

# OS lab8 Report

name: 刘乐奇

sid: 12011327

Ubuntu用户名: lynchrocket

## 1. 使用mutex解决too much milk problem，直接修改milk.c，报告中包含代码截图及运行结果截图。

代码截图

```
C milk.c > mom()
4  #include <string.h>
5  #include <fcntl.h>
6  #include <time.h>
7  #include <sys/stat.h>
8
9  pthread_mutex_t mutex;
10
11 void *mom(){
12     int fd;
13     printf("Mom comes home.\n");
14     sleep(rand()%2+1);
15     printf("Mom checks the fridge.\n");
16
17     pthread_mutex_lock(&mutex);
18     fd=open("fridge", O_CREAT|O_RDWR|O_APPEND, 0777);
19     if(lseek(fd,0,SEEK_END)==0){
20         printf("Mom goes to buy milk...\n");
21         //sleep(rand()%2+1);
22         printf("Mon comes back.\n");
23         if(lseek(fd,0,SEEK_END)>0)
24             printf("What a waste of food! The fridge can not hold so much milk!\n");
25     }else{
26         write(fd,"milk",4);
27         printf("Mom puts milk in fridge and leaves.\n");
28     }
29 }else{
30     printf("Mom closes the fridge and leaves.\n");
31 }
32 close(fd);
33 pthread_mutex_unlock(&mutex);
34 }
35
36 void *dad(){
37     int fd;
38     printf("Dad comes home.\n");
39     sleep(rand()%2+1);
```

```

6 void *dad(){
7     int fd;
8     printf("Dad comes home.\n");
9     sleep(rand()%2+1);
10    printf("Dad checks the fridge.\n");
11
12    pthread_mutex_lock(&mutex);
13    fd=open("fridge", O_CREAT|O_RDWR|O_APPEND, 0777);
14    if(lseek(fd,0,SEEK_END)==0){
15        printf("Dad goes to buy milk...\n");
16        //sleep(rand()%2+1);
17        printf("Dad comes back.\n");
18        if(lseek(fd,0,SEEK_END)>0)
19            printf("What a waste of food! The fridge can not hold so much milk!\n");
20        else{
21            write(fd,"milk",4);
22            printf("Dad puts milk in fridge and leaves.\n");
23        }
24    } else{
25        printf("Dad closes the fridge and leaves.\n");
26    }
27    close(fd);
28    pthread_mutex_unlock(&mutex);
29 }

