Project 6: Banker's Algorithm

白骐硕 518030910102

任务概述

For this project, you will write a program that implements the banker's algorithm discussed in Section 8.6.3. Customers request and release resources from the bank. The banker will grant a request only if it leaves the system in a safe state. A request that leaves the system in an unsafe state will be denied. Although the code examples that describe this project are illustrated in C, you may also develop a solution using Java.

具体实现

全局变量定义

首先介绍需要定义的若干全局变量，即银行家算法需要用到的4个矩阵：

- available
- maximum
- allocation
- need

```
#define NUMBER_OF_CUSTOMERS 5
#define NUMBER_OF_RESOURCES 4

//available
int available[NUMBER_OF_RESOURCES];

//maximum
int maximum[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];

//allocation
int allocation[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];

//need
int need[NUMBER_OF_CUSTOMERS][NUMBER_OF_RESOURCES];
```

main函数

在main函数需要实现的功能为：

- 根据命令行参数和读入文件对上述4个矩阵进行初始化
- 读入用户输入的指令（RQ，RL，*，exit），根据指令调用相应的处理函数

```
int main(int argc, char *argv[]){
    //get commandline input and initialize the avaliable
    if(get_commandline(argc,argv)!=0){
        return 0;
    }
    if(init_maximum() != 0){
```

```
        return 0;
    }
    printf("Initialization Success! \n");

    char command[10];
    printf(">>");
    while(scanf("%s", command) == 1){
        if(strcmp(command, "RQ") == 0){
            //RQ
            request();
        }else if(strcmp(command, "RL") == 0){
            //RL
            release();
        }else if(strcmp(command, "exit") == 0){
            //exit
            break;
        }else if(strcmp(command, "*") == 0){
            //display matrix
            display_matrix();
        }else{
            //Illegal input
            print_usage();
        }
        printf(">>");
    }
    return 0;
}
```

下面对main函数中所调用的各部分功能进行详细介绍。

初始化

首先处理用户的命令行参数，用以初始化available

```
int get_commandline(int argc, char *argv[]){
    if(argc != NUMBER_OF_RESOURCES+1){
        fprintf(stderr, "Incorrect input. Resource number should be %d \n",
NUMBER_OF_RESOURCES);
        return -1;
    }
    for(int i = 0; i < NUMBER_OF_RESOURCES; ++i){
        available[i] = atoi(argv[i+1]);
    }
    return 0;
}
```

其次，从文件中读入maximum矩阵，作为其初始值

```
int init_maximum(){
    FILE *f = fopen("maximum_init.txt", "r");
    if(f == NULL){
        fprintf(stderr, "Failed to open maximum_init.txt \n");
        return -1;
    }
    for(int i = 0; i < NUMBER_OF_CUSTOMERS; ++i){
        for(int j = 0; j < NUMBER_OF_RESOURCES; ++j){
            fscanf(f, "%d", &maximum[i][j]);
```

```
            need[i][j] = maximum[i][j];
        }
    }
    fclose(f);
    return 0;
}
```

至此，初始化完成。

Request（RQ命令）

首先读入用户输入的customer_num，和各资源的请求数。

```
int request[NUMBER_OF_RESOURCES];
int customer_num;
scanf("%d",&customer_num);
for(int i = 0; i < NUMBER_OF_RESOURCES; ++i){
    scanf("%d", &request[i]);
}
```

然后，判断输入是否合法，判断依据：

- customer_num是否属于范围 [0, NUMBER_OF_CUSTOMERS)
- request[i] 是否为非负数
- request[i] 是否小于等于 need[customer_num][i]
- request[i] 是否小于等于 available[i]

如果输入合法，则将资源进行分配，更新矩阵。

对分配后的状态进行安全判断，看当前是否仍处于safe state。根据银行家算法的伪代码，编写下图所示的c语言代码。

```
int check_safe(){
    int work[NUMBER_OF_RESOURCES],Finish[NUMBER_OF_CUSTOMERS];
    memcpy(work,available,NUMBER_OF_RESOURCES*sizeof(int));
    memset(Finish,0,NUMBER_OF_CUSTOMERS*sizeof(int));
    for (int i = 0; i < NUMBER_OF_CUSTOMERS; ++i) {
        int flag = 0;
        for (int j = 0; j < NUMBER_OF_CUSTOMERS; ++j) {
            if(!Finish[j] && is_leq_matrix(need[j],work, NUMBER_OF_RESOURCES)) {
                flag = 1;
                Finish[j] = 1;
                for (int k = 0; k < NUMBER_OF_RESOURCES; ++k) {
                    work[k] += allocation[j][k];
                }
                break;
            }
        }
        if(!flag) {
            return  0;
        }
    }
    return 1;
}
```

如果当前状态仍处于 safe state，则告知用户资源请求成功；否则，恢复刚才已分配的资源，并告知用户资源请求导致 unsafe state，请求失败。

Release（RL命令）

首先，读入用户的输入：

- customer_num
- release[]

