“测测你的GPA”：实时操作系统调度算法评估框架的设计与实现

操作系统专题训练

陈张萌

2021.11.04

目录

项目背景

背景

- 在遥远的青蛤大学，有一个倒霉的小陈同学，她为了毕业必须选很多个学分的课，而且每门课都不能挂科，否则就会毕不了业、失去推研资格，后果十分悲惨。
- 然而课程太多、作业太多，小陈无法把每份作业都完美地完成，只好对每份作业都只完成一部分，以保证自己不能挂科。
- 于是她希望能采取某种写作业调度策略，使得每门课都不挂科，且课程GPA要尽可能地高。

系统定义与设计

写作业模型

- 每一个作业具备2个玩家已知的属性：score和ddl，完成作业的最后得分定义如下：

$$final = score * (grade >= passline?grade : 0)$$

- 每位同学的GPA为 $: \frac{\sum final}{\sum score}$

写作业模型

- score：是“学分数”，score越高意味着这个作业越重要
- ddl：是一个时间，必须在这个时间之前完成的作业才能计算为完成度，ddl之后完成的作业不会算分
- grade：作业的完成度，即：ddl之前作业完成了多少。具体评估方式
- passline：及格线，满分为100，只有ddl之前的完成度达到（指大于等于）及格线时，才能得到对应分数，否则得到的分数为0。在我们的游戏中，passline统一设置为60

写作业模型

- 每份作业还有两个玩家无法提前预知的属性：
- time，表示作业布置时间，只有当作业布置之后玩家才会知道有这个作业。
- time_total，指的是将该作业全部写完需要的总工作量（以时间来衡量），玩家在写完作业之前是无法知道到底要写多久的

写作业模型

- “一个学期”是一轮游戏，有固定的总时间，时间到则游戏结束，开始计算GPA。
- 游戏开始时，只有0个作业。在游戏的任意一个时刻，都可以产生一份新的作业。作业可以分阶段完成，且可以在任意一个阶段停止，去完成另一个作业。小陈（游戏玩家）需要做的事情是：实现一个写作业调度方案，以此决定写作业顺序，使得最终获得的GPA尽可能地高。

系统设计：调度算法评估框架

- 此游戏可以实现为一个实时操作系统调度算法的评估框架。
- 具体地说，每一个作业都作为一个进程出现，用来评估特定的进程调度算法。每个进程都有一个规定的完成时间，因此这是一个实时操作系统。
- 此系统能够提供调度算法接口，对“玩家”给出的进程调度算法使用特定的测试数据进行测试，并给出评价。

系统设计：流程

- 当每产生一次时钟中断时，首先进入进程调度流程。进程调度器实现为一个函数，进程调度器每次能够得到以下信息：
- 一个数组，其中数组包含了当前仍然未运行结束的所有进程（除了初始进程）的相关信息，数组的每一个元素包括：
  - 进程的pid，进程的score，该进程出现的时间，该进程已运行的时长，该进程运行的ddl
- 进程调度器返回给内核的信息是待调度进程的pid，内核以此决定接下来调度哪个进程。
- 当每次发生时钟中断时，有一定概率发生以下事件：增加了一个新的进程。这就意味着，进程调度器需要在每次拿到控制权时，都重新读取当前的进程信息。

系统设计：关于任务完成度评价方案

- 任务完成度 $grade = \frac{time\_run}{time\_total}$
- time_run用来记录在某种特定的调度算法下，该进程分到了多少时间用来执行（不一定将进程执行完）
- time_total，指的是将该进程全部写完需要的总工作量（以时间来衡量）
- 这就需要一个精确计算任务执行时间的方案（参考：[评测鸭JudgeDuck-OS]）

系统设计：关于任务完成度评价方案

- 总的来说，是基于x86架构实现一个实时操作系统，能够对特定场景的调度算法提供一个评估框架
- 特定场景：进程有截止时间，有优先级

相关研究

相关研究

- 对实时操作系统的性能评估模型注重对整体性能评估，例如：
  - RhealStone提供了6个关键操作的时间量。它们是任务切换时间、抢占时间、中断延迟时间、信号量混洗时间、死锁解除时间和数据吞吐时间
  - 对调度算法的评估还不够充分
- 对分时操作系统的调度算法的评估不能简单应用

对实时操作系统的性能评估模型

- 常用的评估方式是：任务错失率，即运行大量任务并调度，来计算有多少任务没有在规定时间内完成。
  - 首先对于已经错过的任务来说，调度时完全可以选择放弃此任务
  - 第二是不同任务的重要程度不同
  - 第三是没有考虑部分完成的情况（对于不可部分完成的任务，只需要将及格线设置为100即可）

进展介绍

进展介绍

- 完成了x86内核的以下功能：
- bootloader和串口输出
- 内存管理
- 中断异常处理
- 用户内核态切换
- 进程管理，给出了简单调度
- 测量调度算法的运行时间
- 调研了JudgeDuck-OS

后续计划

后续计划

- 重新设计进程管理模块，给出良好的接口
- 测量某个进程运行时间
- 对该系统进行测试
  - 调研相关的调度算法测试框架
  - 自己实现一个简单调度算法

谢谢