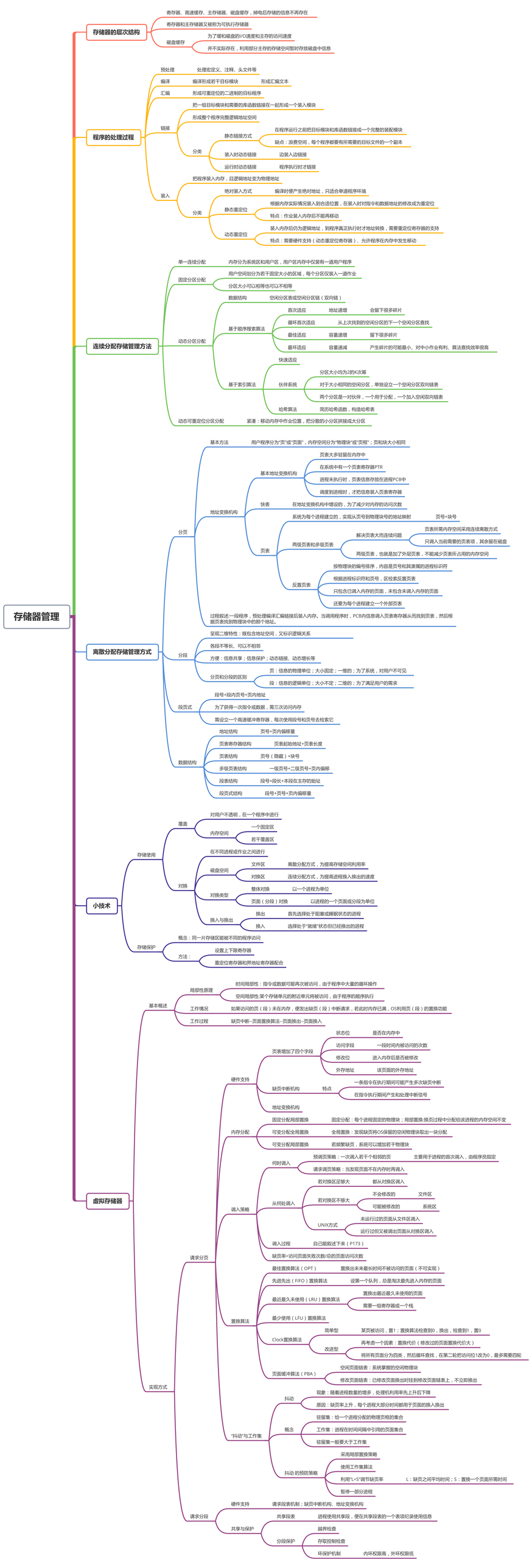
存储器管理
存储器的层次结构
寄存器、高速缓存、主存储器、磁盘缓存，掉电后存储的信息不再存在
寄存器和主存储器又被称为可执行存储器
磁盘缓存
为了缓和磁盘的I/O速度和主存的访问速度
并不实际存在，利用部分主存的存储空间暂时存放磁盘中信息
程序的处理过程
预处理
处理宏定义、注释、头文件等
编译
编译形成若干目标模块
形成汇编文本
汇编
形成可重定位的二进制的目标程序
链接
把一组目标模块和需要的库函数链接在一起形成一个装入模块
形成整个程序完整逻辑地址空间
分类
静态链接方式
在程序运行之前把目标模块和库函数链接成一个完整的装配模块
缺点：浪费空间，每个程序都要有所需要的目标文件的一个副本
装入时动态链接
边装入边链接
运行时动态链接
程序执行时才链接
装入
把程序装入内存，且逻辑地址变为物理地址
分类
绝对装入方式
编译时便产生绝对地址，只适合单道程序环境
静态重定位
根据内存实际情况装入到合适位置，在装入时对指令和数据地址的修改成为重定位
特点：作业装入内存后不能再移动
动态重定位
装入内存后仍为逻辑地址，到程序真正执行时才地址转换，需要重定位寄存器的支持
特点：需要硬件支持（动态重定位寄存器）、允许程序在内存中发生移动
连续分配存储管理方法
单一连续分配
内存分为系统区和用户区，用户区内存中仅装有一道用户程序
固定分区分配
用户空间划分为若干固定大小的区域，每个分区仅装入一道作业
分区大小可以相等也可以不相等
动态分区分配
数据结构
空闲分区表或空闲分区链（双向链）
基于顺序搜索算法
首次适应
地址递增
会留下很多碎片
循环首次适应
从上次找到的空闲分区的下一个空闲分区查找
最佳适应
容量递增
留下很多碎片
最坏适应
容量递减
产生碎片的可能最小，对中小作业有利。算法查找效率很高
基于索引算法
快速适应
伙伴系统
分区大小均为2的K次幂
对于大小相同的空闲分区，单独设立一个空闲分区双向链表
两个分区是一对伙伴，一个用于分配，一个加入空闲双向链表
哈希算法
简历哈希函数，构造哈希表
动态可重定位分区分配
紧凑：移动内存中作业位置，把分散的小分区拼接成大分区
离散分配存储管理方式
分页
基本方法
用户程序分为"页"或"页面"，内存空间分为"物理块"或"页框"；页和块大小相同
地址变换机构
基本地址变换机构
页表大多驻留在内存中
在系统中有一个页表寄存器PTR
进程未执行时，页表信息存放在进程PCB中
调度到进程时，才把信息装入页表寄存器
快表
在地址变换机构中增设的，为了减少对内存的访问次数
页表
系统为每个进程建立的，实现从页号到物理块号的地址映射
页号+块号
两级页表和多级页表
解决页表大而连续问题
页表所需内存空间采用连续离散方式
只调入当前需要的页表项，其余留在磁盘
再级页表，也就是加了外层页表，不能减少页表所占用的内存空间
反置页表
按物理块的编号排序，内容是页号和其隶属的进程标识符
根据进程标识符和页号，区检索反置页表
只包含已调入内存的页面，未包含未调入内存的页面
还要为每个进程建立一个外部页表
过程叙述：一段程序，预处理编译汇编链接后装入内存。当调用程序时，PCB内信息调入页表寄存器从而找到页表，然后根据页表找到物理块中的那个地址。
分段
呈现二维特性：既包含地址空间，又标识逻辑关系
各段不等长、可以不相邻
方便：信息共享；信息保护；动态链接、动态增长等
分页和分段的区别
页：信息的物理单位；大小固定；一维的；为了系统，对用户不可见
段：信息的逻辑单位；大小不定；二维的；为了满足用户的需求
段页式
段号+段内页号+页内地址
为了获得一次指令或数据，需三次访问内存
需设立一个高速缓冲寄存器，每次使用段号和页号去检索它
