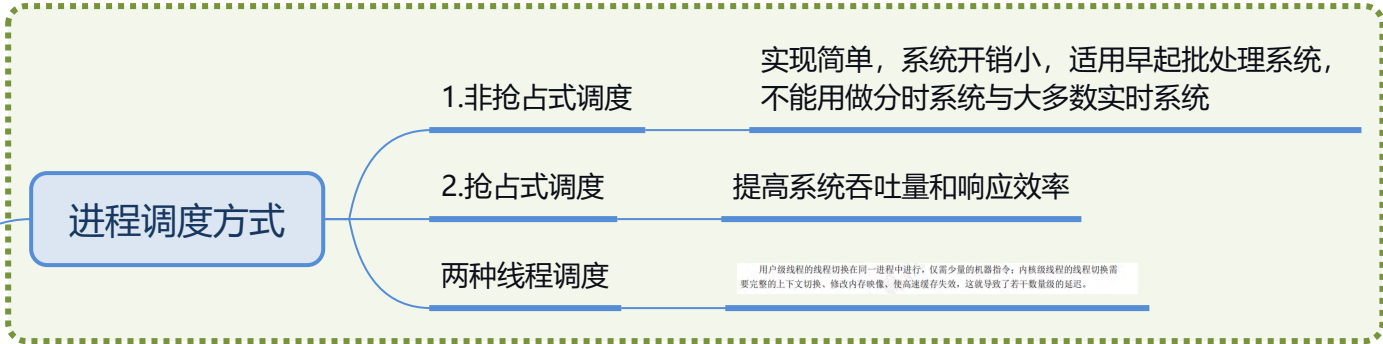
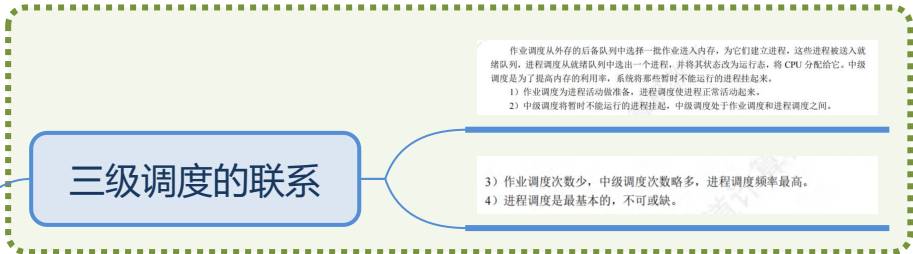
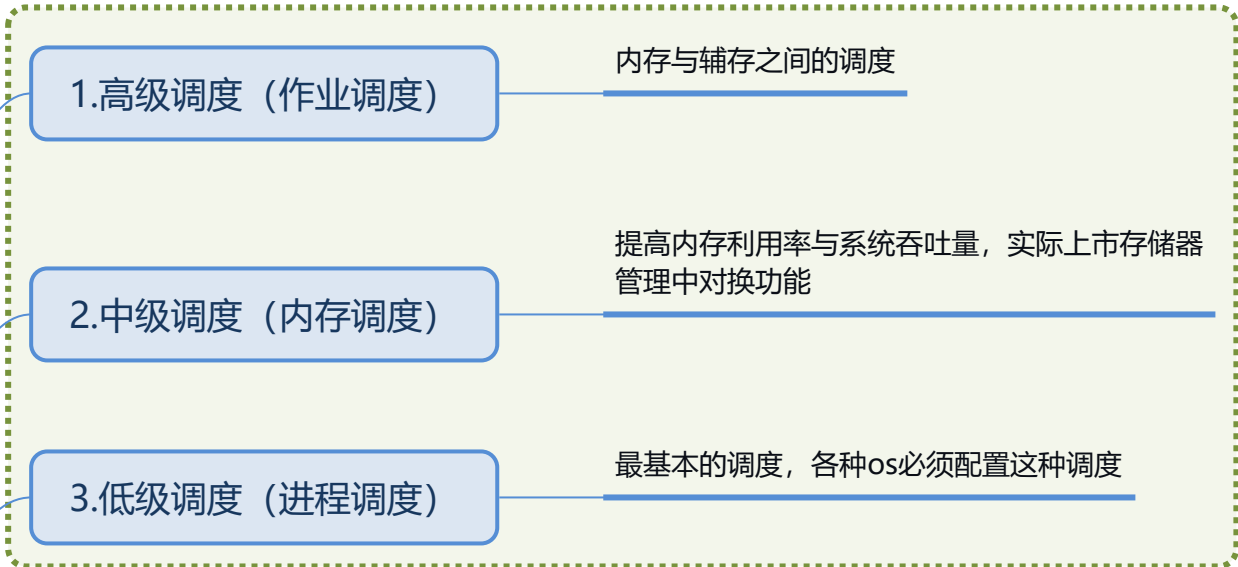


2.2CPU调度



切换 CPU 到另一个进程需要保存当前进程状态并恢复另一个进程的状态，这个任务称为上下文切换。进程上下文采用进程 PCB 表示，包括 CPU 寄存器的值、进程状态和内存管理信息等。当进行上下文切换时，内核将旧进程状态保存在其 PCB 中，然后加载经调度而要执行的新进程的上下文。在切换过程中，进程的运行环境产生实质性的变化。上下文切换的流程如下：

1) 挂起一个进程，将 CPU 上下文保存到 PCB，包括程序计数器和其他寄存器。
2) 将进程的 PCB 移入相应的队列，如就绪、在某事件阻塞等队列。
3) 选择一个进程执行，并更新其 PCB。
4) 恢复新进程的 CPU 上下文。
5) 跳转到新进程 PCB 中的程序计数器所指向的位置执行。