之后，判断输入是否合法，判断依据：

- customer_num 是否属于范围 [0, NUMBER_OF_CUSTOMERS)
- release[i] 是否属于范围 [0, allocation[customer_num][i]]

如果，输入合法，则释放资源，更新allocation，need，和available矩阵；否则告知用户，释放资源失败。

release的完整代码如下：

```
void release(){
    int release[NUMBER_OF_RESOURCES];
    int customer_num;
    scanf("%d", &customer_num);
    for(int i = 0; i < NUMBER_OF_RESOURCES; ++i){
        scanf("%d", &release[i]);
    }
    //check input
    if(customer_num < 0 || customer_num >= NUMBER_OF_CUSTOMERS){
        printf("Illegal customer number! \n");
        printf("Release Denied! \n");
        return;
    }
    for(int i = 0; i<NUMBER_OF_RESOURCES; ++i){
        if(release[i] < 0){
            printf("For resource %d: release less than 0! \n", i);
            printf("Release Denied! \n");
            return;
        }
        if(release[i] > allocation[customer_num][i]){
            printf("For resource %d: release %d, but allocation is %d \n", i,
release[i], allocation[customer_num][i]);
            printf("Release Denied! \n");
            return;
        }
    }
    release_resources(customer_num, release);
    printf("Release Success! \n");
}

void release_resources(int customer_num, int release[]){
    for(int i = 0; i < NUMBER_OF_RESOURCES; ++i){
        allocation[customer_num][i] -= release[i];
        available[i] += release[i];
        need[customer_num][i] += release[i];
    }
}
```

***命令**

只需要将所定义的四个矩阵依次打印出来即可，具体代码在此不再赘述。

非法命令

如果用户的输入命令不为 RQ,RL,*,exit，则为用户提示正确命令的使用方法，并请用户重新输入。

实验结果

测试所用的初始值为：

available: 10 6 7 8

maximum:

6 4 7 3

4 2 3 2

2 5 3 3

6 3 3 2

5 6 7 5

首先，通过*命令打印上述初始值。

```
qsbai@qsbai-virtual-machine:~/os-prj/6$ ./main 10 6 7 8
Initialization Success!
>>*
============================================================
Available resources:
10 6 7 8
------------------------------------------------------------
Maximum resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 6 7 5
------------------------------------------------------------
Allocated resources for each customer:
Customer 0: 0 0 0 0
Customer 1: 0 0 0 0
Customer 2: 0 0 0 0
Customer 3: 0 0 0 0
Customer 4: 0 0 0 0
------------------------------------------------------------
Needed resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 6 7 5
============================================================
```

测试RQ，customer 4 请求 0 6 0 0，并打印请求后的矩阵。

```
>>RQ 4 0 6 0 0
Request Success!
>>*
================================================================
Available resources:
10 0 7 8
----------------------------------------------------------------
Maximum resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 6 7 5
----------------------------------------------------------------
Allocated resources for each customer:
Customer 0: 0 0 0 0
Customer 1: 0 0 0 0
Customer 2: 0 0 0 0
Customer 3: 0 0 0 0
Customer 4: 0 6 0 0
----------------------------------------------------------------
Needed resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 0 7 5
================================================================
```

测试RQ，customer 0 请求 6 0 0 0，结果提示该请求会导致 unsafe state，请求失败。

```
>>RQ 0 6 0 0 0
Request will result in a unsafe state.
Request Denied!
```

测试RL，customer 4 释放 0 5 0 0，并打印释放后的矩阵。

```
>>RL 4 0 5 0 0
Release Success!
>>*
============================================================
Available resources:
10 5 7 8
------------------------------------------------------------
Maximum resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 6 7 5
------------------------------------------------------------
Allocated resources for each customer:
Customer 0: 0 0 0 0
Customer 1: 0 0 0 0
Customer 2: 0 0 0 0
Customer 3: 0 0 0 0
Customer 4: 0 1 0 0
------------------------------------------------------------
Needed resources for each customer:
Customer 0: 6 4 7 3
Customer 1: 4 2 3 2
Customer 2: 2 5 3 3
Customer 3: 6 3 3 2
Customer 4: 5 5 7 5
============================================================
```

测试RL，customer 4 释放 2 0 0 0，结果提示customer 4没有足够的资源满足该释放，释放失败。

```
>>RL 4 2 0 0 0
For resource 0: release 2, but allocation is 0
Release Denied!
```

之后测试若干非法输入：

```
>>RL 4 -1 0 0 0
For resource 0: release less than 0!
Release Denied!
>>RQ 0 -1 0 0 0
For resource 0: request less than 0!
Request Denied!
>>RQ 0 11 0 0 0
For resource 0: request 11, but need is 6
Request Denied!
```