

操作系统

Operating Systems

L14 CPU调度策略

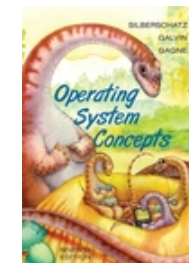
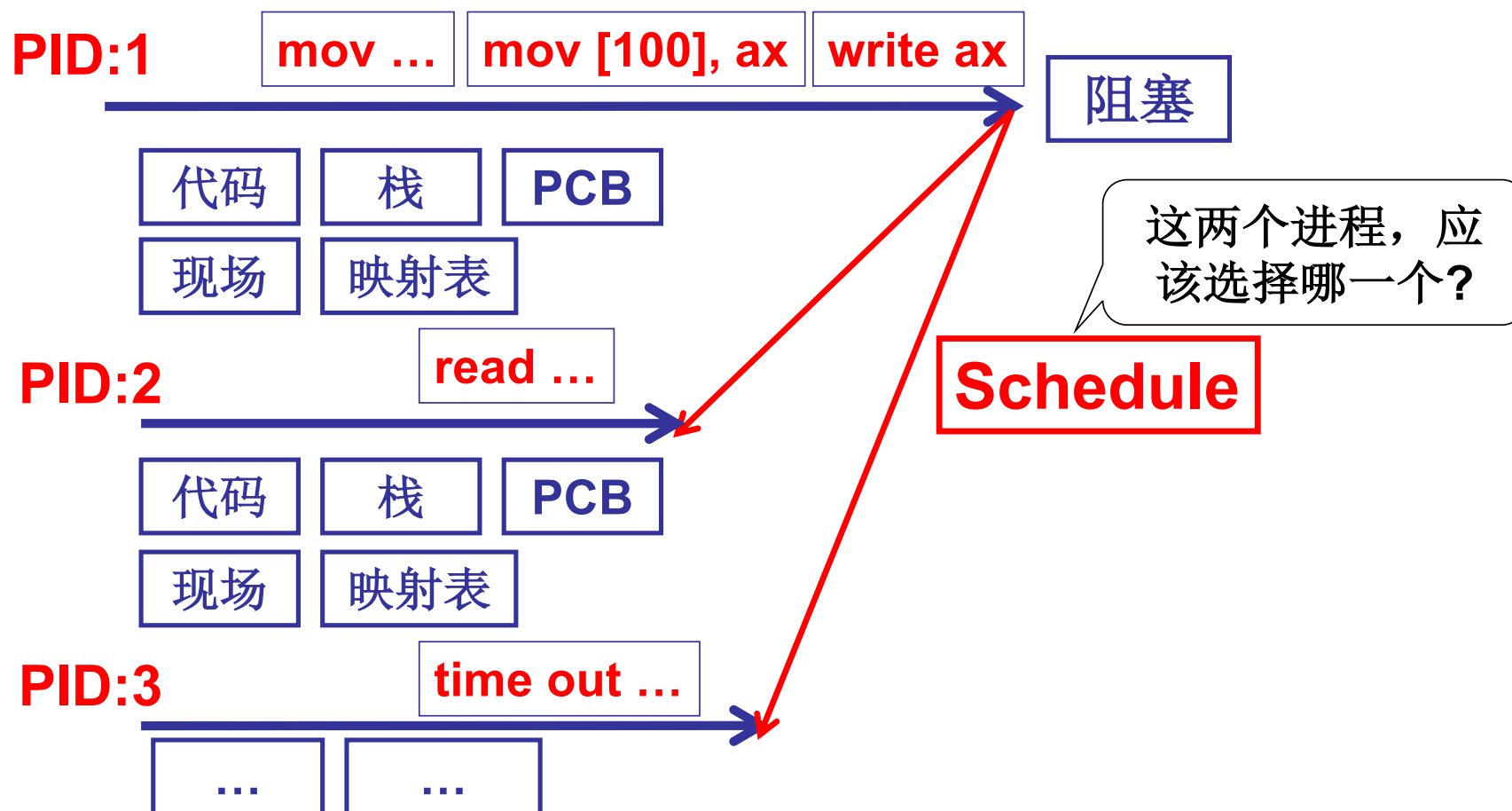
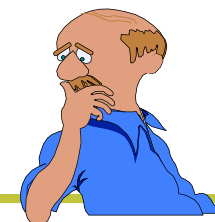
CPU Scheduling

授课教师：李治军

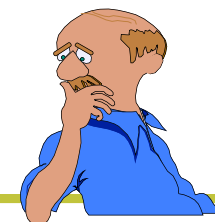
lizhijun_os@hit.edu.cn

综合楼411室

多进程图像与CPU调度



CPU调度(进程调度)的直观想法



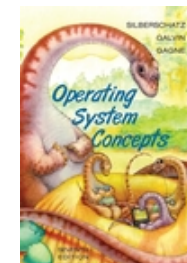
■ FIFO?