```

```

int main(int argc, char * argv[]) {
    srand(time(0));
    pthread_t p1, p2;
    int fd = open("fridge", O_CREAT|O_RDWR|O_TRUNC , 0777); //empty the fridge
    close(fd);

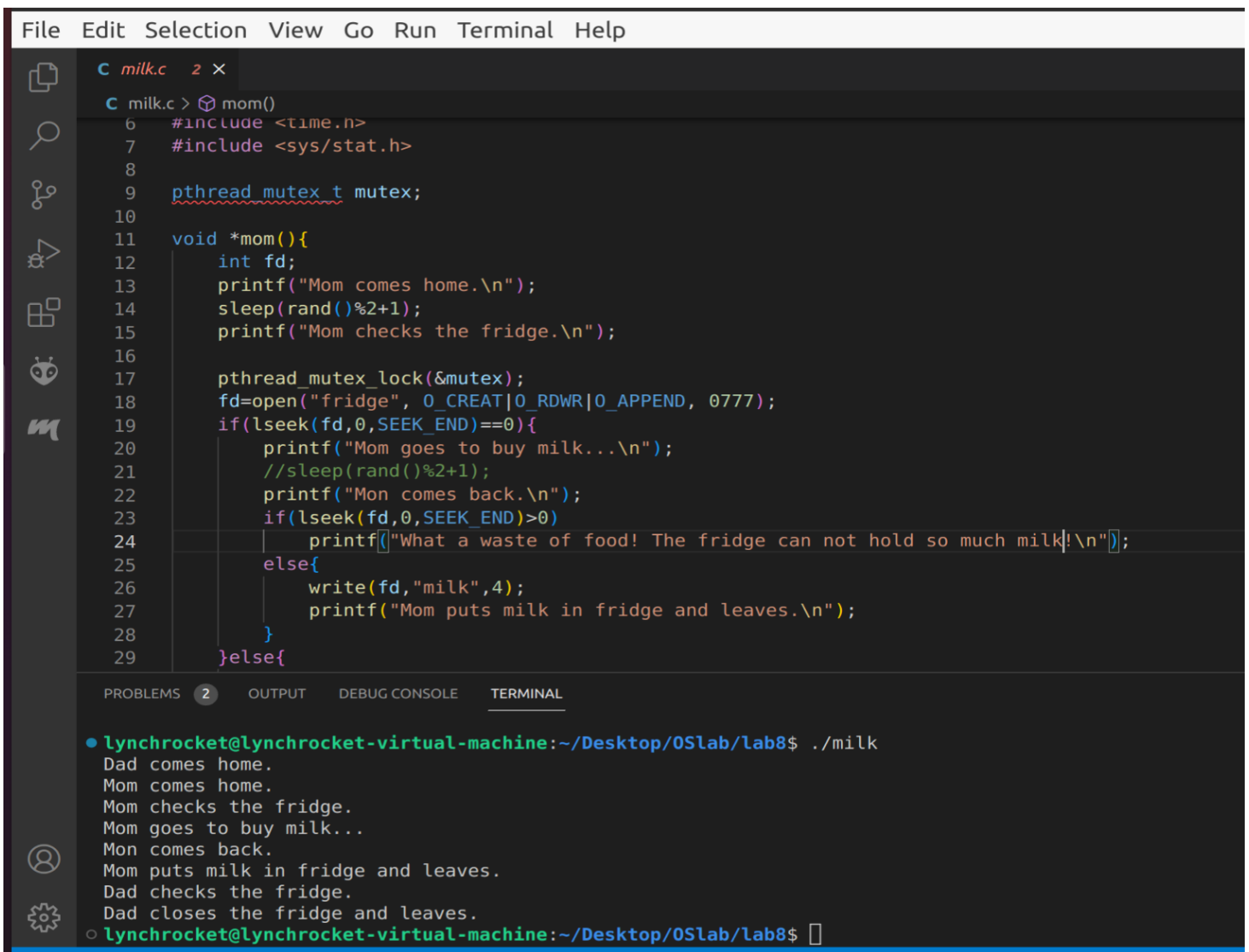
    pthread_mutex_init(&mutex, NULL);
    // Create two threads (both run func)
    pthread_create(&p1, NULL, mom, NULL);
    pthread_create(&p2, NULL, dad, NULL);

    // Wait for the threads to end.
    pthread_join(p1, NULL);
    pthread_join(p2, NULL);

    pthread_mutex_destroy(&mutex);
}

```

运行结果截图



```
File Edit Selection View Go Run Terminal Help

C milk.c 2 x
C milk.c > mom()
6 #include <time.h>
7 #include <sys/stat.h>
8
9 pthread_mutex_t mutex;
10
11 void *mom(){
12     int fd;
13     printf("Mom comes home.\n");
14     sleep(rand()%2+1);
15     printf("Mom checks the fridge.\n");
16
17     pthread_mutex_lock(&mutex);
18     fd=open("fridge", O_CREAT|O_RDWR|O_APPEND, 0777);
19     if(lseek(fd,0,SEEK_END)==0){
20         printf("Mom goes to buy milk...\n");
21         //sleep(rand()%2+1);
22         printf("Mon comes back.\n");
23         if(lseek(fd,0,SEEK_END)>0)
24             printf("What a waste of food! The fridge can not hold so much milk!\n");
25         else{
26             write(fd,"milk",4);
27             printf("Mom puts milk in fridge and leaves.\n");
28         }
29     }else{
30
31     }
32 }
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL

```
lynchrocket@lynchrocket-virtual-machine:~/Desktop/OSlab/lab8$ ./milk
Dad comes home.
Mom comes home.
Mom checks the fridge.
Mom goes to buy milk...
Mon comes back.
Mom puts milk in fridge and leaves.
Dad checks the fridge.
Dad closes the fridge and leaves.
lynchrocket@lynchrocket-virtual-machine:~/Desktop/OSlab/lab8$
```

2. 基于milk.c使用condition variable实现第四步中new problem的solution，报告中需要包含main(), mom(), dad(), you(), sister()的实现代码截图，以及运行结果截图。

变量定义

```
#define MAX_BOTTLE 3
int count = 0;
pthread_mutex_t mutex;
pthread_cond_t cond;
```

main(), mom(), dad(), you(), sister()的实现代码截图

```
int main(int argc, char * argv[]) {  
    srand(time(0));  
    pthread_t p1, p2, p3, p4;  
  
    pthread_mutex_init(&mutex, NULL);  
    pthread_cond_init(&cond, NULL);  
    // Create two threads (both run func)  
    pthread_create(&p1, NULL, mom, NULL);  
    pthread_create(&p2, NULL, dad, NULL);  
    pthread_create(&p3, NULL, you, NULL);  
    pthread_create(&p4, NULL, sister, NULL);  
  
