《计算机操作系统》实验报告

实验题目：操作系统的进程调度

姓名：严昕宇　　学号：20121802　　实验日期：2022.12.02

实验环境：

实验设备：Lenovo Thinkbook16+ 2022

开发环境：CLion 2022.3

实验目的：

进程是操作系统最重要的概念之一，进程调度又是操作系统核心的主要内容。本实习要求学生独立地用高级语言编写和调试一个简单的进程调度程序。调度算法可任意选择或自行设计。例如，简单轮转法和优先数法等。本实习可加深对于进程调度和各种调度算法的理解。

实验要求：

1. 设计一个有 n 个进程工行的进程调度程序。每个进程由一个进程控制块（PCB）表示。进程控制块通常应包含下述信息：进程名、进程优先数、进程需要运行的时间、占用 CPU 的时间以及进程的状态等，且可按调度算法的不同而增删

2. 调度程序应包含 2～3 种不同的调度算法，运行时可任意选一种，以利于各种算法的分析比较

3. 系统应能显示或打印各进程状态和参数的变化情况，便于观察诸进程的调度过程

实验内容：

1. 题目

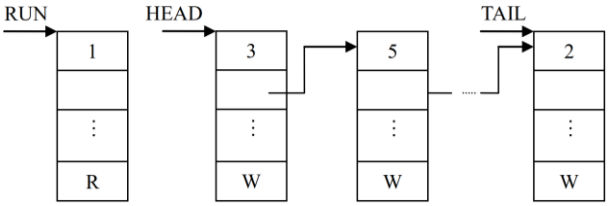
本程序可选用优先数法或简单轮转法对五个进程进行调度。每个进程处于运行 R(run)、就绪 W(wait)和完成 F(finish)三种状态之一，并假设起始状态都是就绪状态 W。为了便于处理，程序进程的运行时间以时间片为单位计算。各进程的优先数或轮转时间片数、以及进程需要运行的时间片数，均由伪随机数发生器产生。

进程控制块结构如下：

PCB：

- 进程标识数
- 链指针
- 优先数/轮转时间片数
- 占用 CPU 时间片数
- 进程所需时间片数
- 进程状态

进程控制块链结构如下：



其中：RUN—当前运行进程指针；

HEAD—进程就绪链链首指针；

TAID—进程就绪链链尾指针。

2. 算法与框图

(1) 优先数法。进程就绪链按优先数大小从高到低排列，链首进程首先投入运行。每过一个时间片，运行进程所需运行的时间片数减 1，说明它已运行了一个时间片，优先数也减 3，理由是该进程如果在一个时间片中完成不了，优先级应该降低一级。接着比较现行进程和就绪链链首进程的优先数，如果仍是现行进程高或者相同，就让现行进程继续进行，否则，调度就绪链链首进程投入运行。原运行进程再按其优先数大小插入就绪链，且改变它们对应的进程状态，直至所有进程都运行完各自的时间片数。

(2) 简单轮转法。进程就绪链按各进程进入的先后次序排列，进程每次占用处理机的轮转时间按其重要程度登入进程控制块中的轮转时间片数记录项(相当于优先数法的优先数记录项位置)。每过一个时间片，运行进程占用处理机的时间片数加 1，然后比较占用处理机的时间片数是否与该进程的轮转时间片数相等，若相等说明已到达轮转时间，应将现运行进程排到就绪链末尾，调度链首进程占用处理机，且改变它们的进程状态，直至所有进程完成各自的时间片。

(3) 程序框图如下图 1 所示。

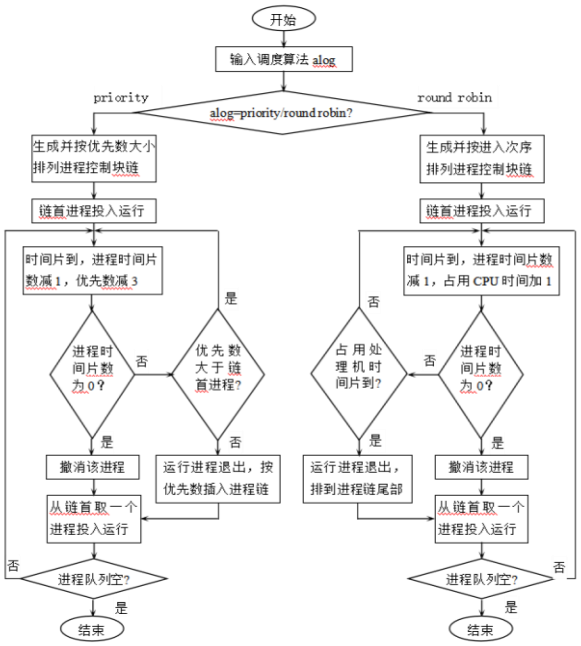


图 1 进程调度框图

3. 程序运行结果格式

(1) 程序运行结果格式

```
TYPE THE ALGORITHM:PRIORITY
OUTPUT OF PRIORITY
==============================
RUNNING PROC.          WAITING QUEUE
                        3   4   1   5
==============================
ID                  1   2   3   4   5
PRIORITY            9  38  30  29   0
CPUTIME             0   0   0   0   0
ALLTIME             3   3   6   3   4
STATE               W   R   W   W   W
NEXT                5   3   4   1   0
==============================
```

```
……
……
……
==============================
SYSTEM FINISHED
```

