Linux存储器管理方式研究

摘要：本文从存储器管理的角度分析Linux系统中存储器管理的实现方式，包括基本内存管理以及虚拟内存管理，并分析其原因。同时分析其存储器管理实现方式变化，研究现代操作系统发展的影响。

关键词： Linux；存储器管理；基本内存管理；虚拟内存管理

1 Linux简介

Linux是类Unix系统，其本质为内核即操作系统最核心的功能部分。一般常见的操作系统如Ubuntu、Fedora、Arch等都是基于Linux内核的。Linux相比较Windows、OS X等系统最显著的特征是其开源性。在Linux操作系统的诞生、发展成长过程中，主要依赖于以下五个重要支柱：UNIX操作系统、MINIX操作系统、GNU计划、POSIX标准与Internet网络。

2 存储器管理

存储器管理的对象是主存，也称内存。它的主要功能包括分配和回收主存空间、提高主存利用率、扩充主存、对主存信息实现有效保护。存储管理方案的主要目的是解决多个用户使用主存的问题，其存储管理方案主要包括连续分配方式、离散分配方式（分页存储管理、分段存储管理、段页式存储管理）以及虚拟存储管理。

2.1 连续分配方式

连续分配方式是指为一个用户程序分配一个连续的内存空间，可以将连续分配方式分为单一连续分配、固定分区分配、动态分区分配、动态重定位分区分配。

2.1.1 **单一连续分配**

这是一种最简单的存储管理方式，但只能在单用户、单任务的操作系统中，将内存分为系统区和用户区，系统区供OS使用，通常放在内存的低地址，用户区是指除系统区以外的全部内存空间，提供给用户使用。

2.1.2 **固定分区分配**

固定分区分配是一种最简单的可运行多道程序的存储管理方式，将内存用户空间划分为若干个固定大小的区域，在每个分区只装入一道作业，这样，便允许多道作业并发执行，当有空闲分区时，便可以再从外存的后备作业队列中选择一个适当大小的作业装入该分区，当该作业结束时，又可再从后备作业队列中找出另一作业调入该分区。

2.1.3 **动态分区分配**

动态分区分配是根据进程的实际需要，动态地为之分配内存空间，在实现可变分区分配时，将涉及到分区分配中所用的数据结构、分区分配算法、分区的分配和回收等。

2.1.4 **可重定位分区分配**

在连续分配方式中，必须把一个系统或用户程序装入一连续的内存空间，若果在系统中只有若干个小的分区，即使他们容量总和大于要装入的程序，但由于这些分区不相邻接，也无法把该程序装入内存。若想装入，则将内存中的所有作业进行移动，使他们全部相邻接，这样，即可把原来分散的多个小分区拼接成一个大分区，这时，就可以把作业装入该区。经过紧凑后的某些用户程序在内存中的位置发生了变化，此时若不对程序和数据的地址加以修改（变换），则程序必将无法执行，为此，在每次紧凑之后，都必须对移动了的数据和程序进行重定向。

2.2离散分配方式

连续分配方式会形成许多“碎片”，虽然可通过“紧凑”方法将碎片拼接成可用的大块空间，但需要为之付出很大的开销。如果可以将一个进程分散地装入许多不相邻的分区中，便可充分地利用内存空间，由此思想产生了离散分配方式。根据离散分配时分配地址空间的基本单位的不同，将离散分配分为分页存储管理、分段存储管理、段页式存储管理。

2.2.1**分页存储管理**

分页存储管理是将一个进程的逻辑地址空间分成若干个大小相等的片，称为页面或页，并为各页加以编号，从0开始，如第0页、第1页等。相应地，也把内存空间分成与页面相同大小的若干个存储块，称为(物理)块或页框(frame)，也同样为它们加以编号，如0#块、1#块等等。在为进程分配内存时，以块为单位将进程中的若干个页分别装入到多个可以不相邻接的物理块中。由于进程的最后一页经常装不满一块而形成了不可利用的碎片，称之为“页内碎片”。

