第二次作业

5.1 为什么区分 CPU 密集型程序和 I/O 密集型程序是重要的？

答：CPU 密集型程序只有少量长 CPU 执行，而 I/O 密集型程序通常有大量短 CPU 执行，区分这两种程序类型，在 I/O 密集型程序在等待 I/O 操作时，通过中期调度程序或短期调度程序使 CPU 密集型程序被换入，减少 CPU 空闲时间，提高 CPU 利用率。

5.4 本章中讨论针对多个内核数据结构的可能竞争条件。大多数调度算法采用一个**运行队列**，用于维护可在处理器上运行的进程。对于多核操作系统，通常有两个选择：

1）每个处理核都有各自的运行队列，或 2）所有处理核共享一个运行队列。

这两种方法的优缺点分别是什么？

答：

1）单队列调度（SQMS）：

优点：实现简单、相对于单处理核的调度算法不需要修改太多内容；

缺点：(1). 缺乏扩展性，程序中存在一些加锁的“原子操作”，但是随着处理核的数目增加，单个锁的争用增加，系统在锁上花费过多时间，CPU 利用率较低。

(2). 缓存亲和性问题，每个进程执行一个时间片，若处理核数目和进程数目互质，则会出现处理核要在不同时间片执行完全不同的进程，造成缓存的大规模不命中。

2）多队列调度（MQMS）：

优点：具有良好的可扩展性（锁的开销较小）和天生良好的缓存亲和性

缺点：负载不均，就如计算机网络中的信道划分，多队列会使有的处理核分到较容易完成的任务，这些核很快完成进程执行之后空闲下来，而其他核心还在执行中，造成 CPU 性能的浪费。

5.5 假设采用指数平均公式来预测下个 CPU 执行的长度。当采用如下参数时，该算法的含义是什么？

$$a.\ \alpha = 0\text{和}\tau_0 = 100ms \qquad b.\ \alpha = 0.99\text{和}\tau_0 = 10ms$$

a. $\alpha = 0$，则由 $\tau_n = \alpha t_0 + (1-\alpha)\tau_{n-1}$，$\tau_n = \tau_{n-1} = \tau_0 = 100ms$ 这意味着最近历史没有影响，当前情形为瞬态，预估的下一 CPU 执行长度总是 100ms。

b. $\alpha = 0.99$，由 $\tau_n = \alpha t_0 + (1-\alpha)\tau_{n-1}$，$\tau_n = 0.99t_0 + 0.01\tau_{n-1}$，最近一次的历史调度时间所占的影响因子为 0.99，说明预测的 CPU 下一执行长度几乎全部依赖于最近一次的调度执行时间长度。

课堂测验：

根据进程状态图分析并回答分时系统与实时系统实现上的区别：

答：

1. 从 CPU 调度（“scheduler dispatch”）来说，分时操作系统具备“中期调度程序”，可以将内存中的进程调入调出，保证各个进程的均匀执行，使得交互性更好，而实时操作系统按照提前计算好的优先级顺序，进行优先级调度或抢占式调度，保证对某些优先级高的进程的

实时响应。

2. 实时操作系统为保证优先级，使用了单调速率调度和最早截止期限优先调度，而分时操作系统为了保证良好交互性则使用了轮转调度或多级队列调度。