- 谁先进入，先调度谁：简单有效 银行、食堂
- 一个只简单询问业务的人该怎么办？

■ Priority?

- 任务短可以适当优先 但你怎么知道这个任务将来会执行多长时间呢？
- 这人的询问越来越长怎么办？
- 那如果一个银行业务很长是因为客户需要填写一个很长的表，该怎么办？

应该还有很多这样的询问...



面对诸多场景，如何设计调度算法？

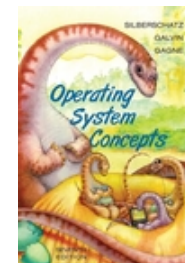
■ 我们的算法应该让什么更好？

- 面对客户：银行调度算法的设计目标应该是 用户满意
- 面对进程：CPU调度的目标应该是 进程满意

■ 那怎么才能让进程满意呢？ 时间...

- 尽快结束任务：周转时间(从任务进入到任务结束)短
- 用户操作尽快响应：响应时间(从操作发生到响应)短
- 系统内耗时间少：吞吐量(完成的任务量)

■ 总原则：系统专注于任务执行，又能合理调配任务...



如何做到合理？ 需要折中，需要综合...

■ 吞吐量和响应时间之间有矛盾...

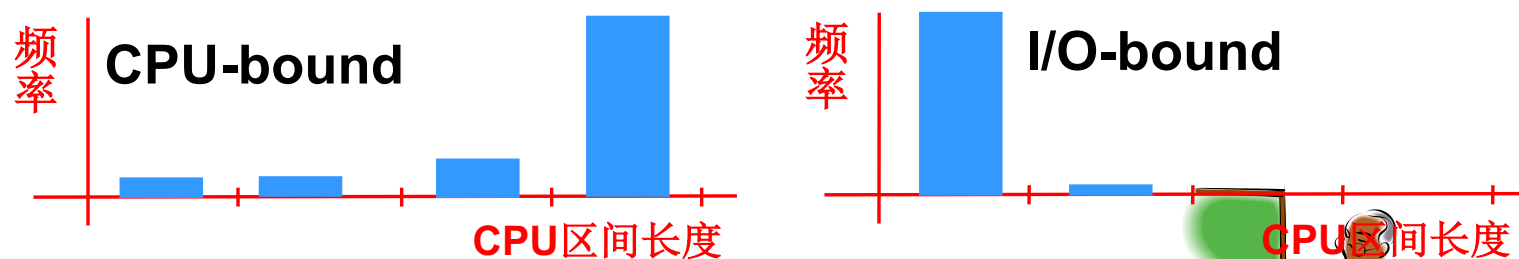
这样的矛盾还有很多...

■ 响应时间小 \Rightarrow 切换次数多 \Rightarrow 系统内耗大 \Rightarrow 吞吐量小

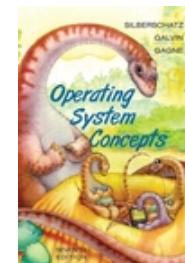
■ 前台任务和后台任务的关注点不同...

■ 前台任务关注响应时间，后台任务关注周转时间

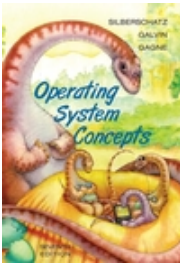
■ IO约束型任务和CPU约束型任务有各自的特点



折中和综合让操作系统变得复杂，
但有效的系统又要求尽量简单...