2.2.2**分段存储管理**

在分段存储管理方式中，作业的地址空间被划分为若干个段，每个段定义了一组逻辑信息。例如，有主程序段MAIN、子程序段X、数据段D及栈段S等。每个段都有自己的名字。为了实现简单起见，通常可用一个段号来代替段名，每个段都从0开始编址，并采用一段连续的地址空间。段的长度由相应的逻辑信息组的长度决定，因而各段长度不等。整个作业的地址空间由于是分成多个段，因而是二维的，亦即，其逻辑地址由段号(段名)和段内地址所组成。

2.2.3**段页式存储管理**

段页式系统的基本原理，是基本分段存储管理方式和[基本分页存储管理方式](https://baike.baidu.com/item/%E5%9F%BA%E6%9C%AC%E5%88%86%E9%A1%B5%E5%AD%98%E5%82%A8%E7%AE%A1%E7%90%86%E6%96%B9%E5%BC%8F)原理的结合，即先将用户程序分成若干个段，再把每个段分成若干个页，并为每一个段赋予一个段名。

2.3 虚拟存储管理

当程序的存储空间要求大于实际的内存空间时，就使得程序难以运行了．虚拟存储技术就是利用实际内存空间和相对大的多的外部储存器存储空间相结合构成一个远远大于实际内存空间的虚拟存储空间，程序就运行在这个虚拟存储空间中．能够实现虚拟存储的依据是程序的局部性原理，即程序在运行过程中经常体现出运行在某个局部范围之内的特点．在时间上，经常运行相同的指令段和数据（称为时间局部性），在空间上，经常运行与某一局部存储空间的指令和数据(称为空间局部性），有些程序段不能同时运行或根本得不到运行。虚拟存储是把一个程序所需要的存储空间分成若干页或段，程序运行用到页和段就放在内存里，暂时不用就放在外存中．当用到外存中的页和段时，就把它们调到内存，反之就把它们送到外存中．装入内存中的页或段可以分散存放．

3 Linux中存储器管理调度与算法研究

3.1 Linux存储管理方式

Linux一般采用页式虚拟存储管理的方式。在页式虚拟存储系统中，将虚拟存储器划分为同样大小的页，称为虚页或逻辑页，主存空间的页成为物理页。

Linux并不将程序中的全部页面装入内存中运行，它只装入少数必要的页面，便启动运行，以后，通过页面调度和页面置换功能，陆续将即将要运行的页面调入内存，同时把暂时不要运行的页面换到外存(磁盘)上。置换时以页面为单位。这也就是通常所说的虚拟存储方式。而不支持虚拟存储器的内存管理，则需要把每个作业都装入内存后才能运行。

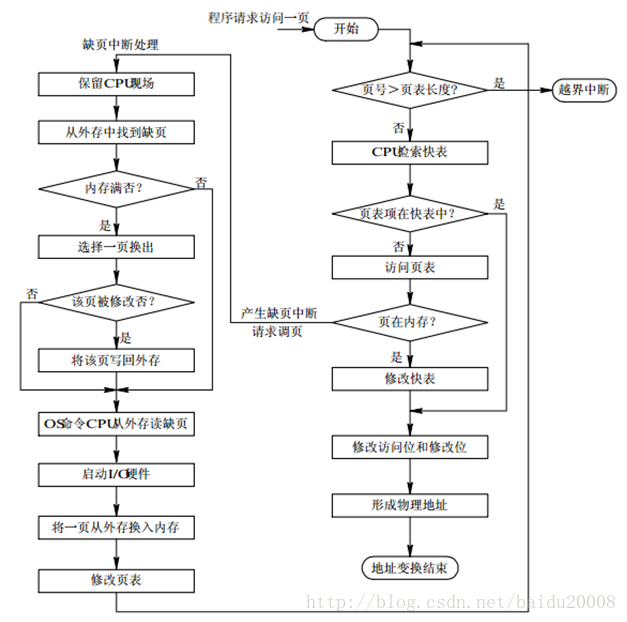


图1 请求分页的地址变换流程

3.2 页表管理

Linux采用三级分页的模型，从2.6.11版本开始采用4级分页模型。4种页表如下：页全局目录（pgd）页上级目录（pud）页中间目录（pmd）页表（pte）对于没有启用物理地址扩展的32位系统使用两级页表，启动了物理地址扩展的32位系统使用三级页表。64位系统使用三级还是四级分页取决与硬件对线性地址的位划分。