(2) 说明

程序启动后，屏幕上显示“TYPE THE ALGORTITHM”，要求用户打入使用何种调度算法。本程序只编制了优先数法（“priority”）和简单轮转法（“RoundRobin”）两种。打入某一算法后，系统自动形成各进程控制块，实施该算法的进程调度算法，并打印各进程在调度过程中的状态和参数的变化。

4. 小结

本实习简单地模拟了进程调度的二种方法，在进程运行时间和进程状态变化方面都做了简化，但已能反映进程调度的实质。通过实习能加深对进程调度的理解和熟悉它的实施方法。

结果：

优先数法：

```
TYPE THE ALGORITHM (1.Priority 2. RoundRobin 3. FB):
1
OUTPUT OF PRIORITY

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PID     Time    Slices  Priority
0       5       3       8
1       8       3       12
2       12      3       15
3       15      3       18
4       18      3       3
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
RUNNING PROC.   WAITING QUEUE:
3               2 1 0 4
RUNNING PROC.   WAITING QUEUE:
2               3 1 0 4
RUNNING PROC.   WAITING QUEUE:
3               1 2 0 4
RUNNING PROC.   WAITING QUEUE:
1               2 3 0 4
RUNNING PROC.   WAITING QUEUE:
2               3 1 0 4
RUNNING PROC.   WAITING QUEUE:
3               1 2 0 4
RUNNING PROC.   WAITING QUEUE:
1               2 3 0 4
RUNNING PROC.   WAITING QUEUE:
2               3 0 1 4
RUNNING PROC.   WAITING QUEUE:
3               0 2 1 4
RUNNING PROC.   WAITING QUEUE:
0               2 3 1 4
RUNNING PROC.   WAITING QUEUE:
2               3 1 0 4
RUNNING PROC.   WAITING QUEUE:
3               1 0 4 2
```

简单轮转法：

```
TYPE THE ALGORITHM (1.Priority 2. RoundRobin 3. FB):
2
OUTPUT OF PRIORITY

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PID     Time    Slices  Priority
0       12      1       1
1       16      1       1
2       1       1       1
3       4       1       1
4       8       1       1
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
RUNNING PROC.   WAITING QUEUE:
0               2 4 1 3
RUNNING PROC.   WAITING QUEUE:
2               4 0 1 3
RUNNING PROC.   WAITING QUEUE:
4               0 2 1 3
RUNNING PROC.   WAITING QUEUE:
0               2 4 1 3
RUNNING PROC.   WAITING QUEUE:
4               0 1 3
RUNNING PROC.   WAITING QUEUE:
0               3 1 4
RUNNING PROC.   WAITING QUEUE:
3               4 1 0
RUNNING PROC.   WAITING QUEUE:
4               0 1 3
RUNNING PROC.   WAITING QUEUE:
0               3 1 4
RUNNING PROC.   WAITING QUEUE:
3               4 1 0
RUNNING PROC.   WAITING QUEUE:
4               0 1 3
RUNNING PROC.   WAITING QUEUE:
0               3 1 4
```

实习题

1．编制和调试示例给出的进程调度程序，并使其投入运行。

答：如之前的实验结果所示。

2．自行设计或改写一个进程调度程序，在相应机器上调试和运行该程序，其功能应该不亚于示例。

【提示】可编写一个反馈排队法(FB方法)的进程调度程序。该算法的基本思想是设置几个进程就绪队列，如队列1……队列i，同一队列中的进程优先级相同，可采用先进先出方法调度。各队列的进程，其优先级逐队降低。即队列1的进程优先数最高，队列i的最低。而时间片，即以此占用CPU的时间正好相反，队列1的最短，队列i则最长。调度方法是开始进入的进程都在队列1中参加调度，如果在一个时间片内该进程完不成，应排入队列2，即优先级要降低，但下一次运行的时间可加长(即时间片加长了)。以此类推，直至排到队列i。调度时现在队列1中找，待队列1中已无进程时，再调度队列2的进程，一旦队列1中有了