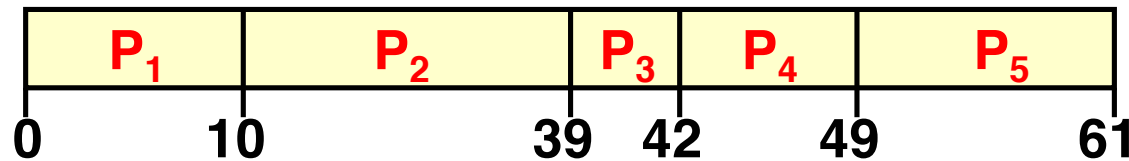
各种**CPU**调度算法



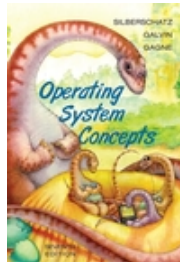
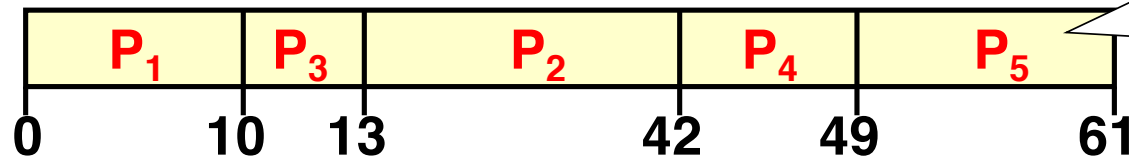
First Come, First Served (FCFS)

任务	到达时间	CPU区间(ms)
P_1	0.001	10
P_2	0.002	29
P_3	0.003	3
P_4	0.004	7
P_5	0.005	12

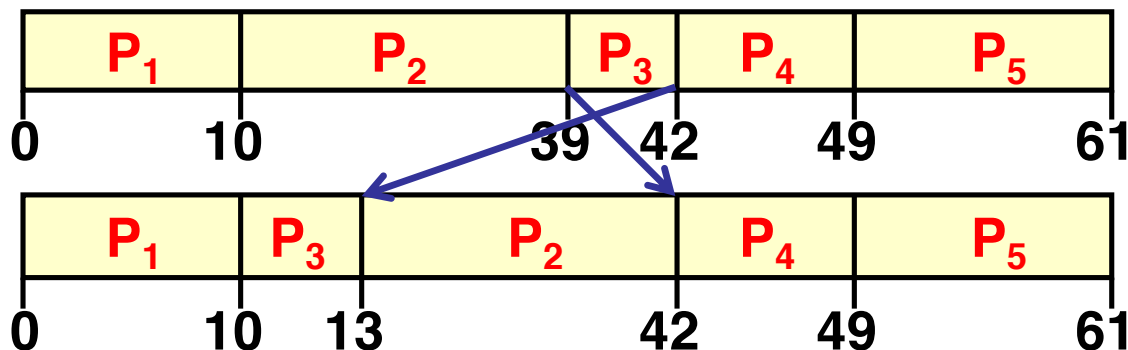
平均周转时间:
 $(10+39+42+49+61) / 5 = 40.2$



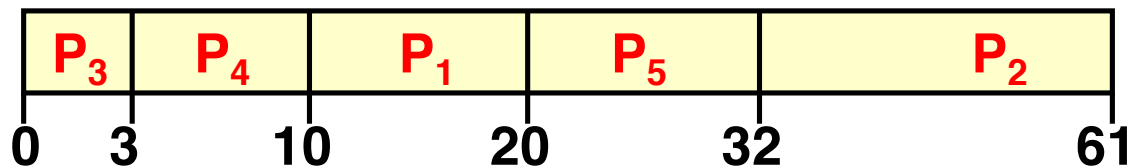
平均周
转时间?



如何缩短周转时间? SJF: 短作业优先



- P₃提前的完成时间较P₂拖后的完成时间要小

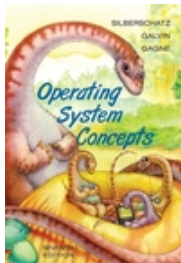


平均周转时间?

- 如果调度结果是 $p_1p_2\dots p_n$, 则平均周转时间为:

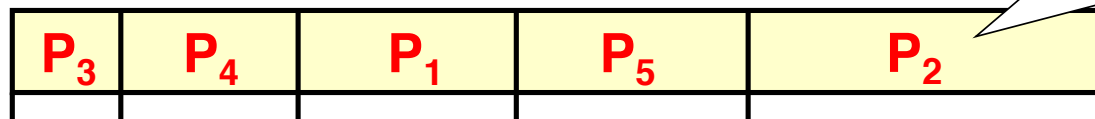
$$p_1 + p_1 + p_2 + p_1 + p_2 + p_3 + \dots = \sum (n+1-i)p_i$$

如果存在 $i < j$ 而 $p_i > p_j$, 交换 p_i, p_j 会怎么样?