3.2.1 页表作用

在使用虚拟地址空间的linux操作系统上，每一个进程都工作在一个4G的地址空间上，其中0~3G是应用进程可以访问的user地址空间，是这个进程独有的，其他进程看不到也无法操作这个地址空间；3G~4G是kernel地址空间，所有进程共享这部分地址空间。

由于每个进程都有3G的私有进程空间，所以系统的物理内存无法对这些地址空间进行一一映射，因此kernel需要一种机制，把进程地址空间映射到物理内存上。当一个进程请求访问内存时，操作系统通过存储在kernel中的进程页表把这个虚拟地址映射到物理地址，如果还没有为这个地址建立页表项，那么操作系统就为这个访问的地址建立页表项。最基本的映射单位是page，对应的是页表项PTE。

页表项和物理地址是多对一的关系，即多个页表项可以对应一个物理页面，因而支持共享内存的实现（几个进程同时共享物理内存）。

3.2.2 页表的实现

实现虚拟地址到物理地址转换最容易想到的方法是使用数组，对虚拟地址空间的每一个页，都分配一个数组项。但是有一个问题，考虑IA32体系结构下，页面大小为4KB，整个虚拟地址空间为4GB，则需要包含1M个页表项，这还只是一个进程，因为每个进程都有自己独立的页表。因此，系统所有的内存都来存放页表项恐怕都不够。

想象一下进程的虚拟地址空间，实际上大部分是空闲的，真正映射的区域几乎是汪洋大海中的小岛，因次我们可以考虑使用多级页表，可以减少页表内存使用量。实际上多级页表也是各种体系结构支持的，没有硬件支持，我们是没有办法实现页表转换的。

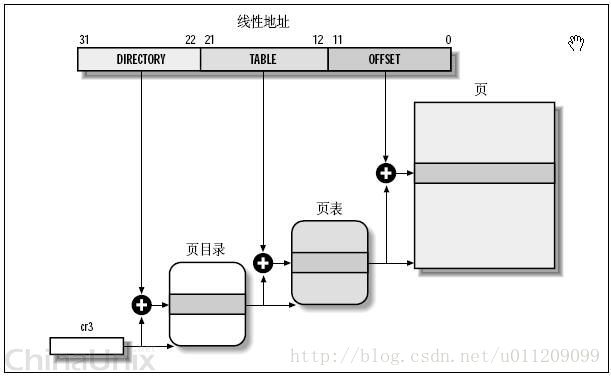


图2 Linux操作系统使用4级页表

3.2.2 地址翻译

首先，对于给定的线性地址，根据线性地址的bit22~bit31作为页目录项索引值，在CR3所指向的页目录中找到一个页目录项。接着找到的页目录项对应着页表，根据线性地址的bit12~bit21作为页表项索引值，在页表中找到一个页表项。然后，找到的页表项中包含着一个页面的地址，线性地址的bit0~bit11作为页内偏移值和找到的页确定线性地址对应的物理地址。这个地址翻译过程完全是由硬件完成的。

6 总结与展望

目前，存储器空间不断扩大，但是对于存储器管理的研究仍在不断进行，Linux作为开源社区的代表作品一直集结着全球各地开源开发者的智慧与力量，对于我们研究现代操作系统的存储器管理有着十分重要的作用。经过这几天的整理学习，发现将来的操作系统将会需要处理更加大型的不同形态作业，引起对内存的激烈竞争。目前来看最好的解决方式就是虚拟存储技术，对虚拟存储器、现代存储器管理的研究将大大帮助现代操作系统的发展。

参考文献

[1]汤小丹，梁红兵，哲凤屏，汤子瀛.计算机操作系统[M].西安：西安电子科技大学出版社2014：5-4

[2] baidu20008. linux的存储管理方式 [blog]. https://blog.csdn.net/baidu20008/article/details/12771235

[3] kernel\_dsn. 深入理解linux内存管理之页表管理 [blog]. https://blog.csdn.net/u011209099/article/details/9248525

[4] leesf. 【操作系统】存储器管理（四） [blog]. https://www.cnblogs.com/leesf456/p/5616041.html

[5] 郦军.存储器管理中的虚拟存储技术[J].微机发展,1997,(第4期).