    // Wait for the threads to end.  
    pthread_join(p1, NULL);  
    pthread_join(p2, NULL);  
    pthread_join(p3, NULL);  
    pthread_join(p4, NULL);  
  
    pthread_mutex_destroy(&mutex);  
    pthread_cond_destroy(&cond);  
}
```

```
void *mom(){
    printf("Mom came home.\n");
    sleep(rand()%2+1);
    int max_bottle = MAX_BOTTLE;
    while(max_bottle--){
        pthread_mutex_lock(&mutex);
        printf("Mom checked the fridge.\n");
        count++;
        printf("Mom brought a milk.\n");
        printf("in producer count is %d\n", count);
        if(count > 0){
            printf("Mom reminded Dad or You.\n");
            pthread_cond_signal(&cond);
        }
        printf("Mom released a lock.\n");
        pthread_mutex_unlock(&mutex);
    }
}
```

```
void *dad(){
    printf("Dad came home.\n");
    sleep(rand()%2+1);
    int max_bottle = MAX_BOTTLE;
    while(max_bottle--){
        pthread_mutex_lock(&mutex);
        printf("Dad checked the fridge.\n");
        if(count <= 0){
            printf("Dad was waiting for milk.\n");
            pthread_cond_wait(&cond, &mutex);
            printf("Dad got milk.\n");
        }
        count--;
        printf("in consumer count is %d\n", count);
        printf("Dad released a lock.\n");
        pthread_mutex_unlock(&mutex);
    }
}
```

```
void *you(){
    printf("You came home.\n");
    sleep(rand()%2+1);
    int max_bottle = MAX_BOTTLE;
    while(max_bottle--){
        pthread_mutex_lock(&mutex);
        printf("You checked the fridge.\n");
        if(count <= 0){
            printf("You were waiting for milk.\n");
            pthread_cond_wait(&cond, &mutex);
            printf("You got milk.\n");
        }
        count--;
        printf("in consumer count is %d\n", count);
        printf("You released a lock.\n");
        pthread_mutex_unlock(&mutex);
    }
}
```

```
void *sister(){
    printf("Sister came home.\n");
    sleep(rand()%2+1);
    int max_bottle = MAX_BOTTLE;
    while(max_bottle--){
        pthread_mutex_lock(&mutex);
        printf("Sister checked the fridge.\n");
        count++;
        printf("Sister brought a milk.\n");
        printf("in producer count is %d\n", count);
        if(count > 0){
            printf("Sister reminded Dad or You.\n");
            pthread_cond_signal(&cond);
        }
        printf("Sister released a lock.\n");
        pthread_mutex_unlock(&mutex);
    }
}
```

运行结果截图



● lynchrocket@lynchrocket-virtual-machine:~/Desktop/OSlab/lab8\$ ./milk

Dad came home.  
Mom came home.  
Sister came home.  
You came home.  
Sister checked the fridge.  
Sister brought a milk.  
in producer count is 1  
Sister reminded Dad or You.  
Sister released a lock.  
Sister checked the fridge.  
Sister brought a milk.  
in producer count is 2  
Sister reminded Dad or You.  
Sister released a lock.  
Sister checked the fridge.  
Sister brought a milk.  
in producer count is 3  
Sister reminded Dad or You.  
Sister released a lock.  
Dad checked the fridge.  
in consumer count is 2  
Dad released a lock.  
Dad checked the fridge.  
in consumer count is 1  
Dad released a lock.  
Dad checked the fridge.  
in consumer count is 0  
Dad released a lock.  
Mom checked the fridge.  
Mom brought a milk.  
in producer count is 1  
Mom reminded Dad or You.  
Mom released a lock.  
Mom checked the fridge.  
Mom brought a milk.  
in producer count is 2  
Mom reminded Dad or You.  
Mom released a lock.  
Mom checked the fridge.  
Mom brought a milk.  
in producer count is 3  
Mom reminded Dad or You.  
Mom released a lock.  
You checked the fridge.  
in consumer count is 2  
You released a lock.  
You checked the fridge.  
in consumer count is 1  
You released a lock.  
You checked the fridge.  
in consumer count is 0  
You released a lock.

○ lynchrocket@lynchrocket-virtual-machine:~/Desktop/OSlab/lab8\$