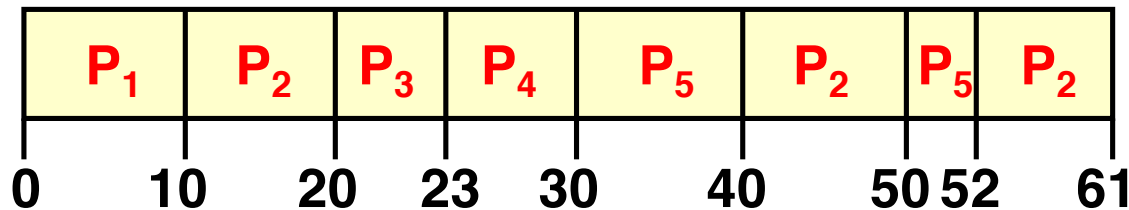
响应时间该怎么办？

P₂用户的操作...



■ 如何解决？开动脑筋...

■ **RR: 按时间片来轮转调度**



■ 时间片大：响应时间太长；时间片小：吞吐量小

■ 折衷：时间片**10-100ms**，切换时间**0.1-1ms(1%)**

■ 如果**CPU**更快，时间片会怎么样？

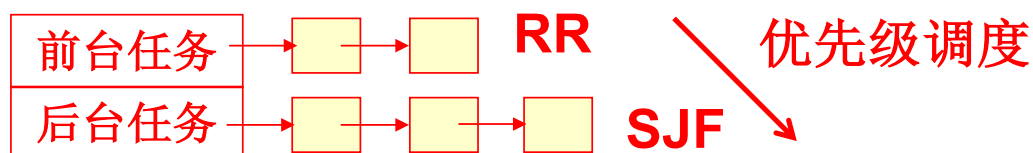


响应时间和周转时间同时存在，怎么办？

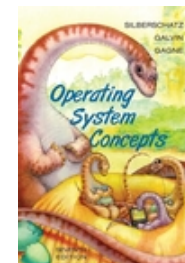
■ Word很关心响应时间，而gcc更关心周转时间，两类任务同时存在怎么办？

■ 一个调度算法让多种类型的任务同时都满意，怎么办？

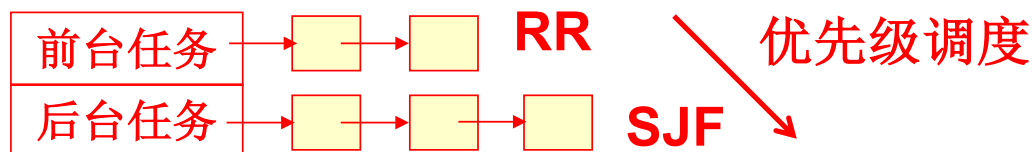
■ **直观想法**：定义前台任务和后台任务两队列，前台**RR**，后台**SJF**，只有前台任务没有时才调度后台任务



■ 但是这会产生很多问题？想一想会出现什么问题？



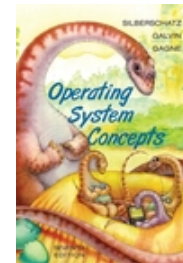
如果一直有前台任务...



一个故事: 1973年关闭的MIT的IBM 7094时, 发现有一个进程在1967年提交但一直未运行

- 后台任务可能一直得不到运行
- 后台任务优先级动态升高, 但后台任务(用SJF调度)一旦执行, 前台的响应时间...
- 前后台任务都用时间片, 但又退化为了RR, 后台任务的SJF如何体现? 前台任务如何照顾?

问题: 分析到此处了, 我们该怎么办?



还有很多问题...

- 我们怎么知道哪些是前台任务，哪些是后台任务，**fork**时告诉我们吗？
- **gcc**就一点不需要交互吗？**Ctrl+C**按键怎么工作？**word**就不会执行一段批处理吗？**Ctrl+F**按键？
- **SJF**中的短作业优先如何体现？如何判断作业的长度？这是未来的信息...

