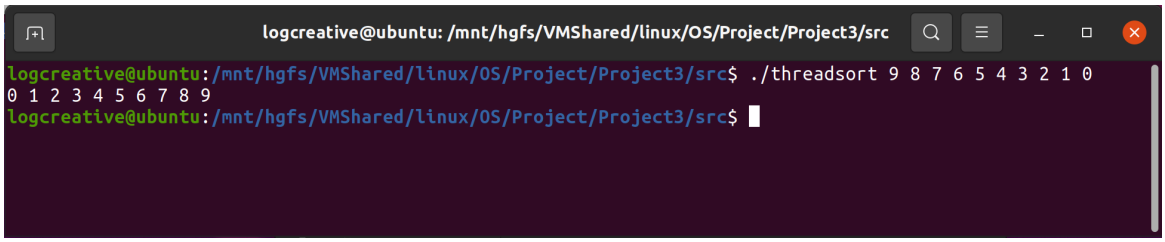
操作系统(D)

# 项目 3

李子龙 518070910095

2021 年 3 月 20 日

## 一 多线程排序程序




使用命令行的参数获取需要排序的数组，使用动态内存分配原数组和排序数组。原数组两边分别开一个线程用冒泡排序，最后归并两个数组到排序数组中。定义了一个结构体用于传递参数：

```
typedef struct {
    int start;
    int end;
} sort_param;
```

**Listing 1:** src/threadsort.c

```
#include<pthread.h>
#include<stdio.h>
#include<stdlib.h>

int* old_list;
int* sort_list;

typedef struct {
    int start;
    int end;
} sort_param;

void bubblesort(sort_param* sp){
    int start = sp->start;
    int end = sp->end;

    int flag = 1;
    for(int i = start + 1; i <= end && flag; ++i){
        flag = 0;
```

```
        for(int j = start; j <= end - i + start; ++j){
            if(old_list[j+1]<old_list[j]){
                int tmp = old_list[j];
                old_list[j] = old_list[j+1];
                old_list[j+1] = tmp;
                flag = 1;
            }
        }
    }

    pthread_exit(0);
}

void mergearray(int mid, int end){
    int left = 0;
    int right = mid + 1;
    int cur = 0;
    while(left<=mid && right <= end)
        if(old_list[left]<=old_list[right])
            sort_list[cur++] = old_list[left++];
        else sort_list[cur++] = old_list[right++];

    while(left<=mid) sort_list[cur++] = old_list[left++];
    while(right<=end) sort_list[cur++] = old_list[right++];
}

int main(int argc, char *argv[]){
    if(argc == 1){
        fprintf(stderr, "Please input the array!\n");
        return 1;
    }

    old_list = (int*)malloc((argc-1)*sizeof(int));
    sort_list = (int*)malloc((argc-1)*sizeof(int));
    for(int i = 1; i < argc; ++i)
        old_list[i-1] = atoi(argv[i]);

    pthread_t sorting_thread[2];
    pthread_attr_t attr[2];

    int mid = (argc-2)/2;

    pthread_attr_init(&attr[0]);
    sort_param *sp0 = (sort_param*) malloc(sizeof(sort_param));
    sp0->start = 0;
    sp0->end = mid;
    pthread_create(&sorting_thread[0],&attr[0],bubblesort,sp0);
    pthread_join(sorting_thread[0],NULL);

    pthread_attr_init(&attr[1]);
    sort_param *sp1 = (sort_param*) malloc(sizeof(sort_param));
    sp1->start = mid + 1;
    sp1->end = argc - 2;
    pthread_create(&sorting_thread[1],&attr[1],bubblesort,sp1);
    pthread_join(sorting_thread[1],NULL);
```

```
    mergearray(mid,argc-2);

    for(int i = 0; i < argc-1; ++i)
        fprintf(stdout, "%d ", sort_list[i]);
    fprintf(stdout, "\n");

    free(old_list);
    free(sort_list);
    return 0;
}
```

## 二　分离–联合排序程序