(2) 上下文切换的消耗

上下文切换通常是计算密集型的，即它需要相当可观的 CPU 时间，在每秒几十上百次的切换中，每次切换都需要纳秒量级的时间，所以上下文切换对系统来说意味着消耗大量的 CPU 时间。有些 CPU 提供多个寄存器组，这样，上下文切换就只需要简单改变当前寄存器组的指针。

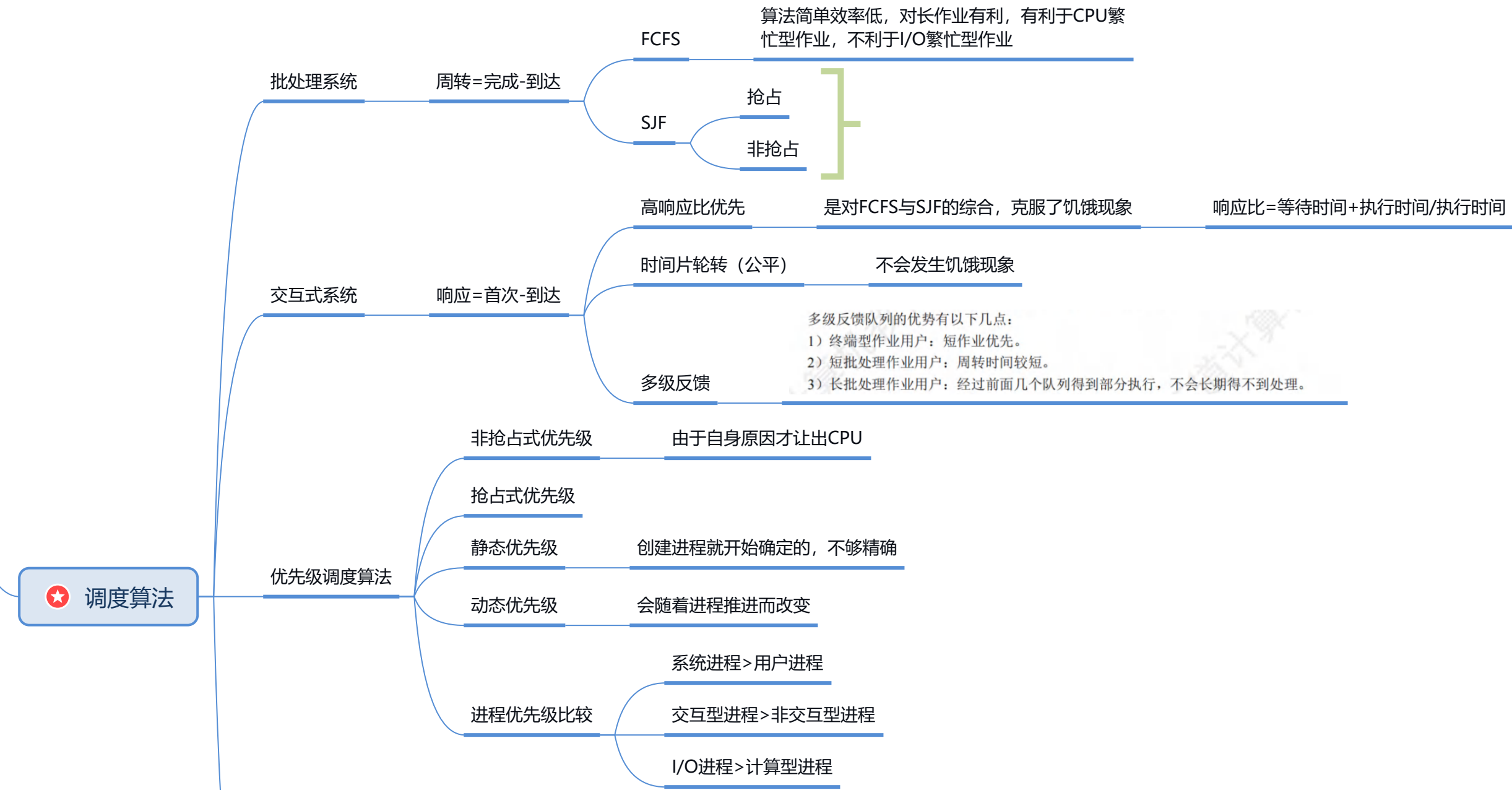
(3) 上下文切换与模式切换

模式切换与上下文切换是不同的，模式切换时，CPU 逻辑上可能还在执行同一进程。用户进程最开始都运行在用户态，若进程因中断或异常进入核心态运行，执行完后又回到用户态刚被中断的进程运行。用户态和内核态之间的切换称为模式切换，而不是上下文切换，因为没有改变当前的进程。上下文切换只能发生在内核态，它是多任务操作系统中的一个必需的特性。

注意

调度和切换的区别：调度是指决定资源分配给哪个进程的行为，是一种决策行为；切换是指实际分配的行为，是执行行为。一般来说，先有资源的调度，然后才有进程的切换。

切换进程时的操作



	先来先服务	短作业优先	高响应比优先	时间片轮转	多级反馈队列
能否可抢占	否	可以	可以	可以	队列内算法不一定
优点	公平，实现简单	平均等待时间、平均周转时间最优	兼顾长短作业	兼顾长短作业	兼顾长短作业，有较好的响应时间，可行性强
缺点	不利于短作业	长作业会饥饿，估计时间不易确定	计算响应比的开销大	平均等待时间较长，上下文切换浪费时间	最复杂
适用于	无	批处理系统	无	分时系统	相当通用