(例如某个已知，被普遍调用的*.so库)具有同一物理地址。