进程，又应返回来调度队列1的进程。这种方法最好设计成运行过程中能创造一定数量的进程，而不是一开始就生成所有进程。

【提示】可综合各种算法的优先，考虑在各种不同情况下的实施方法，如上述FB算法。也可选用有关资料中报导的一些方法，加以分析、简化和实现。

答：

```
TYPE THE ALGORITHM (1.Priority 2. RoundRobin 3. FB):
3
OUTPUT OF PRIORITY

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PID     Time    Slices  Priority
0       7       1       20
1       11      1       20
2       14      1       20
3       17      1       20
4       2       1       20
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
RUNNING PROC.   WAITING QUEUE:
0               2 4 1 3
RUNNING PROC.   WAITING QUEUE:
2               4 1 3 0
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PID     Time    Slices  Priority
5       14      1       20
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
RUNNING PROC.   WAITING QUEUE:
4               1 3 5 2 0
RUNNING PROC.   WAITING QUEUE:
1               3 5 0 2 4
RUNNING PROC.   WAITING QUEUE:
3               5 0 1 4 2
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PID     Time    Slices  Priority
6       17      1       20
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
RUNNING PROC.   WAITING QUEUE:
5               6 0 3 1 4 2
RUNNING PROC.   WAITING QUEUE:
6               0 3 5 1 4 2
RUNNING PROC.   WAITING QUEUE:
0               3 5 6 1 4 2
```

3．直观地评测各种调度算法的性能。

答：如之前的实验结果所示。

算法名称	算法思想	适用调度场景	是否可抢占	优缺点	是否导致饥饿
先来先服务算法（FCFS）	按照作业/进程到达的先后次序来进行调度	作业调度&进程调度	不可抢占	优点：简单、算法简单；缺点：作业的平均带权周转时间太长，对于短作业不友好	不会
短作业/进程优先算法（SJF/SPF）	要求最短的作业/进程先得到调度	作业调度&进程调度	通常不可抢占，但是有可抢占版本：最短剩余时间优先算法（SRTN）	平均周转时间通常是最短的	会，如果一直进入短作业，可能长作业会被饿死
高响应比优先算法（HRRN）	每次调度前就算每个作业/进程的响应比，响应比高的优先得到调度	作业调度&进程调度	非抢占式算法，当前作业/进程主动放弃cpu使用才需调度	综合考虑了等待时间和服务时间，对于长作业来说，随着等待时间增长，响应比也会增长，不会出现饥饿现象	不会
轮转调度算法（RR）	每个进程最多执行一个时间片长度，没执行完就保存结果，并存放到就绪队列，接着执行下一个进程	进程调度	抢占式	非常公平的分配方式，利于进行人机交互	不会
优先级调度算法	根据优先级来进行调度（静态/动态优先级）	进程调度	根据需求，可以是抢占式，可以使非抢占式	考虑了进程任务的紧迫性	会，如果一个进程优先级太低，可能不会被执行
多级反馈队列（multileved feedback queue）（公认较好的进程调度算法）	1.将以往的一个就绪队列设置为多个就绪队列（n个），并使队列的优先级依次递减，时间片依次递增。2. 每个队列采用FCFS算法，若在该时间片内完成就撤离系统，否则加入下一队列，若排到最后的一个队列第n队列，则采用RR算法执行。若高优先级的队列没有任务执行，才能执行低优先级队列中的任务，如果正在执行低优先级的任务，来了一个高优先级的任务，直接暂停低优先级任务，执行高优先级任务	进场调度	抢占式	不必事先知道进程的所需执行时间，思想虽然先进，却较为复杂	通常不会

思考题

1．示例中的程序，没有使用指针型(pointer)数据结构，如何用指针型结构改写本实例，使更能体现 C 语言的特性。

答：示例中和本次实验中所使用的是静态链表，若使用指针型(pointer)数据结构，则需要 PCB 结构体中加入指向下一个结点的 next 指针即可。

2．如何在程序中真实地模拟进程运行的时间片？

答：可以使用 sleep()函数真实地模拟进程运行的时间片(本实验中已实现)。

3．如果增加进程的“等待”状态，即进程因请求输入输出等问题而挂起的状态，如何在程序中实现？

答：可以增加一个用于判断的 if 语句，如果进程有相应的请求输入输出，则挂起，同时保持相关的量不变。如果没有相应请求，则不变。

实验体会

本次实验中，我通过代码模拟了三个进程调度算法。即使这些调度算法已在操作系统的理论课中学习过，但在编写和调试程序过程中也遇到了一些困难和问题。例如，在生成伪随机数时如何保证它们符合一定范围和分布；在插入就绪队列时如何保证按照优先数排序；在计算平均周转时间时如何考虑权重因素等。这些问题都促使我去查阅相关资料，或者参考其他同学或老师给出的代码示例，并通过不断地修改和测试来找出最佳解决方案。

通过本次实验，我加深了对于操作系统中关于多道程序设计、并发性、共享性、虚拟性和异步性等基本概念和特征以及对于操作系统核心之一的多道程序设计技术与处理机分配技术，即处理机调度技术的理解与掌握水平。

总之，这次实验让我收获颇丰，在理论知识方面增长见识，我也了解到，现代操作系统内部的进程调度方法远比所写的要复杂更多，如果要更好的掌握，还需更深入的学习。