动态优先级算法

动态优先数是指在进程创建时先确定一个初始优先数， 以后在进程运行中随着进程特性的改变不断修改优先数，这样，由于开始优先数很低而得不到CPU的进程，就能因为等待时间的增长而优先数变为最高而得到CPU运行。

例如：在进程获得一次CPU后就将其优先数减少1，或者进程等待的时间超过某一时限时增加其优先数的值。

在每次运行你所设计的处理器调度程序之前，为每个进程任意确定它的“优先数”和“要求运行时间”。为了调度方便，把五个进程按给定的优先数从大到小连成队列。用一单元指出队首进程，用指针指出队列的连接情况。

数据结构：设计一个链式队列，链式指针代表按照进程优先级将处于就绪状态的进程连接成一个就绪队列。指针指出下一个到达进程的进程控制块首地址。最后一个进程的链指针为NULL。

#include<stdio.h>

#include<stdlib.h>

typedef struct PCB{

int PID;

char state;

int priority;

int runTime;

int workTime;

struct PCB \*next;

}\*process,ptr;

PCB \*ready=NULL,\*p;

int num;

void PCBsort(){

PCB \*first,\*second;

int flag=0;

if((ready==NULL)||((p->priority)<(ready->priority))){

p->next=ready;

ready = p;

}else{

first=ready;

second=first->next;

while(second!=NULL){

if((p->priority)<(second->priority)){

p->next=second;

first->next=p;

second=NULL;

flag=1;

}else{

first=first->next;

second=second->next;

}

}

if(flag==0)first->next=p;

}

}

void inputProcess()

{

int i;

printf("输入%d个进程的信息（PID、优先级、运行时间）\n",num);

for(i=0;i<num;i++){

p=(PCB\*)malloc(sizeof(PCB));

scanf("%d %d %d",&p->PID,&p->priority,&p->runTime);

p->workTime=0;

p->state='w';

p->next=NULL;

PCBsort();

}

}

int space()

{

int k=0;

PCB\* pr=ready;

while(pr!=NULL){

k++;

pr=pr->next;

}

return k;

}

void showInfo(ptr \*pr){

printf("\nPID\t状态\t优先级\t运行时间\t剩余时间\n");

printf("————————————————————————————\n");

printf(" %d\t",pr->PID);

printf(" %c\t",pr->state);

printf("%d\t",pr->priority);

printf("%d\t\t",pr->runTime);

printf("%d\t",pr->runTime-pr->workTime);

printf("\n");

}

void priorityRun()

{

int len,h=0;

char ch;

inputProcess();

len=space();

while((len!=0)&&(ready!=NULL))

{

ch=getchar();

h++;

printf("\n 运行次数:%d \n",h);

p=ready;

ready=p->next;

p->next=NULL;

p->state='R';

PCB\* pr;

showInfo(p);

pr=ready;

while(pr!=NULL){

showInfo(pr);

pr=pr->next;

}

(p->workTime)++;

if(p->workTime==p->runTime){

printf("\n 进程%d 已结束。\n",p->PID);

free(p);

}

else{

(p->priority)++;

p->state='w';

PCBsort();

}

printf("\n 按任一键继续 ......");

}

printf("\n\n 进程已经完成 .\n");

ch=getchar();

}

int main()

{

printf("—————————————————优先级调度算法—————————————————\n");

printf("输入进程数目：");

scanf("%d",&num);

priorityRun();

}