数据结构
地址结构
页号+页内偏移量
页表寄存器结构
页表起始地址+页表长度
页表结构
页号（隐藏）+块号
多级页表结构
一级页号+二级页号+页内偏移
段表结构
段号+段长+本段在主存的始址
段页式结构
段号+页号+页内偏移量
小技术
存储使用
覆盖
对用户不透明，在一个程序中进行
内存空间
一个固定区
若干覆盖区
对换
在不同进程或作业之间进行
磁盘空间
文件区
离散分配方式，为提高存储空间利用率
对换区
连续分配方式，为提高进程换入换出的速度
对换类型
整体对换
以一个进程为单位
页面（分段）对换
以进程的一个页面或分段为单位
换入与换出
换出
首先选择处于阻塞或睡眠状态的进程
换入
选择处于"就绪"状态但已经换出的进程
存储保护
概念：同一片存储区能被不同的程序访问
方法：
设置上下限寄存器
重定位寄存器和界地址寄存器配合
虚拟存储器
基本概述
局部性原理
时间局部性：指令或数据可能再次被访问，由于程序中大量的循环操作
空间局部性:某个存储单元的附近单元将被访问，由于程序的顺序执行
工作情况
如果访问的页（段）未在内存，便发出缺页（段）中断请求，若此时内存已满，OS利用页（段）的置换功能
工作过程
缺页中断-页面置换算法-页面换出-页面换入
请求分页
硬件支持
页表增加了四个字段
状态位
是否在内存中
访问字段
一段时间内被访问的次数
修改位
进入内存后是否被修改
外存地址
该页面的外存地址
缺页中断机构
特点
一条指令在执行期间可能产生多次缺页中断
在指令执行期间产生和处理中断信号
地址变换机构
内存分配
固定分配局部置换
固定分配：每个进程固定的物理块；局部置换:换页过程中分配给该进程的内存空间不变
可变分配全局置换
全局置换：发现缺页将OS保留的空闲物理块取出一块分配
可变分配局部置换
若频繁缺页，系统可以增加若干物理块
调入策略
何时调入
预调页策略：一次调入若干个相邻的页
主要用于进程的首次调入，由程序员指定
请求调页策略：当发现页面不在内存时再调入
从何处调入
若对换区足够大
都从对换区调入
若对换区不够大
不会修改的
文件区
可能被修改的
系统区
UNIX方式
未运行过的页面从文件区调入
运行过但又被调出页面从对换区调入
调入过程
自己能叙述下来（P173）
缺页率=访问页面失败次数/总的页面访问次数
置换算法
最佳置换算法（OPT）
置换出未来最长时间不被访问的页面（不可实现）
先进先出（FIFO）置换算法
设第一个队列，总是淘汰最先进入内存的页面
最近最久未使用（LRU）置换算法
置换出最近最久未使用的页面
需要一组寄存器或一个栈
最少使用（LFU）置换算法
Clock置换算法
简单型
某页被访问，置1；置换算法检查到0，换出，检查到1，置0
改进型
再考虑一个因素：置换代价（修改过的页面置换代价大）
将所有页面分为四类，然后循环查找，在第二轮把访问位1改为0，最多需要四轮
页面缓冲算法（PBA）
空闲页面链表：系统掌握的空闲物理块
修改页面链表：已修改页面换出时挂到修改页面链表上，不立即换出
"抖动"与工作集
抖动
现象：随着进程数量的增多，处理机利用率先上升后下降
原因：缺页率上升，每个进程大部分时间都用于页面的换入换出
概念
驻留集：给一个进程分配的物理页框的集合
工作集：进程在时间间隔中引用的页面集合
驻留集一般要大于工作集
抖动 的预防策略
采用局部置换策略
使用工作集算法
利用"L=S"调节缺页率
L：缺页之间平均时间；S：置换一个页面所需时间
暂停一部分进程
实现方式
请求分段
硬件支持
请求段表机制；缺页中断机构、地址变换机构
共享与保护
共享段表
进程使用共享段，便在共享段表的一个表项纪录使用信息
分段保护
越界检查
存取控制检查
环保护机制
内环权限高，外环权限低