Project 4: 调度算法

Chentao Wu 吴晨涛
Professor
Dept. of CSE, SJTU
wuct@cs.sjtu.edu.cn

课程目标

- 完成 FCFS，SJF，RR，Priority 和 Priority_RR 调度算法，实现对应的调度程序
  - Project source code: https://github.com/greggagne/osc10e/tree/master/ch5/project/posix
  - 下载代码，完成上述算法。代码文件命名请参考 README
  - 源代码已经定义算法所需要的基本单元，并统一了调用接口
  - 算法原理略，请参考教材

Source Code

- main 函数在 driver.c 里
- 程序从输入文件中获取所有进程信息，添加进调度队列（while 循环）
- 处理完输入后模拟调度算法，开始调度（schedule 函数）

```c
int main(int argc, char *argv[])
{
    FILE *in;
    char *temp;
    char task[SIZE];

    char *name;
    int priority;
    int burst;

    in = fopen(argv[1],"r");

    while (fgets(task,SIZE,in) != NULL) {
        temp = strdup(task);
        name = strsep(&temp,",");
        priority = atoi(strsep(&temp,","));
        burst = atoi(strsep(&temp,","));

        // add the task to the scheduler's list of tasks
        add(name,priority,burst);

        free(temp);
    }

    fclose(in);

    // invoke the scheduler
    schedule();

    return 0;
}
```

Source Code

- schedule.h 文件已经将函数接口给出，请在相应 .c 文件中实现算法
  - 你需要做：
    - 初始化你的 task_list
    - 在 add 函数中向你的 task_list 中添加 task
    - 在 schedule 函数中实现调度。下面以 FCFS 为例：

```
while task_list not empty:
    task_to_run <- task_list.head
    delete(task_list.head)
    run(task)
```

```
// add a task to the list
void add(char *name, int priority, int burst);

// invoke the scheduler
void schedule();
```

- 注：
  - list.c 文件中实现了 list 的基本功能，包括 insert 和 delete
  - CPU.c 文件中实现了 run 函数

作业及评分

自行阅读课本第五章的 Programming Projects 部分，并完成以下三个任务，完成后共计10分。

- （课本习题 5 分）实现 5 种调度算法。
  - 其中，每个 1 分。
- (Bonus 3 分)
  - （1 分）Each task provided to the scheduler is assigned a unique task (tid). If a scheduler is running in an SMP environment where each CPU is separately running its own scheduler, there is a possible race condition on the variable that is used to assign task identifiers. Fix this race condition using an atomic integer.（请参考教材）
  - （2 分）计算每个调度算法的平均周转时间、平均响应时间和平均等待时间。
    - 部分正确得 1 分，全部正确得 2 分。
- （报告 2 分）做一个简单的报告解释你的代码，报告建议不超过2页（防内卷）。